Selecting Impactful Product Features

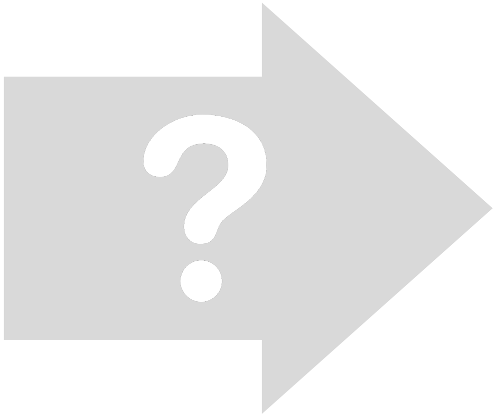
using statistics and machine learning

Ștefan Niculae

CONTENTS

PROBLEM STATEMENT

an application



make it better

PROBLEM STATEMENT

an application

?

make it better

make it more successful

PROBLEM STATEMENT

an application

?

make it better

make it more successful

increase the number of customers

PROBLEM STATEMENT

an application ?

make it better

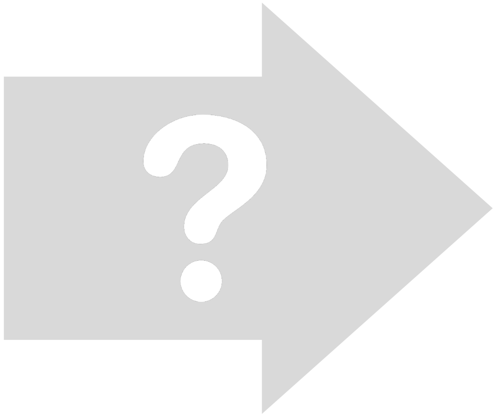
make it more successful

increase the number of customers

increase retention

PROBLEM STATEMENT

an application



make it better

make it more successful

increase the number of customers

increase retention

features that impact retention the most

PROBLEM STATEMENT

an application

usage of an application

?

make it better

make it more successful

increase the number of customers

increase retention

features that impact retention the most

PROBLEM STATEMENT

an application

usage of an application

user logs for an application

?

make it better

make it more successful

increase the number of customers

increase retention

features that impact retention the most

PROBLEM STATEMENT

an application

usage of an application

user logs for an application

make it better

make it more successful

increase the number of customers

increase retention

features that impact retention the most

Machine learning task!

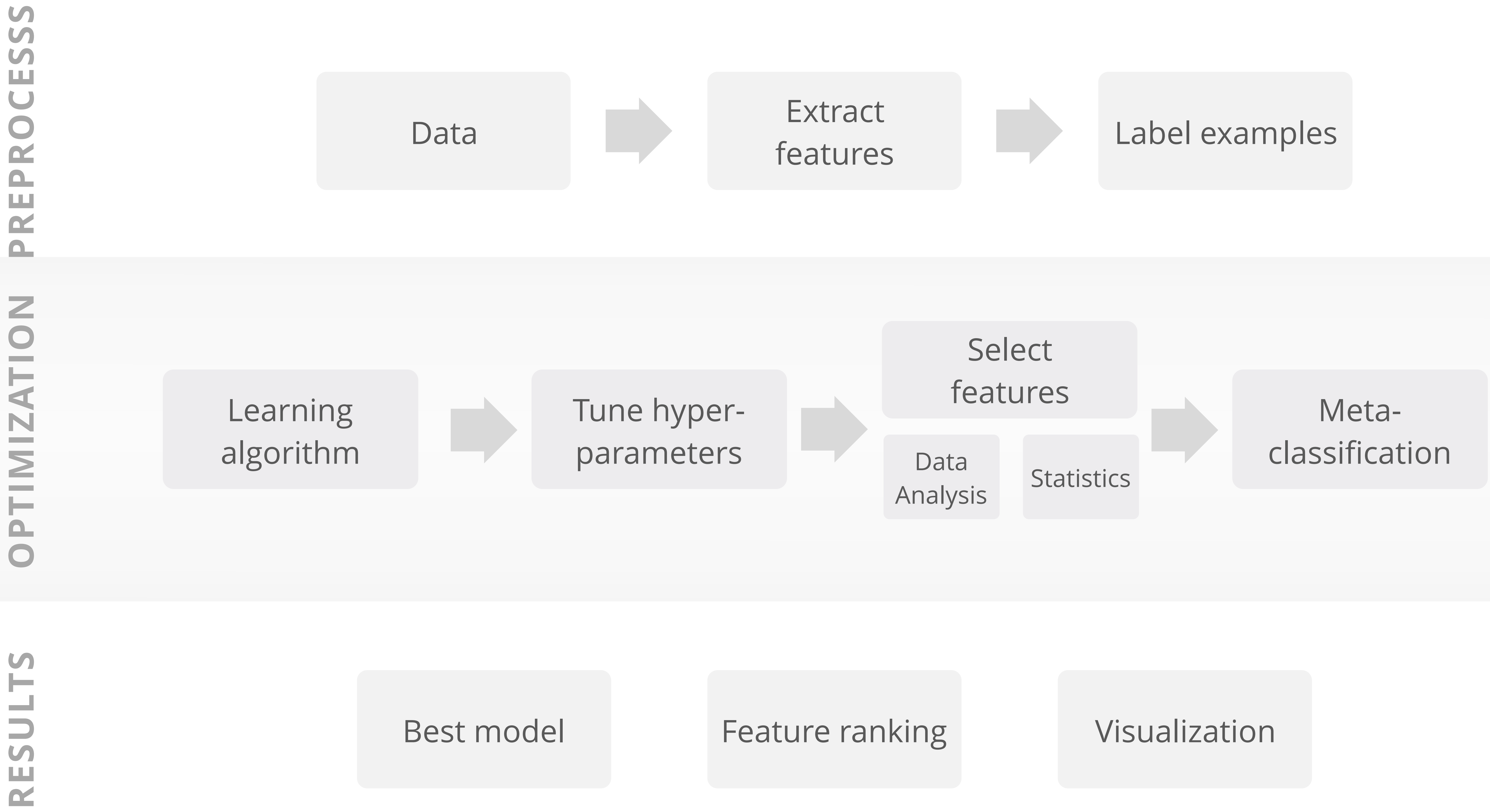
MACHINE LEARNING

train on some labeled examples,
then predict label for a new example

OUR PROBLEM

given a big dataset,
find most discriminatory features

OVERVIEW

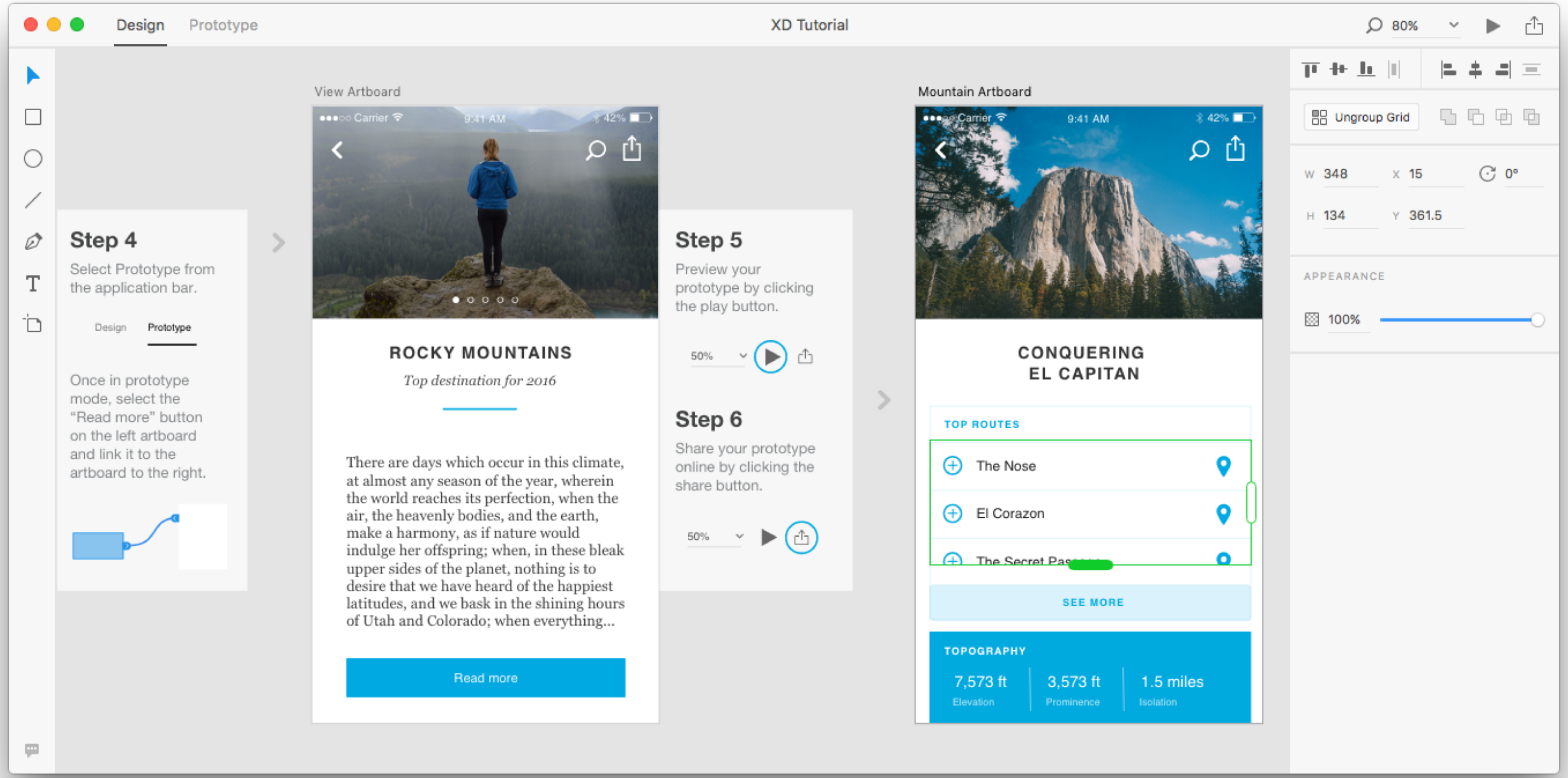

PREPROCESSS
Data
Extract features
Label examples
OPTIMIZATION
Learning algorithm
Tune hyper-parameters
Select features
Data Analysis
Statistics
Meta-classification
RESULTS
Best model
Feature ranking
Visualization

APPROACH

train a model,
it will understand data relationships.
ask it what features helped decide the most

more accurate model,
more valuable its opinion

THE APPLICATION



Adobe Experience Design

BEFORE LEARNING

Dataset → Features → Labels

Get a dataset containing relevant examples

Extract measurements to use for each user

Decide if each user is either a retainer or a churner

QUICK NUMBERS

43k	115k	4.8m
users	sessions	events

no outliers (98 quantile) or accidents (<15s)

EXTRACTED FEATURES

DOCUMENT
opened, created, saved
imported, exported, shared

DRAWS
rectangles, ellipses, lines, paths, text
artboards, repeat-grids, wires

HISTORICAL
total time, number of
launches, days span

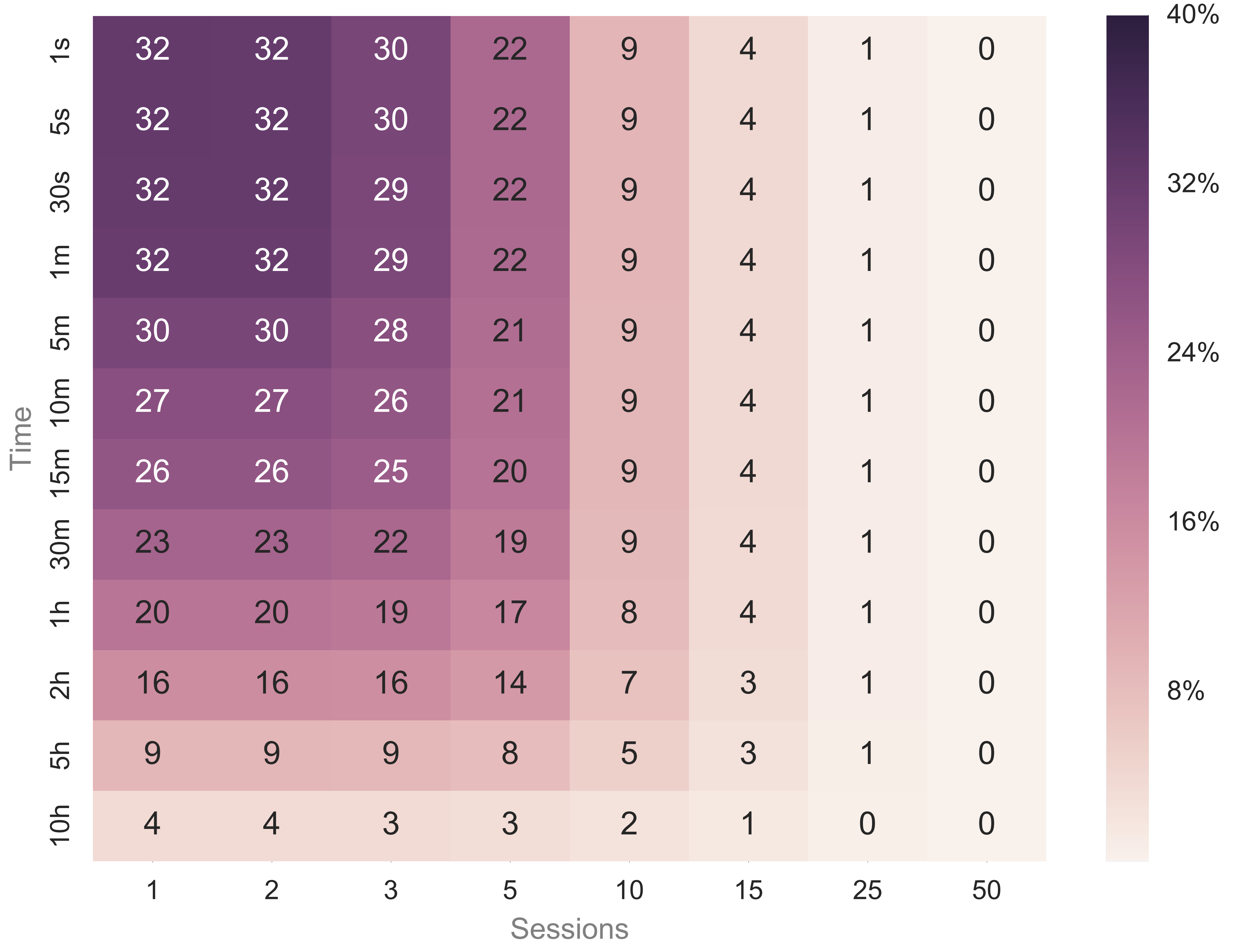
TIMES
in design, prototype, preview,
first session, action frequency

Counts ignore sequentiality.

Build action sequences

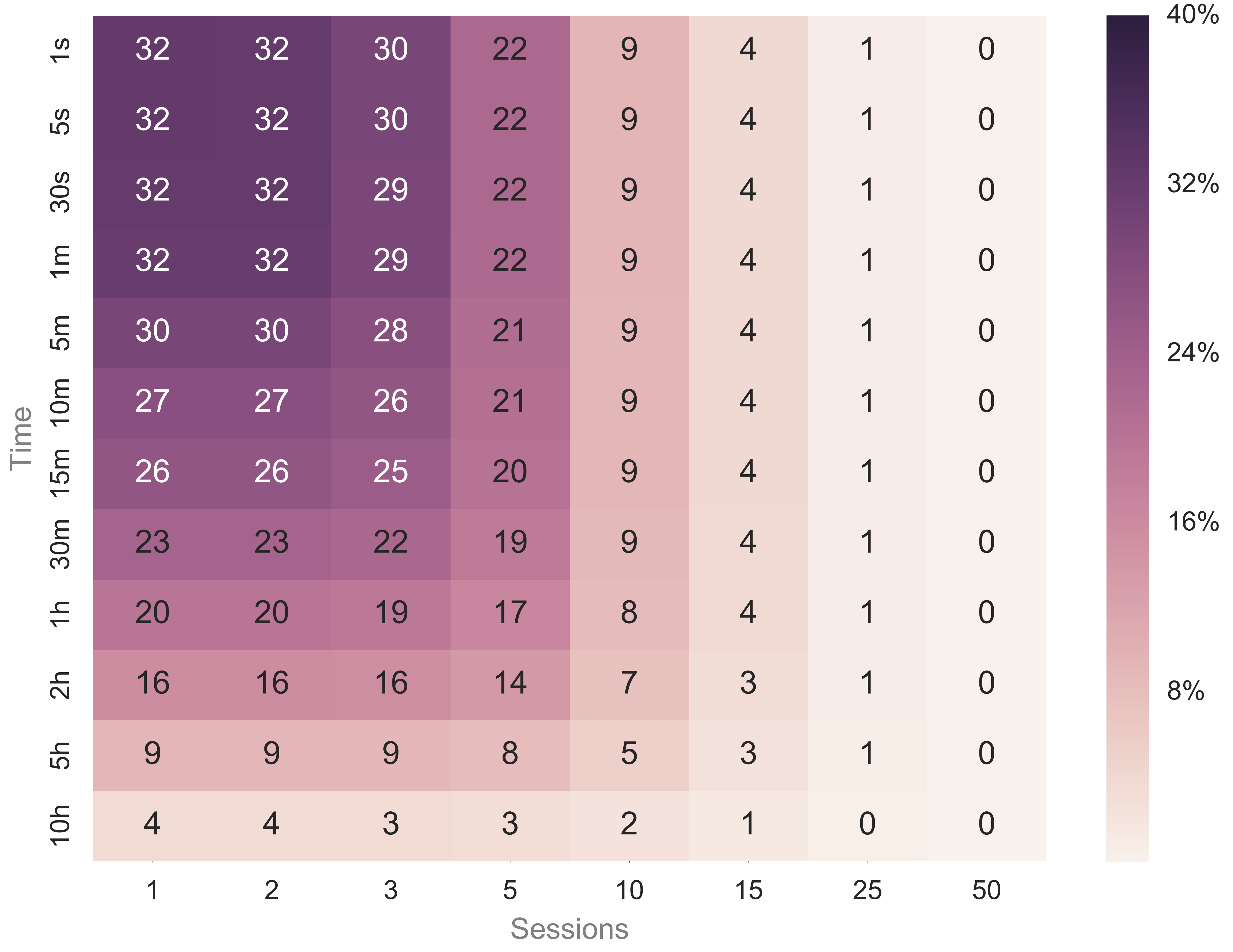
RETENTION DEFINITION

- time
- sessions
- days span



RETENTION DEFINITION

- time: 10m
- sessions: 3
- days span: 15



OPTIMIZE LEARNING

Algorithm → Hyper-parameters → Features

Algorithm

Chose a type of learner, and an actual algorithm

Hyper-parameters

Tune hyper-parameters for best generalization

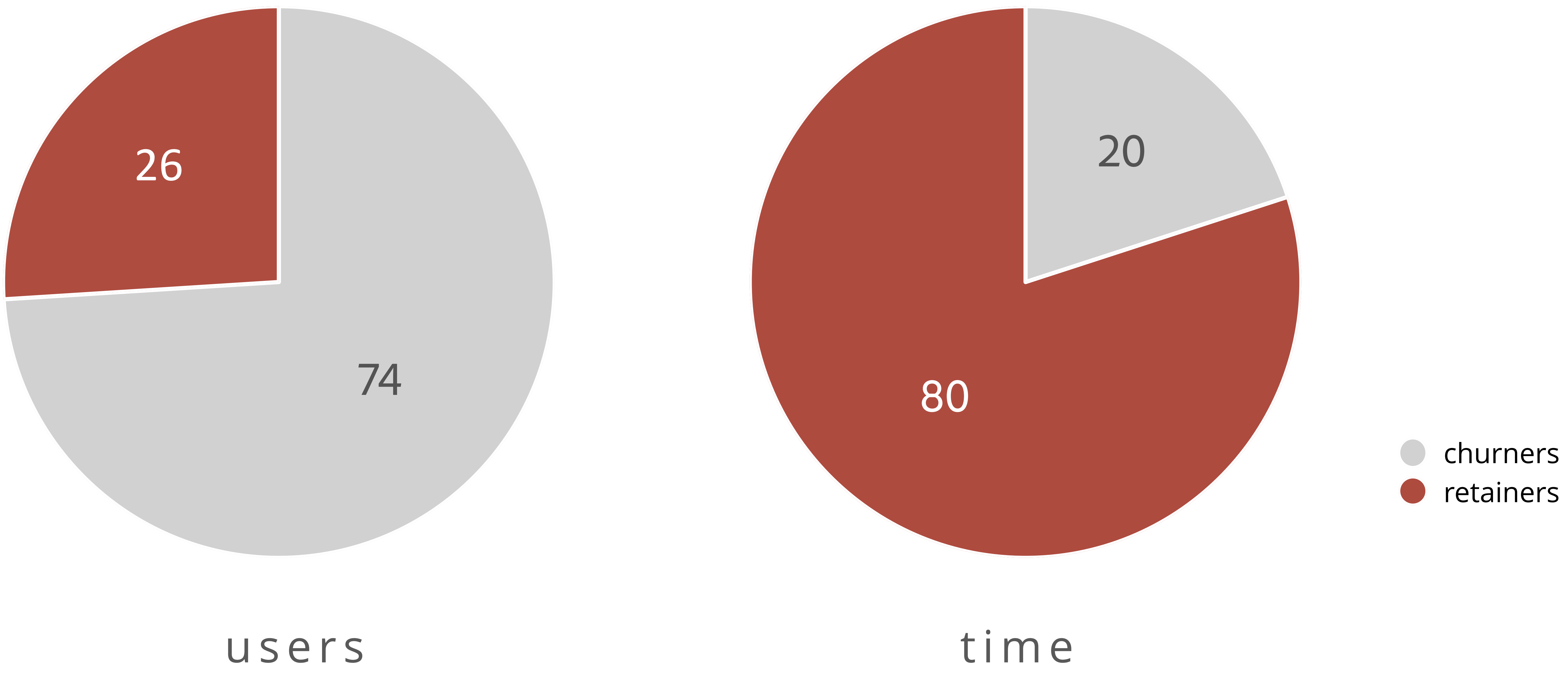
Features

Select the subset that brings the best performance

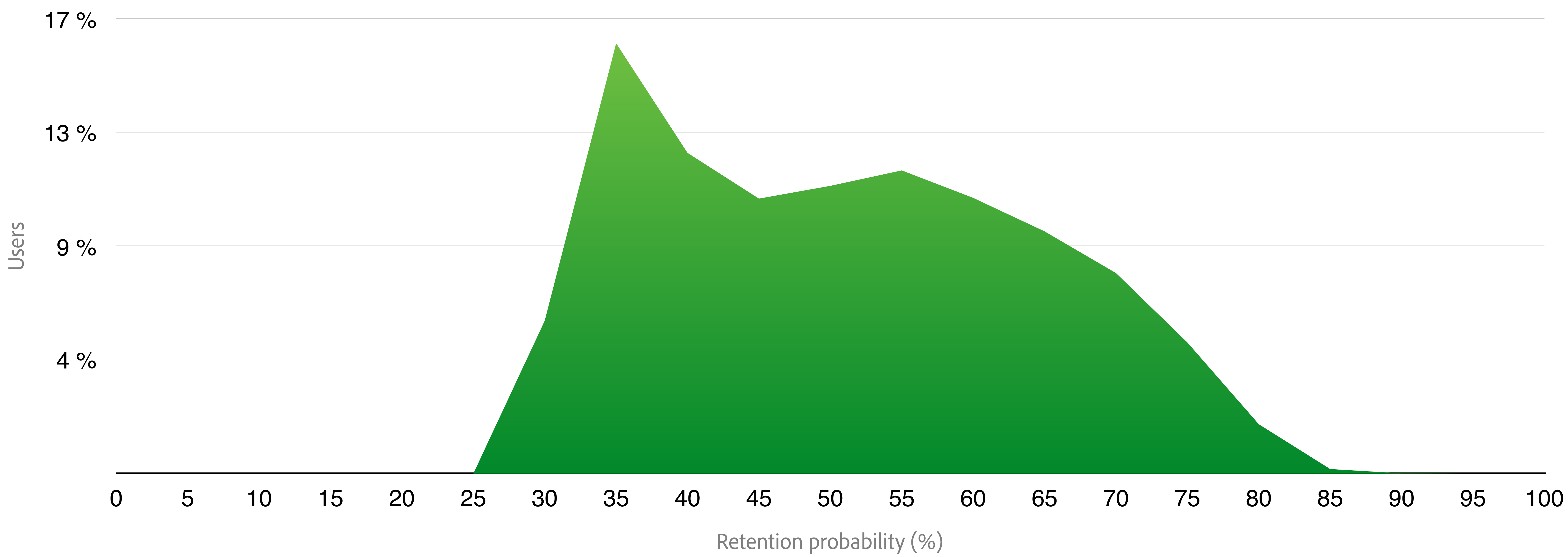
SELECTING A MODEL

- have to know the data beforehand
- need experience to pick,
- even experts need to rely on trial-and-error

PARETO IN PRACTICE



RETENTION DISTRIBUTION



LEARNING ALGORITHMS

- neural networks
- support vector machines
- decision tree forests
- naive bayes
- logistic regression
- gradient descent

- boosting
- bagging

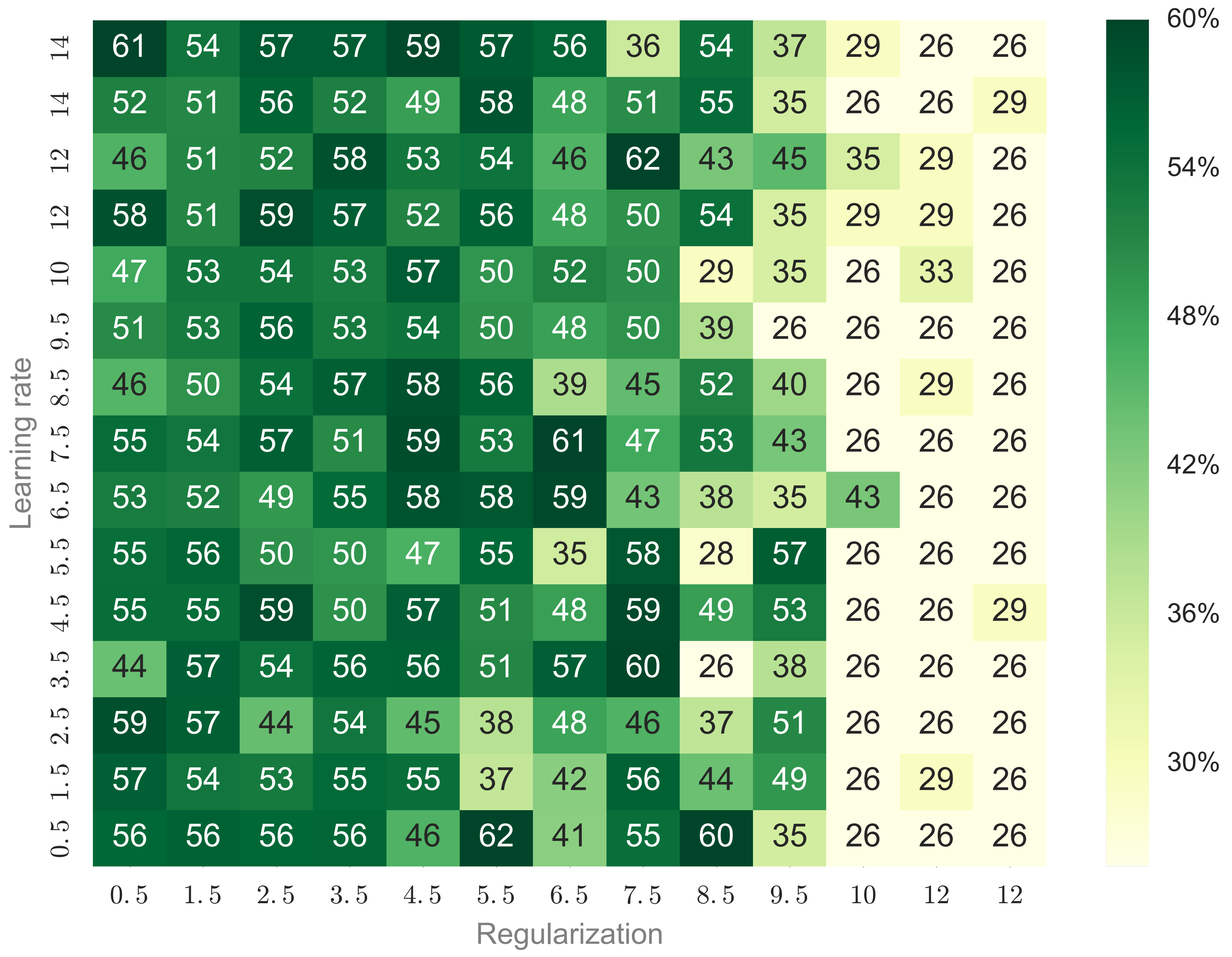
No free lunch theorem

HYPER-PARAMETERS

- models need to be tuned to be effective
- many have multiple hyper-parameters
- can't try every possible combination

$$\prod_{param} \#vals \times tt$$

HYPER-PARAMETER GRID



Horizon effect

HYPER-PARAM SEARCH

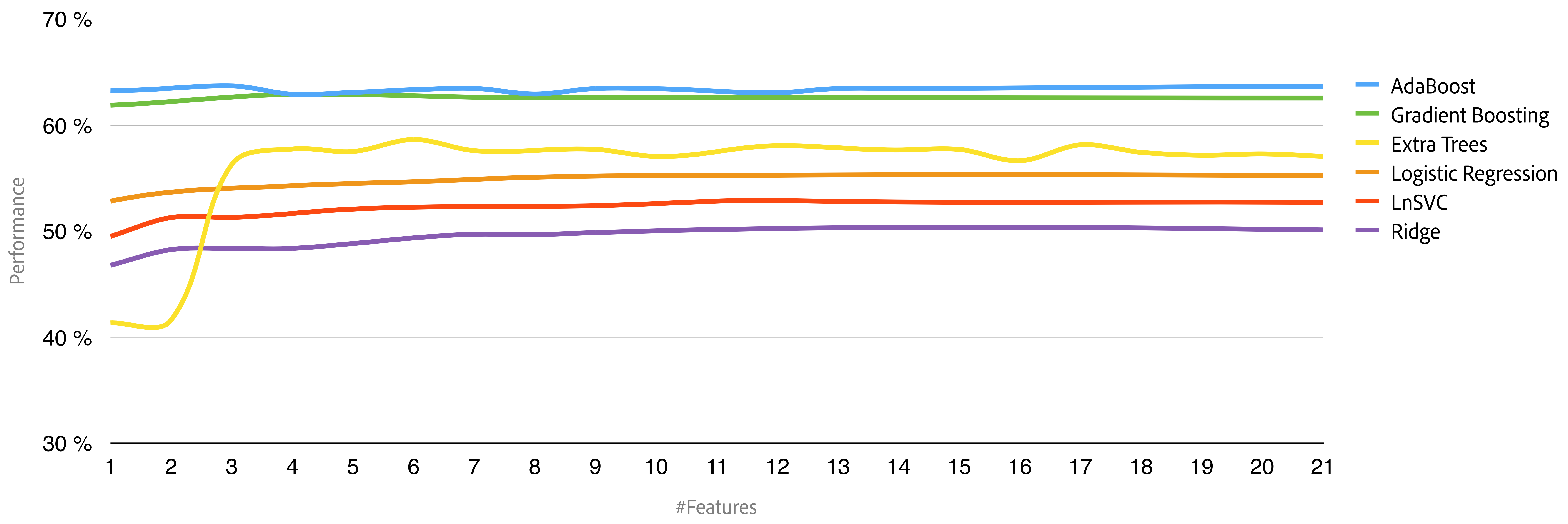
- grid search when model is fast
- randomized search instead
- and for continuous distributions as well

FEATURE SELECTION

- reduces complexity
- easier to interpret
- will learn relationships, not noise
- requires smaller training set

Curse of dimensionality

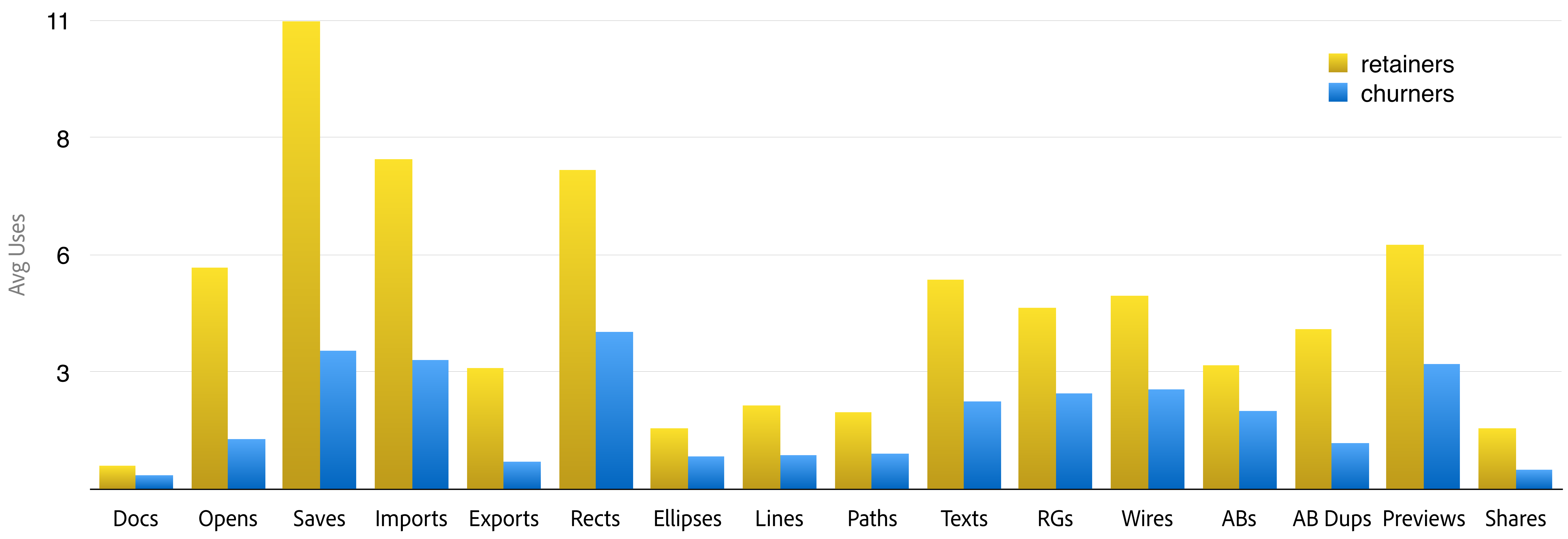
SUBSET PERFORMANCE



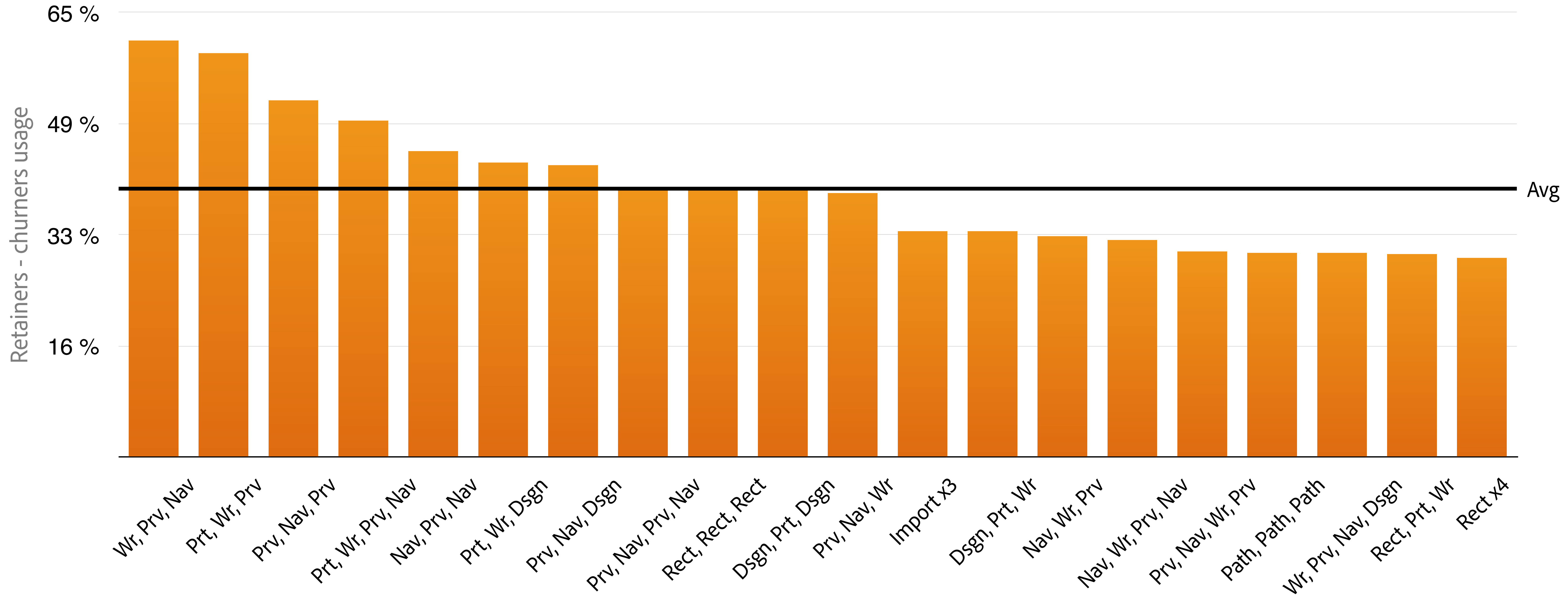
FS TECHNIQUES

- recursive feature elimination
- data analysis
- statistical methods

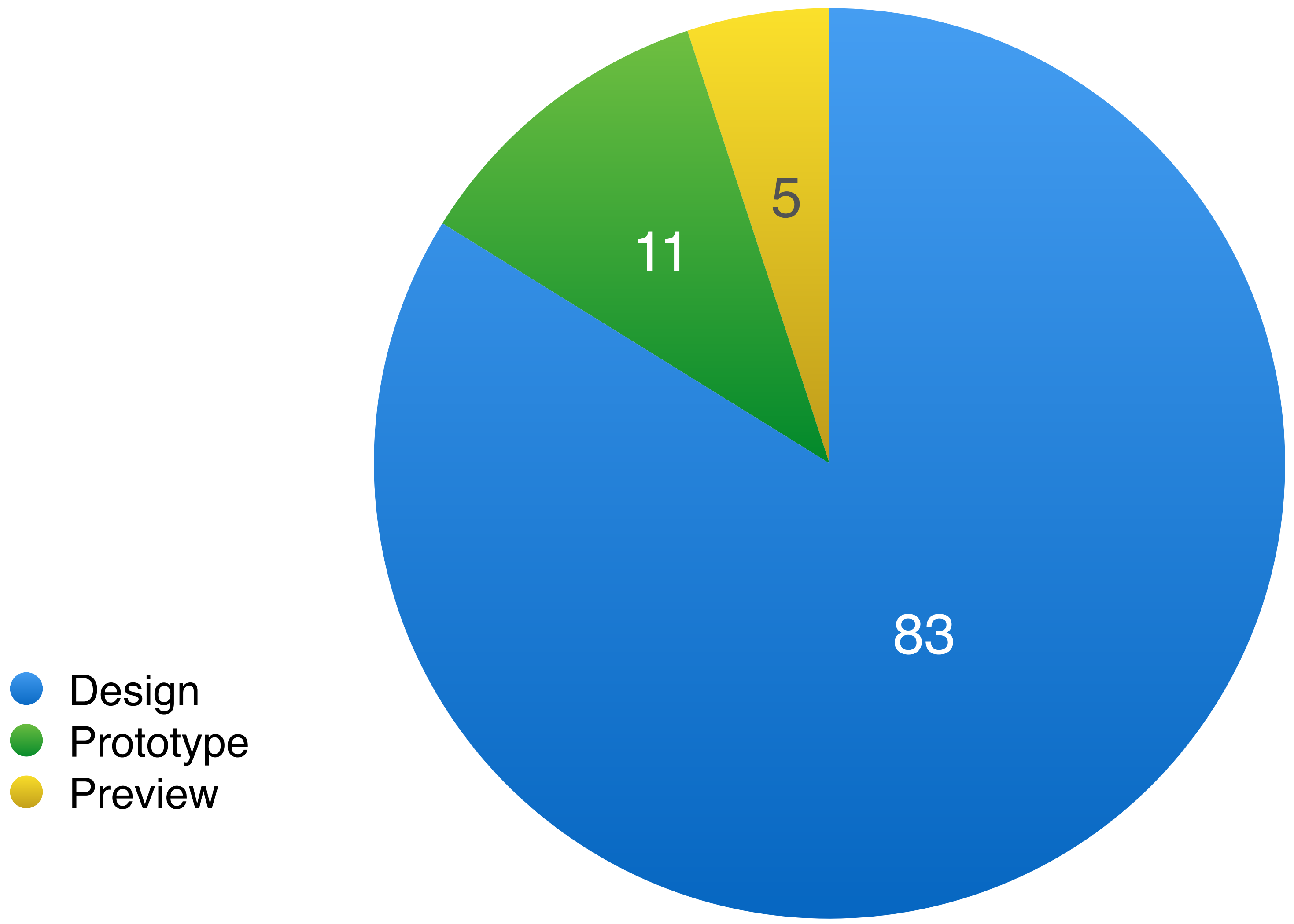
AVERAGE USAGE



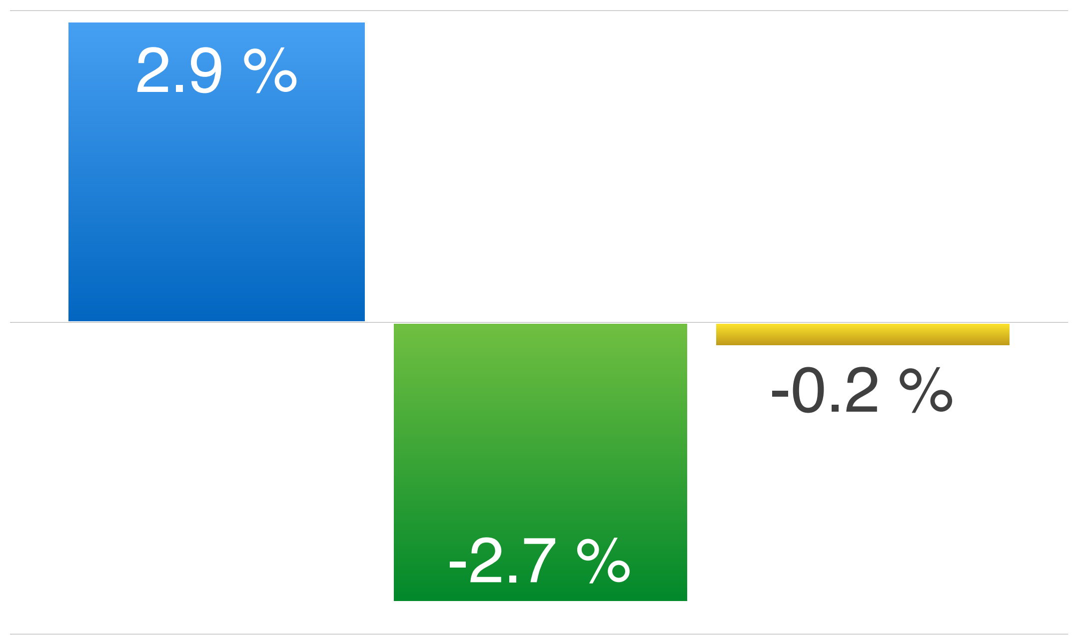
ACTION SEQUENCES



MODE PREFERENCE

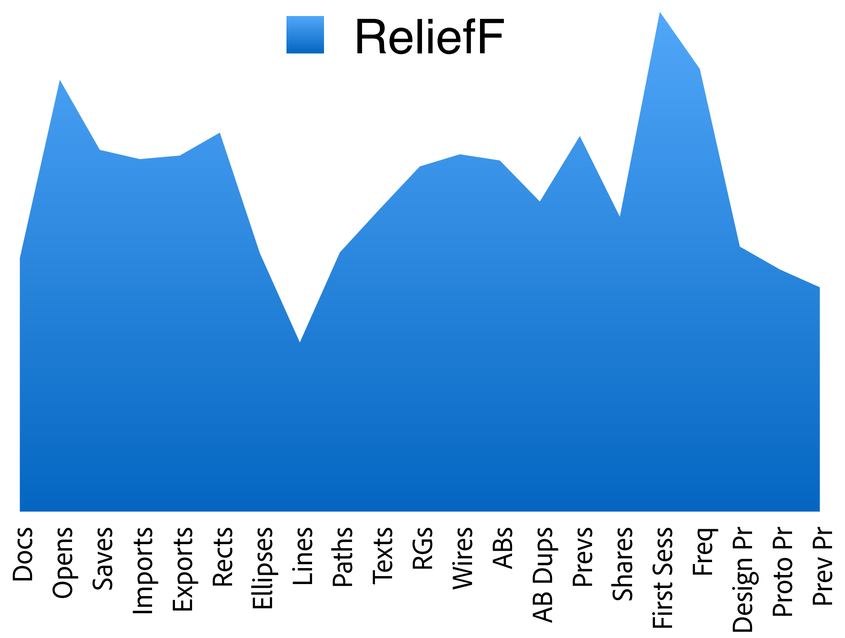


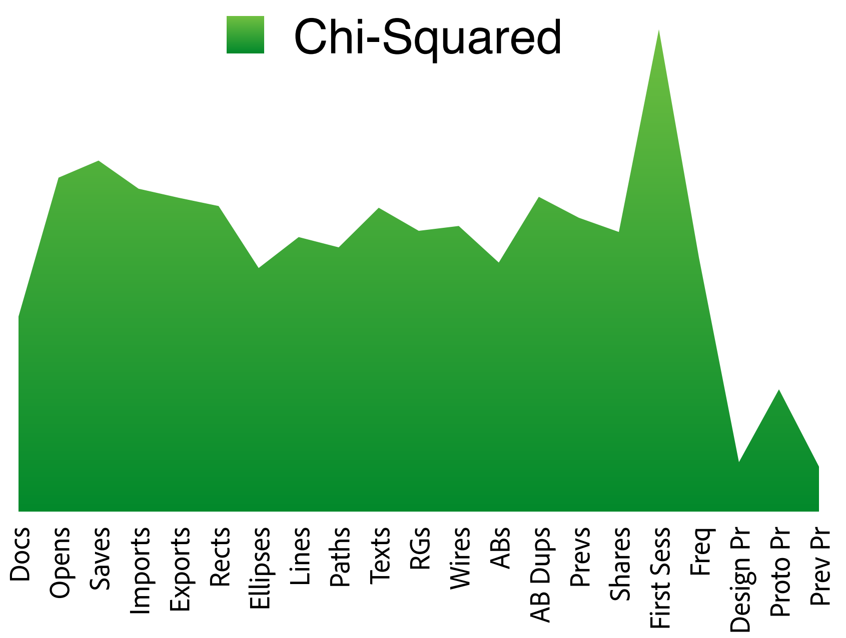
churners time spent

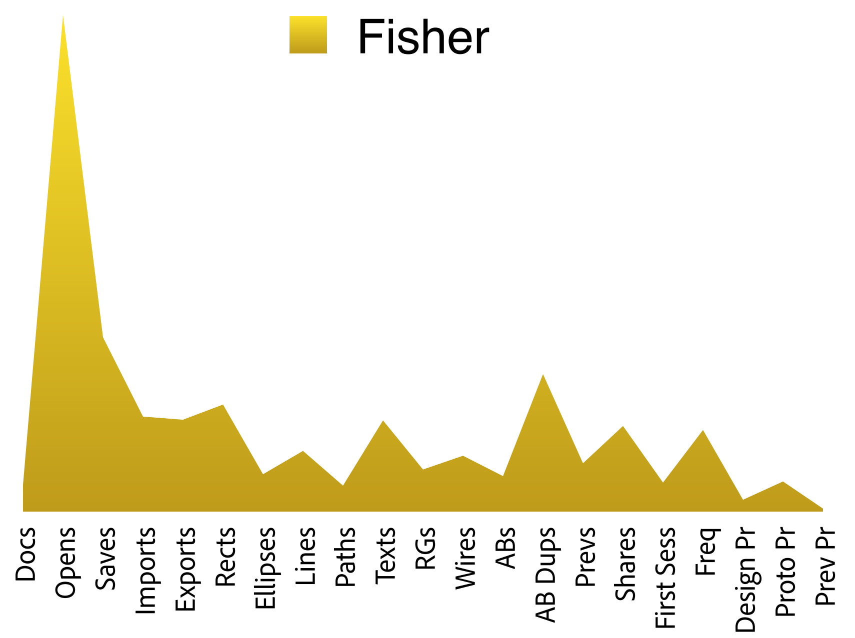


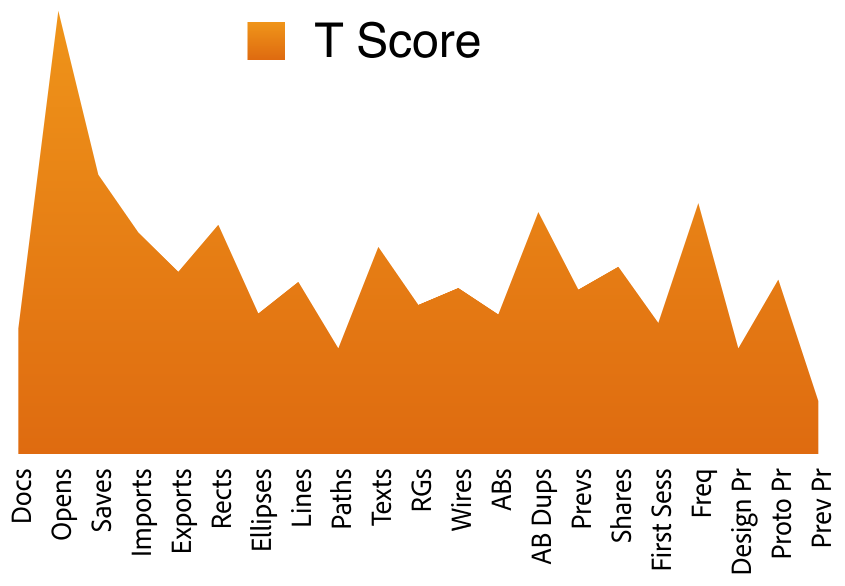
retainers difference

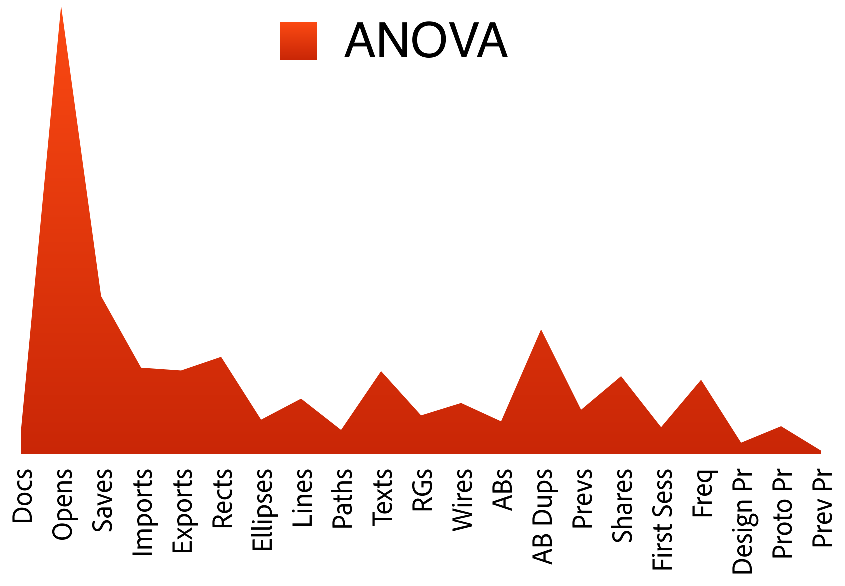
STATISTICAL SCORES

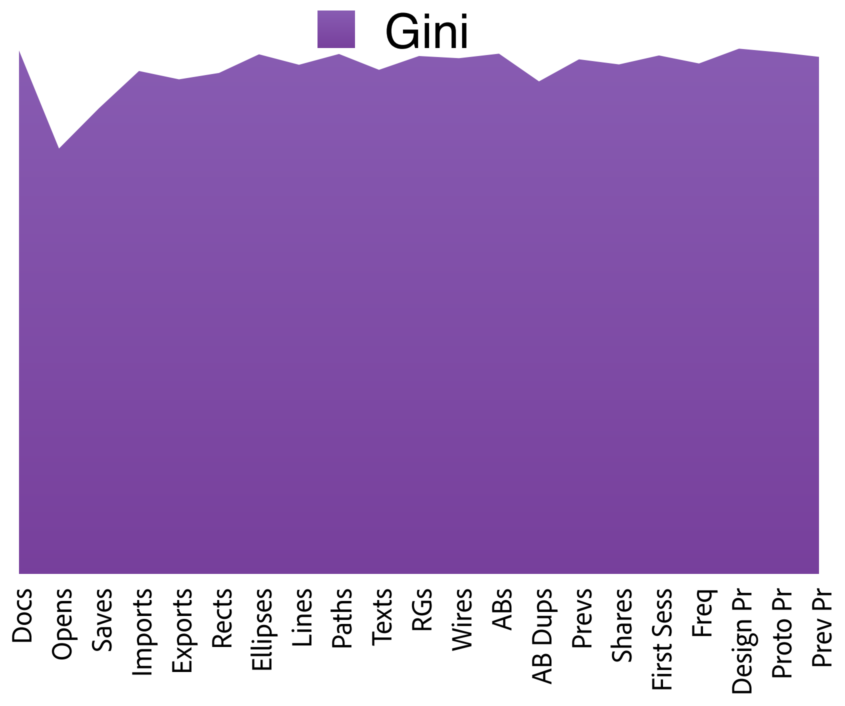




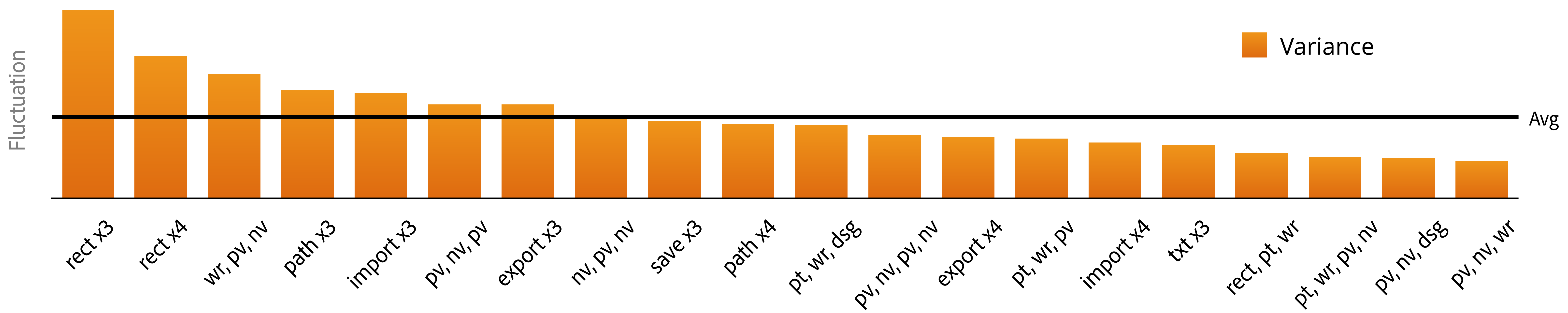


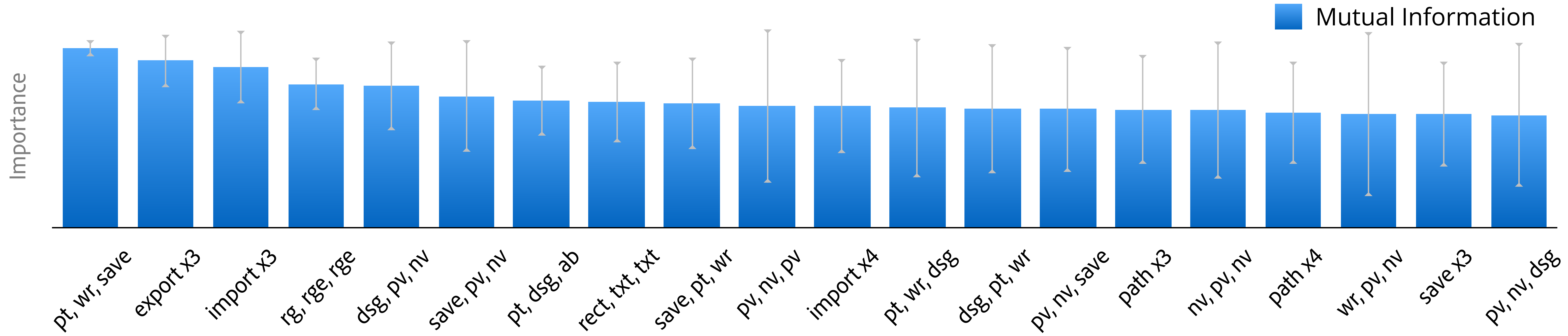






STATISTICAL METHODS





MODEL OPTIMIZATION

PIPELINE

- gives the best model for a learning task
- seeks to minimize human input
- compensates for lack of experience

MODEL OPTIMIZATION

BEFORE

1. Pick applicable learning methods on the task
   Select concrete learning **algorithms** for each method
2. Consider impactful hyper-parameters for each algorithm
   Pick sensible **ranges of values** for each hyper-parameter
3. Run **data analysis** and **statistical methods**
   Propose most impactful feature subsets
4. Decide the desired **duration** allowed for optimization

MODEL OPTIMIZATION

STEPS

1. Run exhaustive search on hyper-parameter grid on a **sample**
   Restrict hyper-parameters iterations and RFE step size based on time

2. Fit **hyper-parameters** for each model
   Using random-search (restricted)

3. Chose best **feature subset** for each model
   Try the whole dataset, proposed subsets and RFE (restricted)

4. Take best models (with diversity in mind)
   Use them as deciders for the **combining classifier**

META CLASSIFIER

classify based on the output of other algorithms,
not on examples themselves

- learn **how** to learn
- already trained many models
- learn how to best combine their decisions

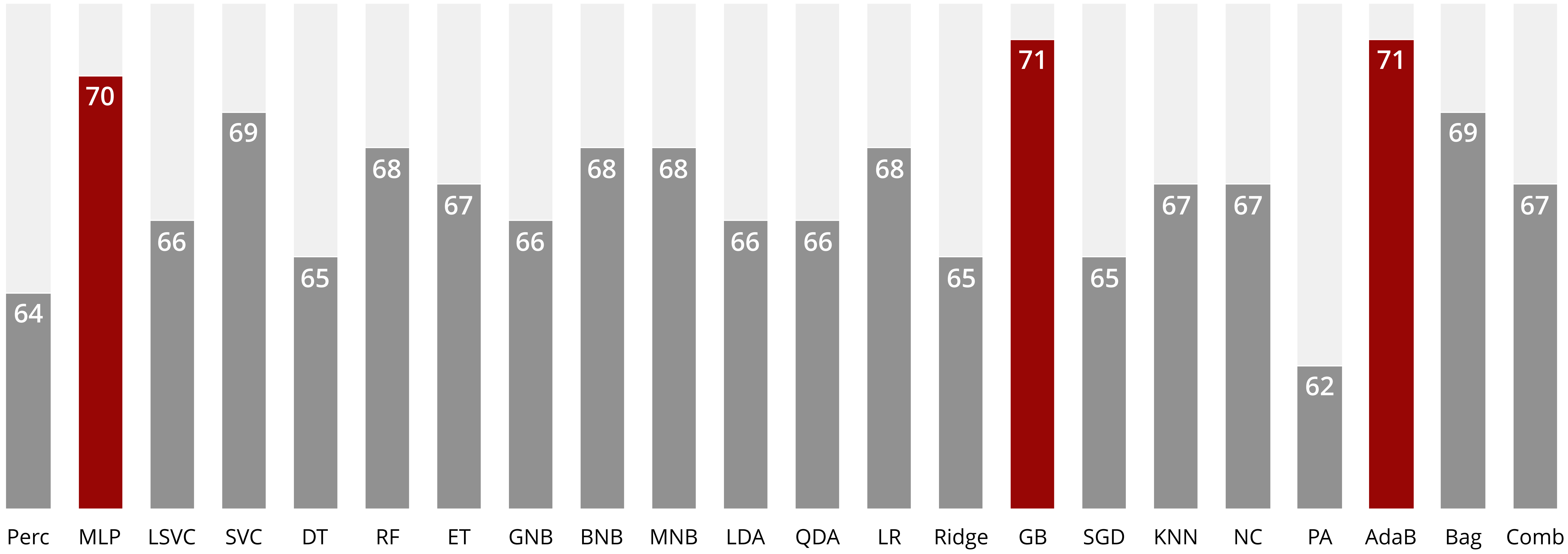
COMBINER

simple model for aggregating:
shallow NN, small RF, linear SVM

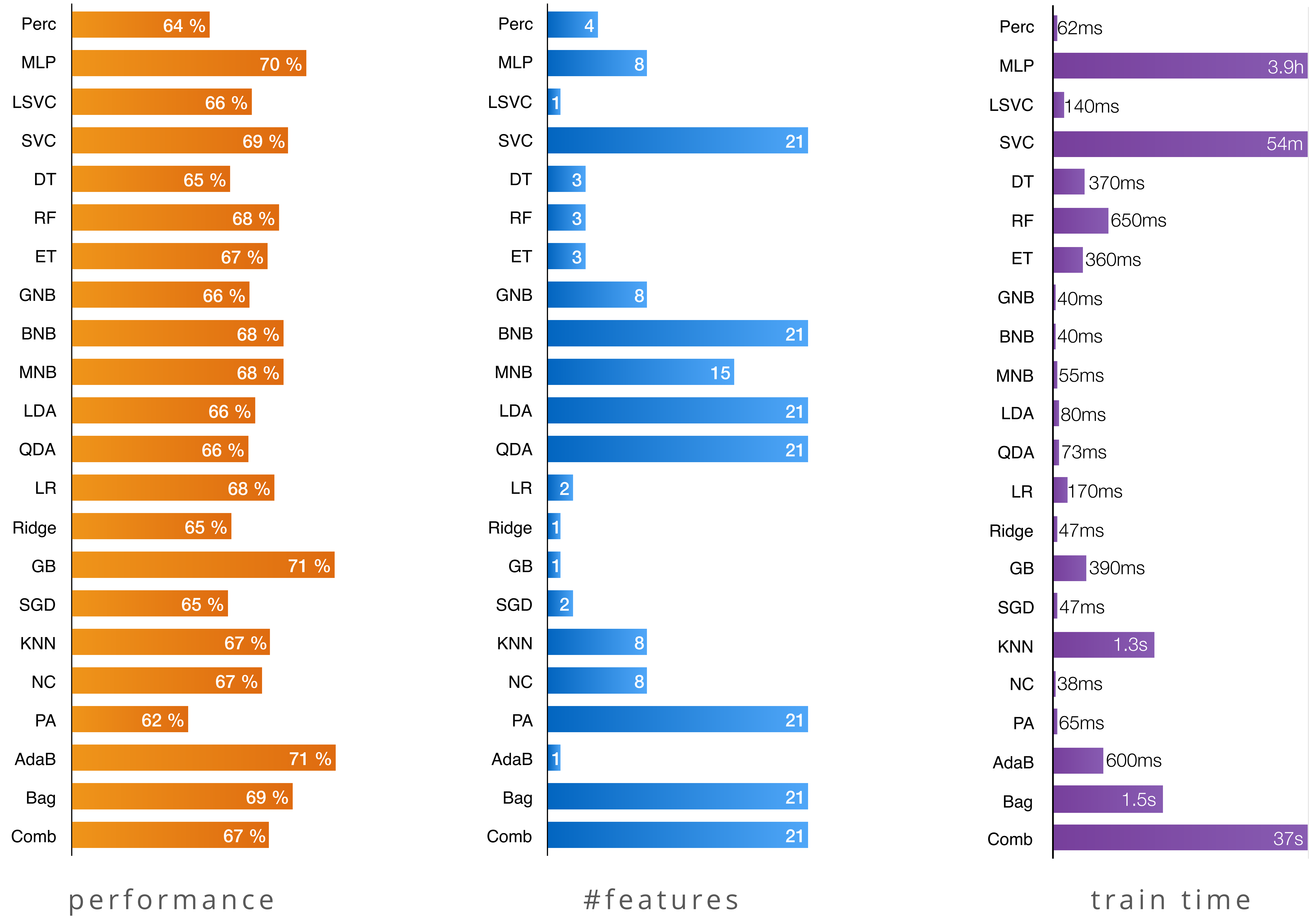
DECIDERS

take best performing models.
each has strengths and weaknesses,
compensate by promoting **diversity**

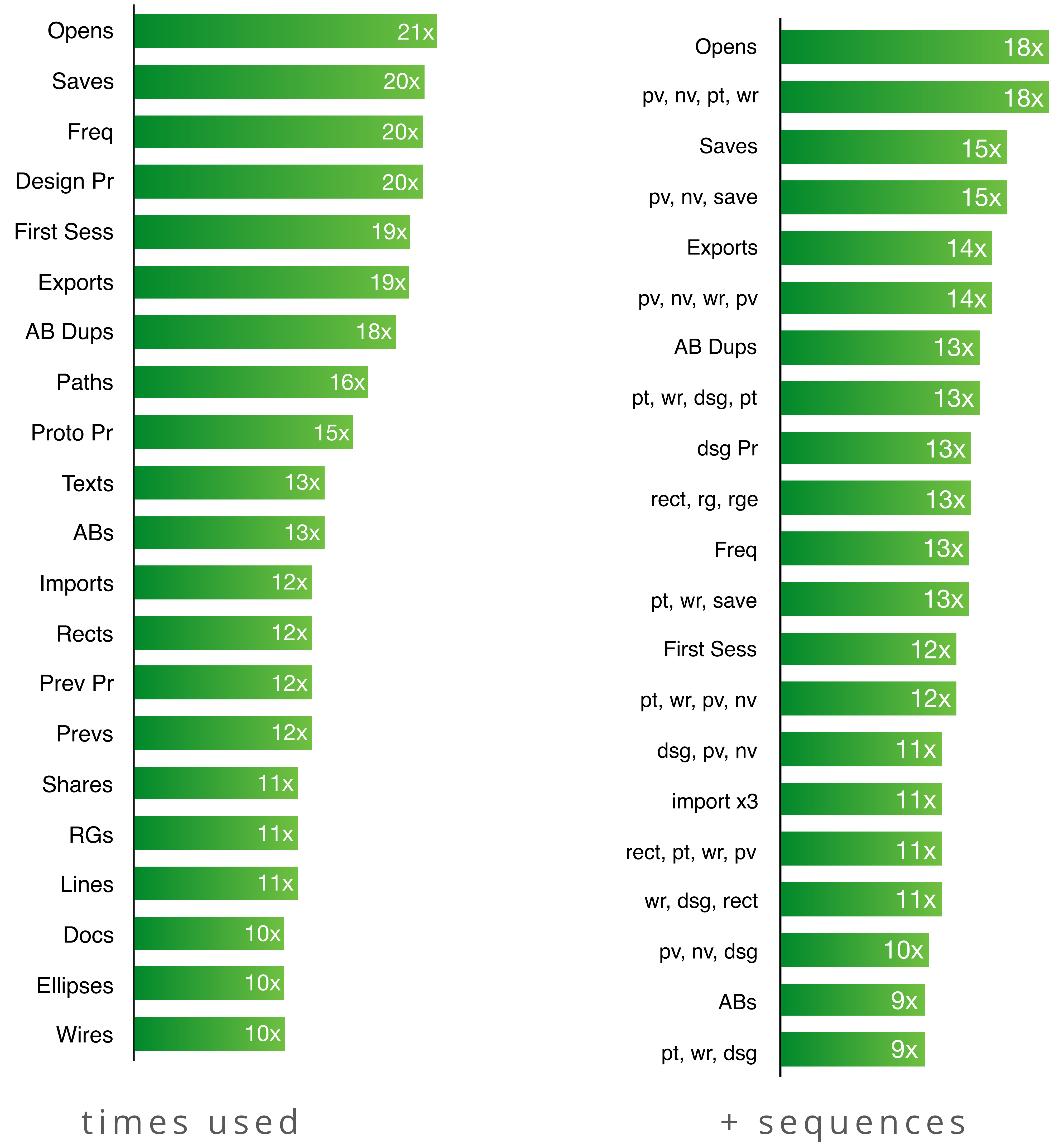
MODEL PERFORMANCE



MODEL COMPARISON



FEATURE RANKING



CONCLUSION

Important features:
- prototype, preview
- RGs, ABs

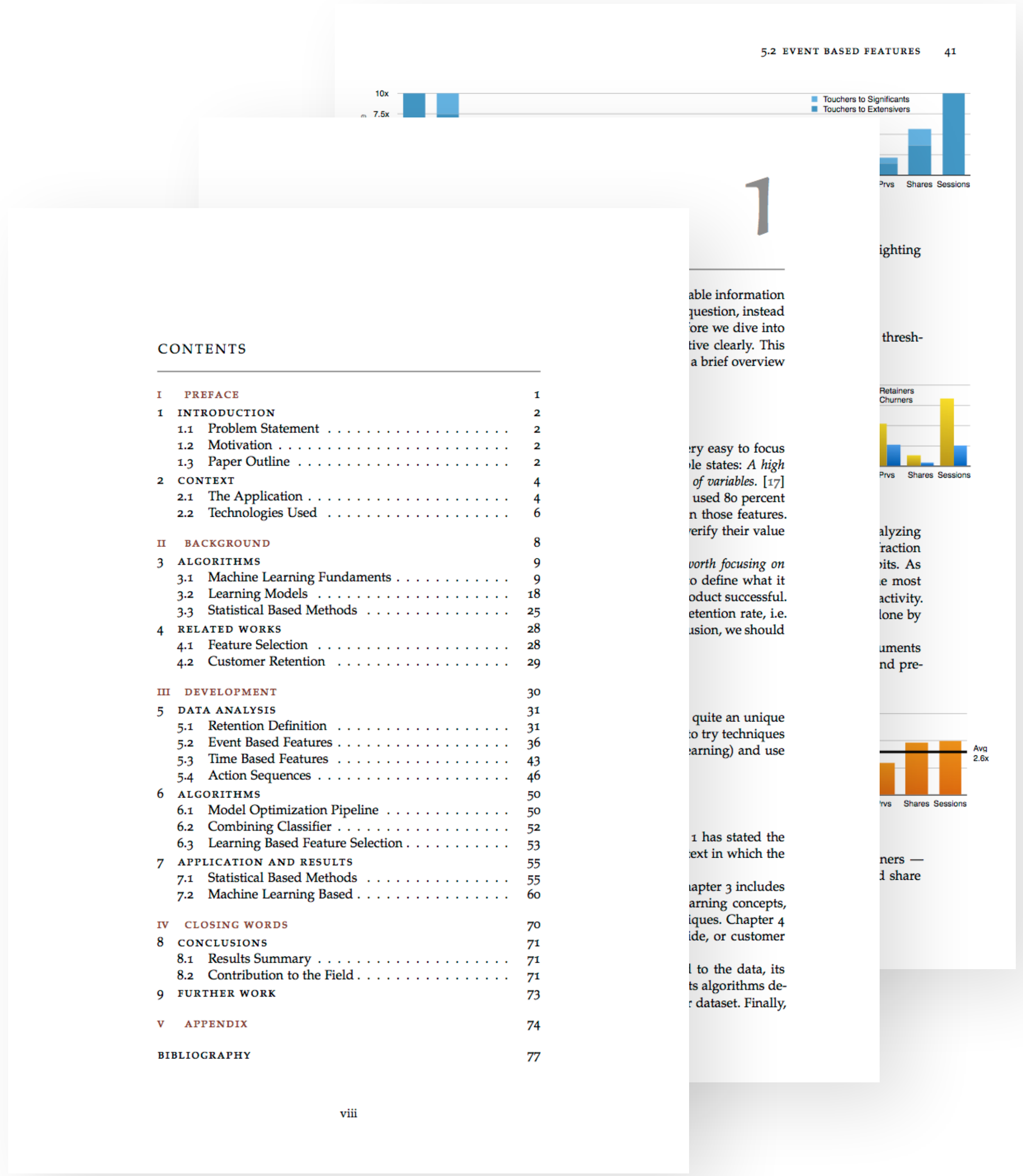
Field contribution:
- model optimization pipeline,
- combining classifier

NEXT STEPS

Technique refinement:
- continuous classification
- deep learning
- word embedding visualization

New direction:
- sequence learning
- predictive system

AVAILABLE ON REQUEST



CONTENTS

viii

THESIS

87 pages

```python
from sklearn.datasets import make_hastie_10_2
from sklearn.metrics import accuracy_score


class CombiningClassifier(BaseEstimator, ClassifierMixin):
    def __init__(self, combiner=Perceptron(), deciders='default'):
        self.combiner = combiner

        if deciders == 'default':
            deciders = [SVC(), ExtraTreesClassifier(),
                        LogisticRegression(), MultinomialNB()]
        self.deciders = deciders


    def check_parameters(self):
        # Checking if the combining and deciding classifiers are
        # actually classifiers
        classifiers = [self.combiner] + self.deciders
        for clf in classifiers:
            # TODO make these exception throwers
            assert issubclass(clf.__class__, BaseEstimator)
            assert issubclass(clf.__class__, ClassifierMixin)
            # TODO use has attr fit and has attr predict
            # pass


    def combiner_input_(self, X):
        # The first row contains the predictions of the first deciding
        # classifier for every point
        # the second row has the second deciding classifier, etc
        deciders_output = [clf.predict(X) for clf in self.deciders]

        # Now on the first row there is the prediction for the first point of
        # every deciding classifier
        # on the second row the prediction for the second point etc
        return np.array(deciders_output).transpose()



    def fit(self, X, y):
        self.check_parameters()

```

CODE

3.9 kloc

ACKNOWLEDGEMENTS

Paul Alexandru Chirița

Ștefan Teodor Craciun

Alexandru-Daniel Mirea

Daniel Dogaru

BIBLIOGRAPHY

C. Bishop - *Pattern Recognition and Machine Learning*
T. Hastie et al - *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*

Full list in paper

Q & A

+ feedback

THANK YOU

for your attention

ȘTEFAN NICULAE

niculae@adobe.com 10C

“An approximate answer to the right problem is worth a good deal more than an exact answer to an approximate problem”

— John Tukey, mathematician