Technische Universiteit Delft

Bayesian networks

in performance of cyclists

M.H.M. Christianen

TU Delft

1

[1] Reference for figure on front page: http://teamsunweb.com/fanzone/wallpapers/

Bayesian networks

in performance of cyclists

(Dutch title: Bayesiaanse netwerken in de prestaties van een wielrenner)

by

M.H.M. Christianen

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended publicly on Friday July 14, 2017 at 10:30 AM.

Student number:	4376153	
Project duration:	March 1, 2017 – July 14, 2017	
Supervisors:	Dr. G. F. Nane,	TU Delft
	MSc. A. H. M. van Erp	Team Sunweb
Other members of the thesis committee:	Dr. J. L. A. Dubbeldam,	TU Delft
	Dr. J. G. Spandaw,	TU Delft

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

0.1. Abstract

This report examines the possibility of modeling the performance of professional cyclists of Team Sunweb using Bayesian Networks. This research has an objective to see how these structures work and how they fit in the complex world of cycling. We want to compare different cyclists of Team Sunweb in the Grand Tours (Giro d'Italia, Tour de France and Vuelta a España) of the year 2016 and build a model for the leader - who is supported by a group of helpers - during different stages in a given race and see if we can predict the pedal power in the crucial part of the race, i.e. the sprint or a last difficult climb. We can conclude that the Bayesian network we created with the combination of help from an expert and the data captures the most common relationships between all variables, but that the model doesn't reveal surprising relationships or good predictions.

Contents

1

INTRODUCTION

1.1. BACKGROUND IN CYCLING

1.1.1. IN GENERAL

Cycling is very popular in some major European countries like France and Germany, but also in the Netherlands this sport is very attractive. Every year sport fans are becoming glued to their tv screens when the Tour de France is held. Only in the Netherlands there are every day 700.000 watchers on average!

Of course, the Tour de France is not the only big event in the world of cycling. It is one of many. To name a few of them: Tour of Flanders, Paris-Roubaix, Critérium du Dauphiné, Paris-Nice, Giro d'Italia, Vuelta a España and the Eneco Tour (our Dutch "Grand Tour"!). About a year and a half ago TU Delft, and in particular the department Sports Engineering, got in touch with the professional cycling team Sunweb, named at the time team Giant Alpecin, that performed in the races mentioned above. They were interested in the scientific contribution of TU Delft to improve their performance in cycling. So what is behind team Sunweb?

1.1.2. TEAM SUNWEB

Team Sunweb is a professional German cycling team that participates in most big cycling events through the year. But don't underestimate the Dutch influence in this crew; right now ten of the twenty-five riders under contract are Dutch! And how could we forget? The team with the cyclist that won the Giro d'Italia 2017 a few weeks ago! Tom Dumoulin, the first male Dutch rider that won the Giro and by this victory the third Dutchman that won one of the Grand Tours ever. But how about Warren Barguil, John Degenkolb and Nikias Arndt? Does these names

ring a bell? If so, you are a fervent fan of team Sunweb and you have been following these men in 2016! And if not, I can ensure you that you will know them after reading this report. In any case I am sure you enjoy reading...

Mag ik dat zo zeggen? Ja, dat mag ik zo zeggen

Mart Smeets

1.2. Overview of the research

The cyclists that I mentioned above are the men that are expected to win the race for Sunweb with the help of the rest of the team. The roles vary from race to race, but in 2016 these five riders were the leaders. The helpers, or so called domestiques, bring water and food from team cars and shield teammates from opponents, but the main job contains protection. This means that much of the rest of the team's effort is to push aside the air in front of them. In this way a leader is riding in the slipstream of another rider and this is easier than taking the lead. Consequently the leaders conserve energy until the last few hundred meters or the last climb of the day. This resulted in an important question from team Sunweb:

Main question:

How good is the performance of the leader during a race and how is this influenced by the helpers?

Our main approach to solve this problem is to use a Bayesian Network to model the performance of the leader and his helpers. This is possible because Team Sunweb collected a huge amount of data of all races for every individual rider. On the way we will run into some difficulties that arise from the data, but we are going to discuss this in the next chapter. After this a comparison of networks will be made when looking at different cyclists during one of the Grand Tours. At last, but not least, we will perform some prediction. To know what is worth predicting we definitely need to have a look at the data. This will also be examined in the next chapter. All this has the effect that we will answer the following sub questions.

Sub questions:

1. What is the influence on the structure of the data caused by the three Grand Tours?
2. How does the structure of the data change when there is a difference in race type?
3. Does the structure of the data change when we encounter a different rider type? If so, how?
4. What is the influence on the structure of the number of days riders are in succession in race?

1.3. DATA

The data that we have at our disposal consists of 63 days in total. In Table 1.1 we can see how the data is distributed over the Grand Tours: Giro d'Italia, Tour de France and Vuelta a España. These three races are similar in format being three week races with daily stages with two resting days. The stages have different types. It can differ from mountain and hill climbs to flat races and team and individual time trials.

Race	Number of days	Number of cyclists
Giro d'Italia	21	9
Tour de France	21	9
Vuelta a España	21	8

Table 1.1: Info about the different races where are examining; the number of days and cyclists in every Grand Tour.

Luckily team Sunweb has been recording a lot of information about these races. There is a small device on every bicycle. This device is from Pioneer and it measures details of the race like altitude and gps-data, but also specific cyclist activity like heart rate and speed.

1.4. METHODOLOGY

To answer the questions that are stated in section 1.2 we are going to use the help of Bayesian networks. A formal definition and a detailed description follow in section 3.2, but let's see what we can say about them in this stage.

To model the complex world of cycling and especially the performance, we need a framework that accounts for uncertainty and depicts relationships between factors that influence each other. As described in Koller & Friedman (2009) [7] a Bayesian network provides a way to do this compactly. In this way we can use a graph-based representation as the basis for encoding a complex distribution. In this representation (a first example can be found in Figure 3.1) the nodes correspond to the variables and the edges to direct probabilistic interactions between them.

A huge advantage of a Bayesian network is the effective construction by learning the model from data. How this can be done is explained in chapter 3.2. The models that are created in this way are usually much better reflections than models that are purely hand-constructed. Sometimes they can reveal surprising relationships. Another advantage is the fact that the distribution can we written very clear, even in cases where the explicit representation of the joint distribution is astronomically large [7].
The last advantage is the graphical representation. As we will see the type of representation is transparent; a Bayesian networks provides us a accurate reflection of our understanding in the 'real world' and so can be used to give us new insights, answers and predictions.

1.5. STRUCTURE OF THE THESIS

In chapter 2, an extensive introduction about the data of team Sunweb is given. This chapter starts with an overview of the data and then the issues that arise from the data are discussed. Chapter 3 is subdivided in the part where we treat the theoretical background of Bayesian networks and the way how to build such models from data and the part where we build our network. In the first part we will discuss two methods for learning the structure of a Bayesian network. Besides that we will investigate the differences between the two ways of learning the structure. In the second part we will then build our own network in cycling by the techniques we learned in this chapter. In chapter 5 we will - based on the model build in section 4.5 - make predictions of the performance of the leader during specific days and see how good these predictions are. In conclusion, the formulated research questions that we presented in section 1.2 are answered in chapter 6. In the appendix we summarize the information of our data in a compact way. At last we put down our code that we used to produce (in Matlab and R) all figures and results in the report.

2

DATA ANALYSIS

2.1. DESCRIPTION OF THE DATA

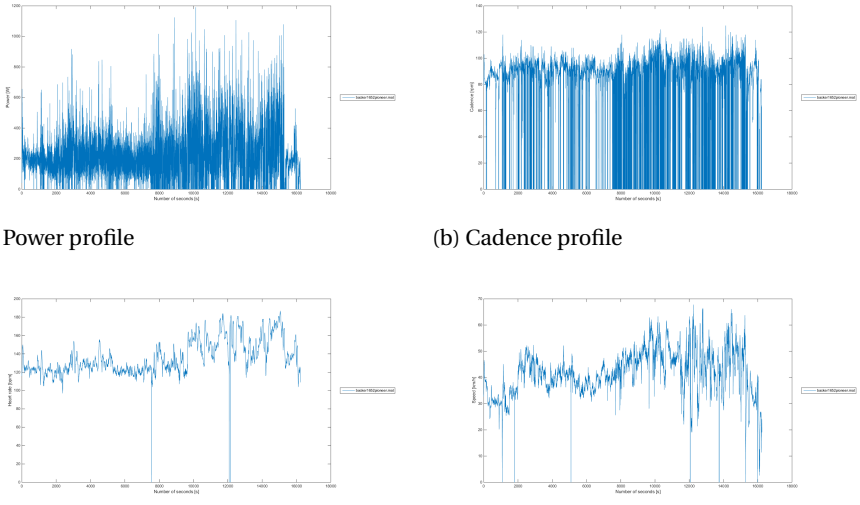
In this chapter we will have a closer look at the available data. As mentioned before - in section 1.3 - there are 69 variables accessible. With the help of an expert in the field of cycling, we decided to use the following 8 variables in the analysis for the performance of cyclists. We make a distinction between variables with details of the race:

- Riding Time: elapsed time from the moment a cyclist turned the device on, in seconds [s].
- Altitude: height above sea level of a location, in meters [m].
- Distance: the total traveled distance so far in meters [m].
- Temperature: the temperature during the race in degrees Celsius [°C]

These race activity variables should be the same for all riders. In this way we are sure that we are comparing data from different cyclists for the same day and race. But as we shall see, in the next section about data issues, this is not always the case. The other group of variables consists of information about cyclist race activity that is very specific to a personal riding style:

- Pedal Power: the activity of riding a bicycle, in Watt [W].
- Cadence: the rate at which a cyclist is turning the pedals, in rate per minute [rpm].

- Heart Rate: the number of contractions of the heart per minute, in beats per minute [bpm].
- Speed: the speed of the cyclist, in kilometer per hour [km/h].



(a) Power profile

(b) Cadence profile

(c) Heart rate profile

(d) Speed profile

Figure 2.1: Illustration of the distribution of rider 1 his race activity variables on 8 May 2016.

In Figure 2.1 we plotted four variables: power, cadence, heart rate and speed against time. The device from Pioneer evaluates all these variables every second, so for one day we have around 16000 measurements. The power profile (a) of rider 1 and of all other riders is very capricious. Notable is the fact that the device measures a lot of zeros. This is because cyclists stop pedaling during a bend or when they are drafting behind each other. In the cadence profile (b) is a clear trend visible. The cadence circulates around 90 rounds per minute and falls back to zero frequently. This is also caused by the lack of pedaling. The heart rate profile (c) is very stable. There are only a few moments when the heart rate is zero, but this is probably caused by a poor signal of the heart rate monitor. The last figure with the speed profile (d) has also a capricious character. The moments when the speed drops to zero could be explained by facing a sharp turn or mistakes from the device. Furthermore it is good to see that some obvious relationships are confirmed by these figures. At the half of the race the pedal power increases, but so do the cadence, heart rate and speed. At the end the power drops to 200 Watt, but such a drop occurs in cadence, heart and speed too.

2.2. DATA ISSUES

The data of rider 1 in the last example was complete, i.e. for every variable we have meaningful information. Unfortunately this is not the case in many data samples; there is missing data. This does not mean a few missing measurements on the whole day, but no information for this variable for the whole day. For example, on 8 May 2016 we don't have a heart rate for rider 4 and on 11 May 2016 we don't have the GPS-location of rider 1. In the case of no GPS-location we decided to leave out this particular cyclist for this day, because of the following issue concerning the device.

This important issue is the moment when a cyclist turns on the device. Every rider can individually decide when to turn the device on, but we want to compare for example the speed and the pedal at the same moment, so we need to align or synchronize the data.

2.2.1. ALIGNMENT

As stated before; one of the more practical problems is that all riders turn the device on the moment they want to. This means that the data is not aligned. For example you can not compare the data in the beginning, because for some riders this is during their warming up and for others during the fifth kilometer in the race. So we have to deal with this problem first.

Luckily we are in possession of the GPS data for every race for almost every rider. Using this data we can find the first common GPS coordinates of all riders. So we are looking for the first moment when they reach the same longitude and latitude coordinates. We are looking for this moment in the first hour of the race, because in this part of the race the riders are together in most races. Now we want to remark that it is not possible to use the GPS coordinates of start and finish, since in almost every race these points - remarkably - did not appear in the coordinates of the riders. Let's see how this works in practice. In Figure 2.2 the GPS coordinates of all riders on 7 July 2016 and the first common point are drawn. For every day we assume that this first common point (in Figure 2.2 the yellow point) is the starting point for all riders. In the next figure, Figure 2.3, we zoom in on the GPS coordinates. Here we can see all different paths for the cyclists and in this way that you need some luck to find common coordinates and especially a common point in the first part of the race.

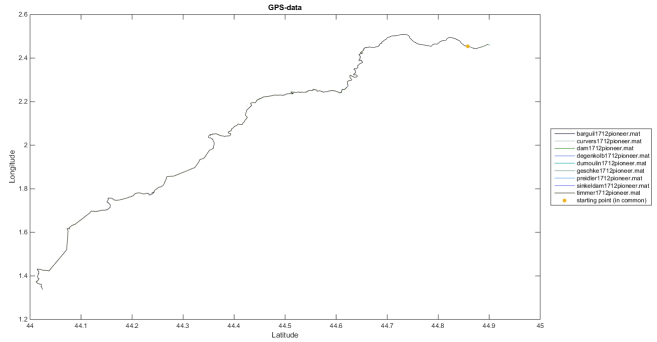


Figure 2.2: Full GPS coordinates of all riders on 7 July 2016 and the first common point.

In this figure only the last plotted coordinates, i.e. the coordinates from rider 2, are visible, because the rest is behind this line. To see that this is really the case, have a look at Figure 2.3.

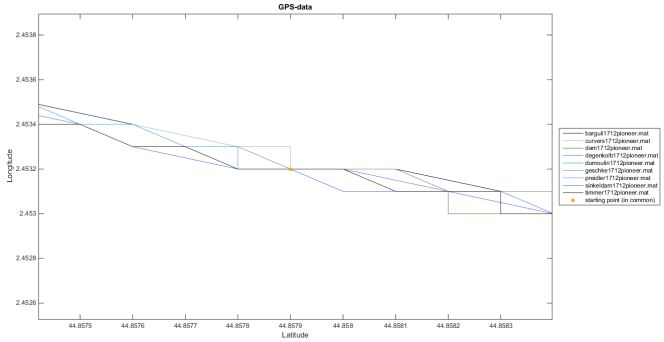


Figure 2.3: Detailed zoom of the GPS data of 9 different cyclists on 7 July 2016.

A remark about the plots above: the legend contains all the names of the cyclists and the starting point of the race. Figures 2.2 and 2.3 are also added to the appendix, Table A.1 and A.2, to have a better look at these figures. In the detailed figure of the GPS data we can see how precise the GPS-coordinates are and how difficult it can be to find a common point. However, in this way we were able to find a starting point for all races and thus a solution for the alignment problem.

But how good does this alignment work? In Figure 2.5 we can make the comparison between the original data and the aligned version based on synchronizing the GPS-coordinates. We compared the altitude, because this is a typical race variable and it should be the same for each rider for a given day. The following two figures are also added to the appendix in Table A.3 and A.4.

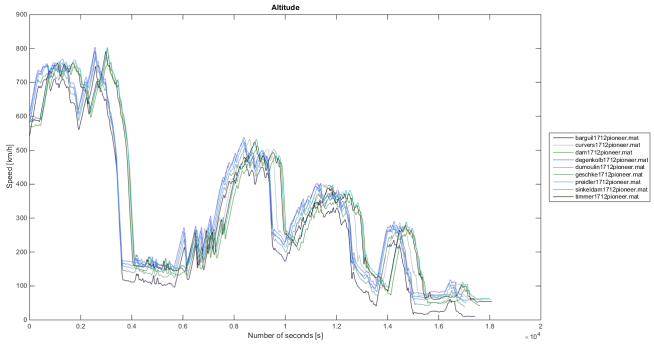


Figure 2.4: The original plot of the altitude on 7 July 2016

It looks like some cyclists turned their device earlier on than others. Compare this figure with Figure 2.5 where we shifted the data based on the first part of the race.

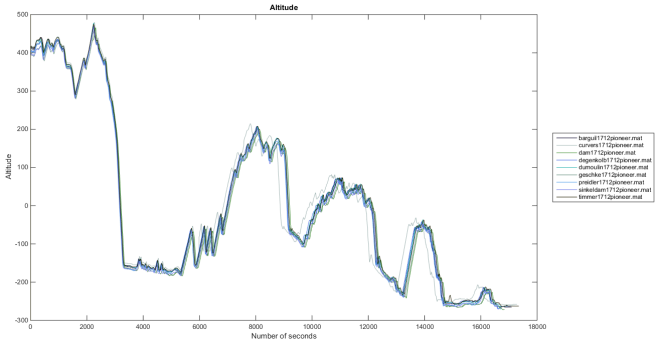


Figure 2.5: The aligned plot of the altitude based on the GPS coordinates on 7 July 2016.

From now on we will use the aligned data, so that we can compare all the race and cyclist variables for the same moment.

2.3. DEPENDENCIES

Now that we have aligned the data, we can return to our variables. Given the selected variables, a first approach is to look at the existing dependencies in the data (because it is an important tool in Bayesian networks). There are several correlation coefficients, but we will have a look at the Pearson product-moment correlation coefficient and Spearman's rank correlation coefficient, because they are generally used.

2.3.1. Pearson's correlation

The Pearson or product moment correlation is most widely used. The Pearson's correlation ρ between two random variables X and Y with expected values μ_X and μ_Y and standard deviations σ_X and σ_Y is defined as

$$\rho_{X,Y} = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

where cov stands for the covariance between X and Y and E is the expected value operator. The Pearson correlation coefficient is sensitive only to a linear relationship between two variables. So nonlinear relationships will not be reflected in the correlation coefficient. One advantage is that it reflects the noisiness and the direction of a linear relationship, but not the slope.

2.3.2. Spearman's correlation

The power of the Spearman's correlation coefficient is the fact that it does not only takes into account linear relationships, but any monotonic relationship.
The Spearman correlation coefficient is defined as the Pearson correlation coefficient between the ranked variables. Thus every element of the original data X and Y is transformed to its rank rg_X and rg_Y and the correlation coefficient is calculated by

$$\rho_{rg_X, rg_Y} = \frac{\text{cov}(rg_X, rg_Y)}{\sigma_{rg_X} \sigma_{rg_X}}$$

The covariance and standard deviation are still defined as before, but applied to the rank variables instead of the original data.

Both correlation coefficients meet three conditions: they are maximal 1 and minimal -1 and the correlation coefficient is 0 means that the variables are uncorrelated, but it needs to be emphasized that uncorrelatedness does not imply independence.

To see if there are some linear or monotonic relationships we can have a look at a scatterplot. This scatterplot is created for cyclist 3 on 8 May 2016, but the same relationships occur in figures like this for all other cyclists. There is one obvious linear relationship between riding time and distance and two monotonic relationships, namely the one between pedal power and cadence and cadence and speed. There seems to be a relationship between distance and temperature, but unfortunately not linear or monotonic (as can be seen in Figure 2.6).

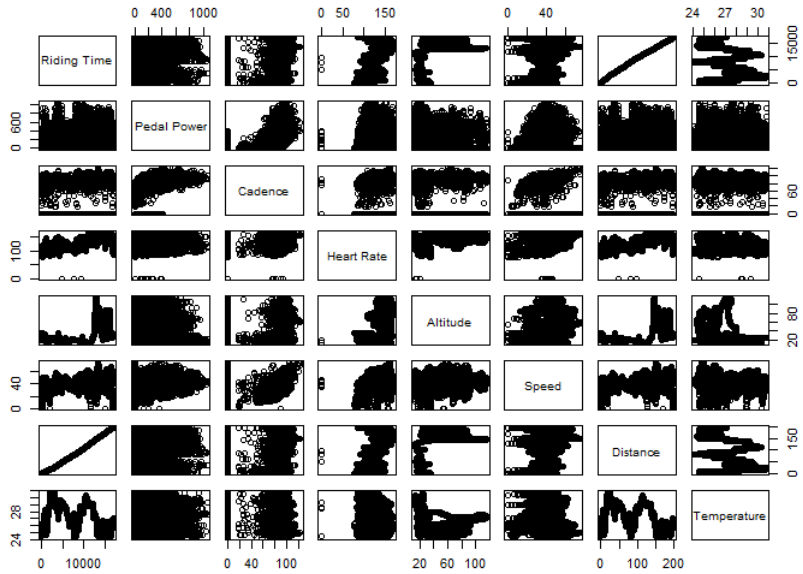
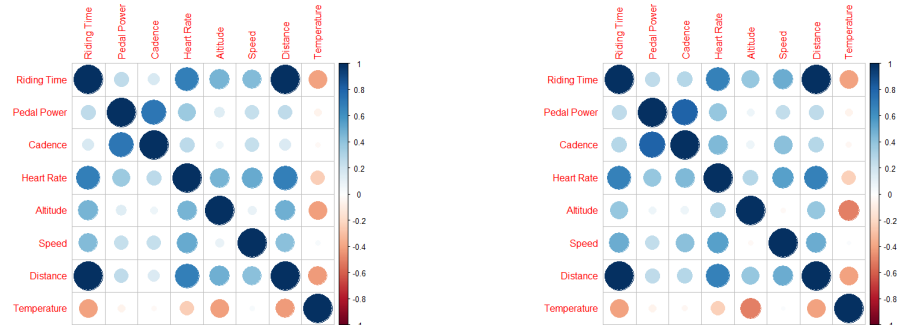


Figure 2.6: Scatterplot for all relevant variables of cyclist 3 on 8 May 2016.

The fact that the relationship between distance and temperature isn't linear or monotonic is captured in the following correlation plots (Figure 2.7).



(a) Pearson

(b) Spearman

Figure 2.7: Correlation plots for the relevant variables for the performance of cyclist 3 on 8 May 2016.

Indeed, in this correlation figures our suspicions are confirmed as stated before and seen in Figure 2.6, but other relationships between heart rate and distance

or between altitude and temperature are clearly visible.

Let us make one last remark: the color intensity and the size of the circle in Figure 2.6 are proportional to the correlation coefficients. So we don't get extra information from the intensity if we have the size of the circles.

3

BAYESIAN NETWORKS

3.1. INTRODUCTION

In general a model becomes useful if it helps understanding the situation we are modeling. Let's say we want to make a probabilistic model in, for example, health care. Everybody is sick sometimes and we really want to know what kind of illness we have. So given the symptoms you have, what is the probability that it is just flu? Or the other way around. You have been told that you are suffering from a disease and you are curious what symptoms you can expect. These are examples of questions that could be solved using a probabilistic network or so called Bayesian network (BN). A BN relates variables, like the symptoms of a disease, by a sort of dependency to other variables.

Suppose we have two kinds of variables that play a role indicating sickness. The fact that you do have a sore throat (yes or no) and your temperature. The main difference between these variables is that they are discrete and continuous. For the question: "Do you have a sore throat?" there are only two possible answers: yes and no. This means this is a discrete variable. For the question: "What is your temperature?" there are much more possible answers. Every number between 35 and 42[1] could be given as an answer.

3.2. THEORY AND EXAMPLES

To illustrate this and to introduce some new terminology we will have a look at a simple discrete BN. After that we will see how the theory changes, when we in-

[1] There could be extraordinary values like 33 or 44, which are taken into the analysis, but these numbers are just to indicate that most measured temperatures will be in this region

troduce continuous variables in a mixed BN, i.e. a network that consists of both discrete and continuous random variables.

The numbers for the example are taken from Kjærulff & Madsen(2003) [3], but we changed the context to health care. We have two possible indicators for sickness: X_1 and X_2. The first indicator X_1 represents a Sore Throat {yes,no} and X_2 represents a Headache {yes,no} and a variable X_3 that indicates if the person in question is sick. So the possible answers are again {yes,no}.

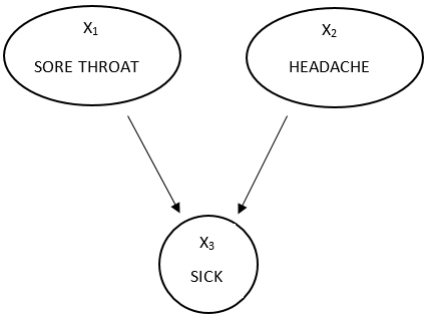


Figure 3.1: A simple Bayesian network consisting of three random variables where X_1, X_2 both represents sickness indicators and X_3 a variable that states of the person in question is sick.

In our example we are working with only three variables. The theory for Bayesian networks works - of course - also for n variables, where $n \in \mathbb{N}$. So let's see how the theory works for n variables and then come back to the example. For a more complete analysis we refer you to Jim Smith [1, p.179]). So let $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ be a joint probability mass function or density function of a vector of random variables. From the standard rules of probability the joint mass function or density $p(x)$ of $\mathbf{X}$ can be written as the product of conditional mass functions or conditional density functions. Thus

$$p(\mathbf{X}) = p(X_1)p_2(X_2|X_1)p_3(X_3|X_1, X_2)\ldots p_n(X_n|X_1, X_2, \ldots, X_{n-1}) \tag{3.1}$$

where $p_1(X_1)$ is the mass function of X_1 while the other probabilities are conditional probabilities based on the values before it. When all components of $\mathbf{X}$ are independent this $p(\mathbf{X})$ becomes the product of all $p_i(X_i), i = 1\ldots n$.

In most interesting models not all variables are independent of each other, like in our simple example. Variable X_3 is clearly dependent on X_1 and X_2, see Figure 3.1. Many of the functions $p_i(X_i|X_1, X_2, \ldots, X_{i-1})$ will often be an explicit function of components of $\mathbf{X}$ whose indices lie in a proper subset $Q_i \subset \{1, 2, \ldots, i-1\}$

Thus suppose

$$p_i(X_i|X_1,\ldots,X_{i-1}) = p_i(X_i|\mathbf{X}_{1,\ldots,i-1}) = p_i(X_i|\mathbf{X}_{Q_i}) \tag{3.2}$$

where the parent set $Q_i \subset \{1,2\ldots,i-1\}$ and let the remainder set $R_i \subset \{\{1,2,\ldots,i-1\} \setminus Q_i$, where we allow both Q_i and R_i to be empty. In the graph we call X_i a parent of X_j (where $i \neq j$ if there is a direct edge from X_i to X_j). The other way around we call, in this case, X_j a child of X_i. From now we put the parents for a variable X_i in the above defined set X_i. For example, in our simple example we call X_1 and X_2 parents of X_3. With our notation we obtain a new factorization formula of (3.1)

$$p(\mathbf{X}) = p_1(X_1)\prod_{i=2}^{n} p_i(X_i|\mathbf{X}_{Q_i}) \tag{3.3}$$

Now return to our problem description. We are dealing with three variables, so let $X = (X_1, X_2, X_3)$. So what is our joint probability function? We know that X_1 and X_2 don't depend on other variables and X_3 depends on both values. Thus we find:

$$p(\mathbf{X}) = p(X_1, X_2, X_3) = p(X_1)p(X_2)p(X_3|X_1, X_2) \tag{3.4}$$

Assume that $p(X_1 = \text{no}) = p(X_2 = \text{no}) = 0.9$ and $p(X_1 = \text{yes}) = p(X_2 = \text{yes}) = 0.1$, i.e. the probability of having a sore throat in the morning is equal to 0.1 and the same holds for having a headache. For the conditional probability $p(x_3|x_1, x_2)$ we put the information in a so called (conditional) probability table, because we need the information of indicators X_1 and X_2 and this is displayed as follows: the first two columns represent the outcomes of indicators X_1 and X_2 respectively and the third and fourth column give the probability of the possible outcomes of X_3.

		Sick	
Sore Throat	Headache	no	yes
no	no	0.98	0.02
no	yes	0.10	0.90
yes	no	0.15	0.85
yes	yes	0.05	0.95

Table 3.1: Conditional probability table for two discrete random variables; given the values {yes,no} for Sore Throat and Headache we find the probability that the person is sick or not.

Now we can give a formal definition of a discrete BN. We cite the definition of a Bayesian network (BN) from Jensen and Nielsen(2007) [2, p.33].

Definition 3.2.1. A discrete Bayesian network consists of the following:

- A set of variables and a set of directed edges between variables.
- Each variable has a finite set of mutually exclusive states.
- The variables together with the directed edges form an acyclic directed graph (traditionally abbreviated DAG); a directed graph is acyclic if there is no directed path $A_1 \to A_2 \to \ldots \to A_n$ so that $A_1 = A_n$.
- To each variable A with parents $B_1, \ldots, B_n$, a conditional probability table $P(A|B_1, \ldots, B_n)$ is attached.

So far, we have considered Bayesian networks for discrete random variables. It is time to extend our horizon and include continuous random variables in a mixed type BN. The standard assumption for the distribution of continuous variables in a BN is the Gaussian (normal) distribution. The density function for a Gaussian distribution with mean μ and variance σ^2 as

$$f(x) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{3.5}$$

We will focus on conditional linear Gaussian distributions. In Scutari & Denis (2015) [5] we find some general info about working with Gaussian BN's. The conditioning effect of the parent nodes is given by an additive linear term in the mean and does not affect the variance. This means that the variance of a random variable does not depend on the values of the parents.

Suppose we have again three random variables X_1, X_2 and X_3, but in this case X_2 and X_3 are continuous variables. To stay in the world of health care we are interested in the influence of the value of Sore Throat {yes,no} (X_1) where we still have $p(X_1 = \text{no}) = 0.9$) and $p(X_1 = \text{yes} = 0.2)$ and Temperature (X_2) on your Heart Rate (X_3). Assume that the marginal distribution of Temperature follows a normal distribution with mean 36 and variance 2 (i.e. $X_2 \sim \mathcal{N}(36, 2)$), see Figure 3.2.

What do we need to fully describe a Gaussian BN? Instead of a conditional probability table we need, for example the following, conditional linear Gaussian distribution function for X_3:

$$P(X_3|no,\ x_2) = \mathcal{N}\Big(70 + (-2 \cdot x_2), 1.1\Big)$$
$$P(X_3|yes,\ x_2) = \mathcal{N}\Big(72 + (2 \cdot x_2), 1.2\Big)^2$$

[2] The numbers in these distributions are fictional and not retrieved from any kind of study

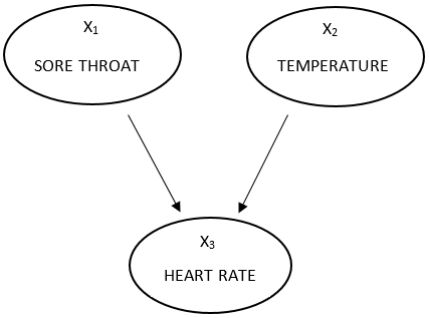


Figure 3.2: A simple Bayesian network consisting of three random variables where we are curious of the influence of a Sore Throat (X_1) and your Temperature X_2 on your Heart Rate (X_3). Notice that X_2 and X_3 are continuous variables.

Definition 3.2.2. A conditional linear Gaussian Bayesian network consists of the following:

- A set of variables and a set of directed edges between variables.
- Each discrete variable has a finite set of mutually exclusive states.
- The variables together with the directed edges form an acyclic directed graph (traditionally abbreviated DAG); a directed graph is acyclic if there is no directed path $A_1 \to A_2 \to \ldots \to A_n$ so that $A_1 = A_n$.
- To each discrete variable A with parents $B_1, \ldots, B_n$, a conditional probability table $P(A|B_1, \ldots, B_n)$ is attached.
- To each continuous variable C with parents $D_1, \ldots, D_n$, a conditional linear Gaussian probability density function.

A BN captures the independencies and conditional independencies in a network. The d-separation concept describes the conditional independence statements in the BN. Now that we have a formal definition of a BN, we want to be able to use the d-separation theorem. This is an extremely helpful theorem that can help us identifying the structure of a BN that we will discuss later. To state this theorem we need more terminology.

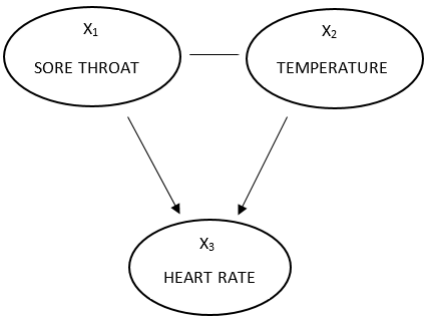
The factorization (3.3) can be expressed as an irrelevance statement about the relationship between the measurement X_i and its parents X_{Q_i} and its remainder X_{R_i} for $i = 2, \ldots, n$ as

$$X_i \amalg X_{R_i} | X_{Q_i}, \quad 2 \leq i \leq n, \tag{3.6}$$

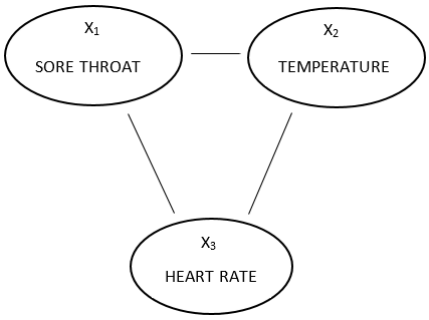
where II represents independence. This relationship reads as follows: X_i is conditional independent from X_{R_i} given X_{Q_i}. This irrelevance statement can also be explained using the Markov property. It essentially states the same; if the conditional probability distribution of future states only depends upon the present state and not on the sequence of events that preceded it, then it has the Markov property. This is what we assumed earlier to come up with the representation in factorization 3.3.

Earlier we introduced the terms child and parent, but we have also ancestor. We call Z an ancestor of Y in a directed G if $Z = Y$ or if there exists a directed path in G from Z to Y. This can also be made to apply to all subsets of $V(G)$. Let X denote a subset of the vertices $V(G)$ in G then the ancestral set of X - denoted by $A(X)$ - is the set of all the vertices in $V(G)$ that are ancestors of a vertex in X. Let's go back to our example (we used the same structure in every example, so it doesn't matter which example we go back to). Here X_1 is an ancestor of X_3. (It it already a parent, so indeed an ancestor).

A graph is said to be mixed if some of its edges are directed and some undirected. The moralised graph G^M of a directed graph G has the same nodes and set of directed edges as G but has an undirected edge between any two vertices $X_i, X_j \in V(G)$ for which there is no directed edges between them in G but are parents of the same child Y in $V(G)$. This is the case for variable X_3: X_1 and X_2 are both parents of X_3, but there is no directed edge between them. So the moralized graph is then given in the next figure. The skeleton $S(H)$ of a mixed graph H is one with the same nodes as H, but all directed edges are replaced by undirected ones. To continue our example, we can take Figure (3.3a) as our H. Then the skeleton of H is given below.



(a) Moralized graph of the network in all our examples



(b) Skeleton graph of the network in all our examples

Figure 3.3: An illustration of the terms 'Moralized graph' and 'Skeleton graph' applied to our examples.

Finally suppose A, B, C are any three disjoint subsets of $\{1, 2, \ldots, n\}$ and X_A, X_B, X_C the corresponding sets of the vertices $V(S)$ of an undirected graph S. Then X_B is said to separate X_C from X_A in S if and only if any path from any vertex $X_a \in X_A$ to any vertex $X_c \in X_C$ passes through a vertex $X_b \in X_B$. In our example it is immediately clear that this is not the case and this means we cannot use the d-separation theorem. But what does it say?

Theorem 1. Let A, B, C be any three disjoint subsets of $\{1, 2, \ldots, n\}$ and G be a valid DAG whose vertices $V(G) = \{X_1, X_2, \ldots, X_n\}$. Then if X_B separates X_C from X_A in the skeleton of the moralised graph $G^M(A(X_{A\cup B\cup C}))$ of the ancestral graph $G(A(X_{A\cup B\cup C}))$ then

$$X_C \amalg X_A | X_B$$

Now that we have seen the basics about Bayesian networks we would like to build a BN based on our data! There are many ways of constructing a BN. In the examples above we worked with 3 variables. But what is the number of possible BN's? There are already 25 possibilities for a network with 3 variables and as can seen in Table 3.2 the number of possible graphs grows very quickly as the number of nodes increases, so it can be very hard to find a model that fits the data.

Number of nodes	Number of possible graphs
1	1
2	3
3	25
4	543
5	29,281
...	...
10	4,175,098,976,430,598,143

Table 3.2: Copied from Jensen and Nielsen [2, p.240]. Shows the number of possible graphs for the number of nodes and the corresponding number of possible graphs.

To do this we need to know how we can find the best BN based on the data. To learn the structure of the data we have the disposal of two groups of algorithms: constraint-based and score-based. In the Journal of Statistical Software [4] Marco Scutari provides a clear description of constraint-based and score-based algorithms:

- Constraint-based structure learning: they learn the network structures by analyzing the probabilistic relations entailed by the Markov property of Bayesian networks with conditional independence tests and then constructing a graph which satisfies the corresponding d-separation statements.
- Score-based structure learning: these algorithms assign a score to each candidate Bayesian network and try to maximize it with some heuristic search algorithm. Greedy search algorithms (such as Hill-Climbing (HC) or tabu search) are a common choice, but almost any kind of search procedure can be used.

We will have a closer look at both types, but first have a look how we can mitigate the structure learning by these algorithms by the influence of an expert.

3.3. Learning the structure from experts

In section 1.4 we shortly mentioned the advantage of the data-driven approach of Bayesian networks. It can be helpful to use an expert approach to provide some rough guidelines on how to model a complex world. By cooperation with Sunweb we have as our expert the scientific expert of team Sunweb Teun van Erp. He would expect to see a strong relationship between power, cadence, heart rate and speed. Furthermore he expects speed, distance and altitude to have some sort of relationship and a strong influence of the temperature on power. This information can be used to build a network more easily, however our first try is

going to be based only on constraint-based or score-based structure learning. Let's see what this entails.

3.4. Learning the structure from data

3.4.1. Constraint-based structure learning

These algorithms try to build a network that best captures the conditional independence statements of the network. By these algorithms we mean popular choices as Grow-Shrink (gs), Incremental Association (iamb) and two variants of iamb named Fast Incremental Association (fast.iamb) and Interleaved Incremental Association (inter.iamb). The algorithms for constraint-based learning are looking for a graph with a desired set of conditional independencies and especially to a set with a minimal amount of edges. We are not going to discuss these algorithms in details. For this we refer you to chapter 3 of Koller and Friedman (2009)[7], however we will give an overview on how the algorithms work in general.

If we think about building such a network it is really time consuming if we would have to check all possible networks. In Table 3.2 we saw that there were already $4 \cdot 10^{18}$ possible graphs for ten nodes, so we need equivalence classes.

Equivalence class

If two graphs have the same skeleton (an explanation can be found in Figure 3.3b) and the same set of v-structures, i.e. they have the same d-separation [2, p.248], then they belong to the same equivalence class [7, p.77].

The term v-structure needs some explanation. Suppose we have three random variables X, Y and Z. If X and Y have a common effect on Z we call the structure $X \rightarrow Z \leftarrow Y$ a v-structure. This is again displayed in Figure 3.4.

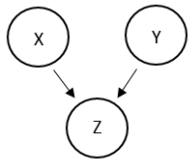


Figure 3.4: v-structure (also called common effect) of three random variables X, Y and Z. Two arcs point, so in the same direction, to the same node.

Now have a look at three other graphs.
If we compare the graphs in Figure 3.5 with the one in Figure 3.4 we notice that these four graphs have the same skeleton, but the difference in the set of v-

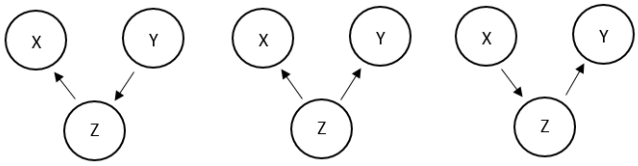


Figure 3.5: Three graphs that belong to an equivalence class.

structures. In Figure 3.4 we have one v-structure and in Figure 3.5 none. This means that three graphs in Figure 3.5 belong to the same equivalence class and the graph in Figure 3.4 not. To learn the structure faster, we use a partially directed graph (PDAG), that is an acyclic graph with some edges undirected as an equivalence class of a set of graph structures [3]. So if we have a way to find the independencies we can find the structure of the Bayesian network using methods described in [7] and faster with the help of the equivalence classes.

INDEPENDENCE TESTS

Suppose we are working with a lot of random variables, then we would like to answer questions like "Is X_1 independent of X_2 and X_3 given X_4?" or more simple "Is X_1 independent from X_2?". This comes down to hypothesis testing. In the case of marginal independence testing between X_1 and X_2, the null hypothesis

$$H_0 : P(X_1, X_2) = P(X_1)P(X_2)$$

against the alternative hypothesis

$$H_1 : P(X_1, X_2) \neq P(X_1)P(X_2)$$

Now the only thing we need is some measure of deviance from the null hypothesis. Common measures are the χ^2 statistic and the mutual information measure. For details we refer you to Koller and Friedmann (2009) in *Probabilistic Graphical Models: principles and techniques*[7].

3.4.2. SCORE-BASED STRUCTURE LEARNING

To learn the structure in the score-based approach the Hill Climbing and Tabu search algorithms are common choices. Such algorithms works as follows: we start with an initial structure and assign a score to it. Then generate a new set of possible networks after performing a simple operation: add a directed edge, reverse a directed edge or delete an edge. Compute the score of each structure in the new set and select the one with the highest score. As long as this score cannot be improved, we do these steps all over again.

So far we mentioned the word 'score' a few times, but what do we mean by that? How do you assign a score to a Bayesian network? The first thing you need is a measure of the quality or goodness of fit of your network as a representation of the data. This measure should also take into account the complexity of the structure, i.e. you don't want a score function that just gives a higher score when there is an extra node or arc. So you need a good mix between quality and complexity.

As in Kjærulff & Madsen [3] we are going to use the Bayesian Information Criterion (BIC) and we compute it as $L - \frac{K}{2} \, log N$ where L is the log-likelihood function of the data given a structure, K is the number of free parameters in the network and N is the number of cases in the database, also known as the sample size. In this criterion we recognize the first term as a measure of goodness of fit, since it maximizes the probability (likelihood) that the network represents the data. The second term in this function penalizes the complexity of the structure. The log-likelihood term will dominate the penalty score when the size of the database grows. This is because the first term grows linearly with N and the second logarithmically.

To see how the Hill Climbing algorithm and the BIC score function work we will discuss the following example including three random variables X, Y and Z. At this point it doesn't matter what the nodes represent, the purpose of the example is explaining the algorithm and the score function.

Assume all variables can take two different values, so X can turn in x_0 and x_1, Y in y_0 and y_1 and Z in z_0 and z_1. Let's say we are given the joint probability function and a sample of six different cases over X, Y and Z. First we will put the joint probability in Table 3.3 below.
Now we assume we have seen a sample of different cases over X, Y and Z - named c_i, where $i = 1 \ldots 6$ - of the experiment over X, Y and Z. We put them in a Table 3.4.
What is the best possible network that fits the data given this joint probability function? We will examine one iteration step of the HC algorithm. We start with an initial guess for the network, see Figure 3.6.
To compare this structure with other possible structures that we are going to examine in a bit, we need a score for the graph in Figure 3.6. Then we have to compute the BIC-score for this current graph. Due to the fact that this is a BN we can compute the probability of an event in the sample data quite easily. Name the current graph G_1. Then

$$P_{G_1}(c_i) = P(X) \cdot P(Z) \cdot P(Y|X)$$

X	Y	Z	P(x,y,z)
x_0	y_0	z_0	0.02
x_0	y_0	z_1	0.005
x_0	y_1	z_0	0.0075
x_0	y_1	z_1	0.0675
x_1	y_0	z_0	0.576
x_1	y_0	z_1	0.144
x_1	y_1	z_0	0.018
x_1	y_1	z_1	0.162

Table 3.3: Joint probability table for three random variables X, Y and Z. It gives for each combination of X, Y and Z the probability that this event will happen. Notice that the sum of all these probabilities sums to one.

c_i	X	Y	Z
c_1	x_0	y_0	z_0
c_2	x_0	y_1	z_0
c_3	x_1	y_0	z_1
c_4	x_1	y_1	z_1
c_5	x_1	z_0	z_0
c_6	x_0	y_1	z_1

Table 3.4: Sample data over X, Y and Z. In this example we have six recordings, $c_1, \ldots, c_6$, of the experiment, i.e. in each row we find the outcomes of a single recording.

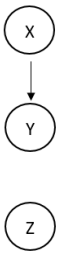


Figure 3.6: Initial guess for a network that we need to assign a BIC-score to if we want to compare this structure with other networks. These other networks appear in the next iteration step in the Hill Climbing algorithm.

To compute this probability - and the rest - we need $P(X)$, $P(Z)$, $P(Y|X)$, $P(Y|Z)$, $P(X|Z)$, $P(Z|X)$ and $P(Z|Y)$. These probabilities can be found by the use of Ta-

ble 3.3. The probability of observing the value x_0, unconditional on the other variables, is the sum of the values that corresponds to a row where x_0 pops up, so $P(x_0) = 0.02 + 0.005 + 0.0075 + 0.0675 = 0.1$. In general we will use the notation $P(X) = (0.1, 0.9)$, i.e. the probability of observing x_0 is 0.1 and x_2 is 0.9. For $P(Z)$ we find $P(Z) = (0.6215, 0.3785)$. A conditional probability like $P(Y|X)$ can be computed as follows. Suppose we observe x_0, what is the probability of observing y_0? Have a look at Table 3.3 again and find the rows where x_0 pop up. Search in these rows for the value y_0 and sum these probabilities. Now we are almost there. We only need to divide this sum by the sum of all probabilities in the first case, so the sum of probabilities of the rows where we found a x_0. In our example

$$P(y_0 \mid x_0) = \frac{0.02 + 0.005}{0.02 + 0.005 + 0.0075 + 0.0675} = 0.25$$

To calculate the BIC we also need N, the sample size, and we assumed we had six of them. So $N = 6$. The last thing we need is K, the number of parameters. We need one for every node in the graph and for each arc an extra parameter. Thus in this case $K = 4$.

$$BIC_{G_1} = \underbrace{\sum_{i=1}^{6} \log P_{G_1}(c_i)}_{\text{-15.5847}} - \underbrace{\frac{4}{2} \log 6}_{3.5835} = -19.17$$

Now continue with the HC algorithm. The next step is to generate a new set of possible networks. If we add, remove or reverse a single arc we find the following six possibilities.

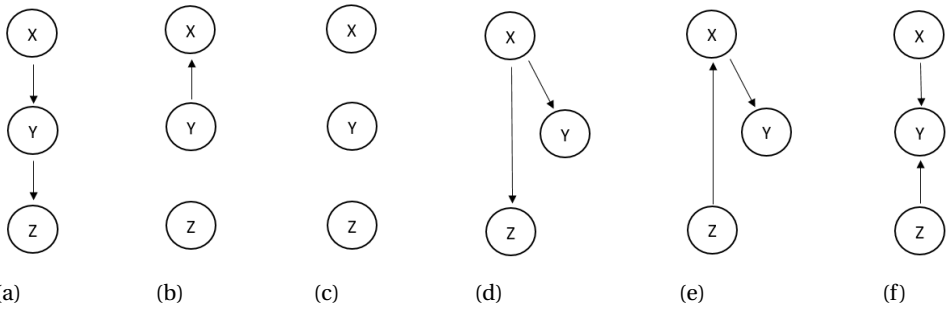


Figure 3.7: A new generation of possible networks in the Hill Climbing Algorithm. We created them by adding, removing or reversing a single arc in the original network (see Figure 3.6).

What are the BIC-scores for networks a to f? To compare all the scores give these structures the names G_2 to G_7.

G_i	Figure	BIC_{G_i}
G_1	3.6	-19.17
G_2	3.7a	-17.99
G_3	3.7b	-19.17
G_4	3.7c	-19.23
G_5	3.7d	-21.19
G_6	3.7e	-21.19
G_7	3.7f	-20.05

Table 3.5: All BIC-scores, i.e. the BIC-score of the original graph (G_1) and the BIC-scores of the graphs in the next generation ($G_2, \ldots, G_7$), compared to each other.

The graph with the highest score is G_2. Now we would have to run the same procedure all over again and see if we can improve this score. In this way we will find a network that fits the data best.

4

CYCLING NETWORK

If we would like to learn the structure of the BN that describes the performance of a cyclist we need a set of random variables that describe or influence this performance. The constraint- and score based structure learning algorithms that we are going to use are implemented in R. We will use one score based algorithm: Hill Climbing (hc) and four constraint based algorithms: Grow-Shrink (gs), Incremental Association (iamb) and two variants of iamb named Fast Incremental Association (fast.iamb) and Interleaved Incremental Association (inter.iamb).

The initial random variables that we are going to consider are the variables that were stored in the data of team Sunweb, i.e. riding time [s], pedal power [W], cadence [rpm], heart rate [bpm], altitude [m], speed [km/h], distance [m] and temperature [°C]. The riding time is only used for finding the starting point of the race.

In the first stage of building this network we will only have a look at one cyclist and his pedal power, cadence, heart rate, altitude, speed, distance and temperature. In the second part we will incorporate the influences of the performances of the other riders.

Thinking about the performance of the leader in the final sprint or on the last climb of the day we could be wondering if it is beneficial to include the following variables:

- Grand Tour, i.e. Giro, Tour and Vuelta.
- Type of race, i.e. flat, hills, mountain and arrival uphill.

- Number of days (in succession) in the race

Thus we are going to model the performance of these cyclists by a Bayesian network and investigate if and how these new variables (Grand Tour, type or race and number of days (in succession) in the race) influence the structure. Since all our data-driven variables are continuous, let's see first if we violate our assumption of normality.

4.1. NORMALITY ASSUMPTION

Like we said before we have to work with continuous variables that we assume to be normally distributed, but is this assumption reasonable? To test this we can use multivariate normal tests like Mardia's, Royston's and Henze-Zirkler's tests. A multivariate normal distribution is just a generalization of the one-dimensional normal distribution to higher dimensions. According to these test the data is not normal distributed. This can also be seen if we plot the histograms with a fitted normal curve for all the variables, see Figure 4.1.

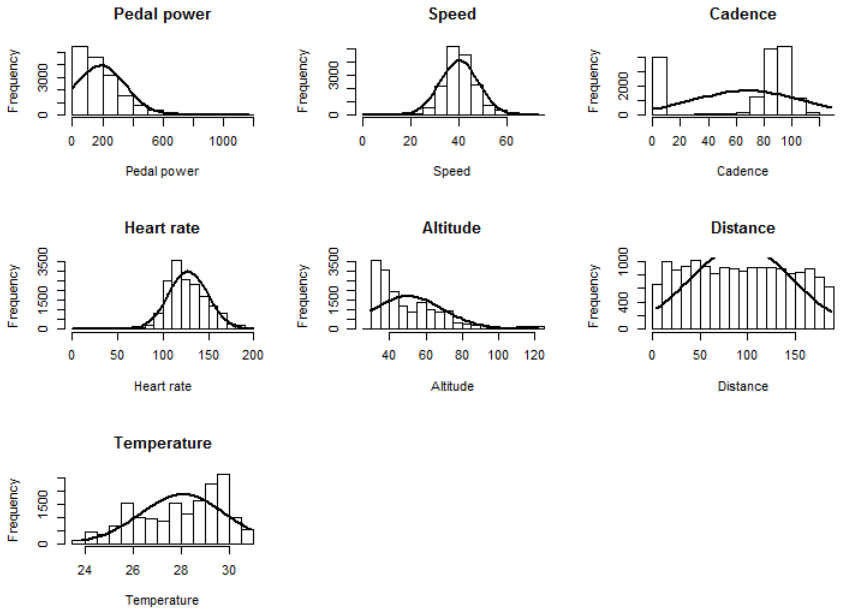


Figure 4.1: Histogram of all 7 variables where each figure has an fitted normal curve over the histogram to check whether the normality assumption is reasonable or not.

One could argue that the variables speed and heart rate are normal distributed,

but the rest of the variables definitely not. So this is a strong violation of the assumption for building a BN.

Despite the violation, we still want to model the performance by a Bayesian network. At first stage we are trying to find some general model for a single rider on a single day on a very local scale. If we could find such a model, we can easily build a model for the whole team, by adding all these small BN's together and find the correlations between the variables of different cyclists to include arcs between the small BN's. If this is not the case, we will have to study the structure on a larger scale, so for example the data of all cyclist during the Tour de France. In this approach we could lose the information that is specific to each rider, but a more general model for the performance of a random cyclist could be an usefull result.

4.2. Structures on a local scale

General remark

In Chapter 2 we made the distinction between variables that give details for the race (altitude, distance and temperature) and cyclist race activity variables (pedal power, cadence, heart rate, speed). It doesn't make sense that any of the personal race activity variables could influence the variables that say something about the race. Thus we don't take these possible arcs into account during the analysis.

One day one rider

If we examine the structure for one rider in a single day, we come to the conclusion that there is no general network for the data of one cyclists on a single day. What could be the reason for this? The first thing we thought about is the fact that the data is too sensitive on noise; if there happened something weird during a race it is immediately captured in this network. The second thing concerns the algorithms we are working with. These algorithms are very sensitive, if one test indicate independence where the the same test in the last structure indicated dependence it changes the whole structure of the network. To avoid the first problem we could make use of more data, so that extraordinary moments in the race disappear by the influence of a huge amount of natural data.

One day all riders

Is there a common structure for all riders for one specific day? Hopefully this is the case, since all cyclists from team Sunweb took part in the same race and they have followed the same tactics - except the leader - during the race, so we would

expect during the same the day the same structure in the BN. And if not, is this caused by the structure of the data of the leader?

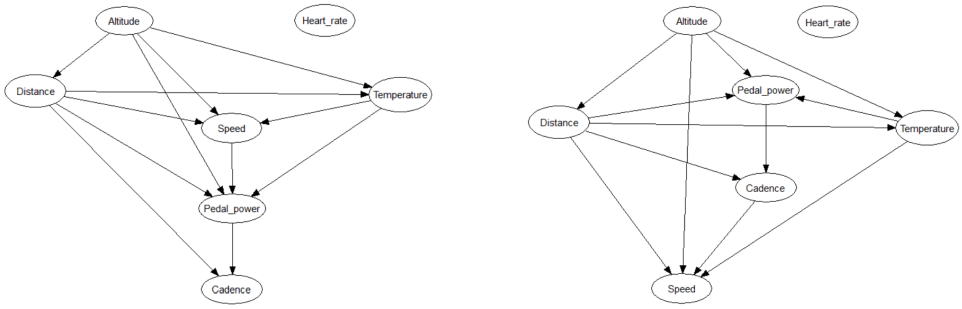
For example; for day 1655 we built for every cyclist a BN and checked which arcs are present in every BN for that specific day. The same procedure is applied for the other days and the results can be seen in Table 4.1.

1652		1655	
From	To	From	To
Distance	Cadence	Altitude	Cadence
Distance	Pedal power	Altitude	Distance
Distance	Speed	Altitude	Temperature
Distance	Temperature	Distance	Temperature
Pedal power	Cadence	Temperature	Speed
Temperature	Speed		
1657		**1663**	
From	To	From	To
Altitude	Distance	Altitude	Pedal power
Altitude	Pedal power	Altitude	Temperature
Altitude	Speed	Temperature	Speed
Altitude	Temperature		
Distance	Pedal power		
Pedal power	Cadence		
Temperature	Speed		

Table 4.1: Comparison of common arcs for all riders during four specific days.

On day 1652,1655,1657 and 1663 rider 4 was the leader of the team and as you can see the only arc that is present every day (Table 4.1) is the arc Temperature - Speed. The rest of the structure is showing a great deal of variety. Below (Figure 4.2) we put two BN's for a given day. For this example we choose day 1652.
It is quite surprising that the node Heart Rate isn't included in the networks at first sight, but when we have a look at the data we see that there is no data available for the heart rates of rider 4 or rider 5. That means we have a column of zeros in the data, so - of course - no correlation with any other variable.

Thus remains the question if there are more common arcs when we leave the data of the leader, in this case rider 4, out. The difference is negligible, only the arc Altitude - Pedal power on day 1657 should be added to the common arcs. This is also the case for other days. From this we might think the structures of



(a) Rider 4 (b) Rider 5

Figure 4.2: Bayesian networks - on day 1652 - that we created using the Hill Climbing algorithm. These figures are included to give an idea how the structures look like.

networks for leaders and helpers are the same, but we will have a look at this in section 4.3 when we are dealing with the rider type. So multiple cyclists on one day didn't result in a clear model for the performance, but maybe are the relationships between all the variables more personal? Does the structure depend on the riding style and performance?

One rider multiple days

Let's have a look at the structure for a specific cyclist at different days. Is there a pattern? As said before we expect a pattern, but unfortunately we can't find one in this way.

For the comparison of networks for cyclists on different days we used five days in the Giro, namely day 1652, 1654, 1655, 1657 and 1663. The BN's for these riders consists of about 12 arcs and the arcs they have in common are in most cases, as can be seen in Table 4.2, arcs that point from race variables to specific rider variables. So for example; for rider 4 we built for every day a BN and checked which arcs are present in every BN for that specific cyclist. The same procedure is applied for the other riders. As it turns out this is on a too local scale, we would like to have a more general model.

Rider 4		Rider 5	
From	To	From	To
Altitude	Distance	Altitude	Distance
Altitude	Speed	Altitude	Speed
Distance	Speed	Altitude	Temperature
Distance	Temperature	Distance	Pedal power
Pedal power	Cadence	Pedal power	Cadence
Temperature	Pedal power	Temperature	Speed
Temperature	Speed		
Rider 6		Rider 3	
From	To	From	To
Altitude	Distance	Altitude	Distance
Altitude	Pedal power	Altitude	Pedal power
Distance	Cadence	Altitude	Speed
Distance	Temperature	Altitude	Temperature
Pedal power	Cadence	Distance	Pedal power
Pedal power	Speed	Temperature	Speed
Temperature	Speed		

Table 4.2: Comparison of common arcs for one rider during five specific days in the Giro.

4.3. STRUCTURES ON A LARGER SCALE

To start rigorous let's combine all possible data, i.e. all data from all riders of all days.

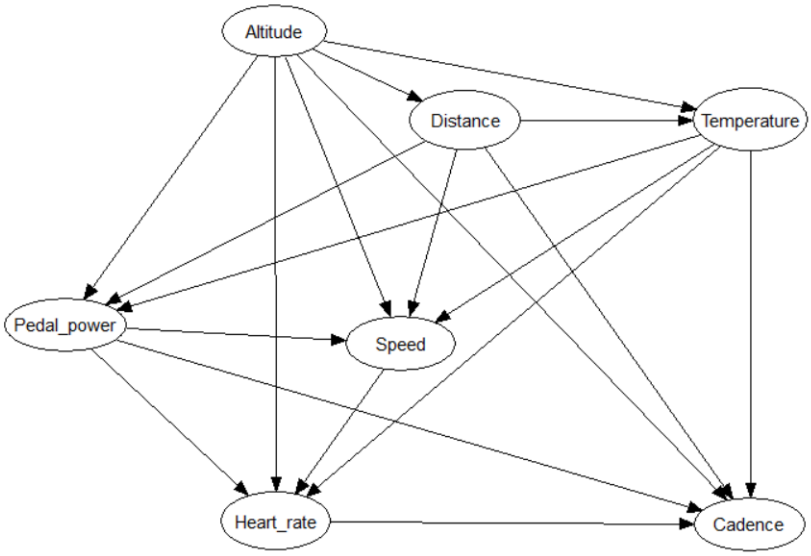


Figure 4.3: Bayesian network based on all possible information via the Hill Climbing algorithm.

The structure doesn't change when we use constraint-based algorithms, so this network is quite consistent. Even when we don't take all measurements into account, so instead of every second a evaluation, every 5, 10 or 20 seconds a measurement the structure doesn't change. In this way we could get rid of 'noise', but since the network doesn't change, we are confident that it doesn't matter if we use all measurements or for example the measurements every 5 seconds. Now it is time to return to our sub questions from section 1.2. All these networks are still be learned from data. There is no expert knowledge used so far.

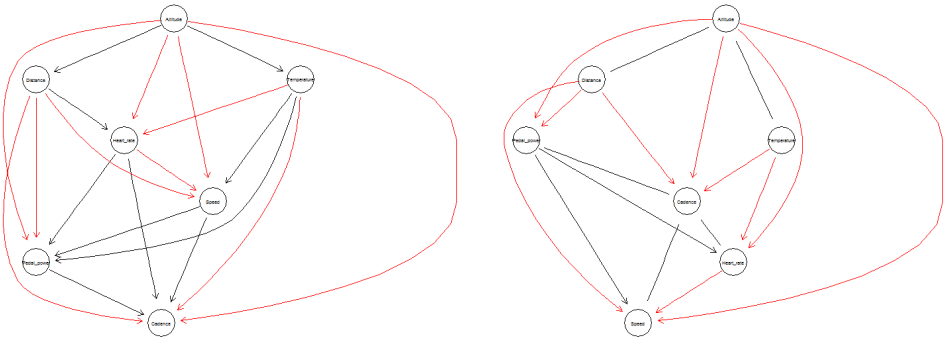
(1) *GRAND TOURS*

What is the influence on the structure of the data caused by the three Grand Tours? (4.1)

Despite the fact that the three Grand Tours are different races in different countries, we still expect to see major similarities between the structures. All cyclists need to divide their energy over three weeks, but they need to perform each day and the tactics over the Grand Tours are more or less the same if there are no dropouts in team Sunweb.

Giro d'Italia

In the following figures we find the structure of the network based on all the available data in the Giro.



(a) Hill Climbing (b) Grow Shrink

Figure 4.4: Side by side comparison of the netwerk structures learned from all the available data in the Giro by the Hill Climbing algorithm (on the left) and by the Grow Shrink algorith (on the right). The arcs present in both networks are highlighted in red.

In Figure 4.4 we can compare the networks generated by the Hill Climbing algorithm and by the Grow Shrink algorithm. All the IAMB algorithms return the same network as Grow Shrink, which is given in Figure 4.4b.

Tour de France

The different networks we get from the constraint-based and score-based algorithms don't differ much from each other, actually the structures for the constraint-

based algorithms are exactly the same. In the following table (Table 4.3) we summarize the performance of the algorithms on the data from the Tour.

	hc	gs	iamb	inter.iamb	fast.iamb
independence tests / network comparisons	151	109	102	102	68
learned arcs (directed/undirected)	21 (21/0)	19 (12/7)	19 (12/7)	19 (12/7)	19 (12/7)
execution time (sec)	45.81669	52.43766	52.05447	57.20332	54.32263

Table 4.3: Performance of learning algorithms on all the available data on the Tour de France, measured in the number of conditional independence tests (for constraint-based algorithms) or network score comparisons (for score-based algorithms), the number of arcs and the execution time [4].

In a score-based approach we used network comparisons to find the network that fits the data best, see section 3.4.2. In this particular example we needed 151 comparisons to get the optimal structure. Instead of network comparisons we need conditional independence tests in the constraint-based algorithms and the number of tests is approximately the same for Grow Shrink, iamb and inter.iamb, but fast.iamb needs significantly less tests to find the same network. As regards execution time; Hill Climbing is the fastest, but there are not very big differences in time.

Vuelta a España

Also in the data of the Vuelta we experience the same behaviour: there are some small differences between the networks created by the score-based approach and the constraint-based algorithms and the algorithms Grow Shrink, IAMB, Inter-IAMB and Fast-IAMB return the same network.

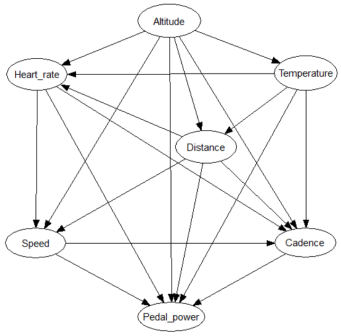
Influence of a Grand Tour

Are there huge differences between the structures in the Grand Tours? The answer to this question is no! If we compare the structures for each Grand Tour generated by the Hill Climbing algorithm we find 15 common arcs and the arcs that are different are the relationships between the variables that are specific to each cyclist, so this could be a possible framework to use in prediction.

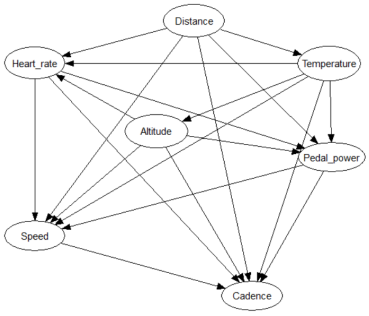
(2) *RACE TYPES*

How does the structure of the data change when there is a difference in race type? (4.2)

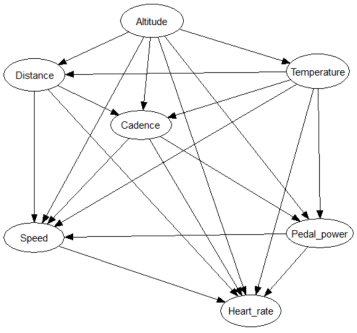
All data



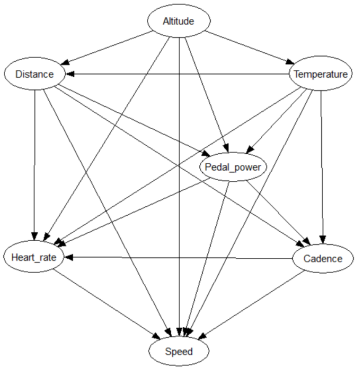
(a) Flat



(b) Hills



(a) Mountain



(b) Arrival uphill

Figure 4.5: Side by side comparison of the network structures learned from all the available data for different race types: flat (a), hills (b), mountain (c) and arrival uphill (d).

To come straight to the point; in Figure 4.5 we put the BN's built for each race type. By race type we mean four different types: flat, hills, mountains and arrival uphill. Each race in the Grand Tours is subdivided into one of these categories.

Normally there are also time trials, but for the purpose of this report it is not very help full to have a look at races where you don't have any help from your teammates.

Per Grand Tour

If we have a look at the race types per grand tour, we find a lot of similarities between these networks. The Hamming distance measures the minimum number of substitutions required to change one skeleton in the other and as can be seen the maximum number of turns is 5. Compared to the 20 arcs in total this is not much. In fact there are a lot of common arcs. To see which arcs are in common we refer you to Table B.1 in the appendix.

	Giro	Tour	Vuelta
Giro	(0,0,0,0)	(2,3,3,x)	(2,1,5,x)
Tour	(2,3,3,x)	(0,0,0,0)	(4,2,4,1)
Vuelta	(2,1,5,x)	(4,2,4,1)	(0,0,0,0)

Table 4.4: Comparison of the four BN's - based on race type (flat, hills, mountain, arrival uphill) - created for each Grand Tour measured by the Hamming Distance.

In the Giro there is no stage classified as arrival uphill. This means that we can't compute the hamming distance between this race type in the Giro. That is the reason there are x's in Table 4.4.

Influence of the race type

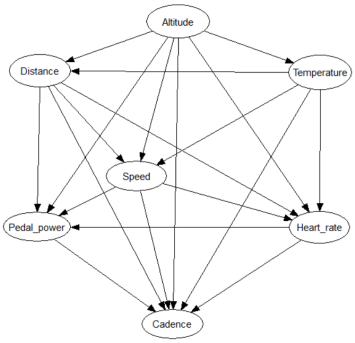
Does the structures above surprise us? No, all arcs that these networks have in common are also present in the global structure we began with in section 4.3. The number of common arcs between the different types is quite low, only 9 of the 20 arcs, but as expected. For example, the performance of a sprinter in a race with a lot of climbs is certainly different from a flat race. This is mainly caused by the fact that the sink nodes, nodes with no outgoing arcs, are different. This means that the direction of the arcs is not the same. However, the hamming distance showed that the difference in structure is not so big if we neglect the direction of the arcs. So it seems that race type definitely influences the direction of the arcs and thus the structure of the BN's.

③ *RIDER TYPE*

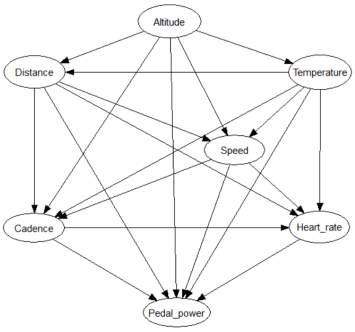
Does the structure of the data change (4.3)
when we encounter a different rider type? If so, how?

Leader and helpers

In section 1.2 we discussed the difference between the role of the leader and helpers. The helpers bring water and food from team cars and shield teammates from opponents, but the main job contains protection from the wind. So we definitely expect a different structure in the BN's for the leader and the helper; if this remains invisible from the structure we would expect to see a different behaviour when predicting their speeds or pedal powers.



(a) Leader



(b) Helper

These BN's are based on the days where team Sunweb assigned a leader to a race day. It is interesting to see whether the structure changes if there is no leader assigned. There are 14 days without a leader and all of these days are in the Vuelta. Rider 7, the potential leader during hills- and climbing days in this race stopped after the first three days. So does the network change for the leader in the sprint stages? And for the helpers? Unfortunately we can't study this problem thoroughly, because for the leader the amount of available data is going to be a problem again. Is the difference in structure caused by the lack of data or by a real difference in performance on these days? If we compare the data of rider 4 with the data of days where there is no leader assigned we lose all information about the heart rates, because there is almost no heart rate for rider 4 and we find only 8 out of 20 common arcs. For the rest of the cyclists, i.e. the helpers, the structure doesn't change.

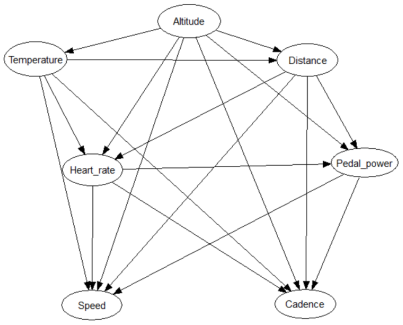
Influence of rider type

We expected a difference between the networks of the leader and the helpers, but this did not appear in the BN's. Furthermore the absence of a leader didn't seem to influence the structure for the helpers and for the leader we can't say anything about it at this moment.

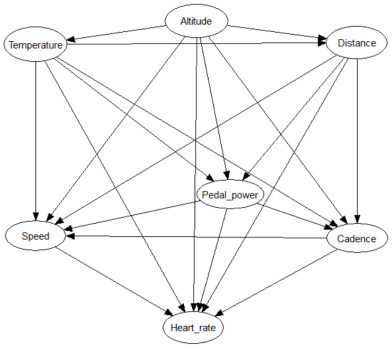
④ *DAYS OF ACTIVITY*

What is the influence on the structure of the number of days riders are in succession in race? (4.4)

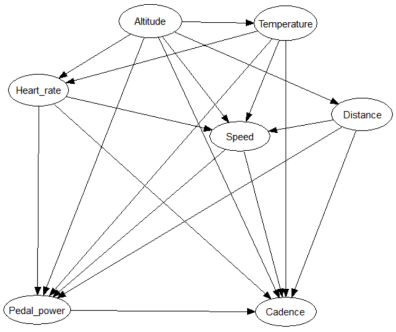
The number of days in succession in the race should definitely have an influence on the performance. Towards the end of a Grand Tour the performances of cyclists decline, but is this reflected in the Bayesian networks or is this only visible in predictions?



(a) Week 1



(b) Week 2



(a) Week 3

Figure 4.6: Bayesian network based on all possible information for each week in the Grand Tours based on the Hill Climbing algorithm.

We divided each Grand Tour into three weeks, because there is too little data to divide the races in separate days. When we combine all information for these Grand Tours in three sets - one for each week - we end up with the BN's above.

The similarities between the three weeks are big; 14 out of 20 arcs are present in every week.

INFLUENCE OF NUMBER OF DAYS IN SUCCESSION

Although this influence doesn't stand out in the Bayesian networks, from a practical kind of view this in one of the most important variables in the model. In this way the term 'fatigue' is somewhat incorporated in the model; of course this is not a perfect measure for fatigue, but it is a start.

4.4. VIEW OF AN EXPERT

One of the advantages of Bayesian networks we mentioned in 1.4 is the fact that the framework makes it possible to construct a model by a human expert or automatically. In chapter 3.3 we summarized the relationships our expert suspected and the network that follows from these comments is given in the figure below.

EXPERT ONLY

Figure 4.7: Bayesian network purely based on an expert view.

However, we will use the human knowledge by specifying the attributes that the model should contain and the details are filled in automatically, by fitting the data to the model. The attributes that should be in the model can be found in chapter 3.3 and are given in Table 4.5. It is hard to say something about the connections between the rider variables, so these will represent our details and based on structures created from all data.

EXPERT AND DATA

The framework where the network is build on is given in Table 4.5. We included both the arc Pedal power - Cadence and Cadence - Pedal power, since the direction of this arc was not clear from the info of the expert or from previous networks.

From	To
Temperature	Speed
Cadence	Speed
Pedal power	Cadence
Cadence	Pedal power
Altitude	Distance
Altitude	Speed
Distance	Speed

Table 4.5: The whitelist (list of arcs that are definitely included) of the Bayesian network based on expert view and data.

The created BN has the following structure:

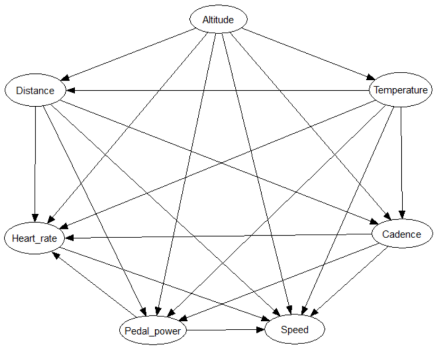


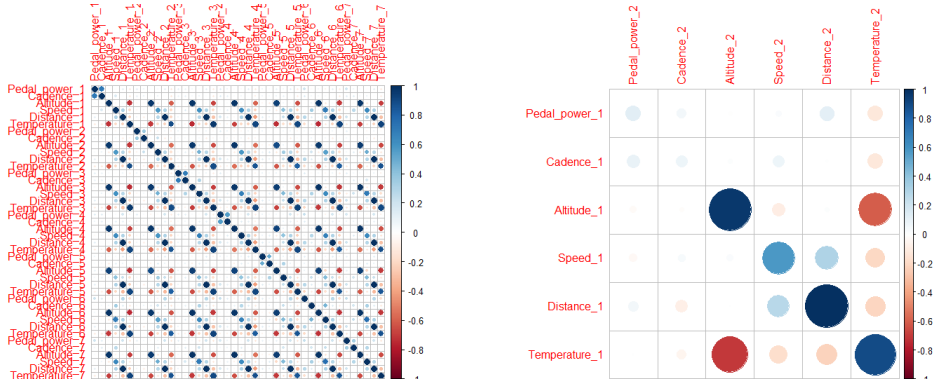
Figure 4.8: Bayesian network based on an expert view and data.

The structure between the race variables is the same as in most BN's we have seen so far, but the structure between the rider variables is again different. In Figure 4.5(d) we find almost the same structure, since the sink node is speed too.

4.5. MULTIPLE RIDERS

To get a feeling of the interaction between different cyclists we are going to have a look at the correlation plot where we compare all (data driven) variables with

each other.



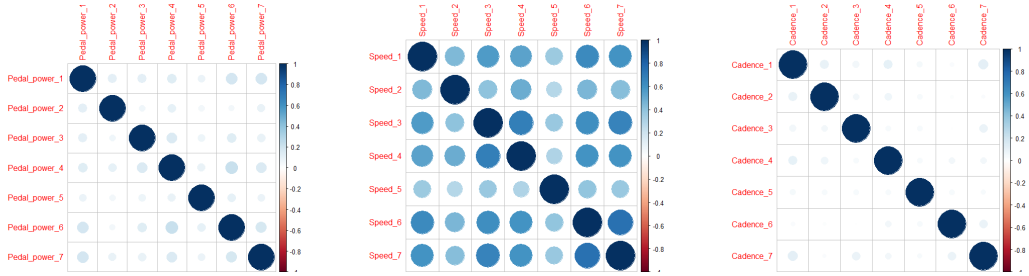
(a) All riders

(b) Two riders compared

Figure 4.9: Correlation plot to find interaction between riders

Although this is the correlation plot for one day, it gives us a reliable image of the structure of all days, since it is for each day more or less the same, see Figure 4.9a. This gives us a lot of information, because the pattern between two different cyclists is always the same! For example, have a look at the correlation plot between cyclists 1 and 2 in Figure 4.9b.

This figure makes us suspect that there is some strong monotonic relationship between the speeds of different cyclists and the altitude and temperature. This last arc between altitude and temperature consists of our external factors which should be the same for all riders, so also no extra information. The correlation between the speed of different riders is clearly visible, but we would suspect the same behaviour between the power and the cadence. Is this the case?



(a) Pedal power (b) Speed (c) Cadence

Figure 4.10: Correlation plot for power, speed and cadence

It is remarkable that the correlation between the power and cadence of different riders is this low. This means that we definitely have to include arcs from the node 'Speed' from one to another in our BN for multiple riders and for 'Pedal power' it could be a good addition to the network if this is learned by the data.

After answering the sub questions, we believe it is beneficial to include four extra discrete variables:

- Grand Tour: Giro, Tour and Vuelta.
- Leader: rider 4, rider 7, rider 8, rider 5 and no leader.
- Week: week 1, week 2 and week 3.
- Race type: flat, hills, mountain and arrival uphill.

The Bayesian network based on the data with the extra variables get really complicated and to emphasize this we included the BN in Figure 4.11. The model we are going to use for predicting in R is based on all measurements except for the ones without any information about heart rate.

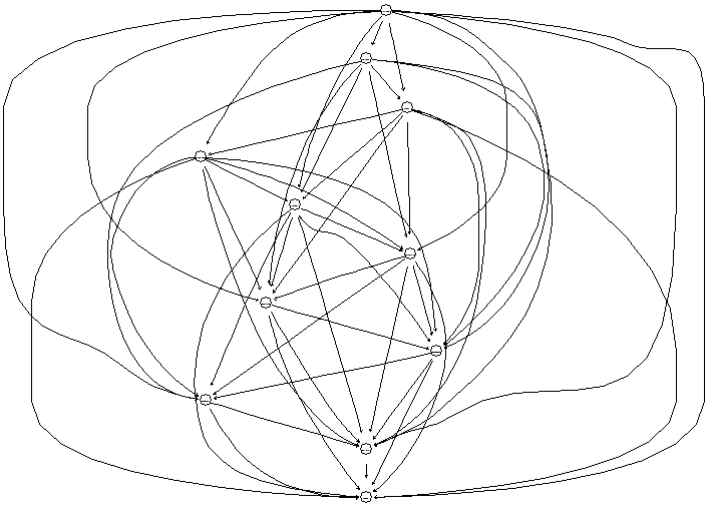
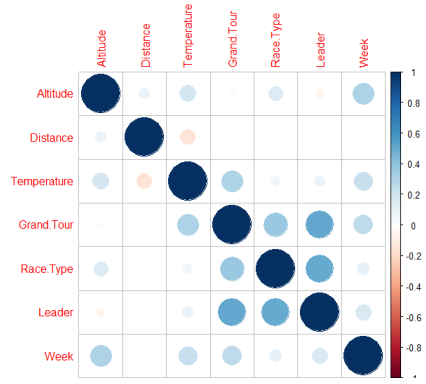
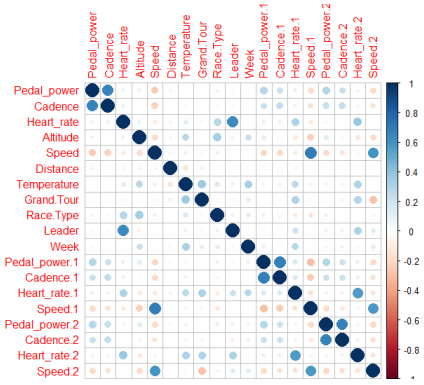


Figure 4.11: Bayesian network for a single cyclist during a specific day in one of the Grand Tours based on the Hill Climbing algorithm.

Luckily in Uninet[1] we are able to create our own BN. We are going to use the network based on the expert and the data found in section 4.4, supplemented by the extra variables for a BN that contains only the variables for the leader and for a BN that takes the influence of two helpers into account.



(a) Race variables



(b) All variables

Figure 4.12: Correlation plot to find interaction between race variables and three cyclists consisting of one leader and two helpers.

[1] Uninet is a statistical software package from the TU Delft[8]

The structure between the four extra variables is based on Figure 4.12a and our view. The structure presented by R was slightly different from this one, but because of the simplicity we preferred our structure, as in Figure 4.13.

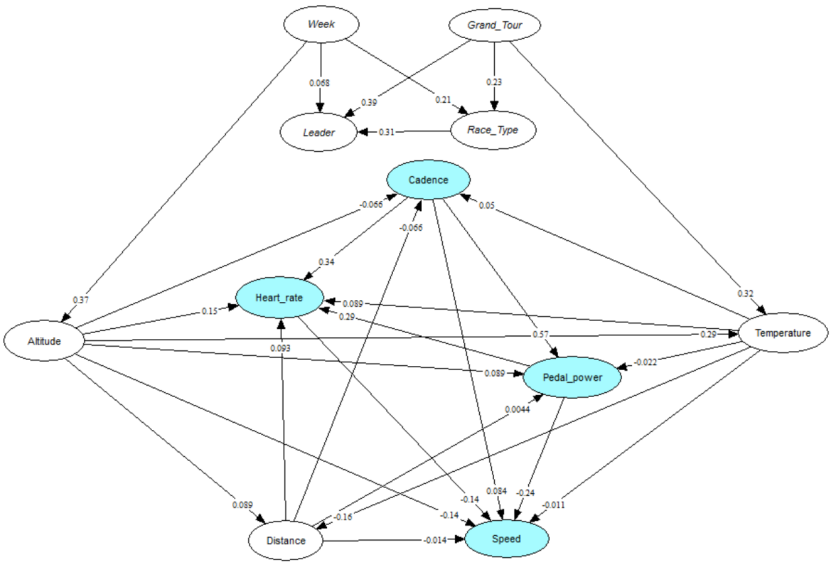


Figure 4.13: Bayesian network for a single cyclist during a specific day in one of the Grand Tours.

The variables that are specific to a cyclist are highlighted in blue. Furthermore the arcs are quantified by rank correlations. For example, ρ(Race_type,Leader) = 0.31 means that the rank correlation between Race_type and leader is 0.31. The correlations are overall quite low and even negative. For example, the rank correlation between Pedal power and Speed is -0.24. This means that if the Pedal power goes up, Speed goes down. We think this is caused by mountain and hills stages, where the cyclists probably need more power to climb the mountain with a lower speed.

The major part of the BN is copied from the Bayesian network for the leader in Figure 4.13. We decided to use Figure 4.10 and Figure 4.12b to complete this BN. When the correlation between two nodes is high, there is an arc added to the existing network. This has led to the following: the influence of the help of two riders in the performance of the leader is modeled by the following Bayesian network:

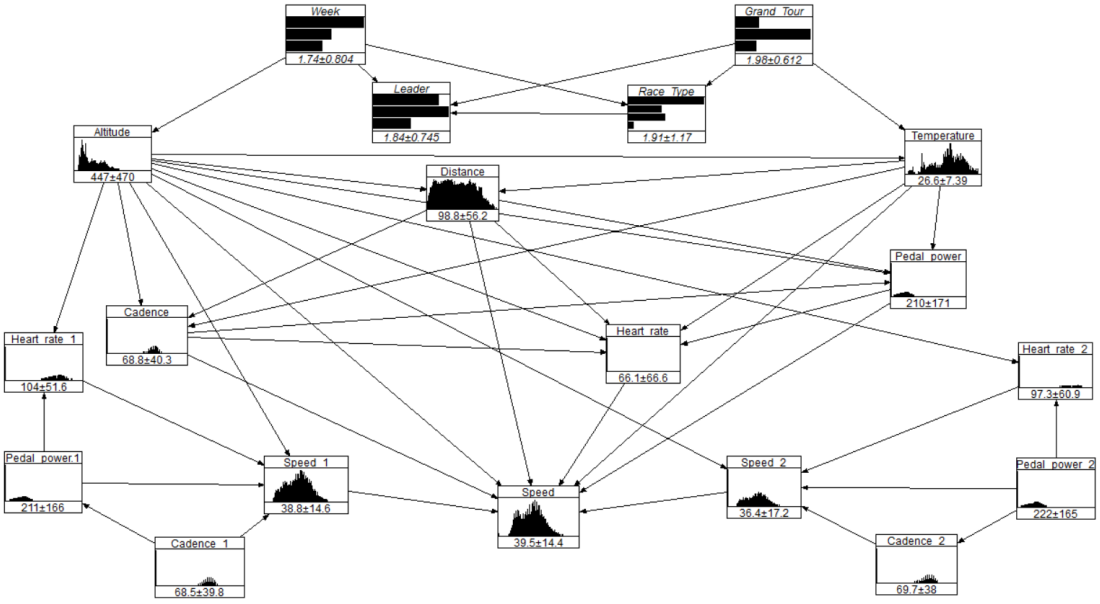


Figure 4.14: Bayesian network for the leader with the assistance of two helpers during a specific day in one of the Grand Tours.

In Figure 4.14 the nodes express the marginal distributions. For the discrete nodes we find bars that indicate the relative frequency in which the factors occur. For the continuous variables we find histograms. Notice that the histograms for the rider variables, except for Speed, are displayed very small. This is because there are a lot of zeros in the other rider variables, which is hard to see in the histograms.

5

PREDICTING PERFORMANCE

5.1. GENERAL INFORMATION

Prediction with Bayesian networks. How does it work? Once we have learned the full structure from data, or from expert judgement or a combination of both we can use the network for prediction and diagnostics (there are other tasks that can be performed, but we will focus on these two features).

We will use the statistical software packages R and Uninet to do this. Both R and Uninet support discrete and continuous data, but R can't handle a combination of the two. In our case it will treat the discrete variables as Grand Tour and Leader as continuous variables, but we can still do some prediction. However, the main difference between R and Uninet is the fact that Uninet supports non-parametric continuous and discrete BN's[8]. Non-parametric means that the model doesn't involve any assumptions for the distributions of the random variables. This is not the case in R. Earlier we spend some time on the normality assumption when we introduced continuous variables, because the package *bnlearn* in R assumed a multivariate normal distribution. Besides that, the user-friendliness and interface of Uninet is a huge advantage over that of R's.

5.2. PREDICTION WITH R

In section 4.3 we discussed our four sub questions and we mentioned that the structures didn't give us satisfactory results. For the race types, rider type and number of days of activity we are going to study if the performance for different factors is different by using the Bayesian networks presented in the corresponding subsections in section 4.3. At last we will make some predictions based on a Bayesian network created by all complete measurements. This means that we

don't take measurements into account without a heart rate.

Our approach for predictions in R is called cross validation. First we split the data in two random subsets: training(60%) and testing(40%). We build the Bayesian network for the data in the training set and we predict the values for speed in the testing set. In this way we can compare the predictions with the actual values. The first case we are going to discuss is race type.

5.2.1. RACE TYPES

In this section some very obvious and clear relationships emerge. The first thing that stands out is that the predicted average speed during a flat stage is higher than during a stage where we encounter hills, mountains or an arrival uphill. Furthermore the standard deviation shows a bigger variation from the average speed in mountain stages than in flat races, which makes sense.

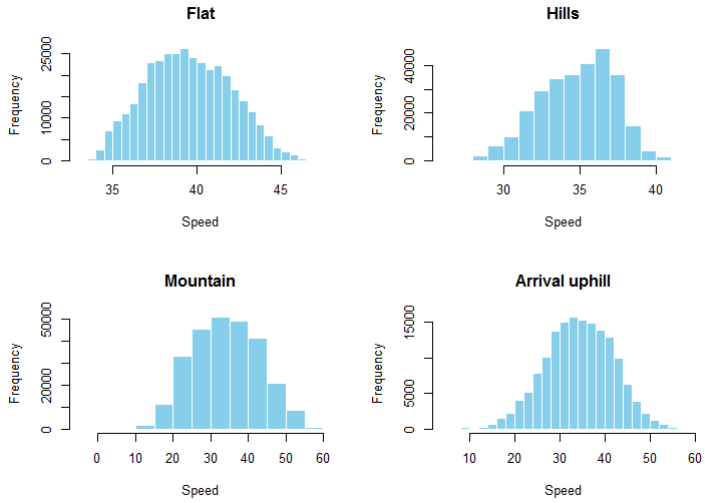


Figure 5.1: Histograms of the predicted values in speed for each race type.

Race Type	Mean	Standard deviation
Flat	39.51198	2.529824
Hills	34.80254	2.436338
Mountain	34.04441	8.869015
Arrival uphill	34.20073	7.548717

Table 5.1: Mean and standard deviation for each race type in the predictions for the speed.

But how are the predictions in comparison with the real values? The measures we are going to use are the Root Mean Squared Error and the Mean Absolute Error. The first one is the square root of the average of squared errors and the second one computes the mean absolute values of errors. So that is:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} \left(\hat{y}_i - y_i\right)^2}{n}},$$

And really straightforward:

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |\hat{y}_i - y_i|,$$

where $\hat{y}_i$ are the predicted values, y_i the real values and n the number of measurements.

Race Type	RMSE	MAE
Flat	13.65144	10.38366
Hills	18.23249	14.56888
Mountain	15.32191	11.82034
Arrival uphill	16.73998	13.41917

Table 5.2: Errors in the predictions for the speed.

If we compare these errors with the magnitude of speed in most races, both measures indicate that the error is large. If we do the same for the pedal power we find extremely large errors. The RSME is 129.4701 and the MAE is 91.3816. We will see in Figure 5.4 that the difference between the predicted values for pedal power and the real values are huge.

5.2.2. Rider type

The histograms for the predicted values for speed are almost the same, see Figure 5.2. The average speed for the leader appears to be higher, but that's the only big difference between rider type. So also in the predictions we don't find big differences when comparing the speed of the leader and the helpers.

Rider type	Mean	Standard deviation
Leader	39.11687	3.008775
Helper	36.8803	3.370341

Table 5.3: Mean and standard deviation for each rider type in the predictions for the speed.

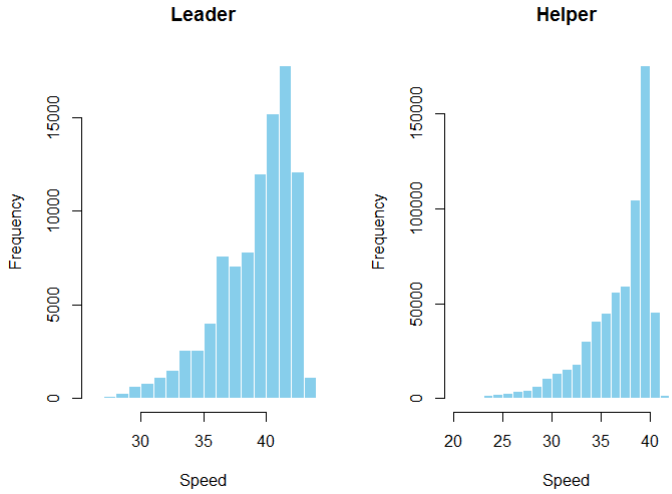


Figure 5.2: Histograms of the predicted values in speed for each rider type.

How about the errors? They are approximately the same as in race types, so it is questionable if we could use the predictions.

Rider type	RMSE	MAE
Leader	14.41618	11.42236
Helper	16.36893	12.69249

Table 5.4: Errors in the predictions for the speed.

5.2.3. Days of activity

The shape of the histograms is the same, but it is remarkable that the predicted average speed increases as the number of weeks increases. We expected this the other way around, because of fatigue during a Grand Tour.

Week	Mean	Standard deviation
1	35.35869	4.946916
2	36.61161	6.779316
3	37.05577	7.822147

Table 5.5: Mean and standard deviation for every week in the predictions for the speed.

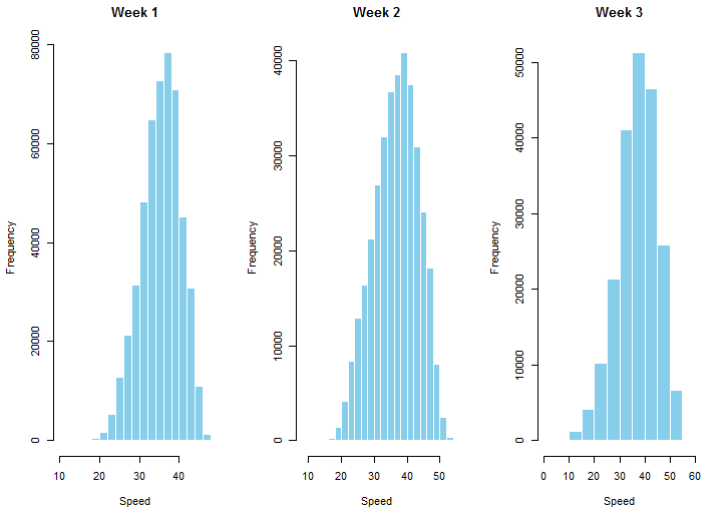


Figure 5.3: Histograms of the predicted values in speed for every week.

The behaviour of the errors is the same as with race- and rider type.

Week	RMSE	MAE
1	16.78276	13.13618
2	15.18715	12.05252
3	14.40703	11.25932

Table 5.6: Errors in the predictions for the speed

If we come back to our original data with the complete measurements we can compare the predictions in speed, cadence, heart rate and pedal power visually in Figure 5.4. In the histogram of pedal power the predicted values are in white instead of blue.

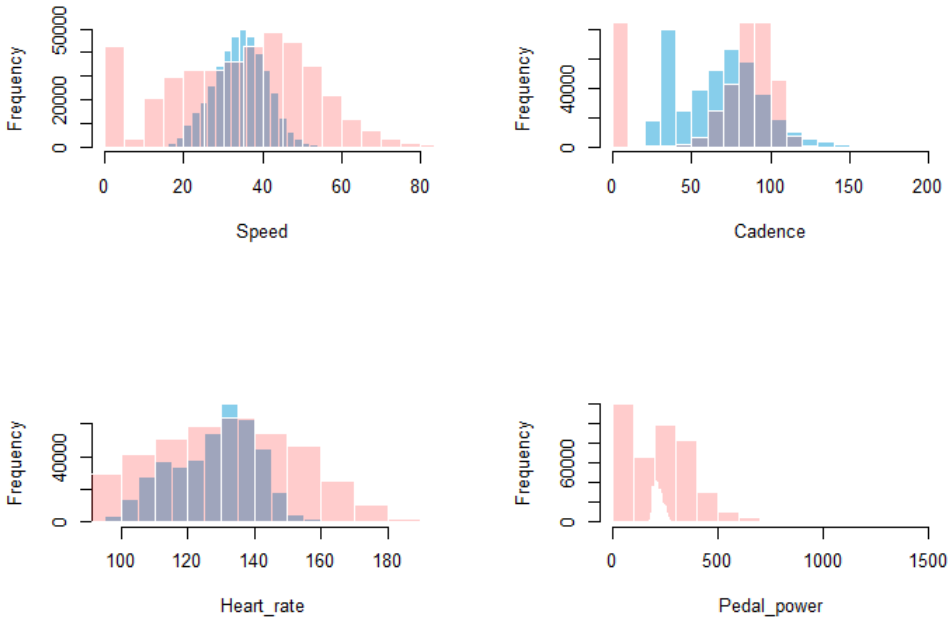


Figure 5.4: Histograms of the rider variables speed, cadence, heart rate and pedal power where we can compare the predicted values (in blue) with the real values (in red).

As mentioned before, the predictions for pedal power stand out. This could be caused by the capricious character of the pedal power that we saw in 2.1 or the violation of the normality assumption. The predicted values for speed, cadence and heart rate don't follow the same distributions as the real values. This can also be seen in the errors in Table 5.7.

Variable	RMSE	MAE
Speed	17.11918	13.61947
Cadence	28.31759	24.17755
Heart rate	21.80642	17.22127
Pedal power	166.2087	133.8682

Table 5.7: Errors in the predictions for the speed, cadence, heart rate and pedal pedal.

5.3. Prediction with Uninet

5.3.1. Predictions

The advantage of using Uninet in prediction is the fact that we can conditionalize on the parameters we want. Let's create a setting.

While writing the thesis, the Tour de France 2017 started a few days ago. In the last week of the Tour there are some heavy stages scheduled. On the 19th of July the riders travel from La Mure to Serre Chevalier and they come across mountains like de Col de la Croix de Fer, Col du Télégraphe and Col du Galibier. The top of the last mountain on 2610 meters is reached after approximately 134 kilometers. Assume the weather forecast is very positive about this day and they forecaste a temperature of 25 °C. Suppose team Sunweb decided to include rider 7 again in their team for the Tour and they assigned him as leader for this stage. What is the influence of these conditions on for example the speed, pedal power or the heart rate of his helpers?

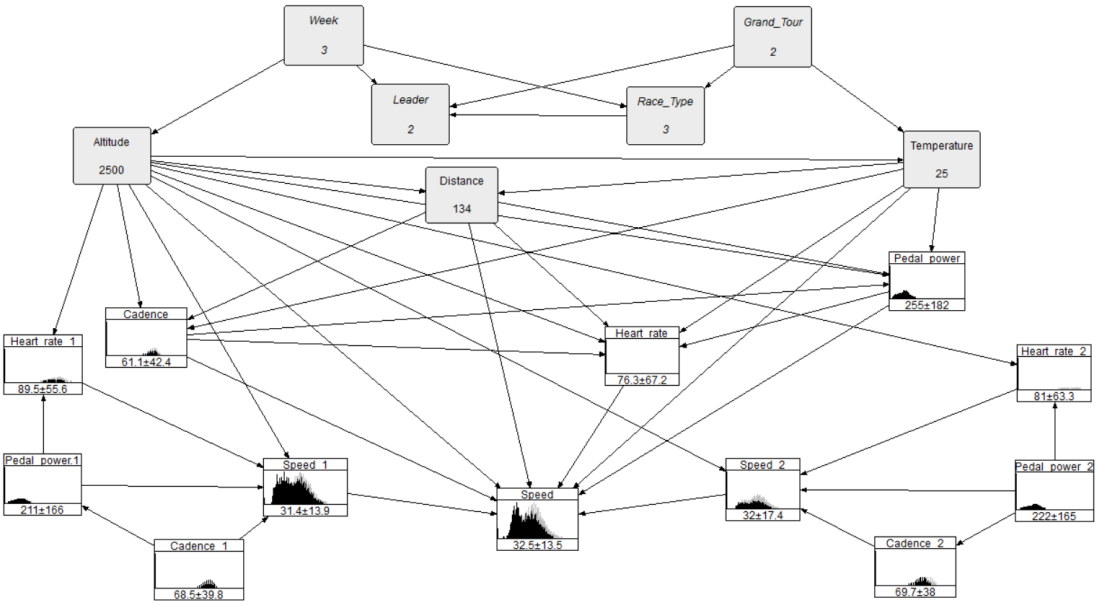


Figure 5.5: Conditionalized Bayesian network for the leader with the assistance of two helpers on the race variables.

In the Bayesian network we can see the consequences of our setting. While the predicted values for rider 7's speed and cadence decline, his pedal power and heart rate increase. For example, the average Pedal power increased from 210

to 255, while the standard deviation changed from 171 to 182. The speed and cadence of the helpers show the same behaviour, but the pedal power and heart rate don't. The pedal powers remain the same and the heart rates decline. We think these last observations are caused by the fact that the there is only one link between the race variables and the variables of helpers.

5.3.2. DIAGNOSTICS

Another feature of prediction is called diagnostics. In cycling this is not an obvious choice, but in for example health care this is an extremely import topic to discover the causes or symptoms of a disease. To create another setting: let's say we lost the information about all race variables, all information that we have left concerns the rider variables. Is the data most likely from rider 4 in the second week of Giro d'Italia? Or from rider 7 in the first week of the Vuelta?

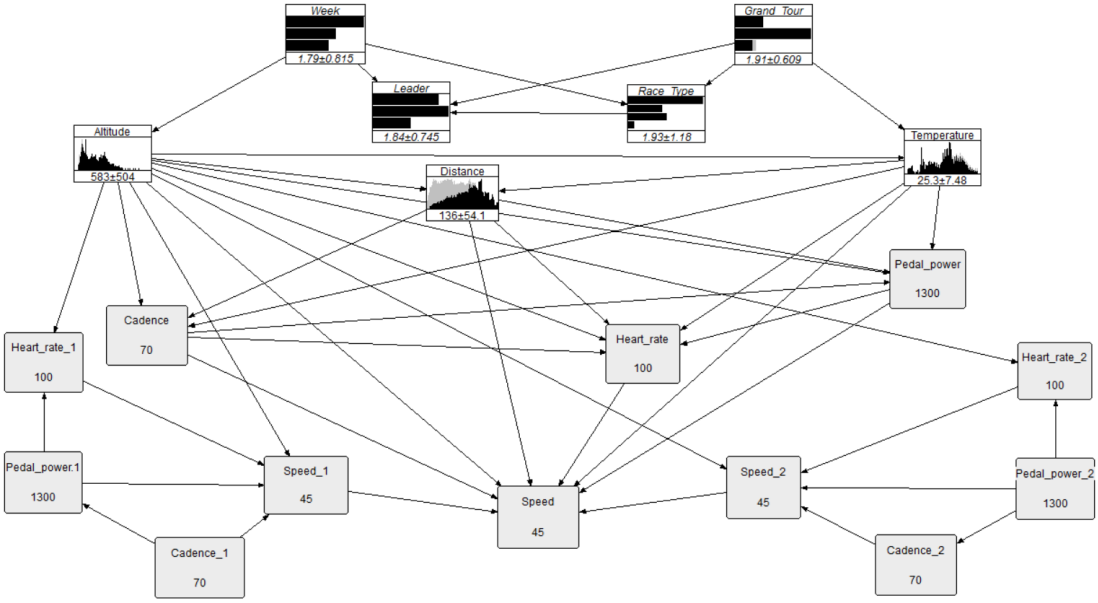


Figure 5.6: Conditionalized Bayesian network for the leader with the assistance of two helpers on the rider variables.

So it is not exactly clear in which Grand Tour, week and race type the leader performed, but is more likely this situation happened at the end of the race and a relative high altitude.

6

CONCLUSION AND RECOMMENDATIONS

6.1. CONCLUSION

In this thesis, the performance of a single rider in one of the Grand Tours and especially with the help of his team was investigated. The research focused on the main question:

> *How good is the performance of the leader during a race and how is this influenced by the helpers?*

We investigated this by the following sub questions:

1. What is the influence on the structure of the data caused by the three Grand Tours?
2. How does the structure of the data change when there is a difference in race type?
3. Does the structure of the data change when we encounter a different rider type? If so, how?
4. What is the influence on the structure of the number of days riders are in succession in race?

We can draw several conclusions from the analysis on the Bayesian networks in cycling, but unfortunately not all of them to our satisfaction. We compared the

different BN's that we created for every Grand Tour. The differences were minimal, so the influence on the structure of the data caused by a Grand Tour is minimal. This was a result that met our expectations.

On the other hand the influence of the race type is not so clear. The differences in the BN's were big, but this did not appear in the predictions. With this in mind we are not sure if the differences in the networks are caused by the algorithms the networks are build with or that there are really different structures.

There doesn't seem to be an influence of the rider type on the Bayesian networks we created. Also in the predictions there was no clear difference. The same happened for the number of days in activity. The influence of this variable appears to be minimal in the creation of the BN's too.

Overall we can say that the Bayesian network we created with the help of an expert and the data captures the most common relationships between all variables, but that the model doesn't reveal surprising relationships or good predictions. It does however provide a graphical tool to visualize the dependencies between random variables.

6.2. Recommendations

Based on the insights gained during this thesis, we are able to make a few recommendations for further research and for team Sunweb. To begin with a general remark: the world of cycling is definitely more complicated than we thought at the beginning of the thesis and we believe there is a way to find the performance of a cyclist. Maybe with other models that account for uncertainty and dependence the performance can be described.

The only thing we were not able to do in the analysis is to find the influence of the fact that there isn't a leader assigned to a stage. This could also be an interesting subject.

Furthermore we want to emphasize that for following studies it is helpfull that there is a heart rate for every rider each day and that the alignment problem could be easily fixed by turning the device for every cyclist on at the same time.

Appendices

A

Description

Some figures from the description of the data are also added here. This concerns the full and detailed GPS coordinates and the original and shifted data to see how the alignment works.

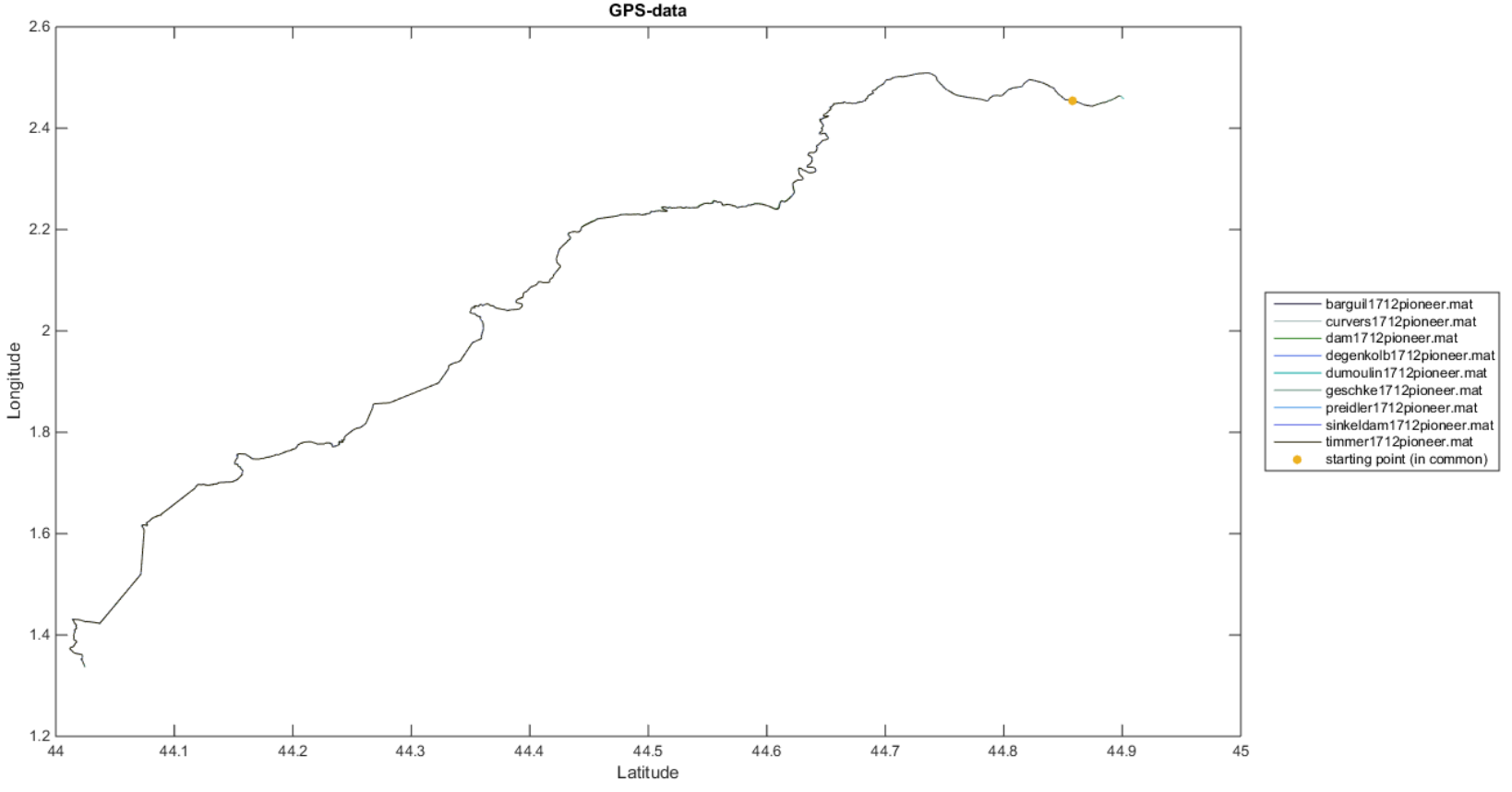


Figure A.1: Full GPS coordinates of all riders on 7 July 2016 and the first common point.

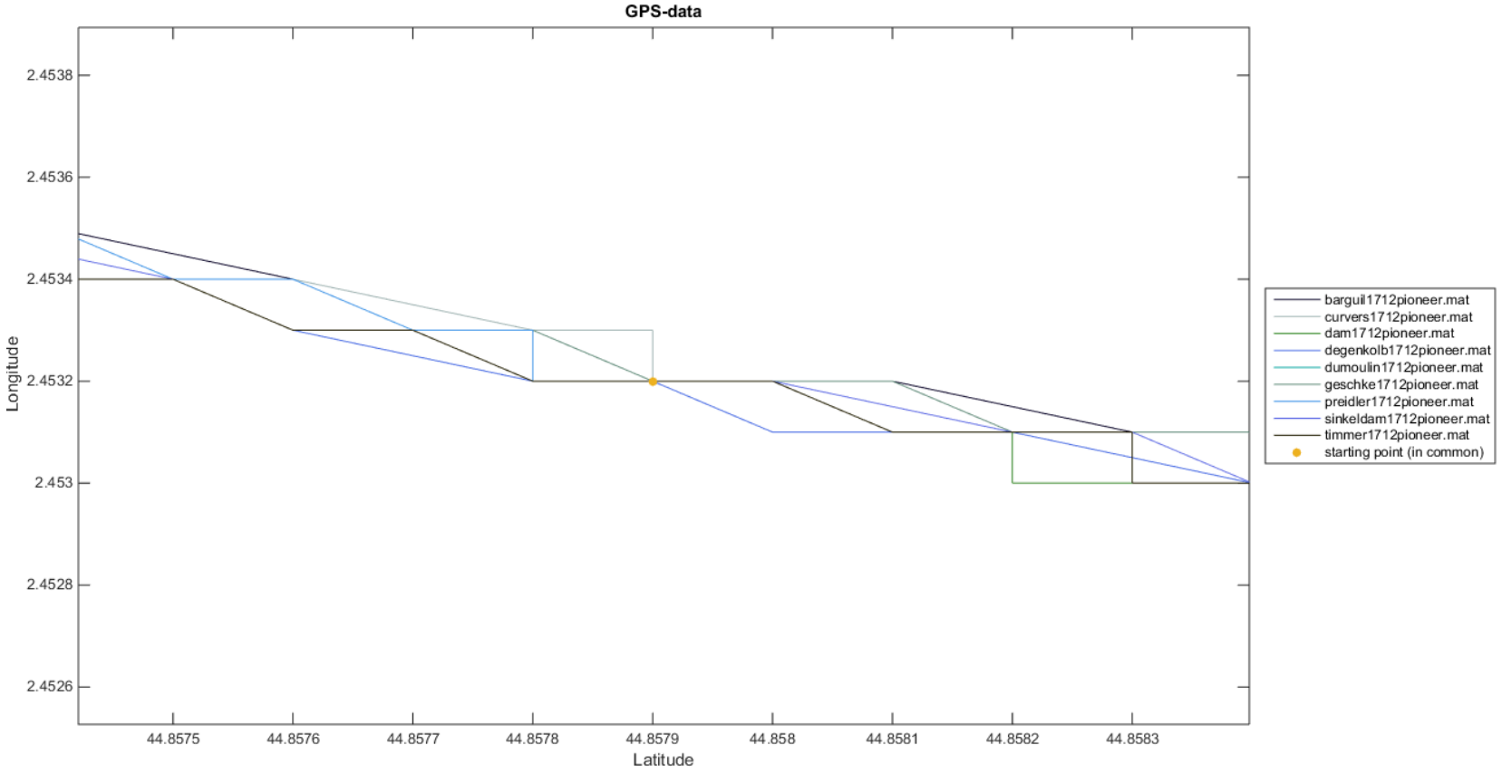


Figure A.2: Detailed zoom of the GPS data of 9 different cyclists on 7 July 2016.

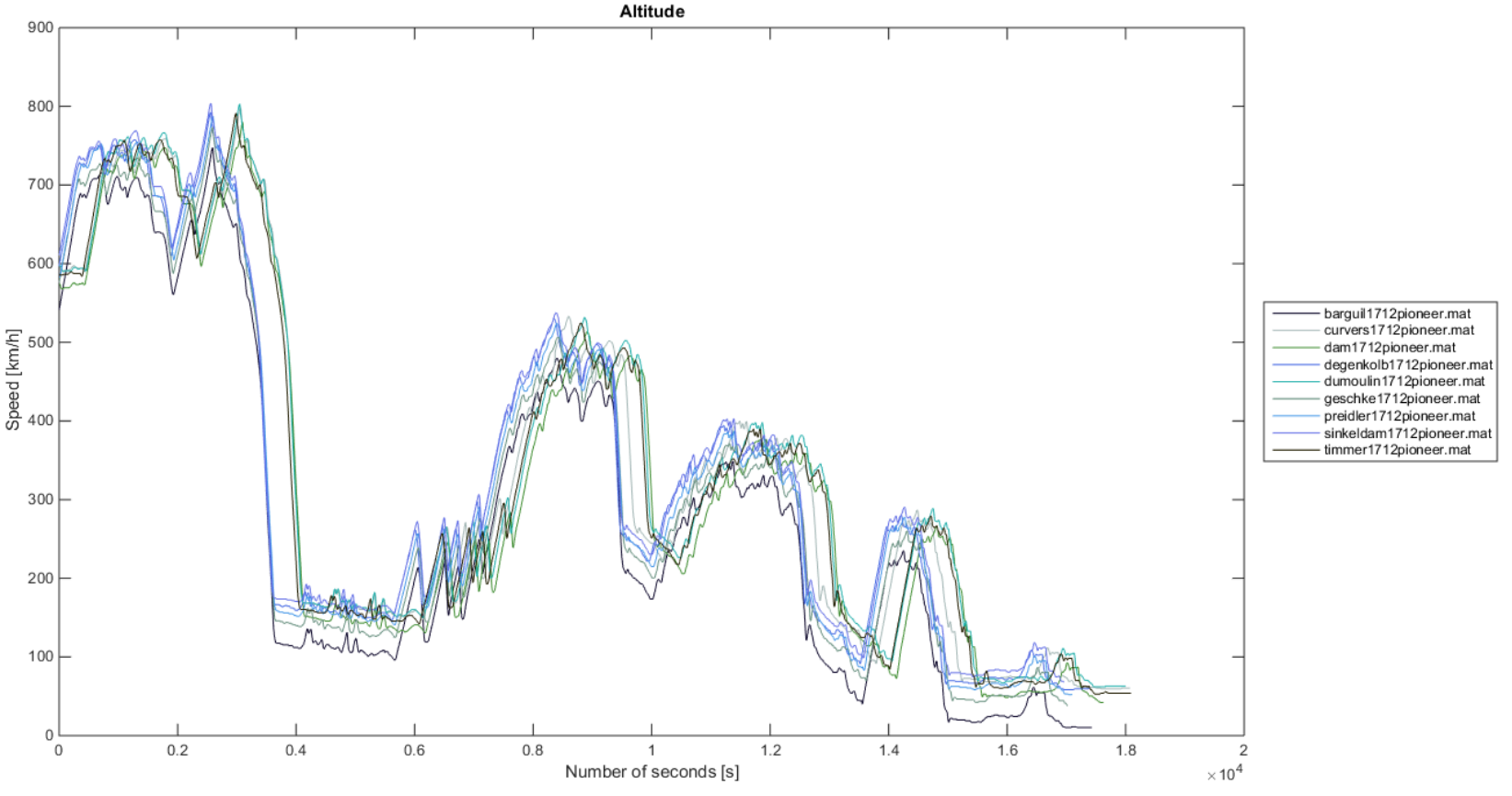


Figure A.3: The original plot of the GPS coordinates.

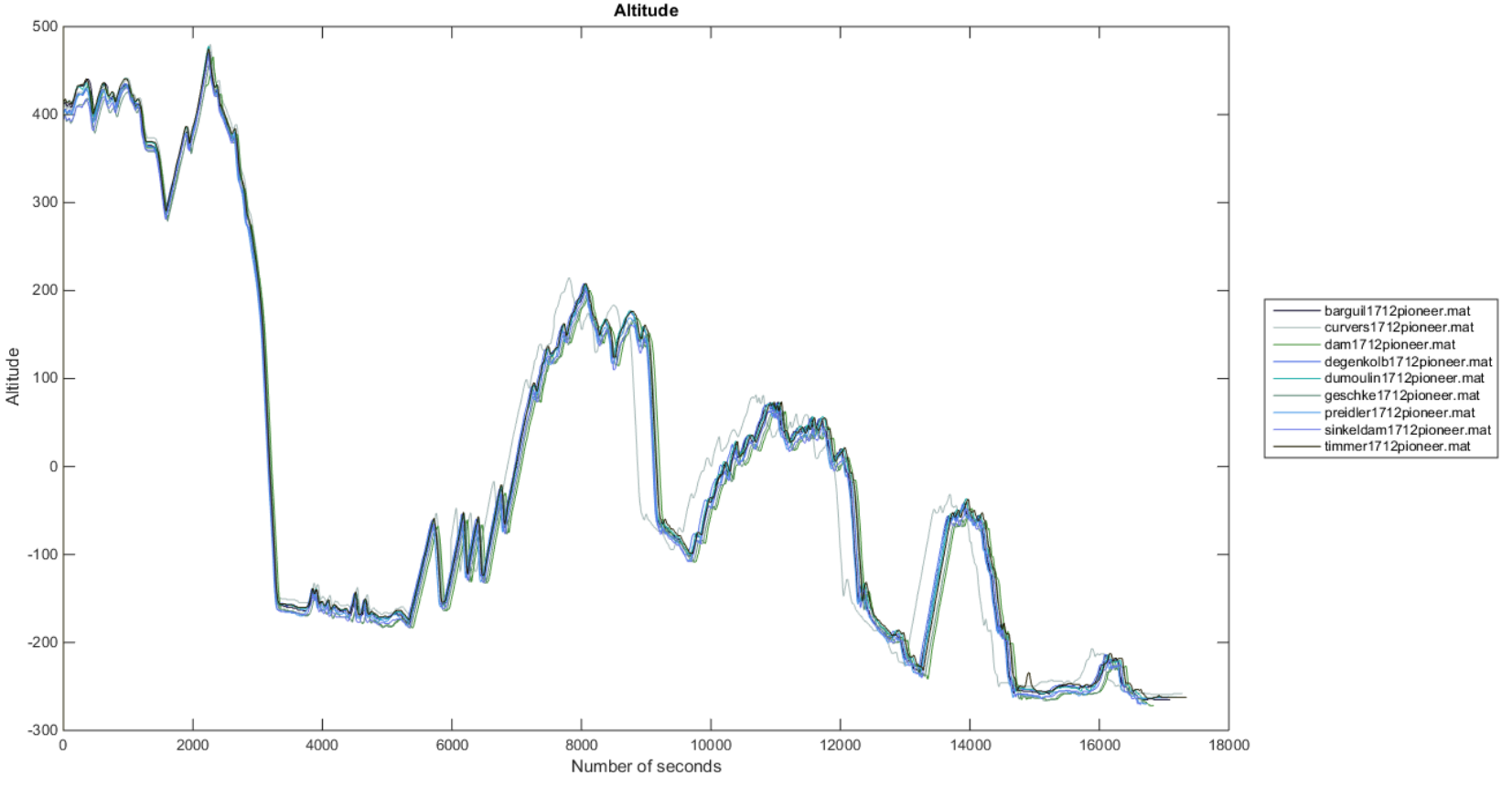


Figure A.4: The aligned plot of the GPS coordinates.

B

Cycling network

For completion the common arcs from the different race types in section 4.3. For each Grand Tour four different BN's are created (flat, hills, mountain and arrival uphill) and the common arcs are represented in the table below.

Giro		Tour		Vuelta	
From	To	From	To		
Altitude	Distance	Altitude	Distance	Altitude	Cadence
Altitude	Heart rate	Altitude	Heart rate	Altitude	Pedal power
Altitude	Pedal power	Altitude	Speed	Altitude	Speed
Altitude	Speed	Altitude	Temperature	Distance	Cadence
Altitude	Temperature	Cadence	Heart rate	Distance	Heart rate
Distance	Cadence	Cadence	Speed	Distance	Pedal power
Distance	Heart rate	Distance	Cadence	Distance	Speed
Distance	Pedal power	Distance	Pedal power	Distance	Temperature
Distance	Speed	Distance	Speed	Speed	Heart rate
Temperature	Cadence	Pedal power	Cadence	Temperature	Cadence
Temperature	Heart rate	Pedal power	Speed	Temperature	Heart rate
Temperature	Pedal power	Temperature	Cadence	Temperature	Pedal power
Temperature	Speed	Temperature	Heart rate		
		Temperature	Pedal power		
		Temperature	Speed		

Table B.1: Comparison of the network structures learned from all the available data for different race types. For each Grand Tour four different BN's are created (flat, hills, mountain and arrival uphill) and the common arcs are represented in this table.

C

MATLAB CODE

C.1. MAIN.M

```
clear all

%%%
%   Main file of synchronization based on gps-data
%%%

race = 'TDF';
day = 1712;

% Get directory to specified race and day
directory = sprintf('C:/Users/mark/Documents/MATLAB/BEP/%s/%d',race,day);
cd(directory)
vals = getFiles();

%% Find the first common gps-data and index for all riders
[vals,H,B,N_vector] = gps_shift(vals);
[index_all] = findIndex(vals,H,N_vector);
fnames = dir('*.mat');

%% Plot gps-data with common starting point
figure
legendInfo = new_legend(fnames,N_vector);
rand('state',149)
C = rand(10,3);

for K = N_vector
    x = vals{K}.data_pioneer(:,61);x(x==0) = NaN;
    y = vals{K}.data_pioneer(:,62);y(y==0) = NaN;
    plot(x,y,'Color',C(K,:));
    hold on
end

% Find first overlapping gps-coordinates
```

```
34 h = plot_first_point(vals,index_all,H);
35
36 % Plot starting point (if exists)
37 if size(H,1) ~= 0
38     plot(H(h(1),1),H(h(1),2),'.','Markersize',20)
39     legendInfo{length(N_vector)+1} = 'starting point (in common)';
40 end
41
42 title('GPS-data')
43 xlabel('Latitude')
44 ylabel('Longitude')
45 legend(legendInfo,'Location','eastoutside')
46
47 %% Look at altitude if synchronization on gps-data works
48
49 figure
50 N_legend = new_legend(fnames,N_vector);
51
52 gps = gpxread('TDF2016_stage2');
53 [start,finish] = gpx(vals,gps,N_vector);
54 dist = zeros(1,size(vals,2));
55
56 for i = N_vector
57     plot(vals{i}.data_pioneer(index_all(i):end,63) -...
58         mean(vals{i}.data_pioneer(index_all(i):end,63)),'Color',C(i,:))
59     hold on
60     if finish(i) == 0
61         dist(i) = 0;
62         continue
63     else
64         dist(i) = vals{i}.data_pioneer(finish(i),65)-vals{i}.data_pioneer(index_all(i
               ),65);
65     end
66 end
67
68 title('Altitude')
69 xlabel('Number of seconds')
70 ylabel('Altitude')
71 legend(N_legend,'Location','eastoutside')
72
73 for i = N_vector
74     plot([finish(i) finish(i)],get(gca,'ylim'),'Color',C(i,:))
75 end
```

C.2. GETFILES.M

```
1  function vals = getFiles()
2  % Get all matlab files from current directory
3
4  fnames = dir('*.mat');
5  numfids = length(fnames);
6  vals = cell(1,numfids);
7  for K = 1:numfids
8    vals{K} = load(fnames(K).name);
9  end
10 end
```

C.3. GPS_SHIFT.M

```
function [vals,H,B,N_vector] = gps_shift(vals)
%%%
%   Find out which riders are interesting to examine and then try to
%   find all overlapping gps-coordinates for given specified race
%   and day.
%
%   Args:
%       vals: the current dataset where we need to find matches (cell)
%
%   Returns:
%       List: containing the latitude and longitude of overlapping
%       gps-coordinates.
%%%

%% Get directory to specified race and day
% directory = sprintf('C:/Users/mark/Documents/MATLAB/BEP/%s/%d',race,day);
% cd(directory)
% vals = getFiles();

% Initialize
t = 3600;                           % maximum time in [sec] to find matches
L = size(vals,2);                   % number of riders this day
nullvec = zeros(1,L);               % dummy variable to test for no gps-data

for i = 1:L
    % Check if all riders drove more than 3600 seconds and if there is any
    % gps-data
    if length(vals{i}.data_pioneer(:,61))> t &&...
            isequal(vals{i}.data_pioneer(1:t,61),zeros(t,1)) == 0;
        nullvec(i) = i;
    end
end

% Index of riders with 'interesting' values
N_vector = nullvec(nullvec ~= 0);

[H,vals] = intersection(vals,N_vector,t);
% [B,vals] = last_intersect(vals,N_vector);
B = [];
end
```

C.4. FINDINDEX.M

```
function [index_all] = findIndex(vals,H,N_vector)
%%%
%   Find index of starting point for all riders given all overlapping
%   gps-coordinates.
%
%   Args:
%       vals: cell with all data of riders
%       H: list of matching gps-coordinates
%       N_vector: list of 'interesting' riders
%
%   Returns:
%       index_all: list of all indices of starting points.
```

```
%%%

index_all = zeros(1,size(vals,2));

for j = N_vector
    [~,index] = ismember(H, vals{j}.data_pioneer(:,61:62),'rows');
    index_all(j) = min(index);
end
end
```

C.5. NEW_LEGEND.M

```
function legendInfo = new_legend(fnames,N_vector)

legendInfo = cell(1,length(fnames));

for i = 1:length(fnames)
    if ismember(i,N_vector) == 1
        legendInfo{i} = fnames(i).name;
    end
end

legendInfo = legendInfo(~cellfun(@isempty, legendInfo));
end
```

C.6. PLOT_FIRST_POINT.M

```
function h = plot_first_point(vals,index,overlap)
[~,F] = find(index ~= 0);
first = min(F);
I = vals{first}.data_pioneer(index(first),61);
h = find(overlap == I);
end
```

D

R CODE

D.1. DATA IMPORT

```
1
2  ## All data is imported, but to save some space we only put down the first and last
       data set.
3  day1651 = read.csv("C:/Users/mark/Documents/MATLAB/Bep/Giro/1652/testdata1.csv",
       header = TRUE); day1651 = col_names(day1651)
4  ...
5  day1767 = read.csv("C:/Users/mark/Documents/MATLAB/Bep/Vuelta/1767/testdata1.csv",
       header = TRUE); day1767 = col_names(day1767)
6
7  ## The data is split per day
8  day1707_tdf = list(day1707[,2:8],day1707[,10:16],day1707[,18:24],day1707[,26:32],
       day1707[,34:40],day1707[,42:48],day1707[,50:56],day1707[,58:64],day1707[,66:72]);
9  ...
10 day1777_vuelta = list(day1777[,2:8],day1777[,10:16],day1777[,18:24],day1777[,26:32],
       day1777[,34:40],day1777[,42:48]);
```

D.2. MAIN

```
1  # Clear the current workspace
2
3  rm(list=ls())
4
5  library(bnlearn)
6  library(lmtest)
7  library(forecast)
8
9  ## TDF
10 tdf = c(day1707_tdf,day1708_tdf,day1709_tdf,day1710_tdf,day1711_tdf,day1712_tdf,
       day1713_tdf,day1714_tdf,day1715_tdf,day1717_tdf,day1718_tdf,day1719_tdf,day1721_
       tdf,day1722_tdf,day1723_tdf,day1725_tdf);
11
12 bn.tdf = BN_rider(tdf); graphviz.plot(bn.tdf)# Hill climbing
13 bn.tdf1 = BN_rider(tdf,algo = gs); graphviz.plot(bn.tdf1)
14 bn.tdf2 = BN_rider(tdf,algo = iamb); graphviz.plot(bn.tdf2)
```

```
bn.tdf3 = BN_rider(tdf,algo = inter.iamb); graphviz.plot(bn.tdf3)
bn.tdf4 = BN_rider(tdf,algo = fast.iamb); graphviz.plot(bn.tdf4)

## Giro
giro = c(day1651_giro,day1652_giro,day1654_giro,day1655_giro,day1656_giro,day1657_
    giro,day1658_giro,day1663_giro,day1669_giro);
highlight.opts = list(arcs = c("Heart_rate","Speed","Altitude","Speed","Distance","
    Cadence","Altitude","Heart_rate","Altitude","Cadence",
                               "Distance","Speed","Altitude","Pedal_power","
                                   Temperature","Cadence","Distance","Pedal_power","
                                   Temperature","Heart_rate"))

bn.giro = BN_rider(giro); graphviz.plot(bn.giro, highlight = highlight.opts) ## HC

bn.giro1 = BN_rider(giro,algo = gs); graphviz.plot(bn.giro1,highlight = highlight.
    opts)
bn.giro2 = BN_rider(giro,algo = iamb); graphviz.plot(bn.giro2)
bn.giro3 = BN_rider(giro,algo = inter.iamb); graphviz.plot(bn.giro3)
bn.giro4 = BN_rider(giro,algo = fast.iamb); graphviz.plot(bn.giro4)

## Vuelta
vuelta = c(day1757_vuelta,day1758_vuelta,day1759_vuelta,day1760_vuelta,day1761_vuelta
    ,day1762_vuelta,day1763_vuelta,day1764_vuelta,day1765_vuelta,day1767_vuelta,
    day1768_vuelta,day1769_vuelta,day1770_vuelta,day1771_vuelta,day1772_vuelta,
    day1774_vuelta,day1775_vuelta,day1777_vuelta)
bn.vuelta = BN_rider(vuelta); graphviz.plot(bn.vuelta)

bn.vuelta1 = BN_rider(vuelta,algo = gs); graphviz.plot(bn.vuelta1)
bn.vuelta2 = BN_rider(vuelta,algo = iamb); graphviz.plot(bn.vuelta2)
bn.vuelta3 = BN_rider(vuelta,algo = inter.iamb); graphviz.plot(bn.vuelta3)
bn.vuelta4 = BN_rider(vuelta,algo = fast.iamb); graphviz.plot(bn.vuelta4)

## All
all_data = c(tdf,giro,vuelta)
bn.all = BN_rider(all_data); graphviz.plot(bn.all)

bn.all_5 = BN_rider(all_data,inc = 5); plot(bn.all_5)
bn.all_10 = BN_rider(all_data,inc= 10); plot(bn.all_10)
bn.all_20 = BN_rider(all_data,inc= 20);
bn.all_30 = BN_rider(all_data,inc= 30);

### Race types

flat = c(day1651_giro,day1652_giro,day1663_giro,day1669_giro,day1707_tdf,day1708_tdf,
    day1709_tdf,day1710_tdf,
         day1718_tdf,day1721_tdf,day1723_tdf,day1757_vuelta,day1760_vuelta,day1772_
             vuelta,day1775_vuelta)
hills = c(day1654_giro,day1655_giro,day1657_giro,day1658_giro,day1711_tdf,day1712_tdf
    ,day1761_vuelta,day1762_vuelta,
          day1768_vuelta,day1769_vuelta,day1771_vuelta)
mountain = c(day1656_giro,day1670_giro,day1713_tdf,day1714_tdf,day1715_tdf,day1717_
    tdf,day1722_tdf,day1725_tdf,
             day1770_vuelta,day1774_vuelta,day1777_vuelta)
arrival = c(day1719_tdf,day1758_vuelta,day1759_vuelta,day1763_vuelta,day1764_vuelta,
    day1765_vuelta,day1767_vuelta)

bn.flat = BN_rider(flat); graphviz.plot(bn.flat)
```

```
59 bn.hills = BN_rider(hills); graphviz.plot(bn.hills)
60 bn.mountain = BN_rider(mountain); graphviz.plot(bn.mountain)
61 bn.arrival = BN_rider(arrival); graphviz.plot(bn.arrival)
62
63 ### Predictions
64 pred_flat = pred(flat)
65 pred_hills = pred(hills)
66 pred_mountain = pred(mountain)
67 pred_arrival = pred(arrival)
68
69 par(mfrow=c(2,2))
70
71 hist(pred_speed,col='skyblue',border = F,xlab="Speed",main = "Flat")
72 hist(pred_speed1,add=F,col='skyblue',border=F,xlab="Speed",main = "Hills")
73 hist(pred_speed2,add=F,col='skyblue',border=F,xlab="Speed",main= "Mountain")
74 hist(pred_speed3,add=F,col='skyblue',border=F,xlab="Speed",main= "Arrival uphill")
75
76 ### Leader and helpers
77
78 kopman_giro = list(day1651[,2:8],day1652[,2:8],day1654[,26:32],day1657[,2:8],day1658
       [,18:24],day1663[,2:8],day1669[,2:8])
79 kopman_tdf = list(day1707[,26:32],day1708[,2:8],day1709[,26:32],day1710[,2:8],day1711
       [,2:8],day1712[,26:32],day1713[,2:8],day1714[,2:8],day1715[,2:8],day1717[,2:8],
       day1718[,26:32],day1719[,2:8],day1721[,26:32],day1722[,2:8],day1723[,18:24],
       day1725[,2:8])
80 kopman_vuelta = list(day1757[,2:8],day1760[,2:8],day1762[,2:8],day1772[,2:8],day1775
       [,2:8])
81
82 kopman = c(kopman_giro,kopman_tdf,kopman_vuelta);
83 bn.kopman = BN_rider(kopman); graphviz.plot(bn.kopman)
84 bn.kopman_gs = BN_rider(kopman,algo=gs)
85
86 helper_giro = c(day1651_giro[-1],day1652_giro[-1],day1654_giro[-4],day1655_giro[-1],
       day1657_giro[-1],day1658_giro[-3],day1663_giro[-1],day1669_giro[-1]);
87 helper_tdf = c(day1707_tdf[-4],day1708_tdf[-1],day1709_tdf[-4],day1710_tdf[-1],
       day1711_tdf[-1],day1712_tdf[-4],day1713_tdf[-1],day1714_tdf[-1],day1715_tdf[-1],
       day1717_tdf[-1],day1718_tdf[-4],day1719_tdf[-1],day1721_tdf[-4],day1722_tdf[-1],
       day1723_tdf[-4],day1725_tdf[-1]);
88 helper_vuelta = c(day1757_vuelta[-1],day1760_vuelta[-1],day1762_vuelta[-1],day1772_
       vuelta[-1],day1775_vuelta[-1]);
89
90 helper = c(helper_giro,helper_tdf,helper_vuelta)
91 bn.helper = BN_rider(helper); graphviz.plot(bn.helper)
92
93 pred_kopman = pred(kopman)
94 pred_helper = pred(helper)
95
96 par(mfrow=c(1,2))
97 hist(pred_kopman,col='skyblue',border = F,xlab="Speed",main = "Leader")
98 hist(pred_helper,add=F,col='skyblue',border=F,xlab="Speed",main = "Helper")
99
100 ### Weeks
101
102 week1_giro = c(day1651_giro,day1652_giro,day1654_giro,day1655_giro,day1656_giro,
       day1657_giro)
103 week2_giro = c(day1658_giro,day1663_giro)
104 week3_giro = c(day1669_giro,day1670_giro)
```

```
105
106 week1_tdf = c(day1707_tdf,day1708_tdf,day1709_tdf,day1710_tdf,day1711_tdf,day1712_tdf
        ,day1713_tdf)
107 week2_tdf = c(day1714_tdf,day1715_tdf,day1717_tdf,day1718_tdf,day1719_tdf,day1721_tdf
        )
108 week3_tdf = c(day1722_tdf,day1723_tdf,day1725_tdf)
109
110 week1_vuelta = c(day1757_vuelta,day1758_vuelta,day1759_vuelta,day1760_vuelta,day1761_
        vuelta,day1762_vuelta)
111 week2_vuelta = c(day1763_vuelta,day1764_vuelta,day1765_vuelta,day1767_vuelta,day1768_
        vuelta,day1769_vuelta,day1770_vuelta)
112 week3_vuelta = c(day1771_vuelta,day1772_vuelta,day1774_vuelta,day1775_vuelta,day1777_
        vuelta)
113
114 week1 = c(week1_giro,week1_tdf,week1_vuelta)
115 week2 = c(week2_giro,week2_tdf,week2_vuelta)
116 week3 = c(week3_giro,week3_tdf,week3_vuelta)
117
118 bn.week1 = BN_rider(week1)
119 bn.week2 = BN_rider(week2)
120 bn.week3 = BN_rider(week3)
121
122 pred_week1 = pred(week1)
123 pred_week2 = pred(week2)
124 pred_week3 = pred(week3)
125
126 par(mfrow=c(1,3))
127 hist(pred_week1,col='skyblue',border = F,xlab="Speed",main = "Week 1")
128 hist(pred_week2,add=F,col='skyblue',border=F,xlab="Speed",main = "Week 2")
129 hist(pred_week3,col='skyblue',border = F,xlab="Speed",main = "Week 3")
130
131 #### Expert view
132 e = empty.graph(c("Altitude","Pedal_power","Distance","Heart_rate","Temperature","
        Cadence","Speed"))
133 bn_expert = c("Altitude", "Pedal_power", "Altitude", "Heart_rate", "Altitude", "
        Distance",
134               "Distance","Pedal_power","Distance","Heart_rate","Distance","Speed",
135               "Pedal_power","Heart_rate","Pedal_power","Speed","Heart_rate","Speed",
                      "Temperature","Distance",
136               "Temperature","Heart_rate","Temperature","Speed","Cadence","Pedal_power
                      ","Cadence","Speed","Altitude","Cadence")
137
138 arc.set = matrix(bn_expert,ncol = 2, byrow = TRUE,dimnames = list(NULL, c("from", "to
        ")))
139 arcs(e) = arc.set
140
141 graphviz.plot(e)
142
143 From = c("Temperature","Cadence","Pedal_power","Cadence","Altitude","Altitude","
        Distance")
144 To = c("Speed","Speed","Cadence","Pedal_power","Distance","Speed","Speed")
145 wl=data.frame(from = From,to = To)
146
147 From = rep(c("Cadence","Speed","Pedal_power","Heart_rate"),3)
148 To = c(rep("Temperature",4),rep("Distance",4),rep("Altitude",4))
149 bl=data.frame(from = From,to = To)
150
```

```
column_names = c("Pedal_power","Cadence","Heart_rate","Altitude","Speed","Distance","
    Temperature")
network = lapply(all_data, setNames, nm = column_names)

network = do.call(rbind,network)
names(network) = column_names
network = na.omit(network)
network = lapply(network,as.numeric)
network = data.frame(network)

bn_expert_all = hc(network,whitelist = wl,blacklist = bl)
```

D.3. FUNCTIONS

```
col_names <- function(dataset){
  x=dim(dataset)[2];
  y=paste(c("Pedal_power_","Cadence_","Heart_rate_","Altitude_","Speed_","Distance_",
      "Temperature_","Energy_"),rep(1:(x%/%8),each=8),sep="")

  column_names = c("Day",y,"Type","Succesive days")
  names(dataset) = column_names
  return(dataset)
}

BN <- function(day,cyclist){
  From = rep(c("Cadence","Speed","Pedal_power","Heart_rate"),3)
  To = c(rep("Temperature",4),rep("Distance",4),rep("Altitude",4))
  bl=data.frame(from = From,to = To)

  z = 8*cyclist
  End = Find.na(day,z)
  network = day[1:End,(z-6):z]
  names(network) = c("Pedal_power","Cadence","Heart_rate","Altitude","Speed","
      Distance","Temperature")
  network = lapply(network,as.numeric)
  network = data.frame(network)

  bn.network = hc(network,blacklist = bl)
}

Compare <- function(networks){
  y = compare(networks[[2,1]],networks[[3,1]],arcs=TRUE)$tp
  for(i in 2:(dim(networks)[1]-1)){
    for(j in (i+1):dim(networks)[1]){
      x = compare(networks[[i,1]],networks[[j,1]],arcs=TRUE)$tp
      y = merge(x,y)
    }
  }
  return(y)
}


BN_rider <- function(foo,algo = hc,inc = 1){
  From = rep(c("Cadence","Speed","Pedal_power","Heart_rate"),3)
  To = c(rep("Temperature",4),rep("Distance",4),rep("Altitude",4))
  bl=data.frame(from = From,to = To)
```

```
41    column_names = c("Pedal_power","Cadence","Heart_rate","Altitude","Speed","Distance"
          ,"Temperature")
42    network = lapply(foo, setNames, nm = column_names)
43
44    network = do.call(rbind,network)
45    names(network) = column_names
46    network = na.omit(network)
47    network = lapply(network,as.numeric)
48    network = data.frame(network)
49    network = network[seq(1,nrow(network),inc),]
50
51    bn.network = algo(network,blacklist = bl)
52    return(bn.network)
53  }
54
55  pred <- function(dataset){
56    set.seed(42)
57    spec = c(train = .6, test = .2, validate = .2)
58    column_names = c("Pedal_power","Cadence","Heart_rate","Altitude","Speed","Distance"
          ,"Temperature")
59    From = rep(c("Cadence","Speed","Pedal_power","Heart_rate"),3)
60    To = c(rep("Temperature",4),rep("Distance",4),rep("Altitude",4))
61    bl=data.frame(from = From,to = To)
62    dataset = lapply(dataset, setNames, nm = column_names)
63    dataset = do.call(rbind,dataset)
64    names(dataset) = column_names
65    dataset = na.omit(dataset)
66    dataset = lapply(dataset,as.numeric)
67    dataset = data.frame(dataset)
68    df = dataset
69
70    g = sample(cut(
71      seq(nrow(df)),
72      nrow(df)*cumsum(c(0,spec)),
73      labels = names(spec)
74    ))
75
76    res = split(df, g)
77
78    bn_dataset = hc(res$train,blacklist = bl)
79    fitted = bn.fit(bn_dataset, res$train)
80
81    pred_speed = predict(fitted,"Speed",res$validate)
82    #cbind(pred_speed,res$validate[,"Speed"])
83    return(pred_speed)
84  }
```

REFERENCES

[1] Smith, Jim Q. *Bayesian Decision Analysis: Principles and Practice.* Coventry: University of Warwick, 2000.

[2] Jensen, Finn V. Nielsen, Thomas D. *Bayesian Networks and Decision Graphs.* Aalborg University: 2007.

[3] Kjærulff, Uffe B. Madsen, Anders L. *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis.* Aalborg University, 2013.

[4] Scutari, Marco *Learning Bayesian Networks with the bnlearn R Package* University of Padova, 2010, downloaded from https://www.jstatsoft.org/article/view/v035i03

[5] Scutari, M. Denis, J-B. *Overview of Bayesian Networks With Examples in R* 2015, downloaded from http://www.ucdenver.edu/academics/colleges/PublicHealth/Academics/departments/Biostatistics/WorkingGroups/Documents/Networks%20Presentation%20With%20Sachs%20-%20032317.pdf

[6] laakkola, T. *course materials for 6.867 Machine Learning* Fall 2006, downloaded from https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-867-machine-learning-fall-2006/lecture-notes/lec21.pdf

[7] Koller, D. Friedman, N. *Probabilistic Graphical Models: principles and techniques* Massachusetts Institute of Technology 2009,

[8] *UNINET Help* downloaded from http://www.lighttwist.net/wp/uninet