Predicting a mouse's behavior in an unpredictable environment

Shay Neufeld
Gil Mandelbaum

December 9th, 2016

how do mice make choices?

exploiting
relying on what you already now

↑ ↓

exploring
an attempt to gain new information

you are at a restaurant, staring at the menu.

How do you decide what to get? Do you pick your favorite dish - one you know will bring satisfaction? Or do you decide to take a risk and try something new? maybe you will discover your new favorite food!

this is the multi arm bandit problem

a scenario first formalized in probability theory. It illustrates the concept of balancing attempts to acquire new information about the world with the desire to optimize ones actions based on existing knowledge.

mult arm bandit task

a gambler in front of a row of slot machines (often aptly referred to as 'one-armed bandits') must decide which machines to play, and in what order, in an effort to maximize earnings over a set period of time.

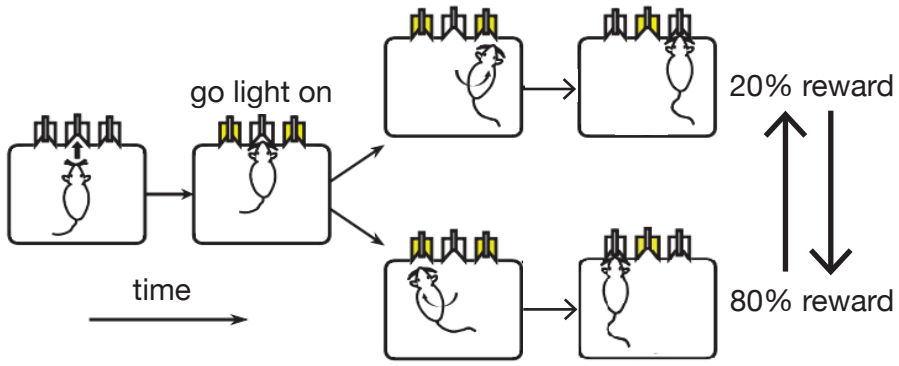
two-armed bandit task adapted for mice

nose ports instead of slot machines and water rewards instead of money prize

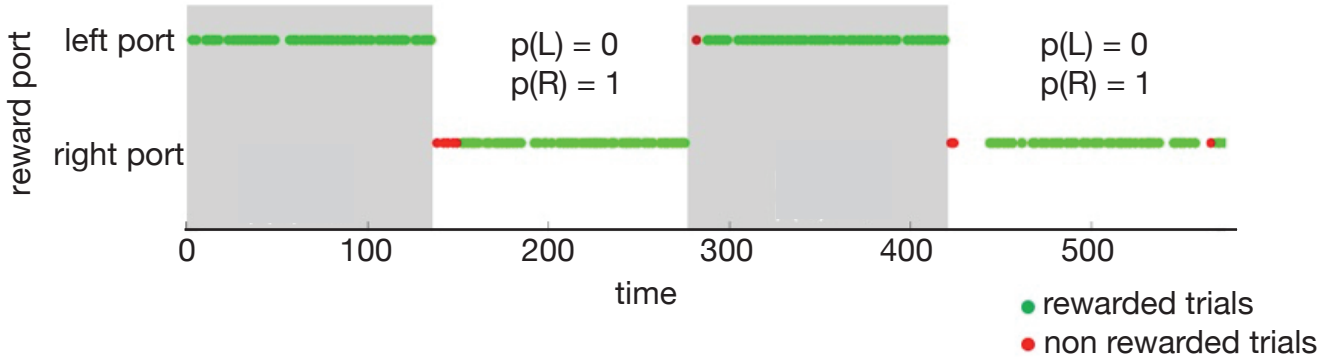
mouse two-armed bandit task rules

to initiates a trial: nose poke in a center port.
to make a choice: move to and poke in a left or right port to possibly receive reward.
get a water reward? reward will be giving probabilistically on the left or right in a block structure.
For example, 80% of pokes on the left and 20% of those on the right will be rewarded.

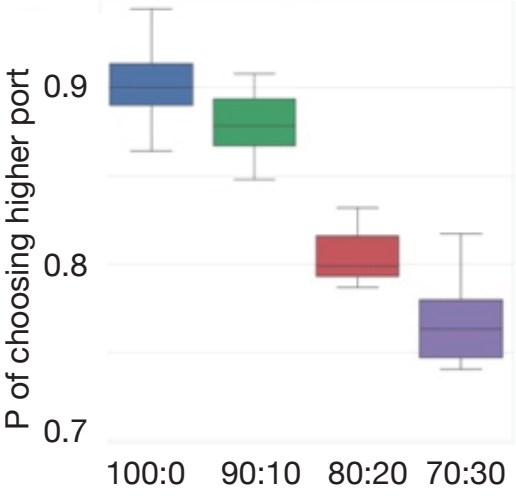
task schematic



example session

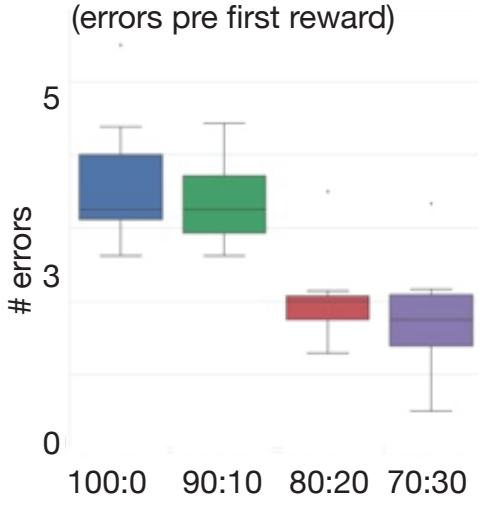


accuracy



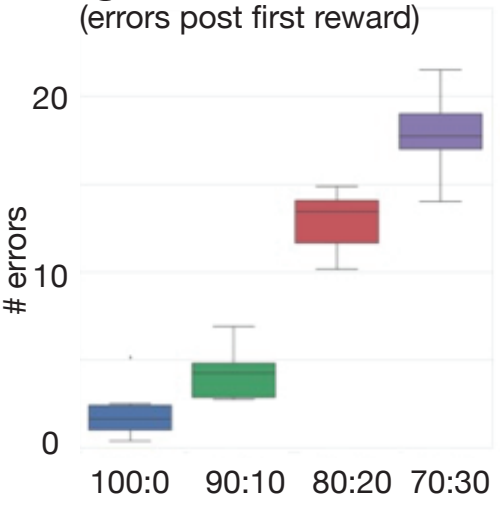
perseverative errors

(errors pre first reward)



regressive errors

(errors post first reward)



data processing

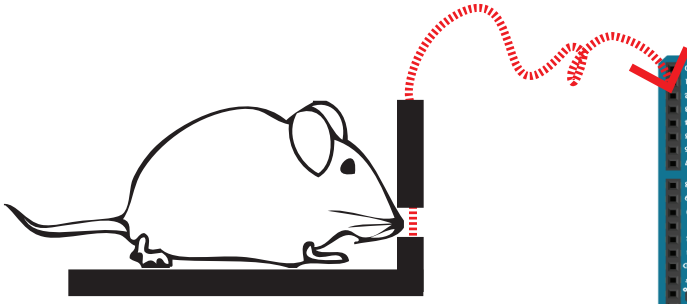
when does the mouse ‘change its mind’ ?

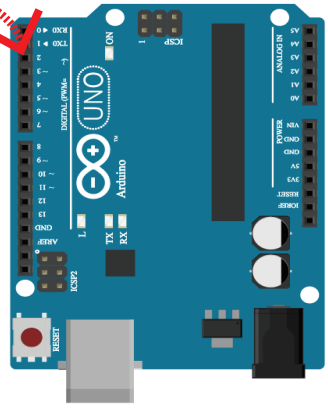
when it switches.

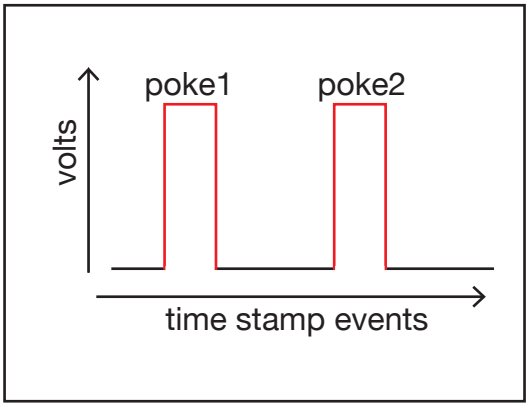
At some point, the mouse accumulates enough evidence (from the reward outcomes) that perhaps things have changed and it should change its behavior. It’s precisely that point that we want to try and model.

what is the mouse is likely to remember and use when making a decision?
past behaviors, past decisions, and past outcomes.
The...

behavioral system run by an arduino







nose pokes are detecting using an IR beam breaker which then sends a 5 volt pulse to the arduino.

raw signal to trial matrix (matlab)

A custom written script that results an csv file imported to python as a pandas dataframe

time from start(s)	to initiate trial (s)	decision time (s)	port selection	(P) reward left port	(P) reward right	port reward delivery
22.045	2.341	0.501	2	0.8	0.2	1

trial matrix to feature matrix (python)

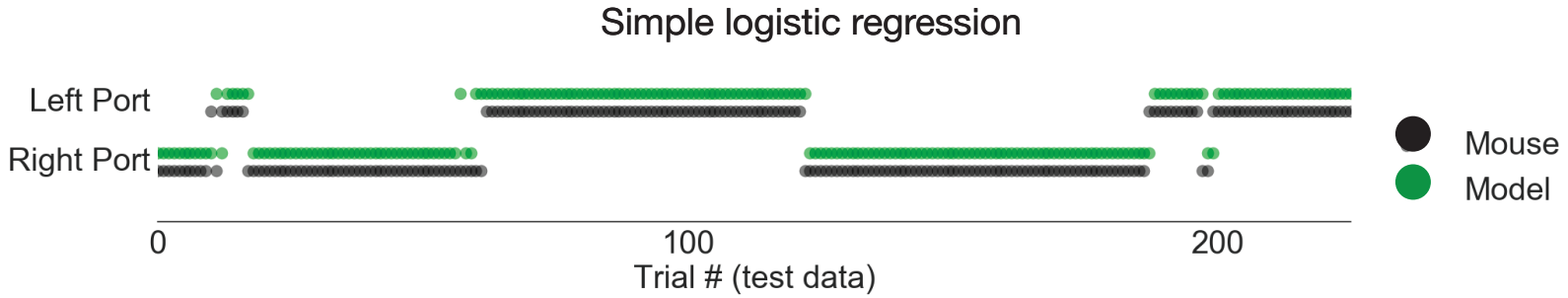
The feature matrix was inspired from the idea that we generally remember things that happened most recently with the most amount of detail, but tend to ‘average together’ events farther in the past to provide a more general ‘feeling’ rather than detailed account of events.

for 5 trials back:

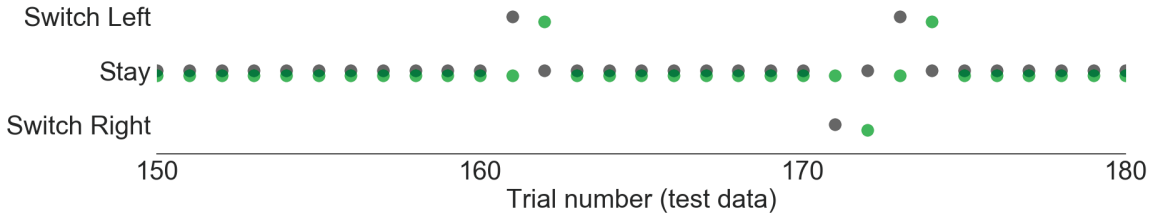
port chosen	rewarded or not	inter-trial interval	decision time
2	1	2.675	0.355

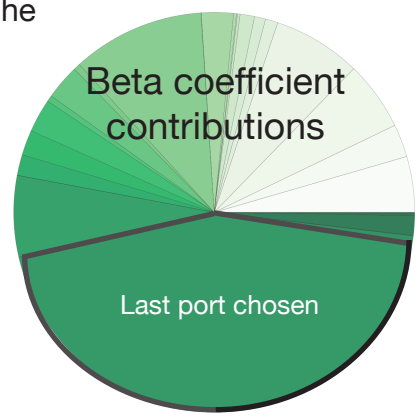
left choice last 10 trials	right choice last 10 trials	rewarded left last 10 trials	rewarded right last 10 trials	current reward streak
6	4	6	1	3

Classifying ‘left’ & ‘right’ decisions



This simple model is ~94% accurate at predicting which port the mouse will choose! However, has it really figured out how the mouse is behaving? The interesting behavior occurs when the mouse *changes* its decision. How does our simple model perform in those trials?

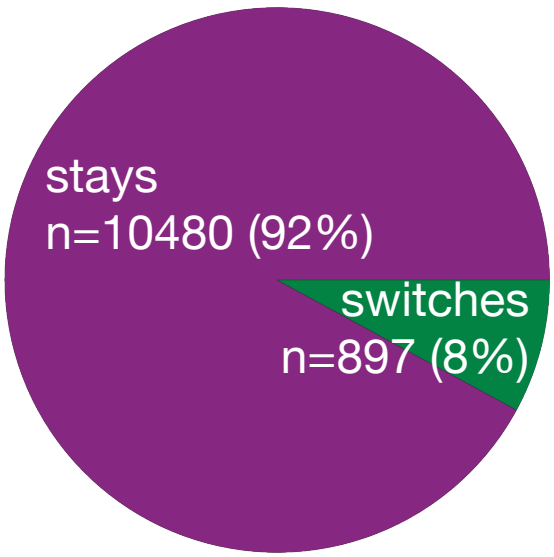




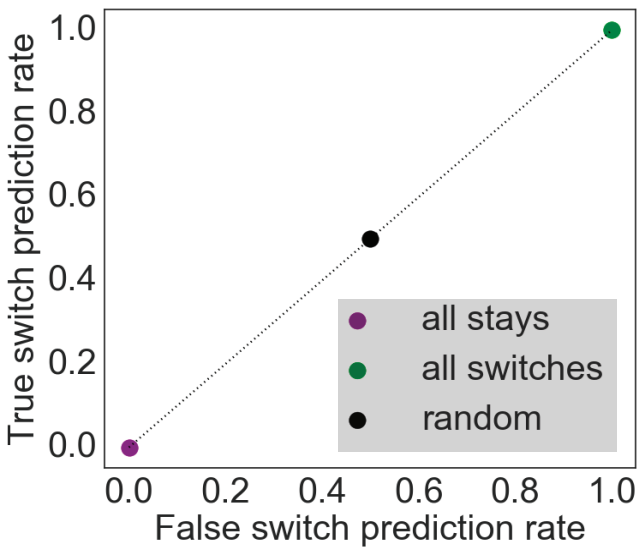
The model consistently lags the actual ‘switch’ trials. This is because the model is relying most heavily on whichever port the mouse chose previously to predict the decision. Therefore, we decided to instead directly model ‘switch’ vs ‘stay’ trials.

‘switch’ (explore) & ‘stay’ (exploit) decisions

distribution of ‘stays’ & ‘switches’ in the data



simplest bench-mark models to beat

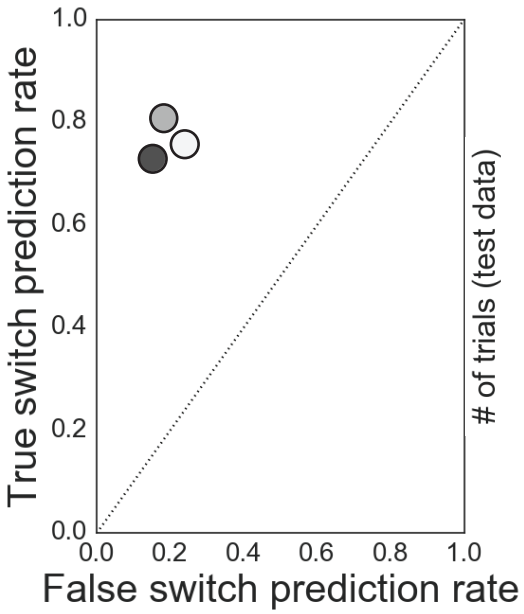


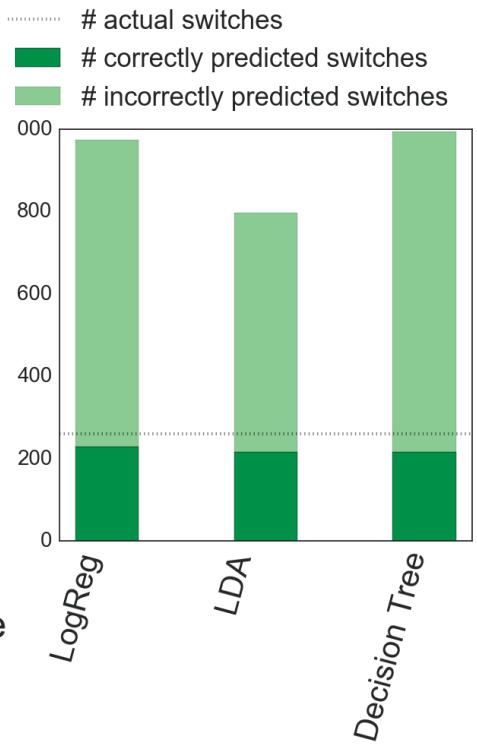
Classifying ‘switch’ & ‘stay’ decisions

Logistic Regression
Regularization : 10e-4
penalty : l2
Class Weight : balanced

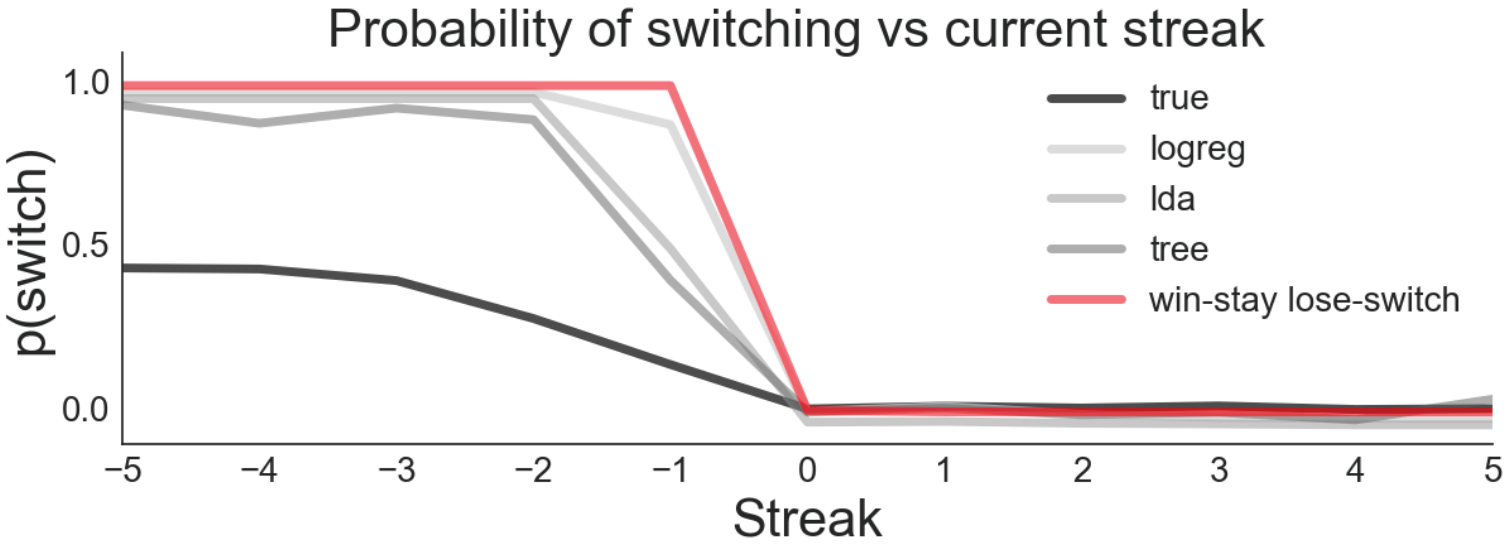
Linear Discriminant Analysis
Priors : {Stay 0.6, Switch 04}
penalty : l2

Decision Tree Classifier
Depth : 5
Class Weight : balanced
criterion : gini

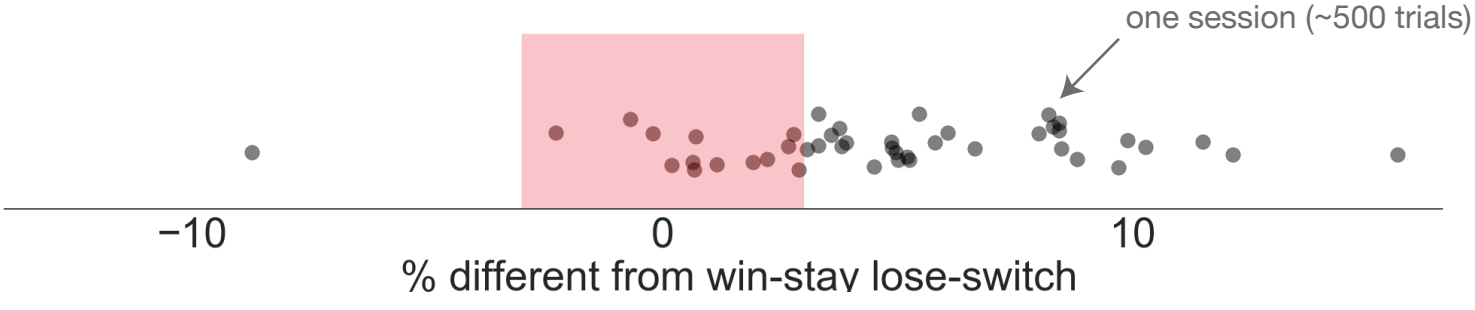




Although these models are ~80% accurate at predicting both switches and stays, the switching accuracy is achieved by *drastically over-predicting the number of switches.* This presents a severe limitation to using model to interpret the behavior (e.g. by studying the beta coefficients). What is causing the models to predict switch when the mice decide to stay?



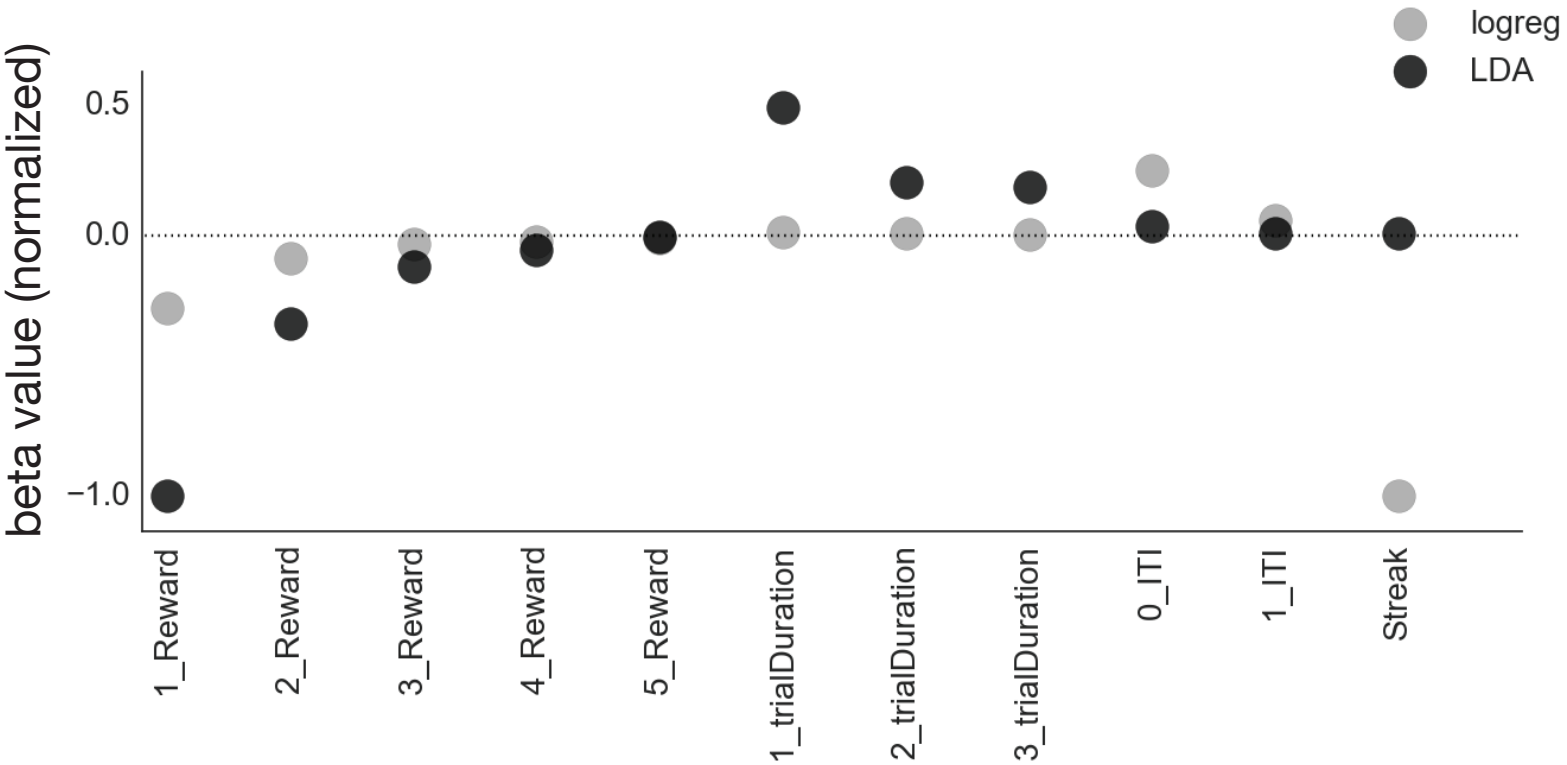
Above we can see that our models are essentially employing ‘win-stay, lose-switch’ strategy. However, in the actual data, mice only switch ~20% of the time following a single ‘no-reward’ outcome. Nonetheless, this result inspired us to take a closer look at the experimental data:



Interpreting the the model coefficients

Not surprisingly, all 3 models heavily depending on the previous reward outcome to predict a switch. Logistic regression fit a high beta to the 'streak' feature, whereas LDA did not. Conversely, LDA appeared to weigh rewards farther in the past more so than the logistic regression. In addition, the LDA used the 3 most recent 'trial durations' (a measurement of how fast the mouse performed the decision) - specifically, slower trial durations predicted switches (we have some hand-wavey intuition for this).

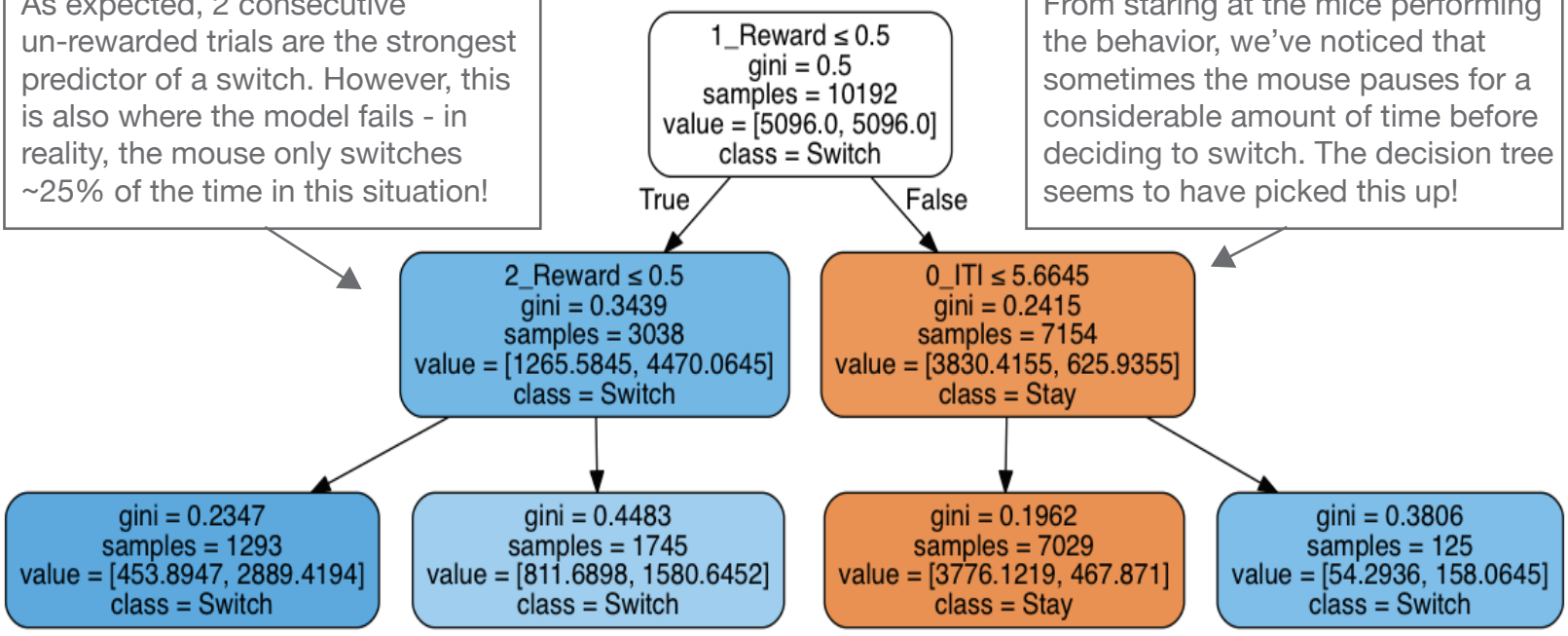
The models are clearly not performing with enough accuracy to make any claim that they are capturing the true strategy of the mouse. Nonetheless, these betas both confirm some of our own assumptions and offer a few interesting insights into how far in the past reward outcomes 'matter' when making a decision, as well as how the previous timing behavior of the mouse might be used to predict future decisions.



Decision Tree

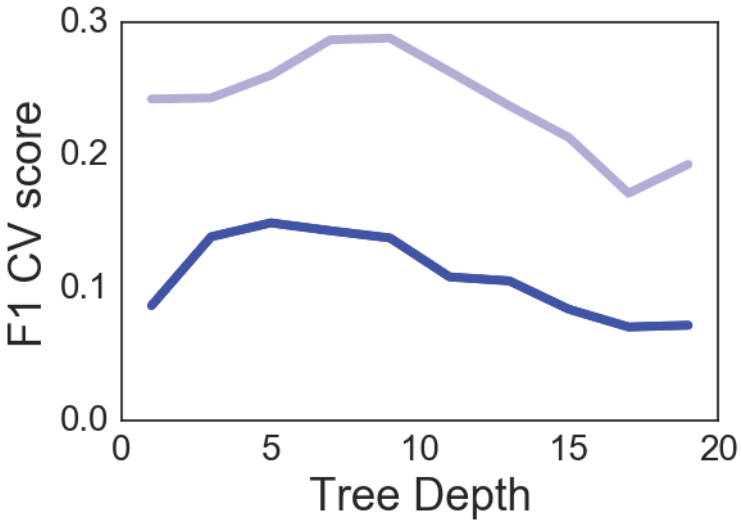
As expected, 2 consecutive un-rewarded trials are the strongest predictor of a switch. However, this is also where the model fails - in reality, the mouse only switches ~25% of the time in this situation!

From staring at the mice performing the behavior, we've noticed that sometimes the mouse pauses for a considerable amount of time before deciding to switch. The decision tree seems to have picked this up!

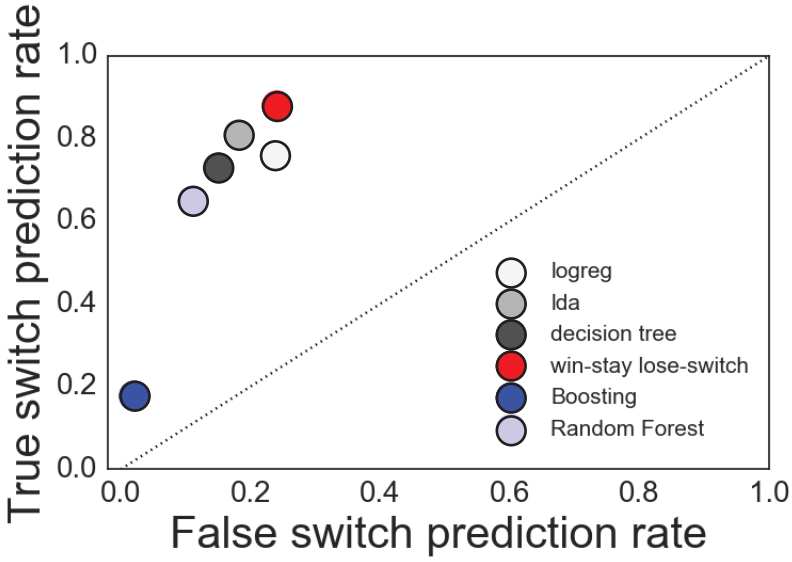


Can we achieve more accuracy at the expense of interpretability?

Tuning Boosting Classifier & Random Forest



Compare to previous models



Brief overview of other attempts...

Train a model to predict ‘switches’ specifically on trials where the previous reward is 0

Boost using all combinations of 1 and 2-depth decision trees.

Model using neural network (multi-layer perceptron)

Briefly explored using a Gaussian Mixture Model (results were extremely similar to LDA)

Created an additional feature ‘fraction of last 5 trials rewarded’

Calculated p(switch) for all possible outcome permuations of last 3 trials, fed in as features

Train and test only on a single mouse

Expand data to include less well performing mice (and increase ‘n’ to ~25,000)

Conclusions

Mice tend to irrationally ‘stay’ even during building evidence that the probabilities have changed. Our models were not able to capture this complicated behavior.

Ensemble methods (like Boosting and Random Forests) did not improve the accuracy of predicting switches.

While the classification models here have limitations, they still offered some insight into the information the animal could be using to make a decision:

- past reward outcomes (decreasing in value over time), which has been demonstrated previously in the field (both in mice and other animals during reinforcement learning tasks)

- the models also indicated that the timing of previous behavior carries some predictive power in determining a future decision.

Future directions

Are there other features we could include that carry predictive information?

Train models during learning, test on expert behavior. Similarly, train models during different timepoints of learning and compare beta coefficients.

Train models on different behavior paradigms (e.g. different reward probability pairs). Do mice adopt a general strategy in this task, or do they adapt to specific parameters.

Train models on different mice - do different mice learn to use different information to decide when to switch? Again, can train & test on different mice, or train on different mice and compare beta coefficients.