Hueck, Andreas:

Classification of Input Devices Using Cursor Trajectories

Bachelorarbeit, Sommersemester 2017

Fakultät für Mathematik, Informatik und Statistik

Ludwig-Maximilians-Universität München

https://doi.org/10.5282/ubm/epub.41005

Department of Statistics
LMU Munich

bachelor's thesis

Classification of Input Devices Using Cursor Trajectories

Author: Andreas Hueck
Supervisors: Prof. Dr. Sonja Greven
Dr. Sarah Brockhaus

May 2017

Abstract

With the immense popularity of the internet and the much lower cost for online surveys compared to in-person interviews, many surveys are now conducted online. One approach to identify whether respondents are having difficulty answering a question is by analysing cursor trajectory data. There are many different input devices, for example computer mouses and touchpads. Respondents' behaviour and thus their cursor trajectories might vary depending on which device is used. As a first step toward tackling this issue, this thesis tries to differentiate between trajectories produced by using a mouse and those made by a touchpad.

There are two classification methods used here; one is using a simple logistic regression model to try and classify the input device, the other is a tree-based approach, more specifically, a Random Forest model. Furthermore, two models with different features are being fitted to the data: a baseline model using features based on previous research as well as a full model utilising several other, new features.

Both methods and models offer a very limited capability to classify respondents' input devices. A comparison between them shows that the full Random Forest model exhibits the best performance of the methods used.

Finally, measures to improve the classification performance and the question whether a differentiation between the input devices is necessary at all, are discussed.

Contents

1 Introduction

Doing online surveys as opposed to collecting information by conventional methods, for example by questioning respondents in person, offers a number of advantages. Firstly and maybe most importantly, online surveys offer a very large reduction in cost by not requiring any manpower once set up. They are also quite adaptable - it is possible to utilise different elements, such as automated skips, text fills or edit messages, to help guide respondents through the questionnaire (Horwitz et al., 2017).

However, because the person asking the questions is a machine and not a human being, additional measures are necessary to detect if and when a respondent is having trouble with the survey. One measure proposed by Horwitz et al. (2017) is tracking cursor trajectories to identify whether a respondent needs assistance. The premise is that cursor movements are different in case a respondent experiences difficulty with a question.

In the study, respondents completed a "typical questionnaire" (Horwitz et al., 2017) utilising a variety of response formats, such as sliders and buttons. Cursor trajectories were recorded using JavaScript. Respondents were also asked to specify the input device they used while completing the survey.

However, the paper only analysed trajectories of respondents who used a computer mouse, as the behaviour might differ depending on which input device was used. The goal of this thesis is to try and identify any possible differences between cursor trajectories of different input devices and subsequently predict the input device, so that further analyses can be adapted accordingly. Different statistical classification methods will be used to try and get as good a classification as possible.

The remainder of this thesis is structured as follows. Section 2 introduces the available data and outlines the steps taken to preprocess the data. Section 3 then goes into detail about the statistical methods and procedures used in this paper. Classification results are presented in section 4 and discussed in section 5. Finally, section 6 provides some ideas on how to progress further with this analysis.

2 Cursor trajectories

The following section will first give some information about the survey that was conducted in general and then focus on the data that was collected (the cursor trajectories), which steps were taken to preprocess the data as well as give a description of the trajectories.

2.1 Online survey

The online survey was conducted by the Institute for Employment Research (IAB) in Nuremberg, Germany. The data were collected in September and October of 2016. The questionnaire consists of four major parts: questions about employment, numeracy, social value orientation and demographic. All in all, respondents were able to answer a maximum of 36 questions; depending on their answers some may have been skipped (Horwitz et al., 2017).

In this thesis, a sample of two questions using different response elements will be analysed.

Figure 1: Question about the respondent's perception of the economic situation

Figure 1 shows a screenshot of the first question to be analysed. It is a question about the respondent's perception of the economic situation in Germany. The response options range from *very bad* to *very good*. The respondent checks exactly one of the radio buttons before moving on to the next question by clicking the submit button titled "Weiter". This question will be referred to as question 1 from this point forward.

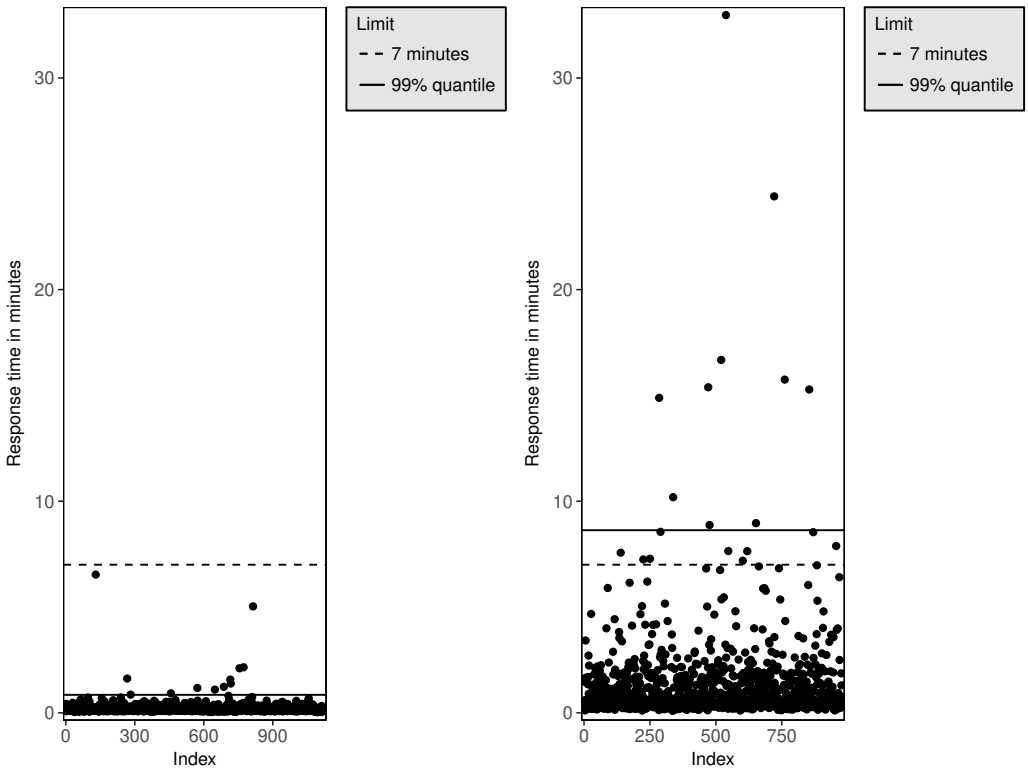
Figure 2: Question about the respondent's salary

Figure 2 shows the second question, in which the respondent is asked about their gross income in the year 2015. It features a slider that ranges from €0 to €100 000 as well as a checkbox indicating that the respondent did not receive any earnings from the particular activity the question refers to (which is defined in an earlier question) in 2015. Here, the respondent either moves the slider to the appropriate position or clicks the checkbox before moving on. This question will be referred to as question 2 going forward.

2.2 Data preprocessing

The data are available in the form of two data sets – one for each question – containing *userid*, *input device*, positions of the area of the input-*form* and *submit button* as well as cursor *position* with corresponding *timestamps* and the position of *clicks* with the respective *timestamps*. Using the package `mousetrap` (Kieslich et al., 2017) the data were imported into `R` (R Core Team, 2016) and some features were calculated (see section 4.2). Respondents were also asked to specify the input device they used.

Subsequently, cases in which respondents took longer than 7 minutes to respond were removed from the data. In total, none were removed for question 1 and less than 2% of cases for question 2 (see also figure 3). Generally, respondents were able to answer question 1 more quickly than question 2.

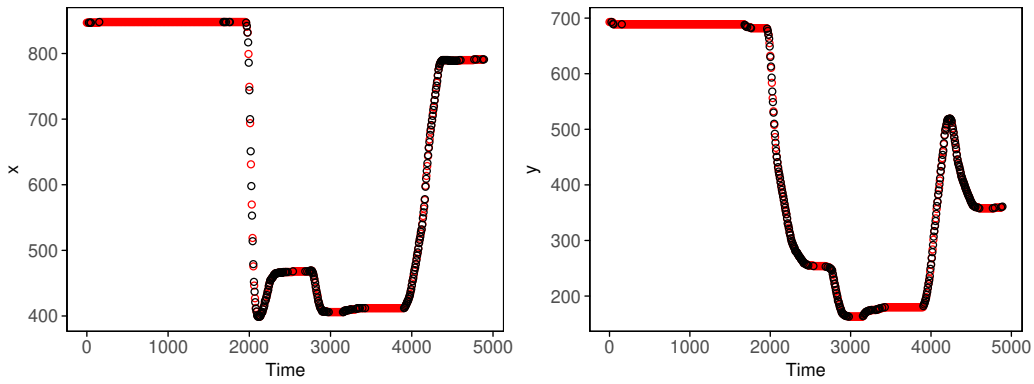


(a) Question 1 (b) Question 2

Figure 3: Scatterplot of response times for all respondents. The solid line shows the 99% quantile and the dashed line the cut-off value at 7 minutes.

Furthermore, cases with extremely low total distance travelled by the cursor were also removed from the data, because either the cursor did not move at all or the movement was miniscule, hindering or making impossible any attempt of classification. For question 1, all cases with a total distance of less than 400 pixels were excluded, while for question 2, the threshold was 200 pixels. In total, 61 cases were removed for question 1 and 43 cases for question 2. Trajectories of all cases removed by this measure can be found in the appendix (Figures A.1 and A.2).

A data point was only created when a change in the cursor's position was detected. This resulted in unevenly spaced timestamps. To ensure temporally equidistant timestamps, trajectories were then modified so that there is a difference of $10ms$ between each timestamp. It was assumed that the cursor remained in the same position in case of timestamps that are more than $8ms$ apart (*constant interpolation*). For timestamps less than $8ms$ apart, *linear interpolation* was used to determine the position of the cursor.



(a) x-coordinates (b) y-coordinates

Figure 4: x and y trajectories for one respondent for question 1. The black points are the raw data; red points show the interpolated data points.

Figure 4 illustrates this procedure. The black points show the raw data for one respondent answering question 1. They are unevenly spaced, because independent of the sampling rate a data point was only created in case the cursor moved. The red points show the resulting observations after interpolation.

2.3 Data description

After preprocessing, a total of 1059 cases remained for question 1, while for the second question 915 cases were left. Table 1 shows the distribution of input devices for each of the two questions. `NA` denotes a missing value, while the category *Other* contains any specified input devices that do not fit in the previous categories.

Input device	Question 1	Question 2
Mouse	804	687
Touchpad	213	194
Touchscreen	26	25
TrackPoint	8	5
Other	5	1
`NA`	3	3
Total	1059	915

Table 1: Number of devices used for each of the two questions

For this thesis, only the two devices *mouse* and *touchpad* are relevant. For a plot of all trajectories using a touchscreen, see Figure A.3 in the appendix. Figure 5 shows a random selection of mouse and touchpad trajectories for each of the two questions.

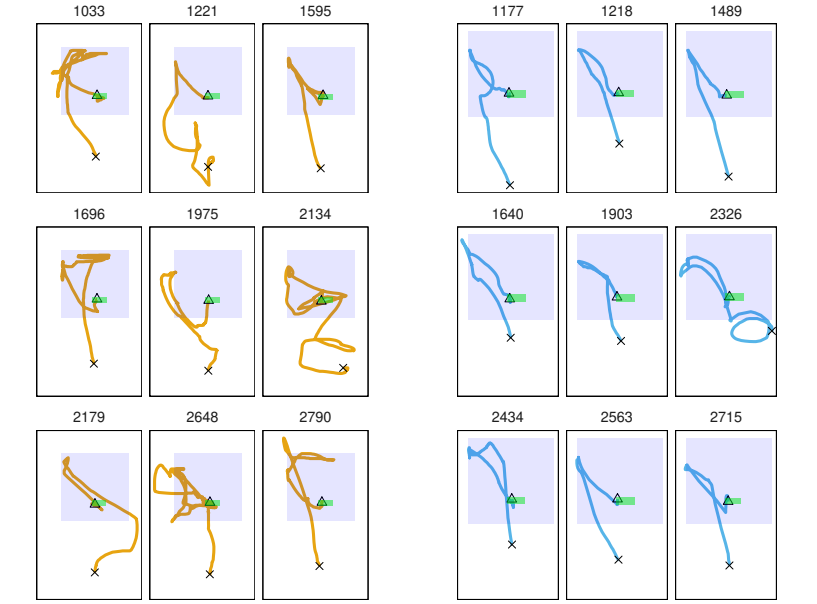
For these plots, the coordinates were standardised to the area of the form (the blue area). The bottom left of the form has the coordinates $(0, 0)$ for each trajectory, while the top right coordinates are $(1, 1)$. Mouse trajectories are shown in orange, touchpad trajectories in blue. The numbers on top are the respondents' IDs.

For question 1, some erratic cursor behaviour can be observed, though it is much more prevalent for mouse users here. For many of the sampled respondents using a touchpad, the trajectory looks similar: they move their cursor to one of the five radio buttons (and click on it, presumably), before moving the cursor to the submit button.

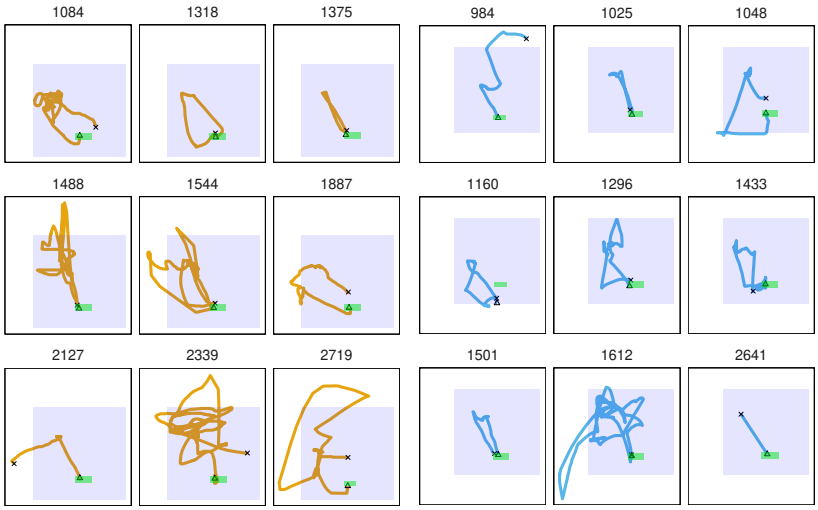
In the samples for question 2, on the other hand, some quite dissimilar trajectories can be seen. This might have to do with the different response elements used, or maybe with the difficulty of the question in general. Some respondents might have just checked the box and might not have moved the slider at all. Again, mouse trajectories seem to show more erratic movements than touchpad trajectories.

Note: The standardisation was not performed for any of the calculated features (see section 4.2), because respondents might have been using differ-

ent screens and resolutions, making it very difficult to generalise. Ideally, this study would have to be performed under laboratory conditions, where a given amount of pixels would always correspond to a fixed length in cm for all respondents.



(a) Mouse, question 1 (b) Touchpad, question 1



(c) Mouse, question 2 (d) Touchpad, question 2

Figure 5: A random sample of standardised trajectories for both questions. The symbols $\times$ and Δ denote the starting and ending points, respectively. The blue area represents the form, while the green rectangle shows the submit button of the form.

3 Methodology

Before going into specifics on the methodology used in this paper, it is necessary to make some notational definitions.

The outcome or the measurement we are interested in is called the *response* Y. Variables that (might) be correlated with the response are also called *features*, denoted as x_i. This is consistent with the notation in Hastie et al. (2009). Regression coefficients are referred to as β_i, with β_0 denoting the intercept.

The question this thesis tries to answer can be interpreted as a classification problem with two classes (mouse and touchpad, respectively). Different methods exist to tackle such a problem, two of which were used here: logistic regression (see section 3.2) and Random Forests (see section 3.4).

Validation of models is frequently done using cross-validation, which is explained in section 3.1 and can be applied to logistic regression models as well as Random Forests. One of the problems that can occur when using logistic regression is multicollinearity, which is addressed in section 3.3. Finally, measures to gauge model performance are detailed in section 3.5.

3.1 Cross-validation

In order to assess prediction accuracy of a model, it is often useful to fit it to only part of the data (a training set) and get predictions on a different, independent subset (validation set) to reduce overfitting and get an idea of how well the out-of-sample prediction works. Because data are scarce, it is usually not viable to use part of the data just for validation.

Cross-validation (CV) is a simple and widely used method to estimate prediction error. In this thesis, *K-fold* cross-validation is used to accomplish this. Algorithm 1 details this procedure.

Algorithm 1 K-Fold cross-validation

1. Split the data set into K roughly equal-sized parts
2. Repeat K times, for $k \in \{1, \ldots, K\}$:
   (a) Keep sample k as validation/test data and the other $K-1$ samples as training data
   (b) Fit the model to the training data
   (c) Make predictions on the test data and calculate the prediction error
3. Calculate the average prediction error

For repeated cross-validation, this algorithm can be repeated several times, each time choosing different subsets.

3.2 Logistic regression

The first classification method used in this paper is logistic regression. It can be applied to a classification problem with two classes. A logistic regression model requires a binary response Y that can take values 0 and 1. In this case the two classes are mapped to the values 0 and 1, respectively. They will be referred to as class 0 and class 1. What is modelled, then, is the posterior probability π for class 1 given the various features x_i. Similarly to the simple linear model, the linear predictor is defined as:

$$\eta = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p. \tag{1}$$

This modelling strategy has the obvious flaw that fitted values can be outside the permissible range for probabilities, $[0, 1]$. To ensure fitted values are within this interval, the linear predictor is linked to the probability using the logistic response function

$$h(\eta) = \frac{\exp(\eta)}{1 + \exp(\eta)} = \pi. \tag{2}$$

Inverting this function yields the logistic link function

$$g(\pi) = h^{-1}(\pi) = \log\left(\frac{\pi}{1-\pi}\right) = \eta, \tag{3}$$

also called logit-link. Thus, the linear predictor η equals the logarithm of the odds or, in short, the log-odds.

So for any given set of features x_i, the probability for class 1 is π and subsequently, the probability for class 0 is $1 - \pi$. Selecting a threshold value for π (often set to 0.5 or determined using ROC, see also section 3.5) makes it possible to predict the class of a new observation.

One of the advantages of logistic regression is that is quite simple to interpret. For any given feature x_j, the corresponding coefficient β_j is the change in log-odds, should x_j increase by 1, other features held constant. Or, on the odds scale: an increase in x_j of 1 yields an increase in the odds for $Y = 1$ by $\exp(\beta_j)$, other features held constant.

Logistic regression models are fitted using a maximum likelihood approach. For further information about fitting logistic regression models, see also Fahrmeir et al. (2009) and Hastie et al. (2009).

In case of nonlinear effects on the response, an extension to this model is possible. Using generalized additive models (GAMs), smooth functions of the features can be used to replace parts of the linear predictor:

$$\eta = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + f_1(x_{k+1}) + \cdots + f_{p-k}(x_p). \tag{4}$$

There are many different ways to model these smooth functions, some of which are detailed in Wood (2006) and implemented in the R package `mgcv` (Wood, 2011). The idea behind the concept is to estimate a smooth function as a linear combination of weighted base functions, for example for B-Splines one has

$$f(x) = \sum_{j=1}^{d} \gamma_j B_j(x), \tag{5}$$

where d is the number of bases B, which are identical but for a shift along the x-axis and γ_j are the respective weights. A detailed explanation about how to construct B-Splines as well as the definition of the B-Spline bases is given in Fahrmeir et al. (2009), p. 303–307. High variance and overfitting can be reduced by penalisation. This can be achieved by adding a penalty term to the equation that will be minimised. For P-Splines (penalised B-Splines, Eilers and Marx, 1996), large differences in neighbouring weights γ_j are penalised to get a smoother function. This process is also detailed further in Fahrmeir et al. (2009).

3.3 Multicollinearity

One of the problems that can arise while using regression models is multicollinearity. This is best demonstrated by a small example:

Consider the simple linear model

$$\mathbb{E}(Y) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 \tag{6}$$

with three features x_1, x_2 and x_3 and perfect collinearity between x_1 and x_2, i.e. $x_1 = a \cdot x_2$. This can happen if two features measure the same thing or something very similar, for example let x_1 be the weight in kilogrammes and x_2 the weight in tons. In this extreme case, the only difference between the two features is the unit of scale and there is a perfect linear association. This means that the two regression coefficients β_1 and β_2 cannot be estimated uniquely:

$$\mathbb{E}(Y) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 = \beta_0 + (\beta_1 \cdot a + \beta_2) x_2 + \beta_3 x_3. \tag{7}$$

In practical applications, there are usually no perfectly linearly dependent features, but rather highly correlated ones (as is the case for this analysis, see section 4.2). This is called multicollinearity and can cause imprecise (high variance) estimates of the regression coefficients (Fahrmeir et al., 2009, p. 170 - 171). One of the most widely used solutions is to simply omit some of the affected features – this is also the approach utilised in this analysis.

3.4 Random Forests

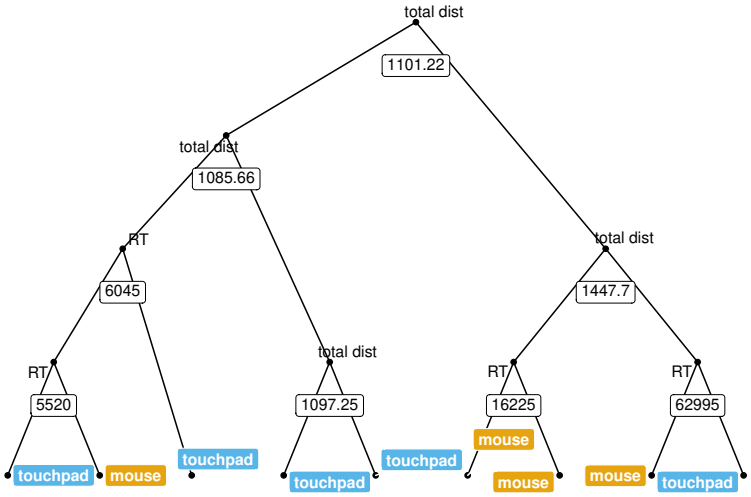
Another modelling strategy that does not suffer from problems with multicollinearity is a tree based approach, more specifically a Random Forest model. Random Forests were first introduced by Breiman (2001) as a modification to bagging (Breiman, 1996), a technique for reducing the variance of a predictor.

Before going into specifics about Random Forests, a short introduction of tree-based methods: Figure 6 shows two binary decision trees using two features of the cursor data, `total_dist` and `RT`. Such decision trees can be used to make class predictions, such as which input device a respondent is using. To classify a new observation x, at each node (except the leaf nodes at the bottom), a threshold decision (binary split) is made, determining which way to go down the tree.

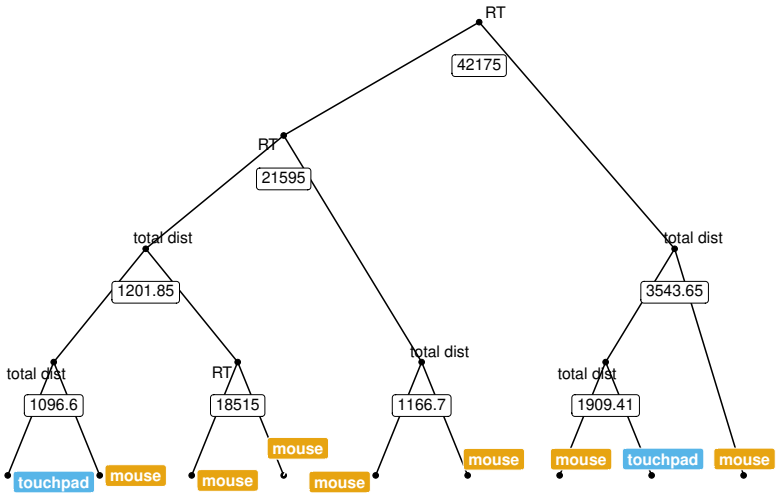
Consider, for example, Figure 6a and a fictional new observation x with `total_dist` = 900 and `RT` = 7000. The first thing to check when applying this decision tree to a new observation is the feature `total_dist`. In this case, the threshold value is 1101.22. As the value of the new observation (900) is less than this threshold, move down the tree to the left (otherwise you would move down one level to the right of the tree). At this node, again the decision is based on the total distance and again, the value is smaller than the threshold, thus moving down to the left. Now, the decision will be made by looking at the value of `RT`. As in this example, the value (7000) is greater than the threshold (6045), move down to the right, reaching a leaf node of the tree. This particular leaf node is labelled *touchpad*, thus the new observation will be classified as such according to this particular decision tree.

Using binary decision thresholds is the preferred method, as using multiway splits would fragment the data too quickly, leaving insufficient data for the lower levels of the tree. Furthermore, multiway splits can be achieved by using multiple binary decision thresholds, rendering them unnecessary (Hastie et al., 2009).

A major problem of decision trees is their instability. Due to the hierarchical structure, an error in one of the upper levels of the tree is subsequently propagated down to the lower levels. This results in a high variance – a small change in the data can have large effects on the tree (Hastie et al., 2009). One possibility of reducing this variance is by bagging (Breiman, 1996), which grows several trees based on bootstrap samples of the data and averages them (for classification, the averaging is done by taking the majority vote of all the trees). Bootstrap samples are generated by randomly drawing a sample with replacement of the same size as the original data from the training data.



(a) A decision tree



(b) Another decision tree with different splits

Figure 6: Sample decision trees using the cursor trajectory data and the two variables total distance (total dist) and response time (RT).

However, in the case of correlated trees, the benefits of bagging on variance reduction are limited: Consider B identically distributed trees, each with variance σ^2 and with positive pairwise correlation ρ. Then, the variance of the average is

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2, \tag{8}$$

see also Hastie et al. (2009), p. 588. Increasing B will make the second term disappear, but the first will not.

Random Forests seek to reduce that first term by generating uncorrelated trees. This is achieved by randomly selecting a subset of features to use at each split.

Algorithm 2, taken from Hastie et al. (2009), outlines the process of growing a Random Forest.

Algorithm 2 Random Forest for Classification (Hastie et al., 2009, p. 588)

1. For $b = 1$ to B:
   (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size N from the training data.
   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.
      i. Select m variables at random from the p variables.
      ii. Pick the best variable/split-point among the m.
      iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x:
Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree. Then $\hat{C}_{\text{rf}}^B(x) = majority\ vote\{\hat{C}_b(x)\}_1^B$.

N is the number of observations in the training data. Typical default values are $\lfloor\sqrt{p}\rfloor$ for m, where p is the total number of features and a minimum node size n_{min} of one (Hastie et al., 2009). For B, a value of 500 is often chosen. Figure 6 shows a couple of trees from an ensemble. These parameters are also called hyperparameters as they need to be selected before the model is fitted. There are methods for tuning these hyperparameters provided by the `mlr` R-package (Bischl et al., 2016).

A subject still not touched upon is how the feature and value for a split at each node is determined. The decision on which feature and value to use for the split is made based on a node impurity measure. In the R-Package `randomForest` (Liaw and Wiener, 2002), which is used here, a weighted Gini index can be utilised to measure node impurity. Since this is a two class problem, the formula for the Gini index is simplified somewhat.

Let p_m be the proportion of observations in the second class of node m. Then, the Gini index is given as

$$Gini = 2p_m(1 - p_m). \tag{9}$$

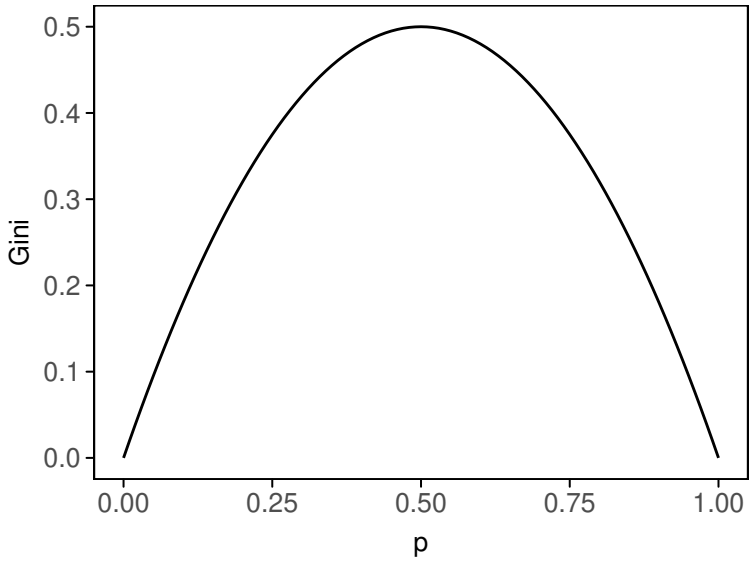


Figure 7: Gini index as a measure of node impurity for two classes as a function of the proportion p in the second class.

Figure 7 shows the Gini index as a function of the proportion p. It has its highest value at $p = 0.5$, i.e. when both classes have the same proportion in a node (node impurity is at its highest), and is 0 if a node contains observations from one class exclusively.

This index is then weighted by the number of observations in the two child nodes generated by splitting node m (Hastie et al., 2009). So, for each split, the feature and value that yield the largest decrease in node impurity are chosen.

Another feature of Random Forests is that the node impurity measure can be used to construct variable importance plots. The improvement in the split-criterion at each split in each of the trees is accumulated over all trees separately for each feature and is attributed to the respective splitting feature. The values are then averaged for each feature. Features with the highest values (highest mean decrease in node impurity) are then considered the most important ones according to this split criterion.

3.5 Classification performance measures

Different measures exist to quantify the performance of classifiers. The ones used in this thesis are the receiver operating characteristic (ROC) curve and corresponding area under the curve (AUC) as well as the mean misclassification error (MMCE) and are discussed in this section.

		Predicted	
		Mouse	Touchpad
True	Mouse	783	21
	Touchpad	169	44

Table 2: Confusion matrix, a cross table of the true input devices and those predicted by the model.

Table 2 shows a confusion matrix for a sample model predicting respondents' input devices. The columns show what was predicted, while the rows show what the actual device was. The values on the main diagonal show what was predicted correctly, while anything else is an incorrect classification. Thus, for a perfect classifier, all cells not on the main diagonal would be 0.

In conjunction with the confusion matrix, sensitivity (or true positive rate) and specificity (true negative rate) are often mentioned. In order to get those, one has to specify a class corresponding to positive and negative, respectively. In medicine, for example, this is often intuitively possible (presence of a particular disease) (Hastie et al., 2009). In this case, however, it is not immediately clear which input device should be considered as positive, the choice is arbitrary. For this thesis, touchpad has been chosen as the positive class. Furthermore, the absolute values have to be converted to relative frequencies: In this example, one gets the following sensitivity and specificity

$$Sensitivity = \frac{44}{44 + 169} \approx 0.21 \tag{10}$$

$$Specificity = \frac{783}{783 + 21} \approx 0.97. \tag{11}$$

These values can also be put into a table (see Table 3), showing true negative rate (tnr), false positive rate (fpr), false negative rate (fnr) and true positive rate (tpr), where $fpr = 1 - tnr$ and $fnr = 1 - tpr$.

As mentioned previously, one can choose the threshold value (the posterior probability threshold controlling when to predict the positive class), yielding different values for sensitivity and specificity for each threshold value.

		Predicted	
		Mouse	Touchpad
True	Mouse	0.97 (tnr)	0.03 (fpr)
	Touchpad	0.79 (fnr)	0.21 (tpr)

Table 3: Confusion matrix with relative values, a cross table of the true input devices and those predicted by the model.

The values calculated for all possible threshold values $[0, 1]$ can be combined to form a ROC curve by plotting sensitivity (true positive rate) against $1-$ specificity (false positive rate). This procedure makes it impossible to see the threshold values in the plot, however.

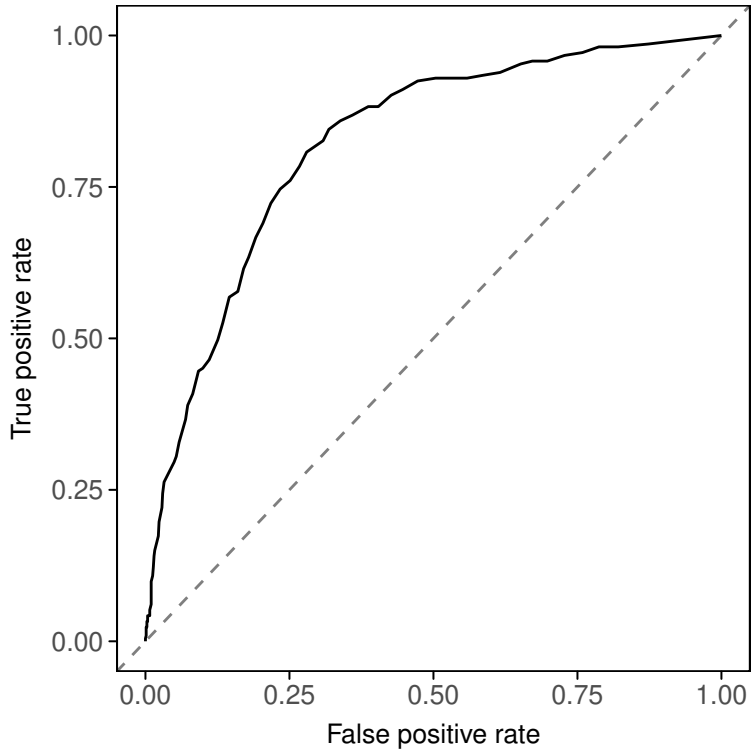


Figure 8: ROC curve showing sensitivity and $1-$ specificity for all threshold values. The dashed line represents a random classifier.

Figure 8 shows the ROC curve corresponding to the same model used to calculate the confusion matrix in Table 2 (with a threshold value of 0.5). The dashed line shows the performance of a random classifier, that is, of random guessing. The farther out the ROC curve is, the better.

An ideal classifier would jump from the lower left corner to the top left corner and then move to the top right corner. This would imply a true positive rate of 1 and a false positive rate of 0. The area under the ROC curve is called AUC (area under the curve) and is also a measure of classification performance. It ranges from 0.5 (random classifier) to 1 (perfect classifier) and can be calculated by integrating the ROC curve. In this example, $AUC = 0.82$.

Another typical performance measure for classification tasks is the mean misclassification error, or MMCE. This is the amount of misclassified observations over all observations. Continuing the example:

$$\text{MMCE} = \frac{169 + 21}{169 + 21 + 783 + 44} \approx 0.19. \tag{12}$$

In the same manner as for sensitivity and specificity, the MMCE can also be calculated for different threshold values. There is also a measure called accuracy, which is the inverse of MMCE ($accuracy = 1-\text{MMCE}$). In this thesis, the MMCE is used.

There are a couple of other measures that can be calculated. Because these are calculated using the same basis, for the sake of brevity and to avoid being bloated and confusing, in this thesis, only the aforementioned ones are used.

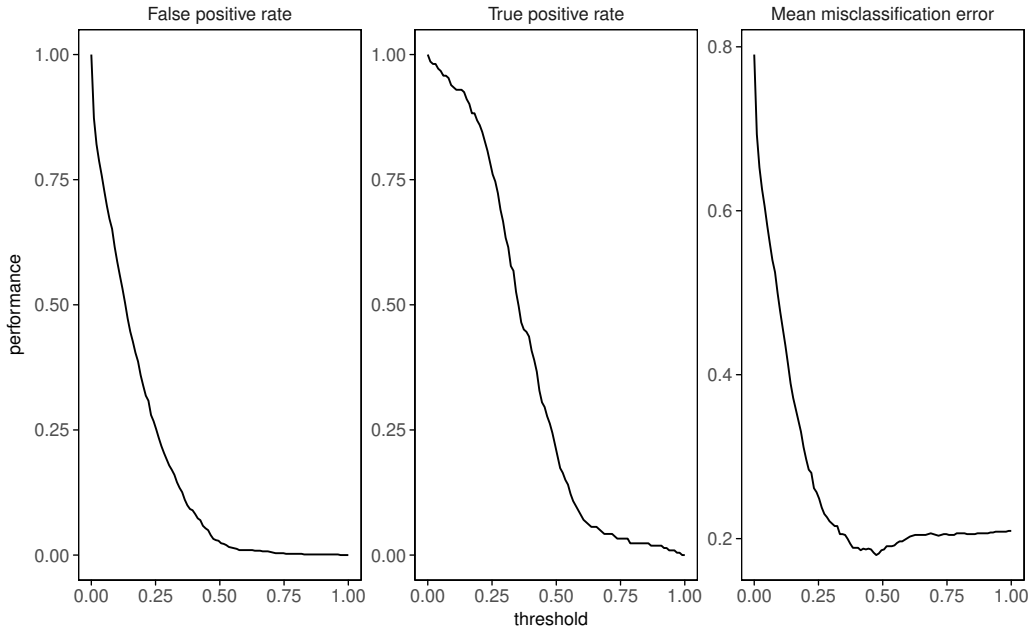


Figure 9: $1-$ specificity, sensitivity and MMCE for different threshold values

Figure 9 shows 1− specificity (false positive rate), sensitivity (true positive rate) as well as the mean misclassification error for different threshold values. False positive and true positive rate always start at 1 for a threshold value of 0 and end at 0 for a threshold value of 1, while the MMCE equals the relative frequency of the negative class for a threshold value of 0 and the positive class for a threshold value of 1. In section 4, only the ROC curve and the MMCE plot will be shown.

4 Results

Classification was performed using the R-Package `mlr` (Bischl et al., 2016), which provides a unified interface for classification tasks and utilises several other R-Packages, such as the `randomForest` package (Liaw and Wiener, 2002), as well as `mgcv` (Wood, 2006) for fitting GAMs. Furthermore, separate models were fitted for each of the two questions. This is because there might be similar trajectories for questions answered by the same respondent. To ensure comparability, the `benchmark` function of the `mlr` R-Package was used for the cross-validated models. It guarantees that the same training and test data sets are used for each model.

4.1 Baseline models

Going at this from a very naïve perspective, as a first step to this analysis, baseline models were established to investigate any potential correlations with the input device used by the respondent.

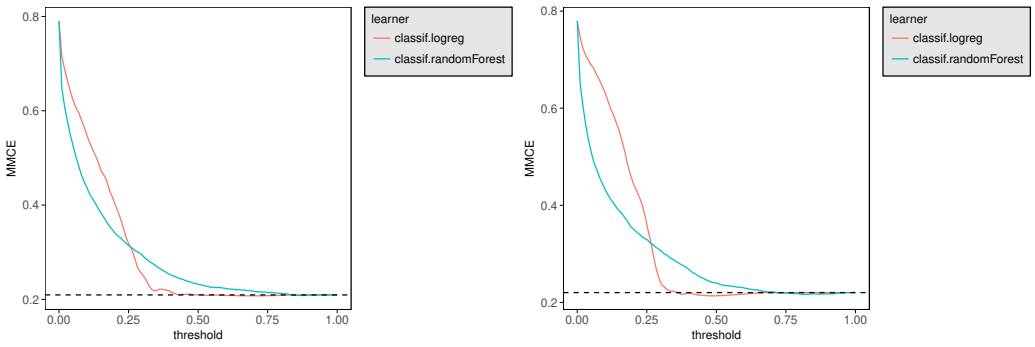
Early work involving cursor trajectories used mainly the total distance travelled, for example to show a connection with data quality (Stieger and Reips, 2010). Furthermore, response time was identified as an indicator of respondents having trouble with a question in an online survey (Horwitz et al., 2016).

These two features were then used in the baseline models to try and predict respondents' input devices. A Random Forest model (rf) and a logistic regression model (logitreg) were fitted using repeated (ten times) tenfold cross-validation, using total distance and response time as features and the input device as the response.

Table 4 shows the average MMCE over all CV iterations for both questions and models at threshold 0.5, while Figure 10 shows the MMCE for all thresholds.

	Model	MMCE	AUC
Question 1	logitreg	0.209	0.763
	rf	0.232	0.693
Question 2	logitreg	0.214	0.738
	rf	0.240	0.700

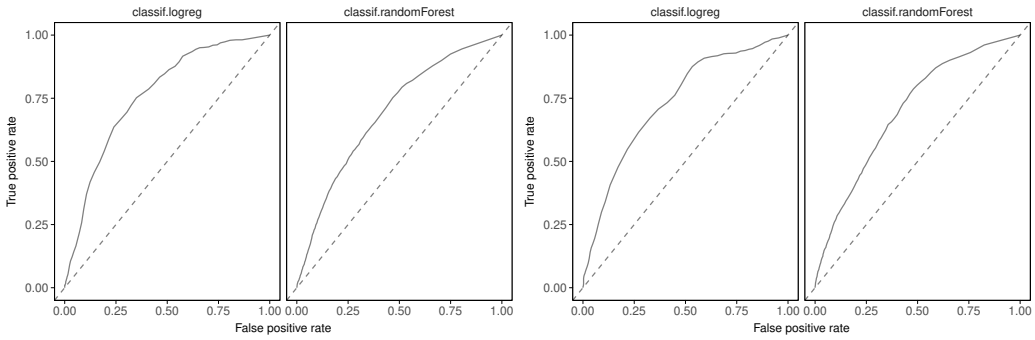
Table 4: Performance of the baseline models. MMCE at threshold 0.5 as well as AUC values.



(a) Question 1 (b) Question 2

Figure 10: MMCE for all thresholds.

The dashed line shows the proportion of the smaller class (touchpad) in the data (0.209 for question 1 and 0.220 for question 2). Any MMCE above this value indicates that classifying all observations as belonging to the majority class (mouse, in this case) would yield a better classification result. The logistic regression model performs slightly better than the Random Forest model in both cases here (for reasonable threshold values), though its performance is still quite meagre, shortly dipping below the dashed line by an extremely small margin.



(a) Question 1 (b) Question 2

Figure 11: ROC Curves for both baseline models and questions.

Figure 11 shows the ROC curves for both models and questions. The corresponding AUC values can be found in Table 4. Again, in both cases, the logistic regression model performs slightly better than the Random Forest.

Overall, the performance of the baseline models is not much or at all better than classifying everything as the larger class, making them impractical to use.

4.2 Key features

Considering the very poor performance of the baseline models, it might be beneficial to glean additional information from the data. The `mousetrap` R-Package (Kieslich et al., 2017) provides the means to compute a number of additional features using the cursor trajectory data.

Feature	Description
xpos_max	Maximum x-position
xpos_min	Minimum x-position
ypos_max	Maximum y-position
ypos_min	Minimum y-position
xpos_flips	Number of directional changes along the x-axis
ypos_flips	Number of directional changes along the y-axis
RT	The total response time in milliseconds
initiation_time	Time at which first mouse movement was initiated
idle_time	Total time without mouse movement
hover_time	Total time of all periods $> 2000ms$ without movement
hovers	Number of periods $> 2000ms$ without movement
total_dist	Total distance covered by the trajectory
vel_max	Maximum velocity
vel_max_time	Time at which maximum velocity occurred
acc_max	Maximum acceleration
acc_max_time	Time at which maximum acceleration occurred
acc_min	Minimum acceleration
acc_min_time	Time at which minimum acceleration occurred

Table 5: Overview and short description of potentially relevant features

Table 5 lists all features calculated by the `mousetrap` R-Package that might be relevant. Considerations such as the correlation between features (see also section 3.3) as well as variable importance need to be taken into account when deciding which features to include in the model.

To reduce correlation between features (for example, the time spent being idle is extremely dependent on the overall response time) and to reduce the overall number of features, a couple of new features were calculated, as detailed in table 6.

The ranges were calculated by subtracting the maximum from the minimum values, respectively. The relative time values were determined by dividing by the total response time. Redundant features were subsequently removed. To alleviate problems with outliers (this only pertains to the logistic regression models), observations with extreme values were truncated to

Feature	Description
xrange	Total range in x-direction
yrange	Total range in y-direction
accrange	Total range in acceleration
rel_idle_time	Relative idle time
rel_hover_time	Relative hover time
rel_init_time	Relative initiation time
rel_vel_max_time	Relative time of maximum velocity
rel_acc_min_time	Relative time of minimum acceleration
rel_acc_max_time	Relative time of maximum acceleration

Table 6: New features calculated by transforming other features

the 1%- and 99%-quantiles, respectively (windsorisation). Figure 12 shows a scatterplot matrix of all remaining features for question 1 (the one for question 2 can be found in the appendix). What becomes very obvious is the extremely high correlation of the velocity features with the acceleration features. Thus, features concerning the velocity were omitted. What might look unusual as well is the rather large quantity of observations that have a relative hover time of 0. This is simply due to the definition of this feature (see table 5). Time spent hovering is only recognised as such if the cursor does not move for at least $2000ms$ at a time.

This still leaves a considerable number of features. Using a single fit of a Random Forest, the variable importance plot pictured in Figure 13 was constructed. It is worth noting that this plot only reflects the importance values of a single fit. Thus, the resulting values might vary a little across multiple iterations. However, over 100 iterations, the top 6 features remained largely the same across both questions. Because of this, these 6 features were then used for subsequent models.

The two most important features were relative initiation time and the total distance travelled by the cursor. The other features selected were relative idle time as well as the ranges in acceleration and x-position and lastly the number of flips in x-direction. Perhaps as a bit of a surprise, response time was not included in the most important features.

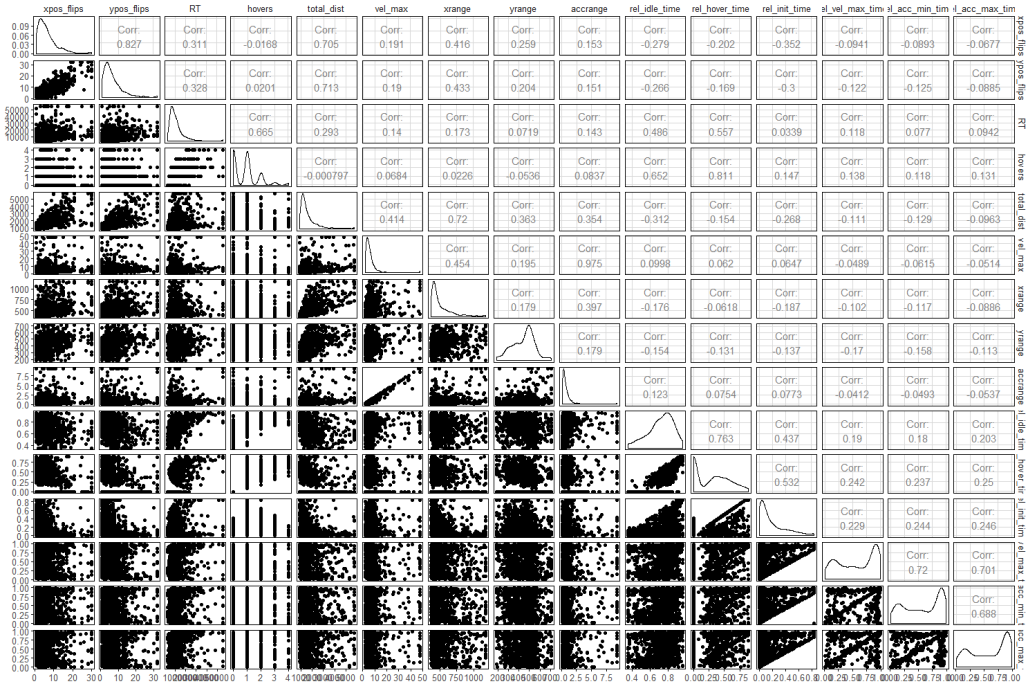
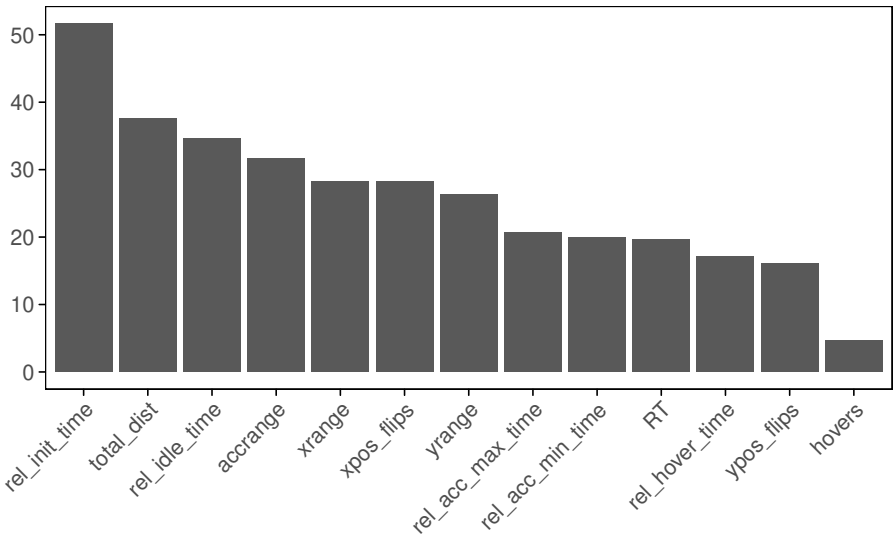
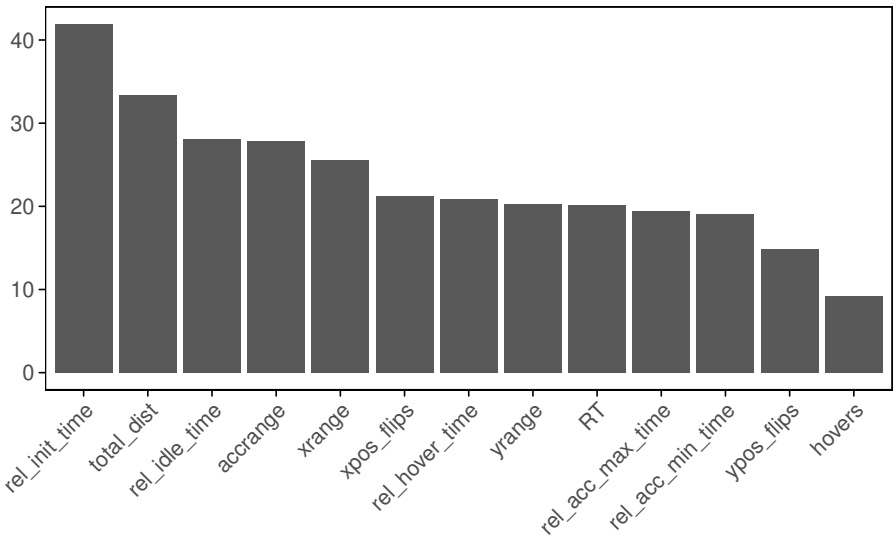


Figure 12: Scatterplotmatrix for remaining features for question 1. Note the high correlation of velocity and acceleration related features. The upper triangle shows the correlations, the lower triangle shows the respective scatterplots. The main diagonal consists of density estimates.



(a) Question 1

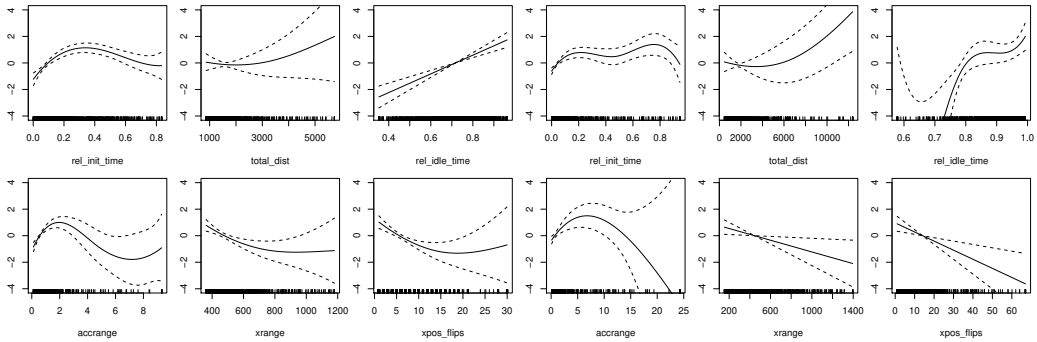


(b) Question 2

Figure 13: Variable importance plot based on GINI impurity and a single Random Forest fit.

4.3 Logistic GAM

To get a better overview of the correlation of the features with the response, a logistic GAM using P-Splines was fitted. This model is based on a single fit to the whole data set and was not cross-validated, thus calculating performance measures such as MMCE is not applicable here.



(a) Question 1 (b) Question 2

Figure 14: Estimated smooths for all features for both questions with log-odds on the y-axis.

Figure 14 shows the estimated smooth effects for each of the features and both questions. They are centred around 0. The dashed lines show the pointwise confidence intervals. The estimated effects look similar for both of the questions, though the ranges of the feature values vary between the two questions. All of the estimated smooths can only be interpreted while holding all of the remaining features constant (ceteris paribus).

The odds for the input device touchpad generally increase with increasing relative initiation time, reaching the maximum at around 0.3 before decreasing again for question 1. There is some more variability for question 2, but the general shape remains the same. With increasing total distance, the odds for touchpad increase as well, taking a parabolic form that is more pronounced for question 2. There is some difference in the estimated smooths for the correlation of relative idle time with the response. In both cases, the odds of the respondent using a touchpad increase with increasing relative idle time, though for question 1 the increase is linear, while for question 2, the estimate has a very high variance and thus, large confidence intervals in the lower ranges. The same is true for the smooth of the acceleration range. An increase in the odds of a touchpad being used is followed by a slight decrease and then a levelling-off for question 2, while the shape is more of a parabola for question 2, but again with an extremely large confidence

interval for higher values of the acceleration range, not allowing any interpretation in this range. An increase in the range in x-position and the number of flips in x-direction both decrease the odds for a touchpad. For question 1, the estimated smooth effects are slightly parabolic, while for question 2 the correlation is linear.

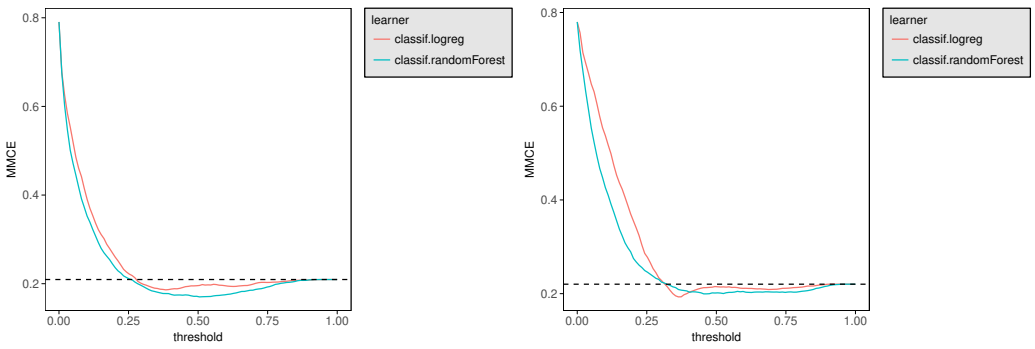
Again, all of the above can only be interpreted while holding the other features constant.

4.4 Full models

Using the features determined in section 4.2, both the logistic regression model and the Random Forest were fitted again, analogously to the models in section 4.1. Again, Table 7 shows the average MMCE over all CV iterations for both questions and models at threshold 0.5, while Figure 15 shows the MMCE for all thresholds.

	Model	MMCE	AUC
Question 1	logitreg	0.196	0.837
	rf	0.171	0.848
Question 2	logitreg	0.215	0.773
	rf	0.201	0.812

Table 7: Performance of the full models. MMCE at threshold 0.5 as well as AUC values.



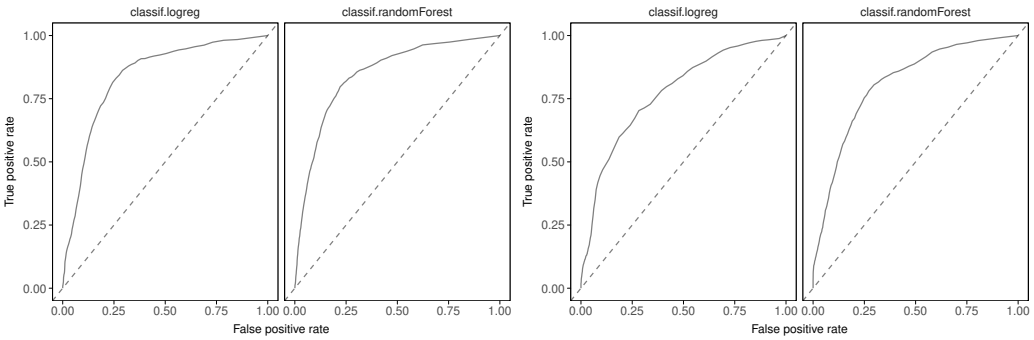
(a) Question 1 (b) Question 2

Figure 15: MMCE for all thresholds.

This time, the Random Forest performs better than the logistic regression question 1 and is outperformed for very few threshold values for question 2.

The dip below the dashed line (proportion of the smaller class) is much more noticeable than in the baseline models, though the MMCE values are still rather high.

Figure 16 shows the ROC curves for both full models, with corresponding AUC values in Table 7. Here, the Random Forest performs better than the logistic regression model, as well.

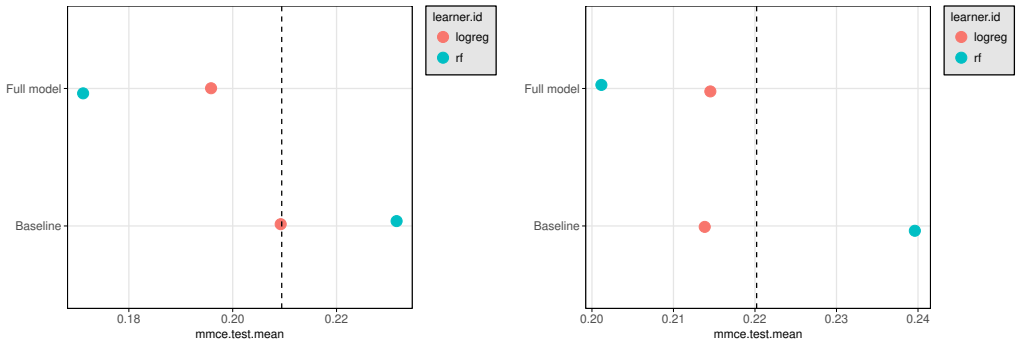


(a) Question 1 (b) Question 2

Figure 16: ROC Curves for both full models and questions.

5 Discussion

All in all, both models and both methods did not perform particularly well and were unable to make a clear distinction between mouse and touchpad trajectories. Using Figure 17, a comparison between logistic regression and Random Forest as well as between the baseline and full models can be made.



(a) Question 1 (b) Question 2

Figure 17: Comparison of the MMCE at threshold 0.5 for both questions and models. The dashed line shows the proportion of the smaller class.

For both questions, using a larger number of features determined by variable importance offered only a slight increase or no increase at all for the logistic regression model, while the change in performance for the Random Forest was noticeably larger. It can also be noted that the logistic regression model performed better than the Random Forest using the baseline features for both questions but was surpassed when using the features of the full model. Any MMCE values to the right of the dashed line can be improved by simply predicting the larger class (mouse) all of the time.

The poor performance might indicate several things. Firstly, there might not be a very large difference in the trajectories of touchpad and mouse users at all, making a differentiation between the two unnecessary in further analyses. On the other hand, the features used in this thesis might be insufficient to detect the differences between mouse and touchpad trajectories. Lastly, the classification methods used here might not be ideal for this specific task.

6 Outlook

The methods and models used in this thesis were unable to make a clear distinction between mouse and touchpad trajectories. This raises the question what else could be done to improve classification.

Other methods, such as boosting or support vector machines could be used to try and get a better classification performance. One could take an over-/undersampling approach to try and get an improvement by choosing an equal number of observations from each input devices for each sample. Hyperparameter tuning could be used to find the optimal hyperparameters for the Random Forest model.

Additionally, some new features could be computed. For example, the difference in length of the most direct path(s), that is, from the start to the radio button (for instance) and then to the submit button, to the actual path taken might provide some additional useful information.

Another option that might yield some additional information is to use functional features, for example velocity curves over time and include these in a functional classification approach.

Finally, the study could be conducted again under laboratory conditions, guaranteeing that all respondents are using the same resolution on the same computer screens with the same computer mouses and touchpads as well as guaranteeing correct information with regards to the input devices used.

7 Supplementary Material

All statistical analyses were performed using R 3.3.3 (R Core Team, 2016) and the R-Packages `mlr` 2.10 (Bischl et al., 2016), `mousetrap` 3.0.0 (Kieslich et al., 2017) as well as `mgcv` 1.8.17 (Wood, 2011), `randomForest` 4.6.12 (Liaw and Wiener, 2002) and many of the functions provided by "the `tidyverse`" (Wickham, 2017).

The supplementary material provided contains the data in `./data/`, some documentation in `./doc/` as well as the R code in `./code/` and all of the plots shown in this thesis can be found in `./img/`.

8 Bibliography

Bischl, B., Lang, M., Kotthoff, L., Schiffner, J., Richter, J., Studerus, E., Casalicchio, G., and Jones, Z. M. (2016). mlr: Machine learning in r. *Journal of Machine Learning Research*, 17(170):1–5.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Eilers, P. H. C. and Marx, B. D. (1996). Flexible smoothing with *b*-splines and penalties. *Statistical Science*, 11(2):89–102.

Fahrmeir, L., Kneib, T., and Lang, S. (2009). *Regression: Modelle, Methoden und Anwendungen*. Statistik und ihre Anwendungen. Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg. 2. Auflage.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Springer series in statistics. Springer, New York, 2 edition.

Horwitz, R., Brockhaus, S., Henninger, F., Keusch, F., Kieslich, P. J., Kreuter, F., and Schierholz, M. (2017). Learning from mouse movements: Improving questionnaire and respondents' user experience through passive data collection. (working paper).

Horwitz, R., Kreuter, F., and Conrad, F. (2016). Using mouse movements to predict web survey response difficulty. *Social Science Computer Review*.

Kieslich, P. J., Wulff, D. U., Henninger, F., and Haslbeck, J. M. B. (2017). mousetrap: Process and analyze mouse-tracking data.

Liaw, A. and Wiener, M. (2002). Classification and regression by randomforest. *R News*, 2(3):18–22.

R Core Team (2016). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Stieger, S. and Reips, U.-D. (2010). What are participants doing while filling in an online questionnaire: A paradata collection tool and an empirical study. *Computers in Human Behavior*, 26(6):1488–1495.

Wickham, H. (2017). *tidyverse: Easily Install and Load 'Tidyverse' Packages*. R package version 1.1.1.

Wood, S. N. (2006). *Generalized additive models: An introduction with R.* Texts in statistical science. Chapman & Hall/CRC, Boca Raton.

Wood, S. N. (2011). Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society (B)*, 73(1):3–36.

A Appendix

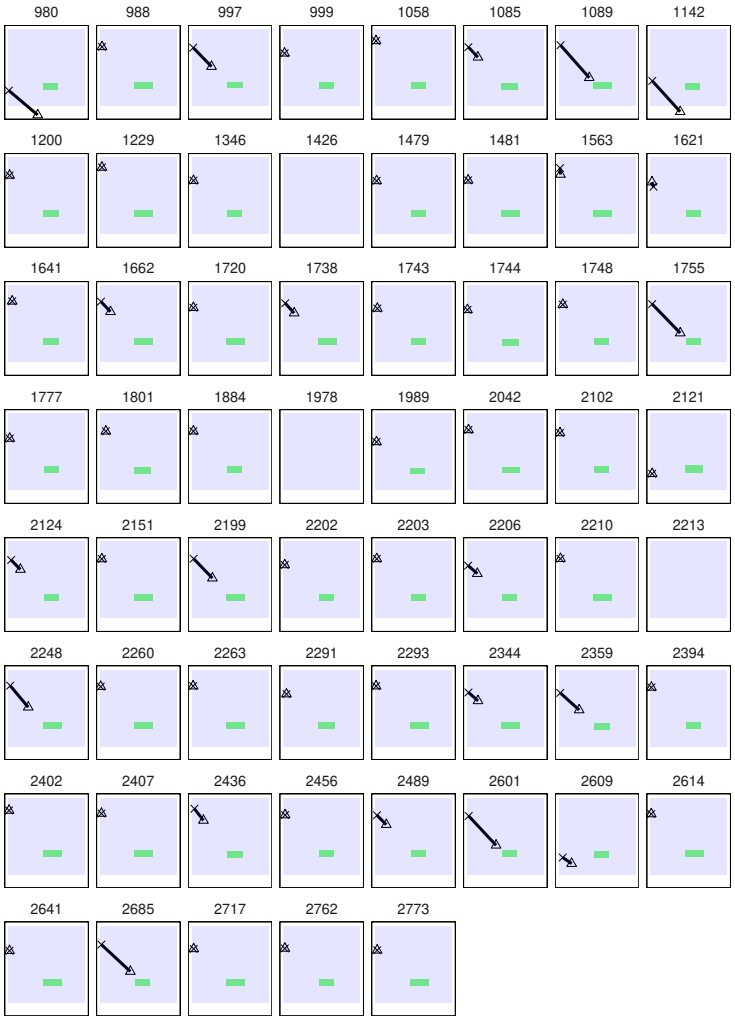


Figure A.1: Trajectories for question 1 with a total distance of less than 400 pixels

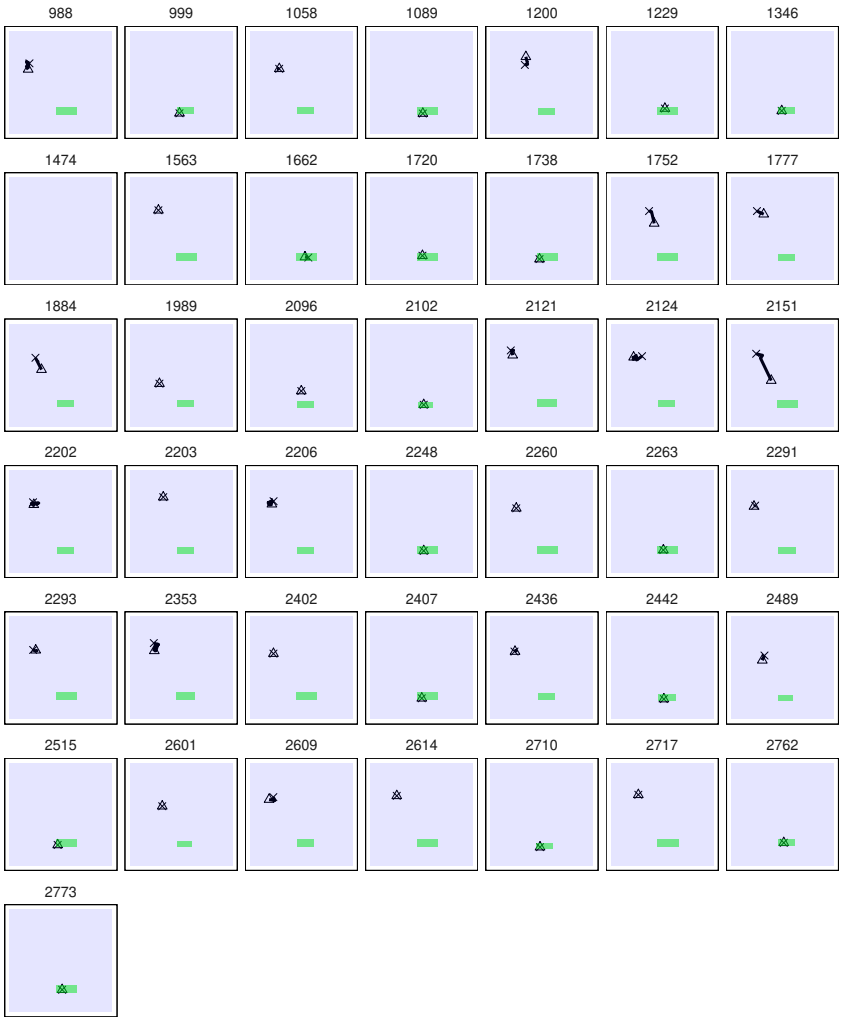


Figure A.2: Trajectories for question 2 with a total distance of less than 200 pixels

(a) Question 1 (b) Question 2

Figure A.3: Trajectories for respondents using a touchscreen for each of the two questions. The symbols $\times$ and Δ denote the starting and ending points, respectively. The blue area represents the form, while the green rectangle shows the submit button of the form.

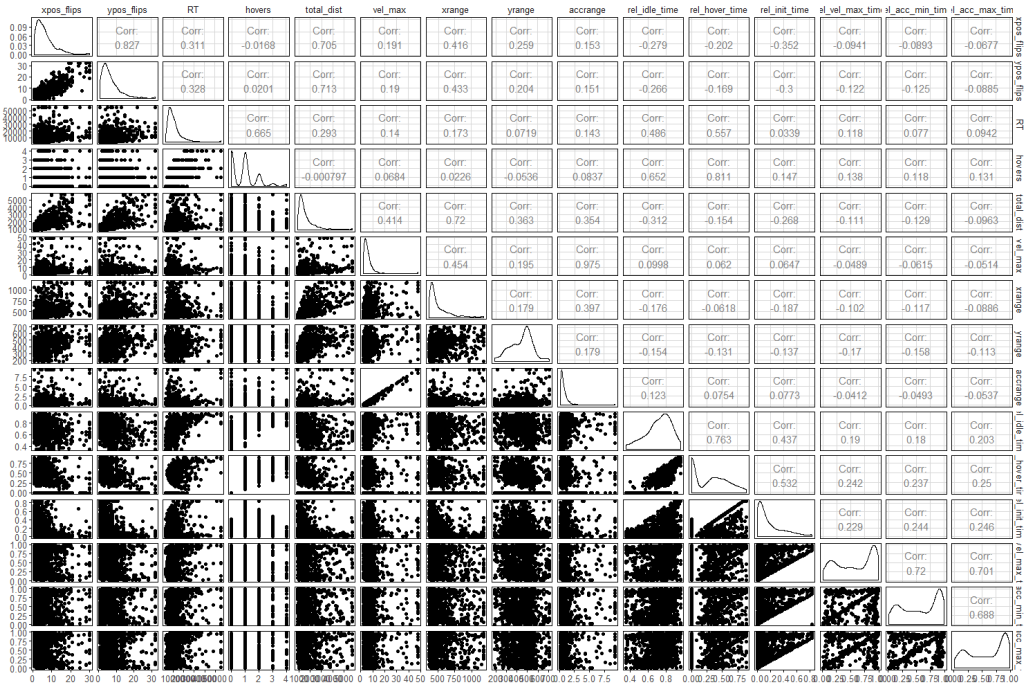


Figure A.4: Scatterplotmatrix for features of question 2. The upper triangle shows the correlations, the lower triangle shows the respective scatterplots. The main diagonal consists of density estimates.