A machine learning approach to the automatic classification of female uroflowmetry measurements

S.P.R. Baas

University of Twente, The Netherlands

June 25, 2016

Abstract

Uroflowmetry is a cheap, simple and noninvasive test that is often first used for patients with possible lower urinary tract dysfunctions. This test is usually automated to a high extend and the result is a graph of the urine voiding speed (ml/s) vs. the time (s). For uroflowmetry measurements obtained from women, these measurements are currently classified subjectively by one or more physicians. The measurements are classified into one of four groups, each indicating a set of underlying dysfunctions. In this research, it is investigated if this classification process can be successfully automated by constructing multiple automatic classification methods. For constructing these classifiers, a dataset of measurements and classifications by hospital staff from the University Medical Center in Utrecht is used. One of the constructed classifiers is the improved questionnaire proposed by van der Kamp [1], the other classification methods are constructed using machine learning methods. All classifiers are evaluated on a set of chosen performance measures. The ultimately chosen classifier is the regression forest classifier, which was shown to have a good overall performance and gives an estimated accuracy of 96.7% for the diagnosis of new patients.

I. Uroflowmetry

Classification of uroflowmetry measurements

Uroflowmetry is often one of the first tests patients with the possibility of having lower urinary tract (LUT) dysfunctions undergo. Because this test is noninvasive, cheap and simple, it was one of the first urodynamic tests to be automated. In the University Medical Center of Utrecht, the uroflowmetry measurements obtained from women are divided into four classes by urologists. This is now done in a completely subjective way. Every class indicates that the patient could suffer from a group of LUT dysfunctions. In this research, it is investigated whether this classification of digital measurements can also be automated. The motivation behind this is that it could give reassurance to hospital staff in evaluating the measurements and could make the classification of the measurements more reliable. This research will build on prior investigations by Brand, van der Kamp, Huizinga and Boele [2], [1], [3], [4].

In the current research, first all available uroflowmetry measurements were collected and stored. After pre-processing the measurements, the questionnaire optimization step proposed in the research of Boele and van der Kamp was replicated and improved to give an optimized questionnaire. Next, machine learning methods were employed to construct automatic classification methods (classifiers). After defining a number of performance measures and evaluating the values of these measures for the classifiers, the regression forest was shown to have the highest performance on uroflowmetry measurement classification with an estimated accuracy of 96.7% for not yet seen measurements.

Clinical urodynamics

Clinical urodynamics is used to measure voiding by the lower urinary tract (LUT for short), which consists of the ureters, bladder and the urethra. The purpose of this is having objective data of voiding by the patient which can help a physician in diagnosing certain LUT illnesses. There are many forms of urodynamics; e.g. videourody-

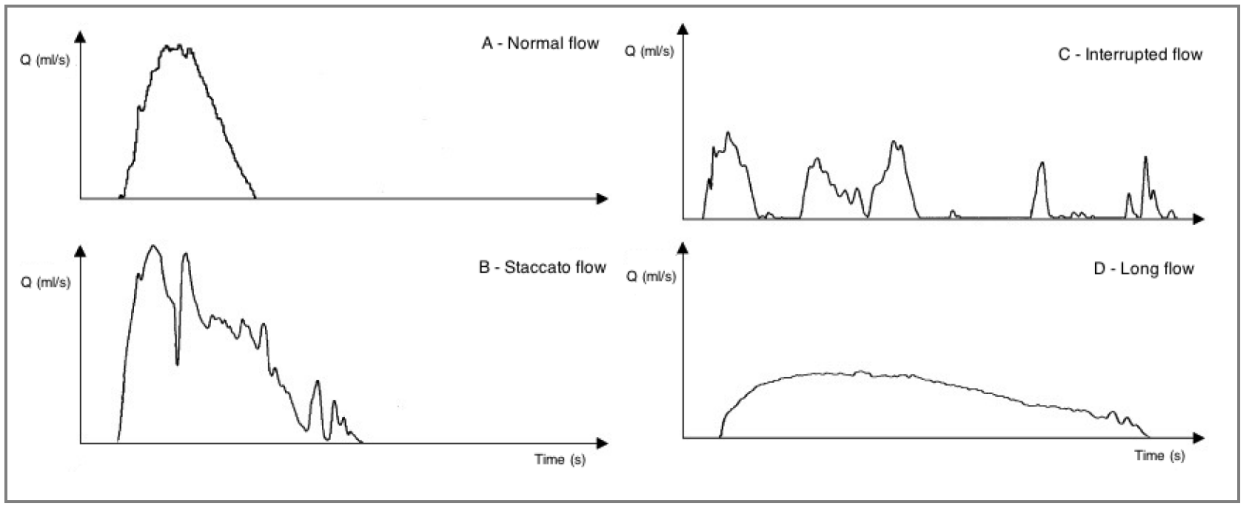

Figure 1: Four examples of uroflowmetry measurements assigned to the classes where the flow (denoted by Q) has a Normal, Staccato, Interrupted, and Long flow. Source: [1].

namics, urethral pressure measurements and systometry. Most of these procedures are invasive or expensive but uroflowmetry, a technique in which LUT voiding speed (ml/s) vs. time (s) is measured, is neither. Because of this, uroflowmetry is often the first test that patients with the probability of LUT dysfunctions undergo. The result of this test is an uroflowmetry measurement from which physicians can draw their first hypotheses about the patient's diagnosis. In the University Medical Center in Utrecht (UMCU for short), and for women above the age of 18, physicians often divide the uroflowmetry measurements in four different groups based on the appearance of these curves (see figure 1). The goal of this research is building an automatic classification method for uroflowmetry measurements that mimics the classification of this data by physicians from the UMCU in the best way possible. The reason behind this is that such an automatic classification method can provide an objective classification of the curves, reassuring the physicians on the one hand and making urflowmetry classifications more reliable on the other [5].

The uroflowmetry measurements considered in this research are obtained from a weight transducer uroflowmeter: Andromeda Ellipse M00101-2 [6]. In a weight transducer uroflowmeter, excreted urine lands in a container and the weight of the urine is measured vs time. From this weight, the volume of the landed urine is calculated and by differentiation, the flow rate. The advantages of the method are that it is relatively simple and the measurements are quite accurate. The disadvantages of this method are the large effects of urine density on the results, the slow response time and the occurrence of artifacts or noise in the data as a result of the aforementioned differentiation step. [7].

Patterns in uroflowmetry

Uroflowmetry curves have several characteristics, like maximum voiding speed or duration, on which a physician can base a diagnosis. Also, and more with women than with men, the shape of the curve is important. In the UMCU, there is consensus that uroflowmetry curves obtained from women can be divided in four groups: normal flow (A), staccato flow (B), interrupted flow (C) and long flow (D) (see figure 1) [1].

The normal flow curves, which are often smooth, approximately symmetrical and have high steepness, reflect a healthy voiding pattern while all the other patterns can be due to mul-

tiple LUT dysfunctions. Staccato flow and interrupted flow are characterized by oscillations. The difference between these two is that interrupted curves have periods during which the voiding speed drops to a very low point. A staccato curve often indicates that the patient suffers from bladder sphincter dyssynergia, a problem in the central nervous system regulation of lower urinary tract muscles, whereas an interrupted curve can also indicate that the patient is suffering from abdominal straining, a tear in the abdominal muscles causing the patient to have pain while urinating. Oscillations seen in the uroflowmetry curves can also be due to artifacts in the measurements. The long flow uroflowmetry curves are characterized by long voiding times, little steepness and low maximum voiding speed. They can be caused by decreased power in the bladder muscles or a constant increased pressure in the urethra. Another reason can be a bladder outlet obstruction which lowers the voiding speed [1],[5].

II. Constructing the classifiers

Data collection and pre-processing

In this research, a database of 1156 uroflow measurements was collected, the dataset was acquired by Dr. Rosier of the urology department of the UMCU. All programming was done in MATLAB R2015b. As already stated, the measurements considered are obtained from women above the age of 18. The measurements are executed with a weight transduced uroflowmeter which measures the voiding speed of urine (ml/s) vs. the time (seconds) with a frequency of 8Hz. Sometimes there are spike artifacts in the data. To account for this, the measurements are filtered with a 2-second moving average filter as advised by the International Continence Society (ICS) [5]. Furthermore, the starting point and endpoint of micturation are determined for each curve and all data outside the start and endpoint is removed. As in van der Kamp's research, the starting point of micturation is defined as the last time the voiding speed is zero before it reaches 20% of its maximal value. The endpoint of micturation is defined as the first point where the voiding speed is zero after the last time 20% of the maximal value is reached.

Besides uroflow measurements, the classification of a subset of the measurements by staff of the UMCU is used in this research. These classifications were obtained from the work of van der Kamp and Boele. The former sent two subsets of 400 and 365 processed measurements to a professor in urology, a urologist and a doctor in functional urology, who classified the curves. Furthermore, the urologists gave a Visual Analogue Scale (VAS) score between 0 and 10 denoting the certainty about their classification. The difference between the two sent subsets was the scaling of the axes, in the first dataset the axes were square and in the second they were scaled according to ICS standards. Boele sent two subsets of 20 and 30 processed measurements to 2 physicians in training and 3 urologists who were asked to give a VAS score denoting the likelihood of membership for all four classes. The measurements in the first subset were given a VAS score from 1 to 5, those of the second subset were given a VAS score from 1 to 3. The scores obtained by van der Kamp only had one class assignment and VAS score, the scores obtained in by Boele were VAS scores from 1 to 3 or 1 to 5 for each class. This difference is accounted for by removing all VAS scores given by the hospital staff approached by Boele, except for the classes with maximal scores. For physician Inter- and intra-observer agreement, a subset of the curves was scored twice in both of the former researches. The double scores of these measurements were also taken into account, resulting in 708 unique scored uroflow measurements. In this database, 94.1% of the measurements came from van der Kamp and 5.9% from Boele.

After all data was collected, the VAS scores and classifications of the curves were used to form likelihood assignments for class membership. These likelihood assignments are 1×4 vectors in which the four elements from left to right correspond to classes A, B, C, and D respectively. Each element denotes the likelihood that the measurement belongs to the respective class, the sum of all elements therefore is always

one. These likelihood assignments were made in the following way: every curve has a set of class assignments with VAS scores denoting the certainty of the people who classified the curves. For every scored curve, for every class assignment, the VAS score of the classification was added to the position in the vector corresponding to the class. After all scores were added, the vector was normalized. This resulted in likelihood assignments of instance membership for each of the four classes in which the certainty of the hospital staff is also taken into account.

As an example, let's say that there are three staffmembers assigning classes and giving VAS scores to a certain measurement. Staff member 1 assigned class A to the measurement with VAS score 6, staffmember 2 assigned class D with VAS score 4 and staffmember 3 assigned class D with VAS score 6. Now the unnormalized likelihood assignment vector would be $[6, 0, 0, 10]$. Dividing by 16 (the sum of all scores) gives $[0.375, 0, 0, 0.625]$, which is the likelihood assignment for this curve. It denotes that there is a 37.5% likelihood of membership in class A and 62.5% likelihood of class membership in class D. If a class had to be assigned to this measurement, it should be class D because the likelihood of membership in this class is the highest.

After computing the likelihood assignments for the scored curves, the set of curves with a likelihood of 100% of belonging to a certain class were stored as well as their assigned classes and sources (van der Kamp or Boele). The resulting set, which is called the golden curves set, contained 428 curves. Of these curves, 71% belonged to class A, 12% to class B, 14% to class C and 3% to class D. Distribution of class frequencies like this in uroflowmetry are also seen in literature [4], [1], [2], [8], [9].

Questionnaire

In the work of Brand, a set of 48 classified uroflow measurements was considered [2]. These flow measurements were obtained from young women who were mostly healthy. For the automatic classification of these measurements, a classification questionnaire was implemented. The questions asked about a measurement in this questionnaire were based on literature regarding flow patterns and insights of experts from the UMCU. The answers to the questions resulted in a class assignment. This questionnaire was shown to have high accuracy, 98% of the 48 classified curves considered in this research were correctly classified.

In the research of van der Kamp, this questionnaire was further investigated. A bigger set of 138 classified curves was employed instead of 48. A performance measure used in his research was the area under the Receiving Operating Characteristic curve (ROC curve). This measure gives more insight in the distribution of false and true positives than accuracy alone, more can be read about this measure in the section about performance measures. For the questionnaire proposed by Brand, the ROC curve area was 0.85. One of the goals of the research of van der Kamp was altering the questionnaire in such a way that the resulting ROC curve area was maximized. The order of the questions was changed and errors in the used MATLAB code were corrected. Furthermore, the questions asked in the questionnaire depended on certain parameters and the parameters that gave the maximal ROC curve area were searched with an interior point algorithm. The modifications of the original questionnaire resulted in a final ROC curve area of 0.99 and an accuracy of 94% on the 138 classified curves [1].

As the first step in the current research, the optimization method proposed by M. van der Kamp is replicated and improved. The code used in the investigations by van der Kamp was obtained and as much code as possible was left unchanged. There are five differences in the way the questionnaire optimization is done when compared to the former research.

The first difference is the size of the dataset on which the parameters are optimized. As already stated, there are now 428 classified golden curves on which the questionnaire parameters can be optimized instead of 138. Using more data is convenient, because using a larger sample (dataset) size often gives rise to more reliable statistical predictions.

The second difference is the function that is maximized in the parameter optimization step: in this research the average value of Cohen's kappa for all four classes is used as the function to maximize. Cohen's kappa value for one class is given as:

$$\kappa = \frac{p_0 - p_e}{1 - p_e} \tag{1}$$

where

$$p_e = \frac{N_T^1 \cdot N_c^1 + N_T^0 \cdot N_c^0}{n^2},\ p_0 = \frac{TP + TN}{n}.$$

In the above equation, n stands for the number of classified measurements. N_T^0 stands for the number of golden curves assigned by staff as not belonging to the class. N_T^1 stands for the number of classified golden curves assigned by staff as belonging to the class. N_c^0 is the number of measurements predicted by the classifier as not belonging to the class. N_c^1 is the number of measurements predicted by the classifier as belonging to the class. TP is the number of measurements correctly predicted to be in that class and TN is the number of measurements correctly predicted as not being in that class by the classifier.
Cohen's kappa is a measure for the agreement of the classifier with the classes assigned by urologists weighted with the probability that this agreement is by random chance. It takes the true positives (TP), true negatives (TN) and probability of random agreement into account and is therefore often considered as a robust measure for classifier performance [10]. In this research, the average over the Cohen's kappa values was optimized instead of the ROC curves because this method corresponds more with the questionnaire optimization step proposed by van der Kamp.

The third difference is the minimization algorithm used in the parameter optimization step. In the research by M. van der Kamp an interior point method was used to find parameter values. In this research, a heuristic method is used, namely the pattern search heuristic. Function evaluations in the case of the questionnaire tend to take a lot of time, and Cohen's kappa, as with accuracy, is a discontinuous function of the questionnaire parameters. Pattern search was chosen because it can be used to find the minimum of a discontinuous function and it uses relatively few function evaluations for finding an optimum as compared to heuristic methods such as particle swarm or simulated annealing [11].

The idea behind pattern search is to start with an initial optimal point (or center) and step size 1. For each parameter, the step size is added and subtracted to generate $2n$ new points (where n is the number of variables) and the function to minimize is evaluated at these new points. If the minimum over both the new points and center is attained at a new point, this point becomes the new center and the step size is doubled. When this is not the case, the old center remains the center, the step size is halved and the above process is repeated. The heuristic stops and returns the center as the optimal point when the method reaches a stopping condition specified by the user. In this research, the stopping criterion was a minimal step size of 10^{-3}. This value was chosen because it was seen that convergence has almost certainly occurred when a mesh size smaller than 10^{-3} is attained (remember that the mesh size doubles when a new optimal point is found, halves when no new point is found and starts at value 1).

The fourth difference between the current and earlier optimization method is the constraints which must hold for the optimal point. In the earlier research, an interval was specified in which the optimal point must lie [1]. In the current research, the initial point is chosen to lie inside this interval, but in the course of calculations the movement of the center point is unrestricted. Because it is a heuristic, pattern search does not give the same optimal point every time. Because of this, the procedure was repeated 50 times and the parameter settings resulting in the lowest average kappa value were chosen.

The fifth and final difference is the implementation of likelihood prediction in the questionnaire algorithm. Every question in the questionnaire can either give an indication that a measurement belongs to one of two classes or just one class (see example below). The likelihood predictions made by the questionnaire are now constructed as follows: stepwise, the answer to

every question in the questionnaire is computed. Now for the class indication that the answer to the question gives, if there is any, the element corresponding to this class in the likelihood vector is raised by one. After all questions have gone through this process, the likelihood vector is normalized, resulting in the ultimate likelihood prediction.

As an example, consider a questionnaire consisting of the questions:

- *Is the maximal flow higher than 10 ml/s?*
- *Does the measurement have an interruption?*
- *Are there staccato peaks in the measurement?*

In the first question, *yes* indicates Normal flow and *no* indicates Long flow. In the second question *yes* indicates Interrupted flow and *no* does not give a class indication. No interruptions could still mean that the measurement belongs to either one of the other three classes. In the third question, *yes* denotes Staccato flow and *no* also does not give a class indication. Now if the answers to the questionnaire were [*yes*, *no*, *yes*], in the unnormalized likelihood vector one point would be given to Normal flow and one point for Staccato flow. The normalized likelihood predictions would then be [0.5, 0.5, 0, 0].

Machine learning approach

As a second step, it is investigated whether good classifiers can be found using machine learning methods. These classifiers could then possibly be used instead of or in combination with the questionnaire. When using machine learning methods, often first some important properties (called features or variables) of the datapoints are extracted from the dataset. These features are often a choice made by the user. After extracting the features, the machine learning method tries to make an as good as possible mapping (or model) from the features to certain targets using an optimization method. This is often called training the classifier and the used set of feature values is often called the training set or the set of training instances. Constructing these mappings is called supervised machine learning and the targets of the machine learning method can be either classes or numbers. Each machine learning algorithm relies on a model template which can be used to map the features to the targets, and this model template directly defines the minimization function and the manner in which this function can be optimized. When optimizing the questionnaire, several aspects of the questionnaire were already fixed (e.g, the number and order of questions asked). In machine learning methods, only the type of model and the set of features used to describe the data are chosen by the user. Because of this, fewer aspects of the model are predetermined and fewer choices have to be made by the user.

If this research is concluded and a classifier is found, this classifier could be used to diagnose new patients. These patients present not yet seen uroflowmetry measurements to the classifier. If it is assumed that future uroflowmetry measurements have similar shapes/feature values as the ones obtained from dr. Rosier, the training set performance of the classifier would roughly be the same as the performance on these new measurements. If this is not the case however, it could be that a classifier that has high training set performance does not perform well on not yet seen instances. The reason for this could be that the built classifier is too specific for the training set data, sometimes also called overfitting. Therefore, it would be useful to investigate classifier performance on not yet seen data. An estimate for the performance on new instances is the 10-fold cross validation performance of a classifier, in the next section this measure will be explained. In the field of machine learning, the classifier that is ultimately chosen is often the classifier that shows the best cross validation performance [12]. The reason for this is that it is more useful to end up with a classifier that is shown to perform well on not yet seen instances as compared to a classifier that is only shown to perform well on the already seen instances. After all, the idea is to use the classifier for diagnosing new patients. For machine learning algorithms, parameters can be

tuned such that the model becomes more general and the performance on new instances becomes higher. What is mostly done in the field of machine learning, is plotting the cross validation performance vs. the parameter value and the parameter value that results roughly in the highest cross validation performance is chosen [12]. For the generation of the questionnaire, it takes a very long time to calculate just one cross validation measure (5 days on the current PC). Therefore, finding good parameter values for the patternsearch heurisctic by constructing the aforementioned plots would take too long. This makes that the questionnaire classifier cannot be further tuned to increase cross validation performance.

As already stated, before using machine learning algorithms, a set of features is determined. The names and descriptions of the features used in this research can be seen in table 1. These features are based on the parameters of the questionnaire used by van der Kamp and Brand [1], [2].

A first idea was to build an as good as possible mapping from the features of the golden curves to their classifications. This can easily be done using the MATLAB classification learner app. This app takes as input the feature values for each measurement and each measurements class. Then, there is a list of machine learning algorithms out of which the user can choose. After a method is chosen, the app automatically generates the corresponding classifier. Furthermore, the user can tune the settings of the machine learning algorithm to get higher cross validation performance. After the parameters are tuned, the model can be exported to the MATLAB workspace and used for classification. The constructed classifiers often can also automatically predict likelihoods for class membership, so that these predicted likelihoods can also be compared with the likelihood assignments given by urologists. Most of the time, there is not one specific machine learning algorithm that must be used for a given problem, a set of multiple machine learning algorithms has to be used and evaluated. As a guideline to determine which machine learning algorithms to use in this research, two often used machine learning cheat sheets are followed [13], [14]. After investigation of these algorithm cheat sheets, the classifiers K-nearest neighbor, support vector machine (SVM) and random forest were chosen for this problem. For more information on K-nearest neighbor, support vector machine and random forest classifiers, see [15], [16] and [17] respectively.

A second idea was to consider all curves with likelihood assignments made by hospital staff and to make an as good as possible mapping from the features of these curves to their respective likelihoods. There are a number of reasons why it is interesting to do this. First, it was seen that a large percentage of the golden curves were classified as A and small percentages for classes B, C and D. It is therefore possible that a classifier trained on the golden curves alone is not presented with enough examples belonging to the classes B, C and D to give a good overall classification performance. However, when the likelihoods assignments are observed, it is seen that 69.2%, 35.7%, 35.7% and 19.3% of the likelihoods for the classes A, B, C and D respectively are larger than 0. The number of curves with likelihoods assignments is also almost twice the size of the set of golden curves. It is therefore expected that when the classifiers are trained on the likelihoods, the classifiers see more examples that correspond to the classes B, C and D to a certain extent and therefore get a better grip on which measurements belong to which class. The second reason to make such a mapping is that it is expected that a classifier trained on the likelihoods will have higher predictive power for the likelihoods in comparison to one trained on the golden curves. Because the golden curves are directly determined from the likelihoods, it is also expected that classifiers trained on the likelihoods will have high classification performance.

The problem of mapping features to likelihoods is a multivariate regression problem with multiple responses, there is no app in MATLAB that handles these problems. Therefore, scripts are built in MATLAB to construct the regression classifiers. For the built classifiers, the class assignment given to a measurement will be the class with the highest predicted likelihood. Af-

ter another inspection of the aforementioned algorithm cheat sheets, support vector regression, regression forest and ridge regression were chosen as classifiers to build for this problem. For more information on support vector regression, regression forest and ridge regression, see [18], [17] and [19], respectively.

Table 1: Names and descriptions of the features used in the machine learning algorithm to generate classifiers for the uroflowmetry measurements.

Name of the feature	feature variable name in MATLAB	description of the feature
Deceleration time divided by acceleration time.	DTAT	This is the time after the maximal flow of the uroflowmetry measurement is reached divided by the time before this happens.
deceleration time divided by maximal flow.	DS	This is the time after the maximal flow of the uroflowmetry measurement is reached divided by this maximal flow.
The number of interruptions in the uroflowmetry measurement.	interruptions	This is the number of times the urine flow drops to a value representing 20% of the maximal flow or less.
The number of stacattopeaks in the uroflowmetry measurement.	number_of_staccatopeaks	This is the number of times the urine flow has a peak with peak prominence bigger than or equal to 20% of the maximal flow.
Maximal measured flow of the uroflowmetry measurement.	QMAX	This is the maximal measured flow of the uroflowmetry measurement.
Average flow divided by the total volume.	QV	This is the average flow divided by the total urine volume measured by the uroflowmeter.
Voiding time	T	The total time between the start-and-endpoints of voiding.
Voiding time divided by maximal flow.	TQ	The total time beween the start-and-endpoints of voiding divided by the maximal urine flow recorded.

Performance measures

It was already stated that accuracy alone cannot fully give insight into the performance of a classifier. Thus, for evaluating the classification methods constructed in this research, a number of performance measures is used. In this research, there is a set of classified curves, called the golden curves, and a set of curves with likelihoods of membership in the four classes given by hospital staff. The agreement of the built classifiers with both these likelihoods and these classifications can be evaluated. The measures taken are *accuracy, Cohen's kappa value for all four classes, ROC curve area for all four classes*, the *area under the MAE curves* and the *10-fold cross validation accuracy*. For all these measures, 1 is the best value and 0 the worst.

- Accuracy is equal to the average sum of the true and false positive rate ($\frac{TP+TN}{n}$ from equation 1) over all four classes [20]. It indicates how well the classifier performs overall on the golden curve set.
- Cohen's kappa value can also be calculated

for the golden curves, the measure is explained in the discussion following formula 1 and more can be read about this measure in [10]. These kappa values are calculated per class and therefore indicate how well the classifier performs for each class.

- ROC curves stem from the idea that, as with the updated questionnaire, some classifiers can give likelihood assignments of instance membership in certain classes [20]. One could now decide that when a likelihood given to a class is greater than some predetermined threshold, the measurement is classified as that class. For each value of this predetermined threshold, the classifier should make a number of good and bad classifications, resulting in a true positive rate ($\frac{N_c^1 - TP}{n}$ from equation 1) and a false positive rate ($\frac{N_c^1 - TP}{n}$ from equation 1) for each class. An ROC curve is now a parametric plot of the true positive rate vs. the false positive rate for certain values of the threshold (see figure 2 for an example). The ROC curve area corresponds to the area under this curve integrated from 0 to 1. It indicates how well the classes assigned to the golden curves can be predicted with the likelihoods given for the four classes by the classifier.

- Another measure is the area under the MAE curves or the MAE area for short. These curves are calculated in the following way: first, 1000 thresholds are defined, these are taken equidistant on the interval [0,1]. Then, for each measurement the classifier predicts four likelihoods for class membership. Now for every class, measurement, and threshold it is checked whether the mean absolute error (MAE) between the predicted and assigned likelihood is smaller than or equal to the given threshold. Per class and threshold, the fraction of uroflowmetry measurements satisfying this condition is stored. Now as a last step, the minimum, mean and maximum over these four fractions are calculated per threshold. These three variables are plotted vs. the thresholds in an MAE curves plot. The MAE area now corresponds to the area under the curve representing the minimum over the fractions. For an example of an MAE curves plot, see figure 3. An MAE curves plot describes the distribution of MAE's between predicted and assigned likelihoods for a classifier. The area under the MAE curves is a measure indicating how well the classifier can predict the likelihoods assigned by hospital staff.

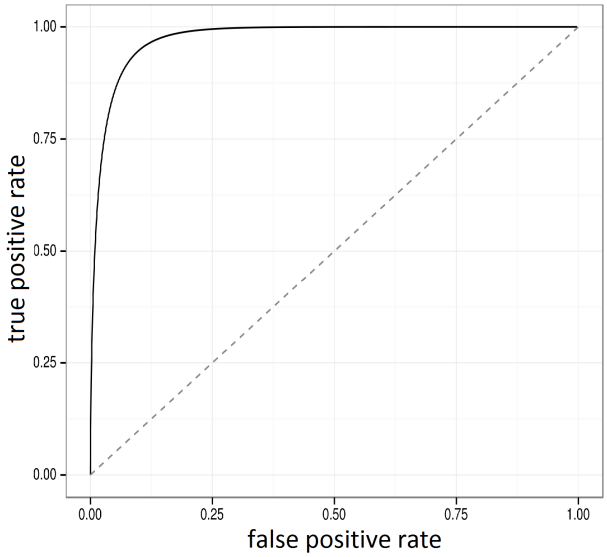

Figure 2: Example of an ROC curve, the true positive rate is plotted vs. the false positive rate for certain values of the classification threshold. Source: [21].

- Another performance measure that is considered is the 10-fold cross validation accuracy. This is an estimate for the overall predictive power of the classifier for not yet seen instances. The dataset is divided into ten subsets, for each subset, the classifier is trained on the complement and the accuracy of the resulting classifier on the former subset is calculated [12]. These accuracies are then averaged over all 10 subsets to give the 10-fold cross validation accuracy.

III. Agreement of hospital staff on the likelihood assignments

In this research, the ability of the constructed classifiers to predict the assigned likelihoods is evaluated. A first thing to investigate is if this evaluation is justifiable. It could be that the staffmembers themselves are incapable of predicting the likelihoods. In other words, it could be the case that there is so much variation in the individual likelihood assignments made by the urologists that coming close to the weighted average of these assignments is not useful. To investigate the agreement of the staffmembers on the likelihood assignments, for each staffmember, the corresponding class assignment and VAS score are left out of the classification database. Now, the MAE curves denoting the agreement between the assignments from this staffmember and the weighted (with VAS scores) average of the scores by the other urologists are constructed. Note however that the likelihood predictions given by single staffmembers are always classifications of 100% certainty about one class. This is because the staffmembers only had to give one classification. In previous researches, some curves were scored twice by physicians (to compute observer inter-and intra-agreement). One could also treat these scores as classifications by different observers and evaluate them. However, this set was relatively small and therefore wasn't taken into account.

Not much variation in the performance of individual urologists was seen. The average standard deviations per threshold for the curve corresponding to the minimum and maximum over the fractions were 0.0260 and 0.0257 respectively for the staffmembers approached by van der Kamp and 0.0265, 0.0448 respectively by Boele. Because the MAE curves for all urologists in both researches were this similar and to avoid presenting a lot of the same results, the MAE curves over all staffmembers in the two researches were averaged. This resulted in two averaged MAE curves and two averaged MAE area's. In figures 3 and 4, the averaged MAE curves are plotted for the hospital staff that evaluated the curves in the research of van der Kamp and Boele respectively.

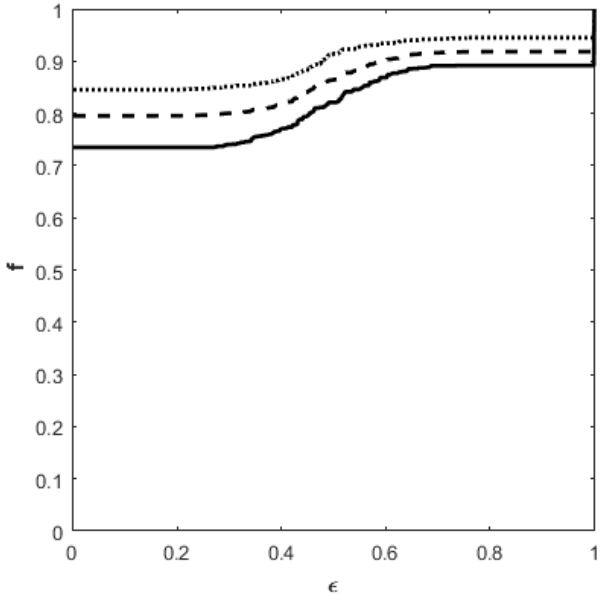

Figure 3: The MAE curves for the average observer approached in the research by van der Kamp. The minimum (solid), mean (dashed) and maximum (dotted) over the fractions (f) of curves over all classes with MAE between true and predicted likelihoods smaller than the threshold are plotted vs. the thresholds (ϵ).

.

The average MAE areas for the staffmembers in the research of van der Kamp and Boele were 0.815 and 0.685 respectively. The observers approached in the research by van der Kamp predicted, on average, the true likelihood for roughly 72% of the measurements (the curve corresponding to the minimum over the fractions starts at 0.72). The curves corresponding to the minimal and maximal fractions lie close to each other, corresponding to small variation in the performance per class. The MAE curves are mostly flat, indicating low deviation of the MAE errors per measurement.

The observers approached in the research by Boele predicted, on average, the true likelihood for roughly 42% of the measurements. In the beginning, the curves corresponding to the minimal and maximal fractions lie far apart from

each other, but as the threshold increases this distance becomes smaller. This corresponds to large variation in the performance per class when small errors are considered and smaller deviation when larger errors are considered.

In both researches, there was (on average) a fraction of measurements with a prediction MAE of 1 for at least one class. This corresponds to the sudden jump to a value of 1 that both the MAE minimum curves have when a threshold of 1 is considered. This is due to the fact that the urologists always gave likelihood predictions of 100% certainty for one class. If some golden curves are then misclassified, the MAE becomes one for both the assigned and predicted class. For both researches, the fraction of measurement with an MAE of 1 lies on average around a value of 0.9 as this is the value attained before the MAE minimum curves jump to a value of 1.

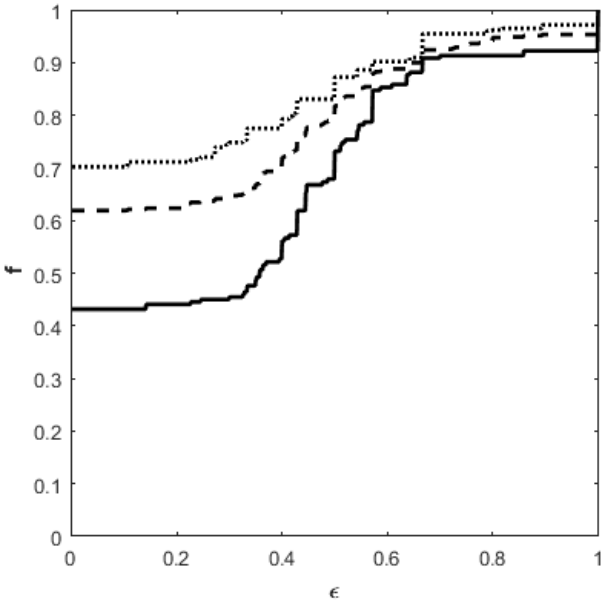

Figure 4: The MAE curves for the average observer approached in the research by Boele. The minimum (solid), mean (dashed) and maximum (dotted) over the fractions (f) of curves over all classes with MAE between true and predicted likelihoods smaller than the threshold are plotted vs. the thresholds (ϵ).

In this section, it was investigated how much the approached hospital staff agreed with each other on the membership likelihoods that they gave to the measurements. It was shown that the staff approached by van der Kamp showed high agreement and the staff approached by Boele showed moderate agreement. As 94.1% of the measurements with likelihood assignments are obtained from the research of van der Kamp, the conclusion is that it is indeed useful to investigate classifier performance on likelihood prediction.

IV. Classifier evaluations

Performance of the classifiers

In table 2, the values of the performance measures on are shown for all constructed classifiers as well as the parameters given to the machine learning model builders. For more information about the construction and ideas behind the machine learning classifiers, see appendix B.

The support vector machine classifier does not give likelihoods for class assignment, therefore the MAE area and ROC curve areas for this classifier remain undetermined. The classifiers that have the highest training set performance are random forest and K-nearest neighbors with an accuracy of 100% and Cohen's kappa values and ROC areas of 1. The regression forest classifier also has training set performance measures around these values. The worst performing classifiers when accuracy and Cohen's kappa are considered are the ridge regression and SVR classifiers, where only the questionnaire classifier has a lower kappa value for class D. The worst performing classifier when the ROC and MAE areas are considered is the questionnaire classifier. This indicates that the correspondence between the predicted likelihoods and both class and likelihood assignments is low for this classifier. The classifier that has the highest MAE area is the regression forest classifier with an area of 0.890, closely followed by the random forest classifier with area 0.881. The regression forest classifier is also the classifier with the highest 10-fold cross validation, 96.7%. This means that this model generalizes well and has high predictive power, as the accuracy for new instances is high and

does not differ much from the training accuracy, which is 99.1%. The questionnaire classifier also seems to generalize well, the training and cross validation accuracies are 98.4% and 96.0% respectively. The K-nearest neighbor classifier performs reasonably well on all training set measures but has the lowest 10-fold cross validation accuracy, which implicates overfitting. Two classifiers that perform well on all measures are the regression forest and random forest models. The support vector machine model also performs well on all measures for classification, but most of the time the performance measures for this classifier are lower than these values for random forest and regression forest.

For some classifiers, it is interesting to further investigate the cross validation performance to fully determine which classifiers perform best on new instances. The classifiers which are interesting to further investigate are the classifiers that performed well on classification, likelihood prediction or both. Furthermore, these classifiers should have a small difference between the 10-fold cross validation accuracy and the training set accuracy because this indicates that the model generalizes well. As often seen in machine learning research, the ultimately chosen classifier will be the one with the highest cross validation performance measures [12]. The reason behind this is that the ultimate classifier is going to be used to diagnose new patients and therefore should have good performance on not yet seen instances.

The classifiers chosen for further investigations were:

- The questionnaire classifier for classification. The 10-fold cross validation Cohen's kappa values are [0.971, 0.930, 0.941, 0.576].
- The SVM classifier on classification. The 10-fold cross validation Cohen's kappa values are [0.953, 0.741, 0.761, 0.743].
- The KNN classifier on both classification and prediction. The 10-fold cross validation MAE area is 0.6177, the cross validation Cohen's kappa values are [0.767, 0.407, 0.543, 0.524] and the cross validation ROC-curve area's are [0.866, 0.701, 0.758, 0.671].
- The regression forest classifier on both classification and likelihood prediction. The 10-fold cross validation MAE area is 0.826, the cross validation Cohen's kappa values are [0.942, 0.756, 0.774, 0.734] and the cross validation ROC-curve area's are [0.988, 0.972, 0.951, 0.738].
- The random forest model on both classification and likelihood prediction. The 10 fold cross validation MAE area is 0.769, the cross validation Cohen's kappa values are [0.943, 0.746, 0.745, 0.686] and the cross validation ROC-curve area's are [0.986, 0.974, 0.940, 0.827].

To estimate the 10-fold cross validation kappa and ROC area values for all classifiers other than the questionnaire, the 10 fold cross validations of these measures is computed 50 times and averaged. The ridge and support vector regression classifiers weren't chosen for further investigation. This is because of the low classification and likelihood predictive performance of these classifiers showed on the training set.

It is seen that both the random forest and regression forest classifiers have good overall cross validation performance. All cross validation kappa values indicate substantial agreement between the predicted and assigned classes [1]. This is also the case for the support vector machine classifier. The cross validation MAE area for the regression forest classifier is 0.826, which is still higher than some MAE area's attained for some classifiers on the training set. The classifier with the highest cross validation Cohen's kappa values (except for class D) is the questionnaire classifier. The value of Cohen's kappa for class D however show only moderate agreement. Furthermore, the standard deviation over 10 runs for Cohen's kappa for class D was 0.393. In comparison to the standard deviations of other Cohen's kappa values (both for the questionnaire and other classifiers) this value is very high. This makes the predictions of this classifier for class D unreliable.

The **regression forest** is the classifier that should be chosen as the ultimate classifier. The random forest, support vector machine and regression forest classifier all have cross validated Cohen's kappa values in the same ranges. When the ROC areas are considered, the random forest wins, as the cross validated area for class D is almost 0.1 higher than that of the regression forest classifier. When cross validated accuracy and MAE area are considered however, the regression forest is the best. Because the regression forest only has a significantly lower cross validated value than the random forest for the ROC area of class D, this classifier is the ultimately chosen one.

Instead of using just one classifier, one could also use two classifiers, one for likelihood prediction and one for class prediction. The only classifier with significantly higher Cohen's kappa values than the regression forest classifier is the questionnaire. However, it was shown that the classification power for class D is very weak for this classifier and therefore it should better not be chosen for classification. Furthermore, a downside to using two classifiers could be that the two classifiers sometimes do not agree with each other on class assignments. Therefore, the predicted likelihoods and class assignments wouldn't correspond, which might cause confusion in hospital staff.

The questionnaire classifier

In the current research, the questionnaire optimization step proposed by van der Kamp was replicated and improved. In this subsection, this ultimately found questionnaire is investigated.

The maximal average kappa value found on the training set was 0.882, the corresponding parameter values are shown in table 3. The found questionnaire performs better on classification as compared to likelihood prediction. The accuracy of this classifier on the training set is 98.4% and the Cohen's kappa values are [0.983, 0.912, 0.900, 0.732] corresponding to nearly perfect agreement for classes A, B, and C and substantial agreement for class D. This classifier has a lower area under its MAE curves (0.604) and ROC curves ([0.982, 0.951, 0.770, 0.724]) as compared to all other constructed classifiers. The 10-fold cross validation for the questionnaire classifier is 96.0%. The cross validated Cohen's kappa values were [0.966, 0.958, 0.931, 0.586] indicating nearly perfect agreement for classes A, B and C but only moderate agreement for class D.

The MAE curves for the questionnaire are plotted in figure 5. The distance between the curves corresponding to the minimal and maximal fraction of curves on average is large, denoting variation in performance per class for this classifier. Also, there are no curves for which the classifier gives the true likelihood assignment as the curve corresponding to the minimum of the fractions takes on the value 0 when the threshold considered is 0.

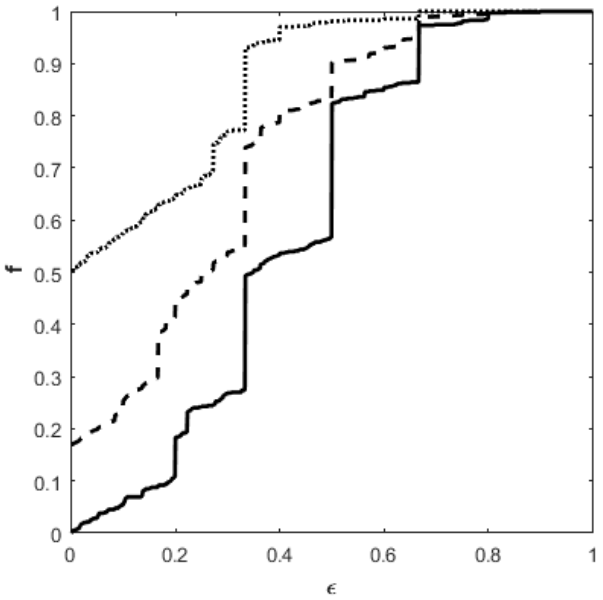

Figure 5: MAE plots for the optimized questionnaire. The minimum (solid), mean (dashed) and maximum (dotted) over the fractions (f) of curves over all classes with MAE between true and predicted likelihoods smaller than the threshold are plotted vs. the thresholds (ϵ).

All MAE values are lower than a value around 0.8, as the MAE curve corresponding to the min-

Table 2: Performance measure values attained by the constructed classifiers, as well as the parameters used for model building.

Classifier name	Accuracy	MAE curve area	Cohen's kappa for class A	Cohen's kappa for class B	Cohen's kappa for class C	Cohen's kappa for class D
Questionnaire	98.4%	0.604	0.983	0.912	0.900	0.732
K-nearest neighbor	100%	0.849	1.00	1.00	1.00	1.00
Support vector machine	98.6%	unknown	0.954	0.904	0.896	0.955
Random forests	100%	0.881	1.00	1.00	1.00	1.00
Support vector regression	93.9%	0.788	0.726	0.607	0.693	0.795
Ridge regression	94.0%	0.7981	0.747	0.613	0.688	0.795
Regression forest	99.1%	0.890	0.965	0.978	0.929	0.955
Classifier	ROC area for class A	ROC area for class B	ROC area for class C	ROC area for class D	10-fold cross validation accuracy	Classification builder parameters
Questionnaire	0.982	0.951	0.770	0.724	96.0%	Pattern search Mesh tolerance: 10^{-3}. 50 repetitions.
K-nearest neighbor	1.00	1.00	1.00	1.00	90.0%	Model: Fine KNN. Number of neighbors :1. City block distance.
Support vector machine	unknown	unknown	unknown	unknown	93.9%	Quadratic kernel. Box constraint level: 1. Automatic kernel scaling. One vs. one method.
Random forest	1.00	1.00	1.00	1.00	93.7%	Bagged trees. Number of trees: 95.
Support vector regression	0.985	0.963	0.961	0.997	93.6%	Polynomial kernel. Epsilon=0.1. Box constraint level: 2.
Ridge regression	0.987	0.960	0.958	0.998	94.6%	Polynomial of order 2. Lambda=1
Regression forest	1.00	1.00	0.998	1.00	96.7%	Regression trees. 250 learners. Polynomial of order 5.

imum over the fractions becomes 1 after a value around 0.8.

The performance of the new questionnaire on the golden curves is higher than for the old questionnaire, an accuracy of 96.7% compared to 87.9%. However, the performance of the questionnaire with the old parameters on the likelihoods is better, a MAE area of 0.697 instead of 0.604. When comparing the values of the parameters for both questionnaires, only the parameters TQ, PIEK, PKcut, Msize, Qmaxcut and QV lie around the same value. The differences in the parameter values are likely due to the fact that the search space for the parameters was chosen to be unbounded for this research.

Table 3: The parameter values for the questionnaire found in this research and the parameter values proposed by van der Kamp, as well as the maximal drops in accuracy when the parameter values are either set to 0 or infinity

Parameter name	New parameter value	Old parameter value	Drop in accuracy when changed to zero or infinity.
DTAT	0.804	2.96	0.230%
DS	2.43	0.608	2.34%
T	41.7	12.0	0.230%
TQ	1.12	1.50	0.230%
PIEK	1.00	1.00	71.3%
PKcut	0.172	0.171	37.9%
Lint	0.0441	2.01	11.2%
VolInt	0.0474	0.0980	11.2%
Nint	-29.0	1.00	11.2%
Msize	0.0956	0.0510	0%
Qmaxcut	6.94	6.11	13.6%
QV	0.0247	0.0300	0.940%

Parameter importance for the questionnaire can be investigated by looking at the biggest change in questionnaire performance when the parameter is either set to 0 or infinity. Most questions namely compare certain features of the uroflowmetry curve with the parameter values. These curve features are always positive. Therefore, if one of these parameter values is set to infinity or zero, the result is a tautology or a contradiction (e.g. $X < \infty$, $X > \infty$, $X > 0$ or $X < 0$). In both cases, this should lead to false classifications lowering the accuracy of the questionnaire. The maximal drop in accuracy for setting the parameter to either 0 or infinity now denotes parameter importance for the questionnaire and in the last column of table 3, these values are shown.

The parameter PIEK seems to be the most important parameter of all with a 71.3% drop in accuracy, and the least important parameter seems to be Msize, with 0% drop. The parameters DTAT, T, TQ, Msize and QV all have drops under 1%, these parameters are therefore likely to be redundant for questionnaire classification.

The regression forest classifier

The regression forest classifier built in this research is shown to have high if not the highest values for all (training and cross validation) performance measures. If one classifier should be chosen for classifying uroflowmetry measurements, it should be this one. In this section, the regression forest classifier is further investigated.

The regression forest classifier consists of a set of automatically built regression trees. These trees are built using a random subset of the features and training instances. Regression trees are binary trees, for every node except the leaf nodes a question is asked about one of the features of a given uroflowmetry measurement. Now there are two branches exiting the node, corresponding to answer *yes* or *no*. From these branches, another node is reached and the process is repeated until a leaf node is reached. When a leaf node is reached, a real number corresponding to this leaf node is the prediction made by the regression tree. For an example of a regression tree, see figure 6.

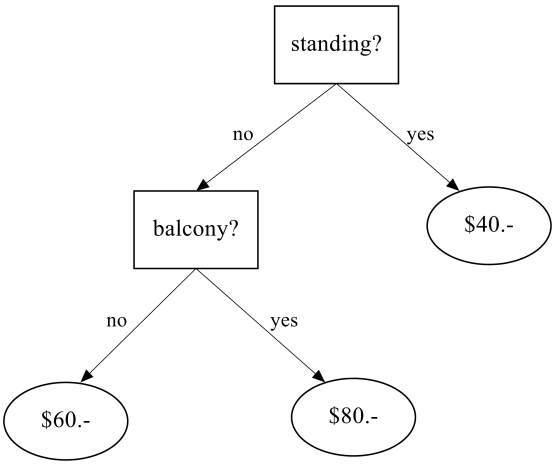

Figure 6: Example of a regression tree made in ClickCharts. The numerical prediction is the price of a concert ticket. Standing is the feature denoting whether the visitor wants to stand up or sit during the concert. Balcony is the feature denoting whether the visitor wants a seat at the balcony.

In a regression forest, each regression tree makes a likelihood prediction and the mean of all these likelihoods is the likelihood prediction made by the regression forest. This procedure generalizes the responses of the regression trees, which often overfit the data. It is often seen that regression forest classifiers compete well with other often used classifiers, while still maintaining high performance on new instances [17]). The TreeBagger() function used for regression forest construction can only construct forests that give one output value. Therefore, four regression forests are constructed to predict the likelihood of curve membership for each of the four classes. When these four different likelihood predictions are obtained, the likelihood predictions that are negative are made zero and the resulting likelihood vector is normalized. Because four regression forests are constructed, whenever the "error" for the regression forest model is discussed, the average of this error over the four regression forest models is meant. For more information on how regression forests work and are constructed, see appendix B.

The regression forest model is the model with the highest training and cross validation MAE area. The MAE curves for the regression forest model are plotted in figure 7. Surprisingly, almost none of the likelihood predictions are spot on, the curve corresponding to the minimum over the fractions starts at a value slightly above zero. However, there is a fast incline seen in the MAE curves of this classifier, denoting high deviation in the MAE's per measurement for this classifier. It is seen that the MAE's for all classes are lower than a value around 0.5, as the MAE curves have converged to a fraction of 1 when the threshold is around 0.5. On average, the curve corresponding to the minimum and maximum over the fractions lie close to each other, denoting similar likelihood predictive performance per class.

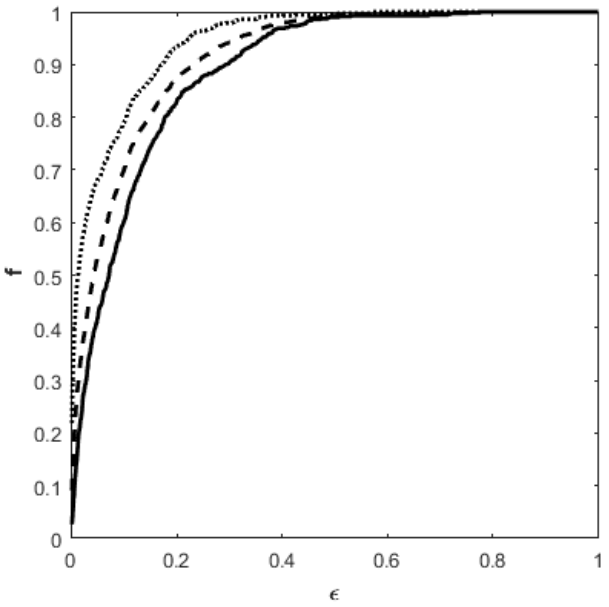

Figure 7: MAE plots for the regression forest model. The minimum (solid), mean (dashed) and maximum (dotted) over the fractions (f) of curves over all classes with MAE between true and predicted likelihoods smaller than the threshold are plotted vs. the thresholds (ϵ).

It can be investigated if more data would improve the cross validation performance of the regression forest model. For investigating this, a curve is plotted for the out of box (OOB) mean squared error of the classifier vs. the training set size. This OOB mean squared error is now explained. As stated earlier, for building a tree in a regression forest, a subset of the total train-

ing set is used. Now for calculating the OOB mean squared error, predictions are done on the instances not included in this subset (out of the box) with the built tree. After this, the mean squared errors between the predicted and assigned likelihoods are calculated. Finally, the calculated mean squared errors are averaged over all trees to give the OOB mean squared error. The OOB mean squared error can be seen as an estimate of the mean squared error of the regression forest classifier on not yet seen instances.

Now, for constructing the aforementioned plot, the total training set is permuted and divided into 50 parts. The first part is initially used for training the regression forest and the OOB mean squared error is calculated and stored. The next training set part is now added and again the model is constructed and the error is calculated and stored. This process is repeated until the total training set is used. After this, the total process beginning with the permutation of the training set is repeated another 9 times and the calculated OOB mean squared errors are averaged over these 10 runs. The permutation of the dataset has the effect that the order of the training instances has no influence on the appearance of the aforementioned plot. The plot of the OOB mean squared error vs. the training set size for the regression forest classifier is shown in figure 8.

It is seen that the error has a lot of variation when small training set sizes are used and becomes more stable as the training set grows. There also seem to be oscillations in the overall trend of the plot. This is curious, as one would expect that the overall trend of the OOB error would only decrease in the training set size. The OOB error also seems to be very small when the smallest training set size is used, this was also not expected. As it is seen that the OOB error still shows some variation when the total training set is used, more training data is probably needed for the OOB error to fully converge. However, this would probably only result in changes of an order of magnitude -3 as for the last 200 instances added the OOB mean squared error stays in the interval [0.115, 0.12]. Because the likelihoods themselves are of order of magnitude -1, it is probably not useful to add more data.

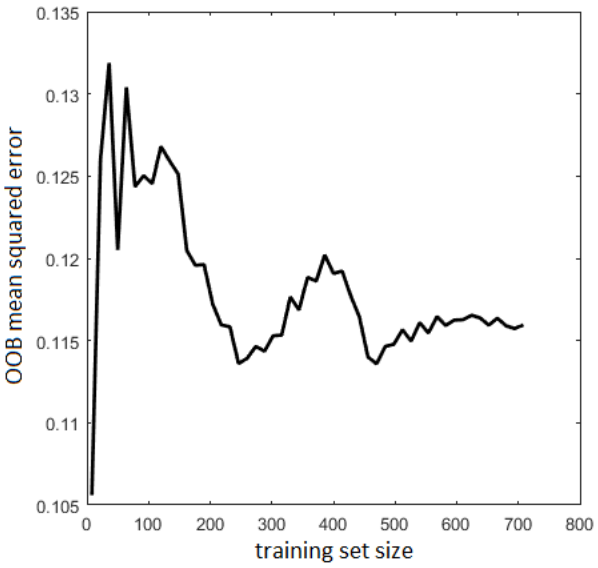

Figure 8: Plot of the OOB mean squared error vs. the training set size for the regression forest classifier.

Another thing to investigate using regression forests is the importance of each feature that is fed to the model. This can be done using the out of box permuted predictor delta error. For any variable, this measure is the increase in OOB mean squared error if the values of that feature are permuted across the observations. A larger error corresponds to a more important feature. In table 4, estimates of the values of the OOB mean squared error, averaged over 10 models, are shown for each feature. *The number of staccato peaks, number of interruptions, voiding time divided by maximal flow* and *deceleration time divided by maximal flow* are the variables that seem to be the most important. The other variables all lie around an error of 0.6. The reason behind this is probably that only the number of interruptions may indicate Interrupted flow behavior and the number of staccato peaks distinguishes Staccato flow behavior, but all the other variables distinguish Long from Normal flow behavior and therefore basically indicate the same thing.

Table 4: Estimates of the out of box permuted predictor delta error for the variables given to the regression forest builder, the errors are averaged over 10 built models.

Parameter name	Out of box permuted predictor delta error
Deceleration time divided by acceleration time	0.736
Deceleration time divided by maximal flow	0.640
Number of interruptions	1.34
Number of staccato peaks	2.00
Maximal flow	0.562
Average flow divided by total volume	0.658
Voiding time	0.606
Voiding time divided by maximal flow	0.769

Something that can also be investigated is the effect of the parameters chosen in the preprocessing of measurements on the regression forest classifier. An averaging filter with a sliding window of 2 seconds was used, the starting-and endpoints of micturation depended on a percentage of 20% of the maximal flow and furthermore the minimal staccato peak prominence was chosen as 20% of the maximal voiding speed. It is now investigated how much the performance of the regression forest classifier depends on these values.

In figure 9, the value of the window size of the averaging filter is plotted vs. the OOB mean squared error for the regression forest classifier. To account for randomness, the average is taken over 10 runs.

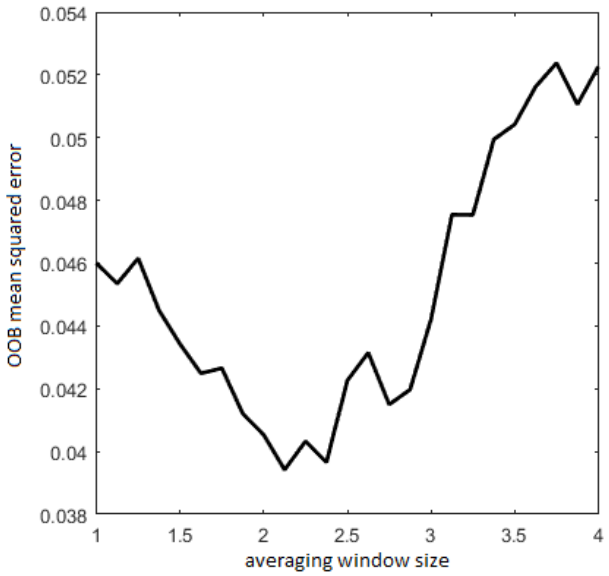

Figure 9: Plot of the OOB mean squared error vs. the windowsize of the averaging filter, averaged over 10 regression forest models.

It is seen that in the range [2, 2.5], the error does not change that much. Outside of this interval, the error gets larger and the regression forest model performs worse. The reason behind this is probably that outside this interval, the averaging filter window gets too large or too small. When the averaging filter window gets too large, the filter also leaves out the low frequency oscillations in the Staccato and Interrupted flow measurements. This has as effect that the regression forest falsely gives a higher probability for Normal flow for these measurements. Also, when the averaging filter window gets too small, the high frequency oscillations stay in the measurement, resulting in more normal flow curves that falsely have a higher predicted likelihood for Staccato or even Interrupted flow behavior.

In figure 10, the values of the voiding threshold percentage, which define the start and endpoints of micturation, are plotted vs. the OOB mean squared error of the regression forest

model.

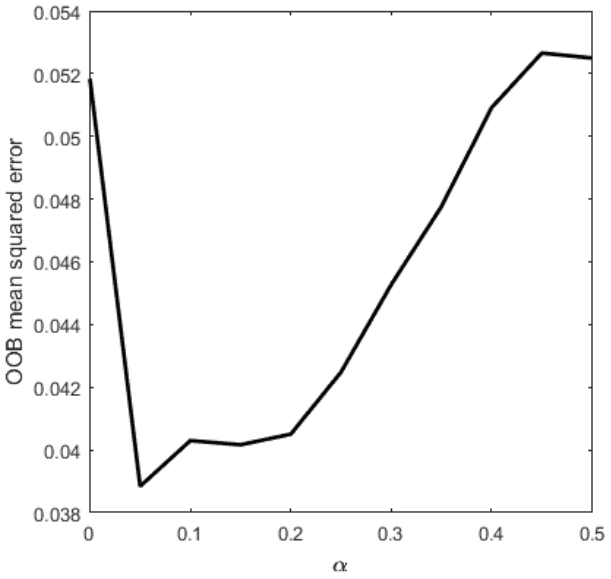

Figure 10: Plot of the OOB mean squared error vs. the percentage defining the voiding threshold (α), averaged over 10 regression forest models.

To account for the randomness of the models, the average is taken over 10 runs. It is seen that in the range [0.05, 0.2], the error does not change that much but outside of this interval the error grows bigger. The reason behind this is that when the voiding threshold becomes too big, some small peaks in the Interrupted curves vanish and some of them might falsely get a higher predicted likelihood to be a Normal flow curve. Also, when the voiding threshold becomes too small, some very small peaks outside of what one would consider as the real voiding period are not removed from the measurement, resulting in Normal curves falsely getting a higher likelihood prediction to be an Interrupted curve.

In figure 11, the percentage of maximal voiding speed defining the minimal staccato peak prominence is plotted vs. the OOB mean squared error. To account for the randomness of the models, the average is taken over 10 runs. It is seen that in the range [0.1, 0.25], the error does not change that much but outside of this interval the error grows bigger. This is likely due to too much or too little staccato peak detection, resulting in measurements getting a too high or low likelihood prediction to be a Staccato curve respectively.

It is observed from these last three plots that the parameters window size, voiding threshold percentage and staccato peak percentage are chosen so that the OOB mean squared error is reasonably close to a local minimum value (which is quite possibly a global minimum value). However, it is also seen that the size of the ranges that the errors take on in these plots all lie around a value of 0.02. The likelihoods themselves are in the order of magnitude -1, so an increase in mean squared error of 0.02 is not really that bad. It is therefore concluded that the built classifier is dependent on the chosen pre-processing parameters, but if these parameters were to change, the obtained values for the performance measures wouldn't change drastically. This makes the regression forest a robust model.

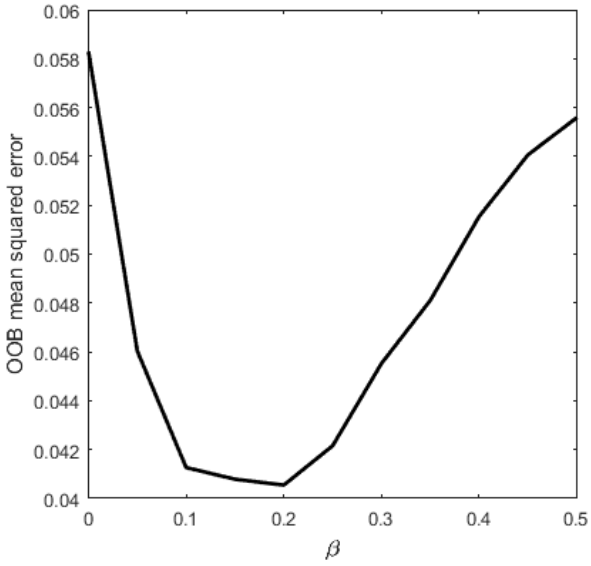

Figure 11: Plot of the OOB prediction error vs. the percentage of maximal voiding speed denoting minimal staccato peak prominence (β), averaged over 10 regression forest models.

V. Classification Display

In the research of van der Kamp, besides a questionnaire, a classification display was constructed in MATLAB. In this research, this display was altered to also show the likelihood pre-

dictions given by the classifier in the form of a pie chart. Furthermore, the colors shown in the display were changed, the axes of the plot of the measurement were changed and the curve characteristics (such as maximal voiding speed) are rounded according to the ICS standards. The result is a function that takes as input the classifier and a raw (unprocessed) uroflowmetry measurement and outputs the display. Finally, in red or green, advice is given on whether the patient should be sent to a physician. An example of the shown display for a certain uroflowmetry measurement is given in appendix A.

VI. Recommendations for further research

In this research, a number of steps are made in constructing an automatic classification system for uroflowmetry measurements. There are still a few more interesting subjects to investigate, which are left for further research.

In this research, machine learning methods are used to find classifiers for uroflowmetry measurement classification. These methods leave fewer choices to the user when compared to the questionnaire optimization because the methods find the best form of the chosen classifier for the data with built in optimization methods. Two things that were left to the user however were the choice of features extracted from the data and the pre-processing step that each measurement had to undergo before classification. Another (less ad hoc) strategy would be to directly obtain features from the data. One example would be to scale all uroflowmetry measurements to the square $[0, 1] \times [0, 1]$ and divide the voiding speed axis into n regions. The number of times the voiding speed measurements lie in each of these regions could then be used as n features for classification. Another idea is to use deep learning methods. Deep learning methods try to learn the optimal representations of data with the use of neural or other networks. This would remove the choice of feature selection in this research because features would then be extracted automatically [22].

Also, it would be interesting to investigate if the automatic classification method proposed in this research could be applied on a larger scale (i.e. in more hospitals than the UMCU only). In order to investigate this, uroflowmetry measurements and classifications have to be obtained from other hospitals.

VII. Conclusion

In this research, it was investigated if the classification of uroflowmetry measurements obtained from women by staff from the University Medical Center in Utrecht could be automated. The questionnaire optimization step proposed by van der Kamp was replicated to give a questionnaire for automatic classification. Furthermore, machine learning algorithms were used to generate a set of other classification algorithms. All constructed classifiers were evaluated on a set of performance measures. The classifier that had the highest overall performance was the regression forest classifier with an estimated accuracy of 96.7% on not yet seen uroflowmetry measurements. Further research could focus on uroflowmetry measurement classification outside of the University Medical Center.

Aknowledgements

The author would like to thank Eliene Starreveld-Brand, Mattiènne van der Kamp, Erik Huizinga, Denise Boele and dr. Rosier from the University Medical Center in Utrecht for their work and collaboration in this research.

VIII. References

[1] M. van der Kamp BSc, "Automatic uroflowmetry analysis of women, technical medicine report.," 2015.

[2] R. Brand, "Uroflowmetry curve analysis of healthy young women, towards a standardized curve pattern recognition algorithm for uroflowmetry curves, technical medicine report.," 2015.

[3] E. Huizinga, "Pattern recognition in uroflowmetry in women, technical medicine report.," 2015.

[4] D. Boele, "Towards an objective classification algorithm for uroflowmetry curves, technical medicine report.," 2015.

[5] W. Schäfer, P. Abrams, L. Liao, A. Mattiasson, F. Pesce, A. Spangberg, A. M. Sterling, N. R. Zinner, and P. v. Kerrebroeck, "Good urodynamic practices: Uroflowmetry, filling cystometry, and pressure-flow studies**," *Neurourology and urodynamics*, vol. 21, no. 3, pp. 261–274, 2002.

[6] *ANDROMEDA*. Available at: "http://www.andromeda-ms.com/en.php?product/urodynamiksysteme/ellipse/ellipse_details.html", accessed on 14-06-2016.

[7] J. Caffarel, C. Griffiths, R. Pickard, W. Robson, and M. Drinnan, "Flow-how far can you go?," *Urodinamica*, vol. 16, no. 4, p. 259, 2006.

[8] S. Wenske, J. P. Van Batavia, A. J. Combs, and K. I. Glassberg, "Analysis of uroflow patterns in children with dysfunctional voiding," *Journal of pediatric urology*, vol. 10, no. 2, pp. 250–254, 2014.

[9] A.-M. Kajbafzadeh, C. A. Yazdi, O. Rouhi, P. Tajik, and P. Mohseni, "Uroflowmetry nomogram in iranian children aged 7 to 14 years," *BMC urology*, vol. 5, no. 1, p. 1, 2005.

[10] J. Kohen, "A coefficient of agreement for nominal scale," *Educ Psychol Meas*, vol. 20, pp. 37–46, 1960.

[11] A.-R. Hedar and M. Fukushima, "Heuristic pattern search and its hybridization with simulated annealing for nonlinear global optimization," *Optimization Methods and Software*, vol. 19, no. 3-4, pp. 291–308, 2004.

[12] S. Arlot, A. Celisse, *et al.*, "A survey of cross-validation procedures for model selection," *Statistics surveys*, vol. 4, pp. 40–79, 2010.

[13] *Scikit cheat sheet for machine learning algorithms*. Available at: "http://peekaboo-vision.blogspot.nl/2013/01/machine-learning-cheat-sheet-for-scikit.html", accessed on 12-06-2016.

[14] *Cheat sheet for machine learning algorithms*. Available at: "http://i.imgur.com/ryOuViG.png", accessed on 14-06-2016.

[15] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[16] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[17] A. Liaw and M. Wiener, "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[18] D. Basak, S. Pal, and D. C. Patranabis, "Support vector regression," *Neural Information Processing-Letters and Reviews*, vol. 11, no. 10, pp. 203–224, 2007.

[19] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[20] C. E. Metz, "Basic principles of roc analysis," in *Seminars in nuclear medicine*, vol. 8, pp. 283–298, Elsevier, 1978.

[21] *ROC curve example*. Available at: "https://commons.wikimedia.org/wiki/File:Threshold_roc.stack_overflow_answers.svg", accessed on 14-06-2016.

[22] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[23] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*. CRC press, 1984.

[24] *SVM example*. Available at: "https://upload.wikimedia.org/wikipedia/commons/f/fd/SVM_margins.png", accessed on 21-06-2016.

Appendix A Classification display example

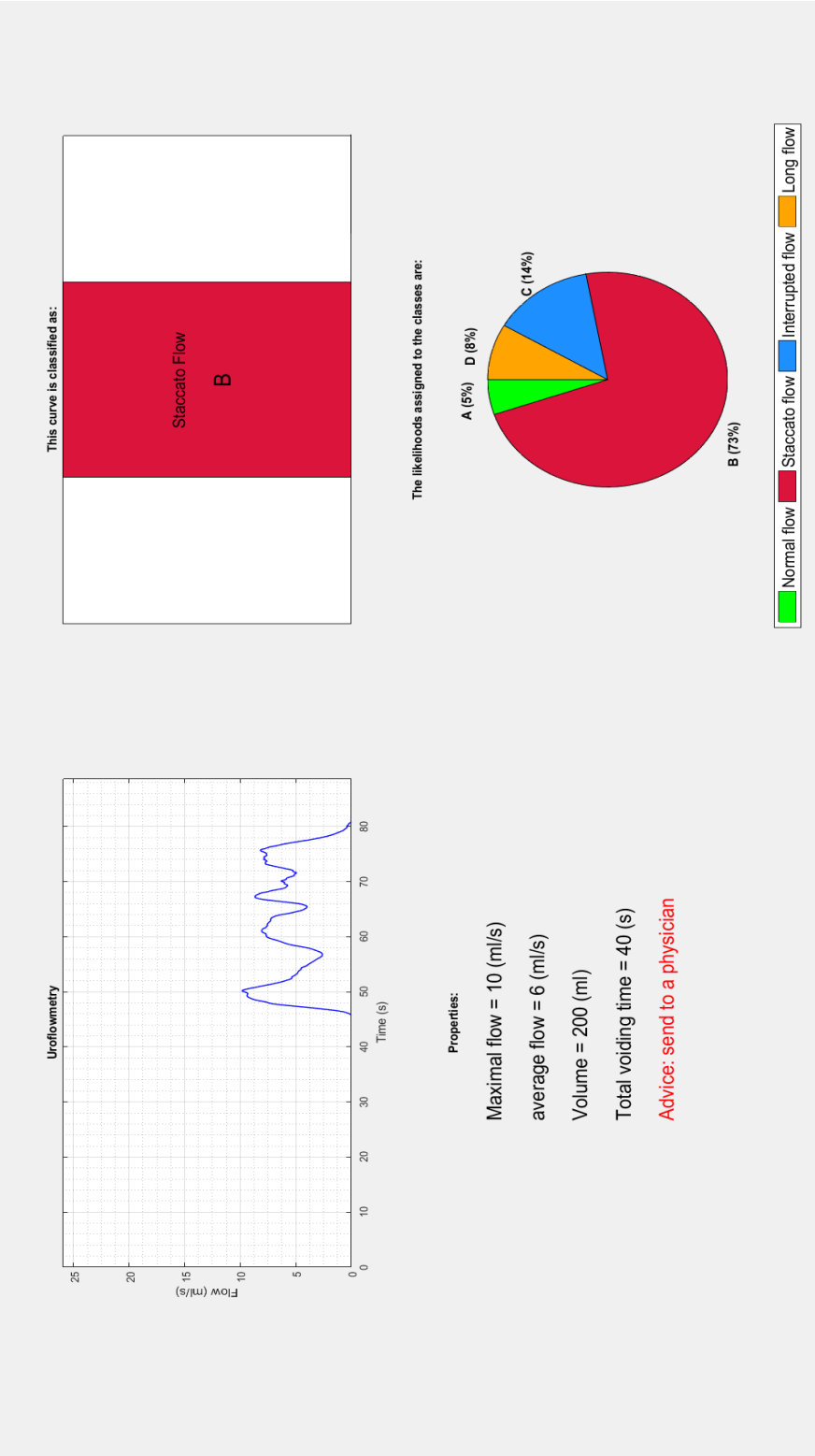

Figure 12: Example of the classification display given for an uroflowmetry measurement.

Appendix B More information about the machine learning classifiers and their construction

In this appendix, the ideas behind the machine learning classifiers constructed in this research are outlined, as well as the way in which they were constructed. For the construction of the classifiers, a feature matrix was used. In this matrix, every column corresponds to a feature and every row corresponds to a training instance. Before a classifier is constructed, all columns of the feature matrix are scaled to the interval [-1, 1] by subtracting the mean and dividing by the maximal value in the column (there are no negative values). This often increases the efficiëncy of the machine learning algorithms. When new instances are classified, the features of this new instance are of course also scaled in the same way.

B.I Ridge regression

Ridge regression is actually an extension of multivariate regression, which tries to make an as best as possible polynomial fit from the features to numerical outputs [19]. In multivariate regression, a matrix of variable values $X \in \mathbb{R}^{m\times(pn+1)}$ is constructed where m is the number of instances, n is the number of features and p is the number of powers of the features used as additional variables. The first column of X consists of ones so that a constant can be added to the polynomial regression fit. Furthermore, a vector of numerical outputs $Y \in \mathbb{R}^{m\times 1}$ is also used. The aim of multivariate regression is now to find a matrix $\Theta \in \mathbb{R}^{(pn+1)\times 1}$ such that the mean squared error between $X \cdot \Theta$ and Y is minimized, i.e.:

$$\min_{\Theta\in\mathbb{R}^{n\times 1}} (X\Theta - Y)^2. \tag{2}$$

Where v^2 denotes $v^T v$ when $v \in \mathbb{R}^{s\times 1}, s \in \mathbb{N}$. The difference between regression and ridge regression is that the coefficients in front of the features in the polynomial fit are penalized using some parameter $\lambda \in \mathbb{R}$. The effect of this is that the extent to which the features are used for prediction becomes smaller and the regression model becomes more simple, which can reduce overfitting. A new minimization function is used:

$$\min_{\Theta\in\mathbb{R}^{n\times 1}} (X\Theta - Y)^2 + \lambda\Theta^2. \tag{3}$$

Analyzing the derivative of this function at it's root gives the optimal Θ matrix:

$$\Theta = (X^T X - \lambda I_{pn+1})^{-1} X^T Y. \tag{4}$$

In MATLAB, ridge regression is done by calling the ridge(Y, X, K, 0) built-in function. The Y matrix is the same as the one in the description above, the X matrix is the X matrix above with the first column, which only contains ones, deleted. K is an $1 \times k$ vector of values of λ to be considered. For each of the values in K, the ridge() function makes a ridge regression model. The zero denotes that the feature columns in the X matrix do not have to be scaled and centered. After completion, the ridge() function outputs k different ridge regression models. After all parameters are chosen, four regression models are made, each one for predicting the likelihood of measurement membership in one of the four classes. If all four likelihood predictions are obtained, the negative likelihood predictions are converted to zero and the resulting likelihoods are normalized to give the resulting likelihood vector. Because four models were constructed, all cross validation mean squared errors considered in the parameter selection step were averaged over all four ridge regression models.

There are two parameters which have to be chosen to build the ridge regression model, namely the powers of the features added to X and the λ parameter for regularization. The first chosen

parameter is the number of powers of the features added to X. When powers of the features are added to X, ridge regression can make a higher polynomial fit for Y. The 10-fold cross validation mean squared error (MSE) is plotted vs. the added feature powers in figure 13. It is seen that for a power of 2, the cross validation MSE is the smallest and so this parameter is set to 2.

The value of the 10-fold cross validation MSE (averaged over 10 runs) is plotted vs. the value of λ in figure 14. It is seen that after $\lambda = 1$, the cross validation MSE has more or less converged. Therefore, λ is set to 1.

After determining suitable power and λ values, the ridge regression classifier is trained on all uroflowmetry data with the chosen parameters. The results for this model are shown in table 2.

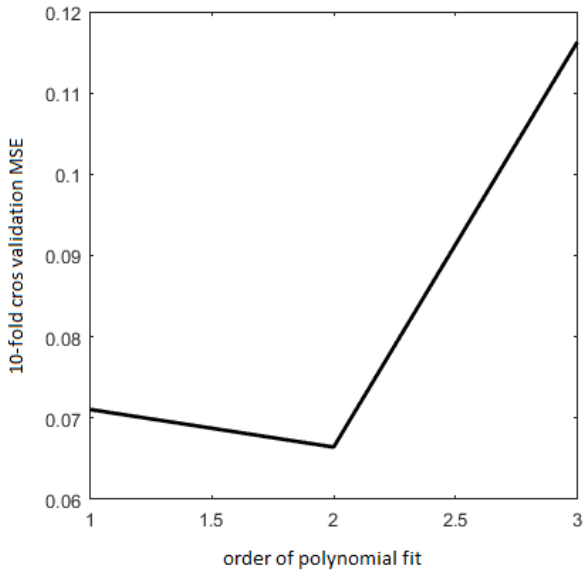

Figure 13: Plot of the 10-fold cross validation mean squared error for the ridge regression model vs. the number of feature powers added.

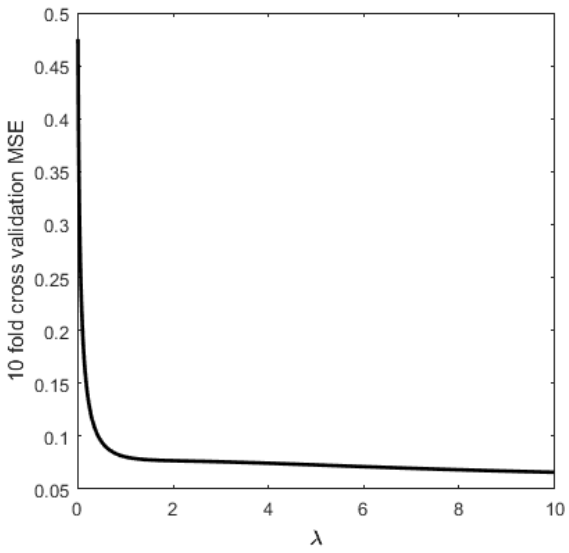

Figure 14: Plot of the 10-fold cross validation mean squared error (averaged over 10 repetitions) for the ridge regression model vs. the value of λ.

B.II K-nearest neighbor

The idea behind the K-nearest neighbor (KNN) classifier is to let the data speak for itself [15]. In order to classify a new instance, the classes of the k-nearest neighbors of the instance are observed. The class that is observed the most in these k neighbors is predicted. Typically, k is chosen to be odd to avoid ties. Two choices have to be made using the (Fine) K-nearest neighbor classifier, namely the parameter k denoting the number of neighbors observed and the distance measure used.

The K-nearest neighbor classifier used in this research is constructed using the classification learner app from MATLAB. It was observed that the 10-fold cross validation accuracy descended when k was chosen to be higher than one, so k is set to 1. After setting k=1, changing the distance measure to city block (Manhattan) distance increased the cross validation accuracy, so this distance measure is chosen. The resulting classifier is exported from the classification learner app and the results for this model are shown in table 2.

B.III Classification and regression trees

Classification and regression trees are used to predict responses from data, for classification trees, these responses are classifications and for regression trees these responses are numbers [23]. For response prediction, the decisions in the tree should be followed from the root down to a leaf node. The leaf node gives the response. For an example of a regression tree, see figure 6. Decision and regression trees are built in the four following steps:

- Start with the total training set. For every feature, sort all features values in order of magnitude and look at all possible ways that the resulting set of values can be split in two. For example, let's say that the feature *number of staccato peaks* has values {0,1,4} for instances 2, 1, and 3, these values can then be split in {0}, {1,4} and {0,1},{4}. Every split results in two new subsets of training instances.

- Now for every possible split, the split that resulted in the best optimization criterion is chosen. The optimization function for constructing classification trees is Gini's diversity index summed over all subsets. This function has to be maximized. Gini's diversity index is an estimator for the probability that two instances taken from a set have the same class. It is calculated as $\sum_{i=1}^{c} p_i^2$ where c is the number of classes in the subset and p_i is the fraction of instances in the subset belonging to class i. All instances in a subset get the same prediction, which is the most frequently occurring class over all subset instances. For the construction of a regression tree, the optimization criterion is the minimization of the mean squared error between the predicted and actual numerical values assigned to the subset instances summed over all subsets. Again, all instances in the subset get the same numerical prediction, which is the mean of all numerical responses of the instances in the subset.

- After the best split is chosen, the split is imposed.

- Now for every resulting subset of instances, the whole process is repeated. The process halts when all subsets have size less than 2 for classification trees and 5 for regression trees.

Now, for prediction on new instances, the splits are followed until the subset that the new instance should belong to is found. The prediction for this new instance is then the response corresponding to that subset.

B.IV Random forests and regression forests

For the construction of a random or regression forest, a number of trees is build which is specified by the user. For constructing these trees, a random subset of both the training instances and the n features is selected with size $\sqrt{n}$ for classification and size $\frac{n}{3}$ for regression [17]. Now, using this random subset of features, a classification or regression tree is built for a random forest or regression forest respectively. Prediction is done for new instances by outputting the class with the highest frequency among all trees in a random forest and outputting the mean of all tree responses in a regression forest. The averaging of the tree predictions has as effect that the regression or classification forest is more general than just one classification or regression tree, while not becoming too simple. The number of trees to build for the regression or classification is a free parameter, an optimal value for this parameter can be found by observing the cross validation performance when the number of grown trees changes. The number of trees resulting in the highest cross validation performance should be chosen.

Random forests are built in MATLAB using the classification learner app. It was seen that

building 95 trees resulted roughly in the highest cross validation accuracy so the number of trees is set to 95. The resulting classifier is exported from the classification learner app and the results for this model are shown in table 2.

The classification learner app cannot build regression forests, so this classifier had to be built in a script using the TreeBagger(NoTrees, X, Y, 'Method', 'regression') function. Four regression forest models are built, as the TreeBagger() function can only predict one dimensional responses. Each regression forest predicts likelihoods for instance membership in one of the four classes. Because four models were made, all out of box mean squared errors considered in the parameter selection step were averaged over all four regression forest models. In the TreeBagger() function, NoTrees stands for the number of trees, X stands for the feature matrix and Y is a vector in which each row corresponds to the likelihood for membership in one of the four classes assigned to an instance. When the TreeBagger function is called, it automatically builds a regression forest model mapping the features to the likelihoods. In figure 15, the out of box mean squared error is plotted vs. the number of trees, it is seen that the error descends much slower when the number of trees is bigger than 250, so this value is chosen for the number of trees.

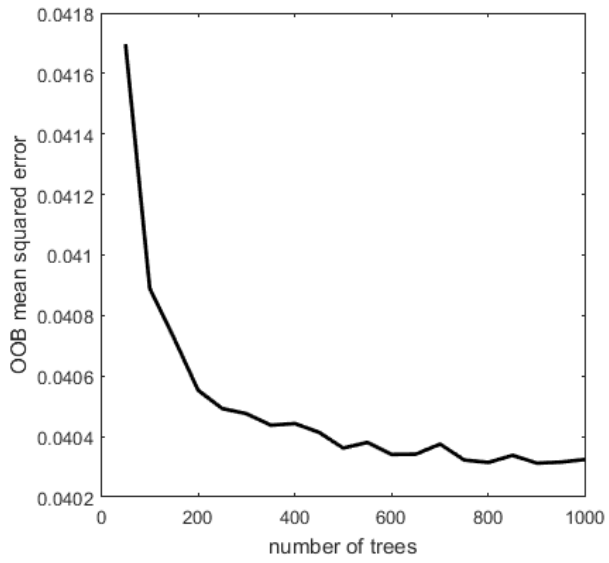

Figure 15: Plot of the out of box mean squared error vs. the number of built trees for the regression forest.

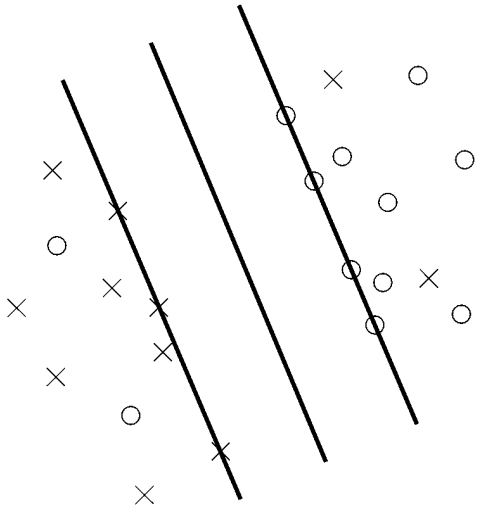

Figure 16: A 2-dimensional example of the separating hyperplane made for a support vector machine classifier. Source: [24].

For determining the ultimate likelihood vector, first the four regression forest classifiers are used to predict the likelihoods of membership in the four classes. These likelihoods are put in a 1×4 vector. Then, the likelihoods smaller than zero are made zero and the resulting likelihood vector is normalized. The results for the regression forest classifier are shown in table 2.

B.V Support vector machines for classification and regression

The idea behind the support vector machine (SVM) for classification is to build a hyperplane separating differently classified data, see figure 16 for a 2-dimensional example. This hyperplane serves as a classification boundary where instances on different sides of the boundary get different class predictions. The goal is to attain as much generalization as possible by maximizing the distance

between the hyperplane and the training instances. This is done by constructing two additional parallel hyperplanes and maximizing the distance between the two outer hyperplanes while still excluding instances from the inner region. This problem is solved by constructing a quadratic optimization problem, subject to certain constraints, which can be solved by the Lagrange multiplier method. If this quadratic optimization method does not have a feasible solution, it is said that the data is not *linearly separable*. In the aforementioned quadratic optimization problem, the number of misclassified instances is also minimized to a certain extent. This extend is determined with a parameter C$\in \mathbb{R}$, which is a free parameter and should be chosen by the user.

If the data is not linearly separable by a hyperplane, kernel tricks are often used. Kernel tricks are transformations to make the data linearly seperable. Mostly, a set of size k of functions is chosen, which are mappings from the features of the instances to the real numbers. The values of these functions can then be used as (possibly linearly separable) new features for SVM training. The kernel trick is also something to be chosen by the user. For more information on classification support vector machines, see [16].

Classification SVM classifiers can be constructed in MATLAB with the classification learner app. First, out of the often used linear, polynomial and Gaussian kernels, the cubic kernel function was chosen as this function resulted in the highest 10-fold cross validation accuracy. After determining the kernel function, the parameter C (called the "box constraint level" in MATLAB) in the quadratic optimization problem was investigated. This parameter was chosen to be equal to 1, as the cross validtation accuracy only descended when bigger values were chosen. The performance of the resulting SVM classifier is shown in table 2. As the SVM classifier does not predict likelihoods for class membership in the four classes, the MAE area and ROC areas remain undetermined for this classifier.

In support vector regression (SVR), the middle hyperplane corresponds to the responses given to training instances. The idea is now actually to get as many instances as possible inside the two outer hyperplanes to obtain an as small as possible mean squared error (see figure 17). The difference with SVM classifiers is now that the distance between the two hyperplanes is another parameter to be tuned by the user. Again, a quadratic optimization problem is proposed which can be solved using the Lagrange multiplier method. Also, a parameter C is defined by the user to define how far training instances may lie outside of the two outer hyperplanes. For making a higher order fit to the data, kernel functions can be used to transform the nonlinear relation between the output and the original features to a linear one. The kernel is another parameter to be decided for support vector regression classifiers. For more information on support vector regression, see [18].

The classification learner app in MATLAB cannot build support vector regression models, so this model had to be build using a MATLAB script. Four SVR models were build, one corresponding to each class. For calculating the ultimate likelihood vector, all predicted likelihoods that were negative are made zero and the resulting likelihood vector is normalized. All cross validation mean squared errors considered in the parameter selection step were therefore averaged over all four SVR models. As already stated, the kernel function, distance between the two hyperplanes (denoted by ϵ) and the parameter C (called the "box constraint level" in MATLAB) have to be chosen by the user to construct a SVR model. When comparing the performance of linear, polynomials and Gaussian kernels on the cross validation mean squared error, the polynomial kernel was chosen. After the kernel function was chosen, the 10-fold cross validation mean squared error was plotted vs. the distance between the two hyperplanes (ϵ). It is seen that an ϵ of 0.1 resulted in the lowest error so this value is chosen. In figure 19, the 10-fold cross validation mean squared error is plotted vs. C (the box constraint level). It is seen that C=2 resulted in the lowest cross validation error, so this value is chosen. The performance measures for the classifier constructed with these parameters are shown in table 2

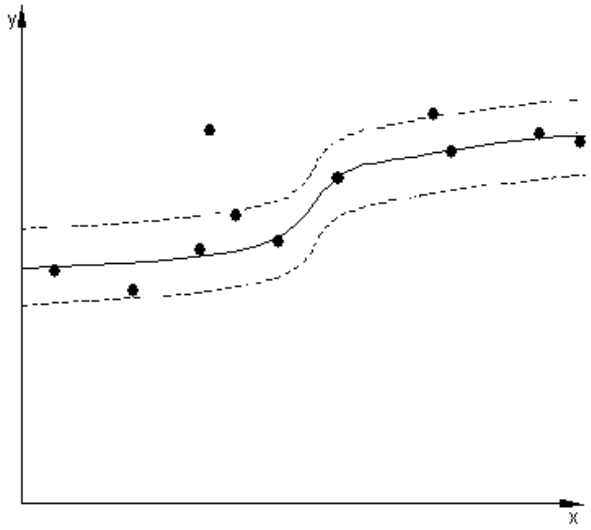

Figure 17: A two dimensional example of a support vector regression classifier.

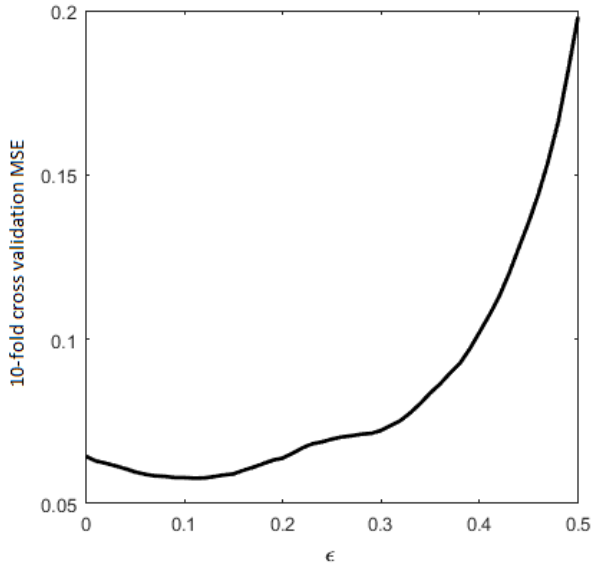

Figure 18: Plot of the 10-fold cross validation mean squared error vs. the distance between the outer hyperplanes (ϵ) for the support vector regression model.

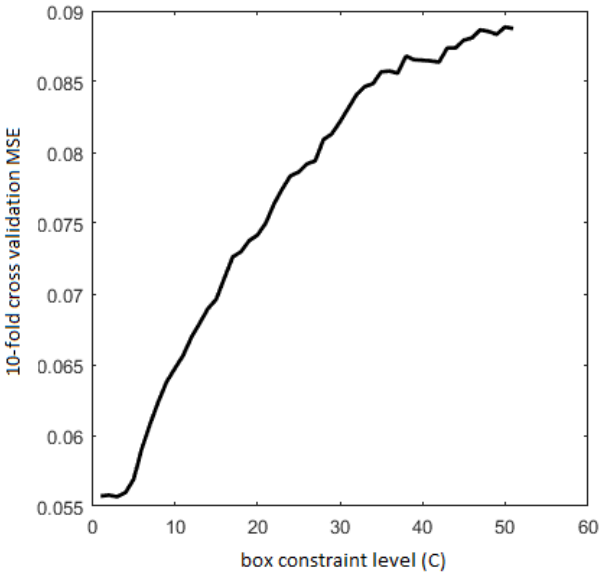

Figure 19: Plot of the 10-fold cross validation mean squared error vs. the box constraint level (C) for the support vector regression model.