# Mouse-Tracking a window intro decision making.

We have argued for mouse-tracking as an excellent tool for obtaining a rich, contextual and continuous measure for cognitive processes, avoiding averaging techniques used in the otherwise similar paradigm of eye-tracking (Spivey et al., 2005). Mouse-tracking analysis procedures are currently very ambiguous, lacking a supported clear analysis pipeline. Such ambiguity has emerged several diverging paths of mouse-tracking analysis, thereby weaking the entire field and making comparative analysis difficult (Schoemann et al., 2021). An attempt to standardize mouse-tracking analysis while still keeping the core flexibility that mouse-tracking contains was conducted by Maldonado et al., (2019). Maldonado’s et.al., (2019) contribution to the field of mouse-tracking analysis consist mainly of linear-discriminant-analysis (LDA) classifier trained on a forced-switch task simulating the expected mouse trajectory following a decision task with a two-step processing. By replicating Dale & Duran's, (2011) study on linguistic negation as two-step process they obtained data for testing their LDA classifier using [Area Under the Receiver Operating Characteristic curve](https://en.wikipedia.org/wiki/Receiver_operating_characteristic#Area_under_the_curve) [[1]](#footnote-1)(AUROC) as a metric for performance. We will be investigating the possibilities for applying their framework to a broader scope of the mouse-tracking literature by replicating a known experiment on dynamical processing using mouse-tracking ie. (Spivey et al., 2005) work on decision processing in phonological similarity. Followed up by a discussing advantage and disadvantages of LDA AUROC and First we will walk through the experimental setup and ideas behind (Maldonado et al., 2019), followed up by an introduction to our experimental idea and design.

## Calibration Data:

The first step of Maldonado et al., (2019) were to create a baseline for the LDA classifier to train on. An “ideal” mouse-trajectory a two-step processing decision task were simulated using a forced dual-choice switch task. The experiment is initiated with two possible answer boxes (blue) & (red) and a frame at the edge of the window with the same possible colors. The job of the participants is to repeatedly select the box with the matching color of the frame at the edge across the 88 trials. To induce an element that would simulate a two-step processing task twenty-four of the eighty-eight trials had a “switch” condition with the remaining being “straightforward” with no manipulation. “Switch” trials involved the frame changing color to the opposite, forcing participants to change mouse-trajectory, and thereby simulating a switch decision making. Point of change were dependent on the y-coordinate which varied between 40%, 70% and 90% of the way to the target on the y-axis allowing for the LDA classifier to have data with varying temporal aspects of the switched decision (Maldonado et al., 2019).

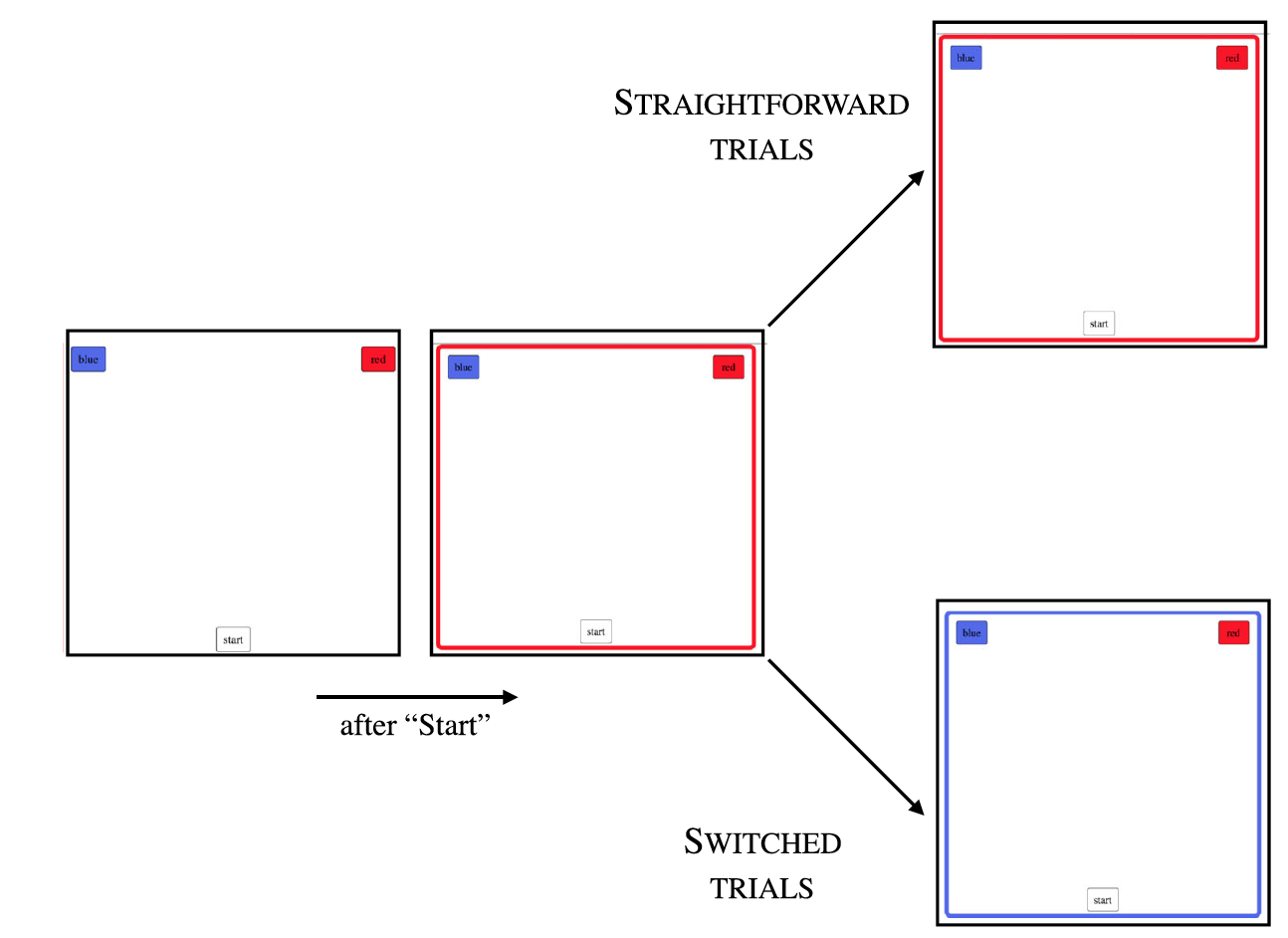


Figure 1 (Maldonado et al., 2019), Experimental design of the Calibration experiment.

### Training of LDA on calibration data.

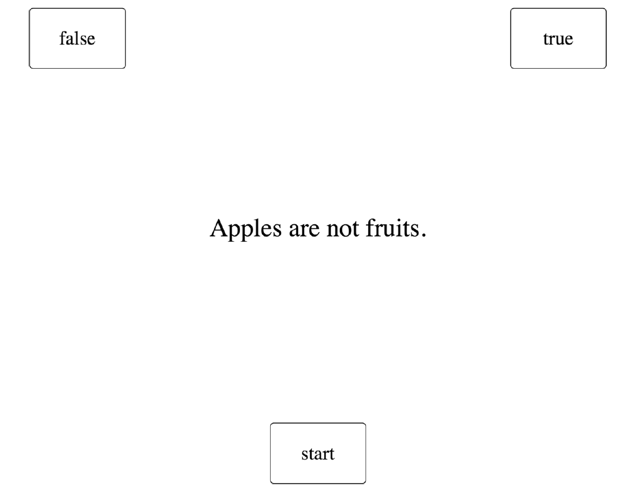
A pipeline for training the LDA classifier is as follows, **1)** Normalize and flip (X, Y) coordinates by setting start point = (0,0), end point = (1,1). Based on the spatio-temporal information calculate the non-linear Euclidean velocity and acceleration. Finally, time-normalize the data into 101 steps. **2)** Even though Maldonado et al., (2019) didn’t report it in their paper we ran through their analysis script and found them to standardize all features based on the Z-distribution with a mean = 0 and SD = 1 prior to the conducting of the principal component analysis (PCA) (*Importance of Feature Scaling*, u.å.). This is important since is gives equal weight to all features when calculating the covariance matrix used for PCA. The PCA removed any collinearity which otherwise would be an issue for any regression or classification model, leaving us with 13 PCA features to feed to the LDA classifier. 3) The 13 PCA features fed to the LDA classifier were based on a varying X matrix containing one of the combinations showed in table 1. An in-depth review of LDA and PCA will be take place later in the article. By comparing their classifiers with different PCA features extracted from the varying combination of predictors shown in table 1 were Maldonado et al., (2019) able to gather information on which mouse-features were the most important for classifying decision processes. A 10-fold cross-validation(cv) on all their models concluded that the most important feature for classifying “straight” vs. “switched” trial were the coordinates.

Table 1 (Maldonado et al., 2019)

Et billede, der indeholder bord

Automatisk genereret beskrivelse

## Extending the LDA classifier to negation data:

For testing whether the classifier is a useful tool for analyzing mouse-tracking data it must generalize beyond the “simulated” decision data in the forced switch task. To test the generalizability of their classifier Maldonado et al., (2019) had to test it on more ecological mouse-tracking data. They choose to replicate Dale & Duran, (2011) study on sentence negation with an hypothesize that if sentence negation followed the same two-step processing as in the forced calibration task the LDA performer would perform similarly. Participants were with a screen like figure 2. A true or false sentence would be presented in the middle of the screen either including or excluding a negation, i.e., “elephants are not small” or “elephants are not large” (Dale & Duran, 2011). The job of the participants was to evaluate the statement and select the corresponding false or true statement. Maldonado et al., (2019) managed to replicate the original studies finding of a significant increase in x-flips when the statement was truth and included a negation.

Having verified the replication of the original Dale & Duran, (2011) findings the LDA classifier could now be tested to evaluate its generalizability and advantages.

Figure 2, (Dale & Duran, 2011) Negation study.

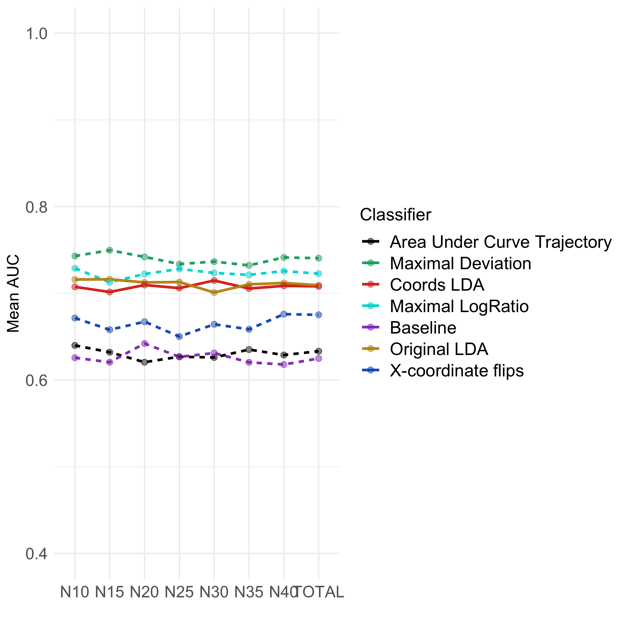
All data were subset to only include the true statements condition as the negation only influenced mouse trajectories in said condition. Wrong responses were also excluded. Remaining data were pre-processed in the same manner the calibration data allowing for a trained model on the calibration data to be tested on a similar feature matrix in the test data. Bootstrapping was conducted on various sample sizes which was then used for testing the LDA. The results showed a similar pattern to that of the calibration. Coordinates were the most predominant predictor of those included in the LDA classifier following AUROC as the metric for performance, Maximal Deviation (MD) and Maximal LogRatio both performed slightly better than the LDA classifier. However, Maldonado et al., states, *“The classifier is still a better choice from a conceptual point of view, since it does not make any specific assumptions about how the change of decision should be reflected by mouse trajectories beyond the observed ones.”* In the following section an argumentation for and against this statement will be presented.

Figure 3, (Maldonado et al., 2019) Results of CV LDA performance on different bootstrap samples.

# LDA & PCA/ (methodology of Maldonado et al.,

LDA has originally been thought of as a feature extractor through dimensionality reduction. Feature extraction is especially useful when working with a high dimensionality data set such allowing for the same information to be conveyed in lower sub dimensional space which maximizes separability, where Separability is defined in terms of deviation in statistical measures of mean value and variance. The technique has been employed in a broad range of research areas i.e. EEG (Subasi & Ismail Gursoy, 2010), face-recognition (Jin et al., 2001), medical bioinformation recognition (El-Feghi et al., 2004). A more recent employment has been to use LDA as a classifier by testing the separability of different classes mean and variance on the new LDA subspace. While this technique is not novel (Kim et al., 2007) or pioneered by Maldonado et al., (2019) it is novel to the field of mouse-tracking. By doing so does Maldonado et al., (2019) introduce a machine-learning approach for evaluating mouse data, and testing hypothesis. As highlighted in an earlier section the predominant mouse-tracking methodology has largely been occupied by multiple t-test and regular hypothesis testing on a varying selection of features. Some of the most common being t-test of x-flips, MD, AUC, Maximal LogRatio and X-coordinates in specific time-intervals. Regardless of research field of inquiry, be it linguistic negation or phonological similarity, the trend has been to look for significant difference in MD, AUC etc. between conditions. A significant difference in means would then support the hypothesis of the experimental manipulation having an effect. There is two main issues with such an approach which both can be summed up the a lack of reproducibility and best-practice methodology. 1) One of the fallacies of ordinary least squares is the dichotomous thinking of significance testing (Dienes, 2008). Just because a test is significant is not the same as the magnitude of difference being larger. One of the most common misinterpretations is that smaller p-values equals a larger effect which also leads to increased focus on p-values. An un-scientific tradition of p-hacking[[2]](#footnote-2) has started to emerge across scientific-fields (Head et al., 2015) as an off-spring from the increased focus on p-values has. One of the major assumptions of hypothesis testing is that the hypothesis and planning of statistical test must take place prior to any collection or investigation of data. Even though such an assumption is difficult to control for, is there specific fields with larger way room when it comes selecting their analysis. Making it even more important in those fields to state specific hypothesis and plans for testing prior to data investigation, which leads us to the second issue in mouse-tracking analysis.

2) One of the main advantages of mouse-tracking is its flexibility, as simulated in the calibration experiment temporal patterns can vary from experiment to experiment, the temporal and spatial aspects when the trajectory changes can vary. The change can be early or late both temporally and spatially, with either a smooth change (dynamical) or sharp and edgy (dual system). Different differences result in changing patterns. One experimental condition could influence AUC greatly but not x-flips and vice-versa. Mouse-tracking analysis can accommodate these differences by comparing different measures, MD, AUC, Maximal LogRatio, X-flips or temporal differing X-coordinates but such flexibility also results in varying analysis and a lot of options of p-hacking. By introducing a machine-learning-like analysis apparatus for mouse-tracking will we reduce the options for selectively only comparing x-coordinates slices or other mouse-tracking measures that are different in a specific experimental setup. Rather by creating a “standardized” model trained on simulated data and tested on experimental-specific data will we be able to avoid dichotomous thinking and p-hacking while also introducing an easy way to compare results across papers.

Maldonado et al., (2019) has chosen LDA as for classification with PCA for feature extraction prior to model training and testing. Combinatory effect dubbed PCA plus LDA, has been test and validated in multiple experimental settings (Liu & Wechsler, 2000)(Yu & Yang, 2001)(Belhumeur et al., 1997). High dimensional data is computationally expensive for LDA, a PCA reduction will therefore improve the computationally expensive of LDA, but the question is how much information is lost in process. This is dependent on multiple things, among others the proportional amount of PCA n-components being used compared to the total available. Yang & Yang, (2003) asked the questions *“Why select PCA for dimensionality reduction beforehand? Is there any important discriminatory information lost in the PCA process since the criterion of PCA is not identical to that of LDA?*” (Yang & Yang, 2003). They concluded that stacking PCA upon LDA did not result in any additional significant information loss. To sum up, there is support in statistical academia both towards moving from hypothesis testing to a machine-learning framework and for the application of PCA plus LDA. But currently has the Maldonado et al., (2019) framework only been tested on mouse-tracking data from a linguistic negation experiment. To implement such framework in the broader field of mouse-tracking the model must generalize to other areas of research and be robust against changing temporal and spatial mouse-patterns.

# Hypothesis/ continued work:

1. To evaluate if the LDA successfully generalize it must perform close to the topline and well above chance following AUROC measure of success.
   1. Perform better than traditional mouse-tracking measures.
2. Is 13 PCA features the optimal amount.

# Design:

# Methods & Results:

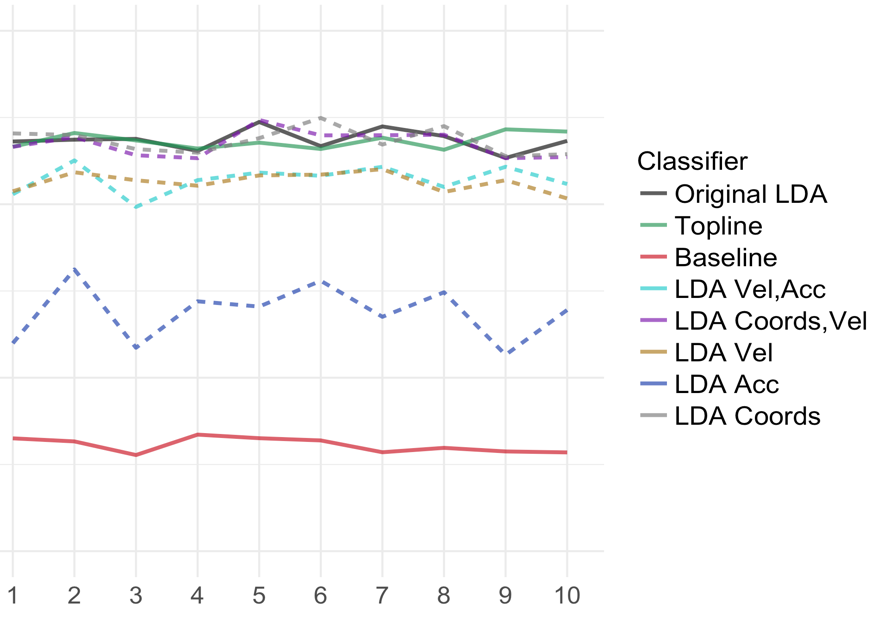
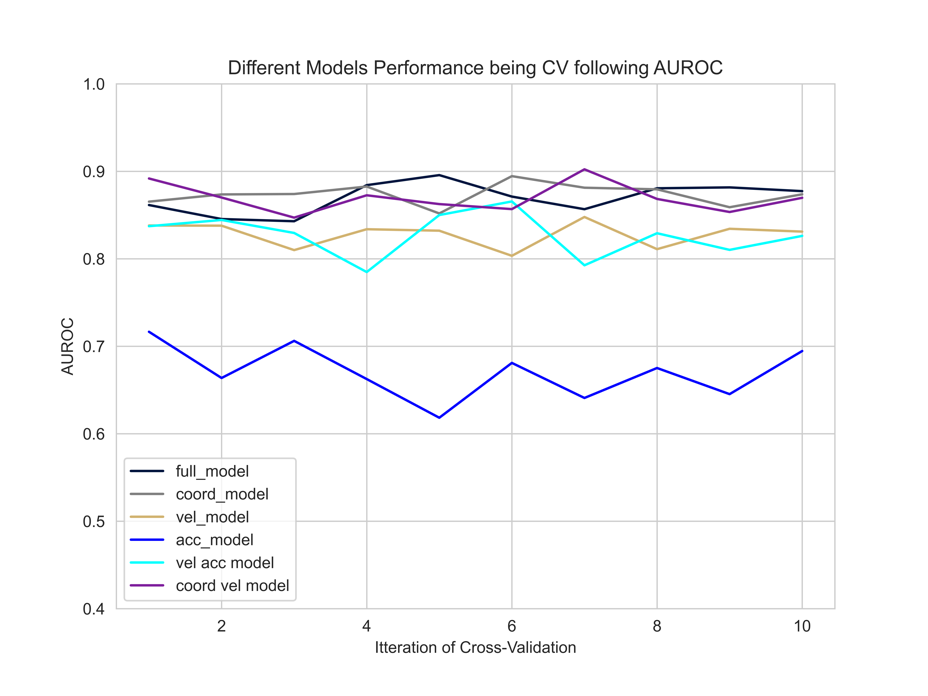
Maldonado et al., have conducted their data analysis in R-studio (RStudio Team, 2020) and posted all R scripts[[3]](#footnote-3) and data for others to replicate. Due to out-of-date R libraries and a few missing self-made functions the R script could not fully run. We conducted our data preprocessing and analysis partly in R-studio and Python (Rossum et al., 2009) using VS code as our integrated development environment (IDE). By replicating Maldonado et al., (2019) results using VS code we ensured that changing programming language and IDE would not be an interfering factor for further testing of their framework. We followed the preprocessing pipeline as highlighted in the earlier section on calibration and validation. Six different subsets of data was created from original calibration data provided by Maldonado et al., (2019), 1) Containing all features, 2) Coordinates, 3) Velocity, 4) Acceleration, 5) Velocity & Acceleration, 6) Coordinates and Velocity. All subsets were split into 10 splits using stratified k-fold with shuffle. AUROC cross-validation was performed to test the LDA’s classifier performance on the different subsets with varying features. Across all subsets were feature columns standardized based on the z-score distribution from the train portion (9/10 splits) using the StandardScaler() function in sci-kit. Followed by a PCA() to reduce our standardized data to thirteen dimensions again fitted and transformed on the train split while the remaining test split was only transformed to avoid data leakage between test and training splits (Buitinck et al., 2013). For all six different subsets an AUROC score was computed for when each of the ten splits was being used as the test split. The resulting AUROC scores were plotted and compared to the findings of Maldonado et al., (2019). Maldonado et al., (2019) didn’t use a specific seed for their random generator so there will be slight differences in the splitting and shuffling of the ten splits. Slightly different results are to be expected due differences in splits so what we’re looking for is the general performance across all the splits and a comparison between the different LDA models. Figure 4 and figure 5 shows the same tendencies, so we can finally conclude that conducting the data modelling work in python and VS code will not be a cause for conflict or different results compared to working with the incomplete script provided by Maldonado et al., **(2019).

Figure 4, Our replication of the CV on calibration data.

Figure 5, (Maldonado et al., 2019) original cross-validated graph.

## Data treatment:

Data from our replication of (Spivey et al., 2005) were preprocessed in R studio (RStudio Team, 2020) using the mousetrap package (Kieslich & Henninger, 2017). Incorrect responses and training trials were excluded. For the pre-trained LDA being able to predict “straight/control” vs “switched/cohort” our data representation must be an exact match; this means replicating every step of the preprocessing. So, all X-coordinates were flipped and normalized mapping the start point onto (0,0) and the center of the target pictures onto (1,1), all mouse-coordinates prior to initiation of mouse-movement were excluded. Different response time results in a varying number of mouse-coordinate data points per trial. We therefore time normalized every trial into 101 proportionally equal time steps including the first and last data point. To proceeded to next trial, participants had to click anywhere within the boundaries of the target pictures. Close to non-information is retained in varying endpoints within the target stimuli, what we care about is really what happened on the way to the picture i.e., the trajectory. “*Different decisions (i.e., decision patterns) have a different impacts on mouse trajectories”* (Maldonado et al., 2019). So, all endpoints were aligned to have the exact same starting and endpoint within target. Allowing for a better comparison of trajectories. Data were subset into 6 different categories like those seen in figure 4.

## Extension of the LDA classifier:

Having matched our mouse-tracking data structure to that of Maldonado et al., we were able to apply the exact same framework to our Spivey et al., (2005) replication as Maldonado et al., (2019) used for testing the LDAs extension to linguistic negation data. This involved training the LDA classifier repeatedly on different subsets (Coordinates, Velocity etc.) of the preprocessed calibration data and testing it on matching Spivey et al., (2005) replicated data. LDA was used as a supervised classifier projecting all input features onto a k-1 space (k = n levels of the dependent variable). Both Spivey et al., and Maldonado et al., were dealing with a binary decision task cohort/switched or control/straight, so k = 2. In our case the output of the LDA will be a 1-dimensional space with one single scalar value per observation/trial resulting in the density distribution shown in **SEE FIGURE XX**. If the two density distributions, cohort, and control, are perfectly separable in the right direction it will be equal to having an AUROC of 1. Our LDA classifier will be fitted on data from two-step forced switch calibration experiment (Maldonado et al., 2019) and fitted/tested on our data. Maldonado et al., selected the two top performing CV models on calibration data shown in figure 5 for testing on the linguistic negation data. We’ve continued with the same two models (full model and coordinate model) for evaluating if the framework can meaningfully be applicable to other areas of research i.e., phonological similarity.

Cross-validation is not possible in framework where training and testing data has different sources of origin. Instead, we choose to bootstrap 2000 samples based on varying subsets of participants in our study. Bootstrapping works as a good alternative to CV and allows us to make better inference about our data when the sample size is small (STINE, 1989). The random seed were set to 10 for all pseudo random processes.

Topline:

A topline which represents the best possible fit on the bootstrapped data were found. Data were standardized based on its own mean and variance and the 13 PCA features found based on the data’s own eigen- values and -vectors. An LDA classifier was fitted, and the decision function found based on the same data used to fit. Based on the values in the decision function and true labels (Cohort = 1, Control = 0) an ROC curve were computed using the roc\_curve() function from scikit-learn (Buitinck et al., 2013). Finally, the model’s performance was evaluated based on the AUROC across all different bootstraps.

Baseline:

A baseline was created by simulating a completely stochastic classifier. The array containing true labels were copied and shuffled randomly using seed = 10. An ROC was found using the shuffled and true labels which were used to compute an AUROC score for a completely stochastic process. The procedure was repeated for all bootstraps.

LDA:

All LDA models were trained on the calibration data and tested on our data. Both calibration data and our data matrices were standardized based on the mean and variance of the calibration data. Similarly, was the 13 PCA features in both data sets constructed upon the eigenvalues and eigenvectors found by using the PCA.fit() function from scikit (Buitinck et al., 2013) on the standardized calibration data. We used the trained LDA model and found the decision function for the phonological similarity data. The decision function values, and true labels was used to compute the ROC with its respective AUC score. This was done for all bootstraps.

Traditional Mouse-Tracking Features:

To evaluate the classification performance of traditional mouse-tracking features an ROC was found for each of the following feature’s AUC, MAD, X-position flips, and X-position reversals. All features were standardized with a mean = 0 and std = 1. The features form two 1-dimensional density curves like that of the LDA, one for cohort and one for control. If the density distributions are perfectly separable in the right direction the classifying will be optimal and result in an AUROC score of 1. The ROC was found for each feature and the AUC calculated again using scikit (Buitinck et al., 2013).

# Report results from analysis:

# Discussion:

## Discussion of results:

* Comparing topline & baseline other traditional mouse-tracking measures to our LDA performance.
* Performance issues why?
* Implications of results for the framework.
  + LDA performs like trad. Mouse-tracking but Maldonado argues LDA Is better. Do we agree? Why, why not?
* Discussing their general methodology and choices.
  + LDA as the model choice.
  + 13 principal components.
* Our choice of mouse-experimental design (OUR FUCK UPS WHY??). **(NIELS)**
  + Mouse sensitivity settings.
  + Mouse acceleration.
  + Dynamical vs static start procedure.
* Dual-system vs dynamical.
  + Different kinds of data.

1. For information on AUROC https://towardsdatascience. com/understanding-auc-roc-curve-68b2303cc9c5. [↑](#footnote-ref-1)
2. For a quick explanation of p-hacking see [this](https://scienceinthenewsroom.org/resources/statistical-p-hacking-explained/). [↑](#footnote-ref-2)
3. [Link to Maldonado et al., R-scripts and data.](https://osf.io/rbx3m/?view_only=7d557aa8931c4a0886e7ce2442a77895) [↑](#footnote-ref-3)