Introduction to model validation

MODEL VALIDATION IN PYTHON

Kasey Jones
Data Scientist

What is model validation?

Model validation consists of:

- Ensuring your model performs as expected on new data
- Testing model performance on holdout datasets
- Selecting the best model, parameters, and accuracy metrics
- Achieving the best accuracy for the data given

scikit-learn modeling review

Basic modeling steps:

```
model = RandomForestRegressor(n_estimators=500, random_state=1111)
model.fit(X=X_train, y=y_train)
```

```
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
           max_features='auto', max_leaf_nodes=None,
           min_impurity_decrease=0.0, min_impurity_split=None,
           min_samples_leaf=1, min_samples_split=2,
           min_weight_fraction_leaf=0.0, n_estimators=500, n_jobs=1,
           oob_score=False, random_state=1111, verbose=0, warm_start=False)
```

Modeling review continued

```
predictions = model.predict(X_test)
print("{0:.2f}".format(mae(y_true=y_test, y_pred=predictions)))
```

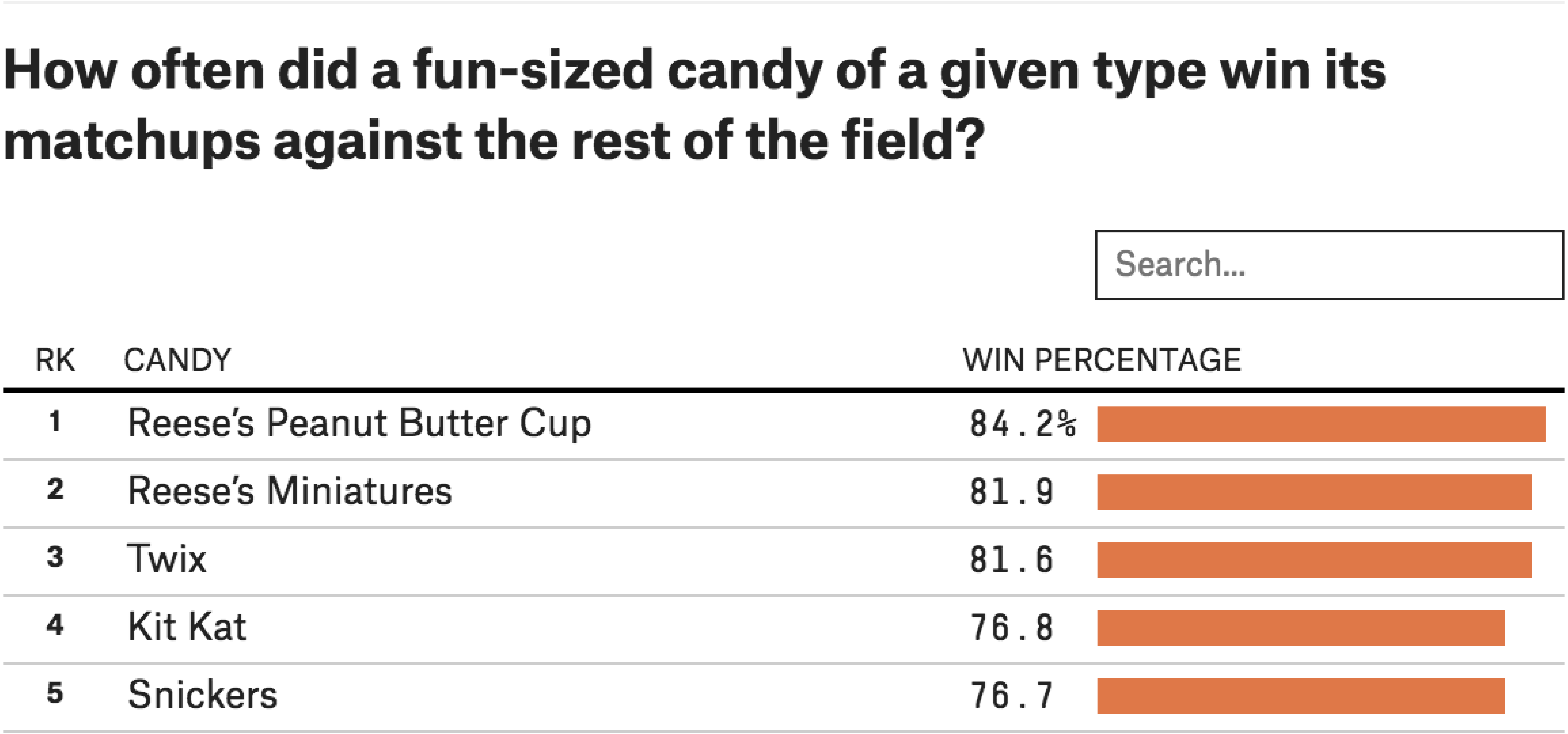
```
10.84
```

Mean Absolute Error Formula

$$\frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n}$$

Review prerequisites

- Intermediate Python
- Supervised Learning with scikit-learn



How often did a fun-sized candy of a given type win its matchups against the rest of the field?

RK	CANDY	WIN PERCENTAGE
1	Reese's Peanut Butter Cup	84.2%
2	Reese's Miniatures	81.9
3	Twix	81.6
4	Kit Kat	76.8
5	Snickers	76.7

Seen vs. unseen data

Training data = seen data

```
model = RandomForestRegressor(n_estimators=500, random_state=1111)
model.fit(X_train, y_train)
train_predictions = model.predict(X_train)
```

Testing data = unseen data

```
model = RandomForestRegressor(n_estimators=500, random_state=1111)
model.fit(X_train, y_train)
test_predictions = model.predict(X_test)
```

Let's begin!

MODEL VALIDATION IN PYTHON

Regression models

MODEL VALIDATION IN PYTHON

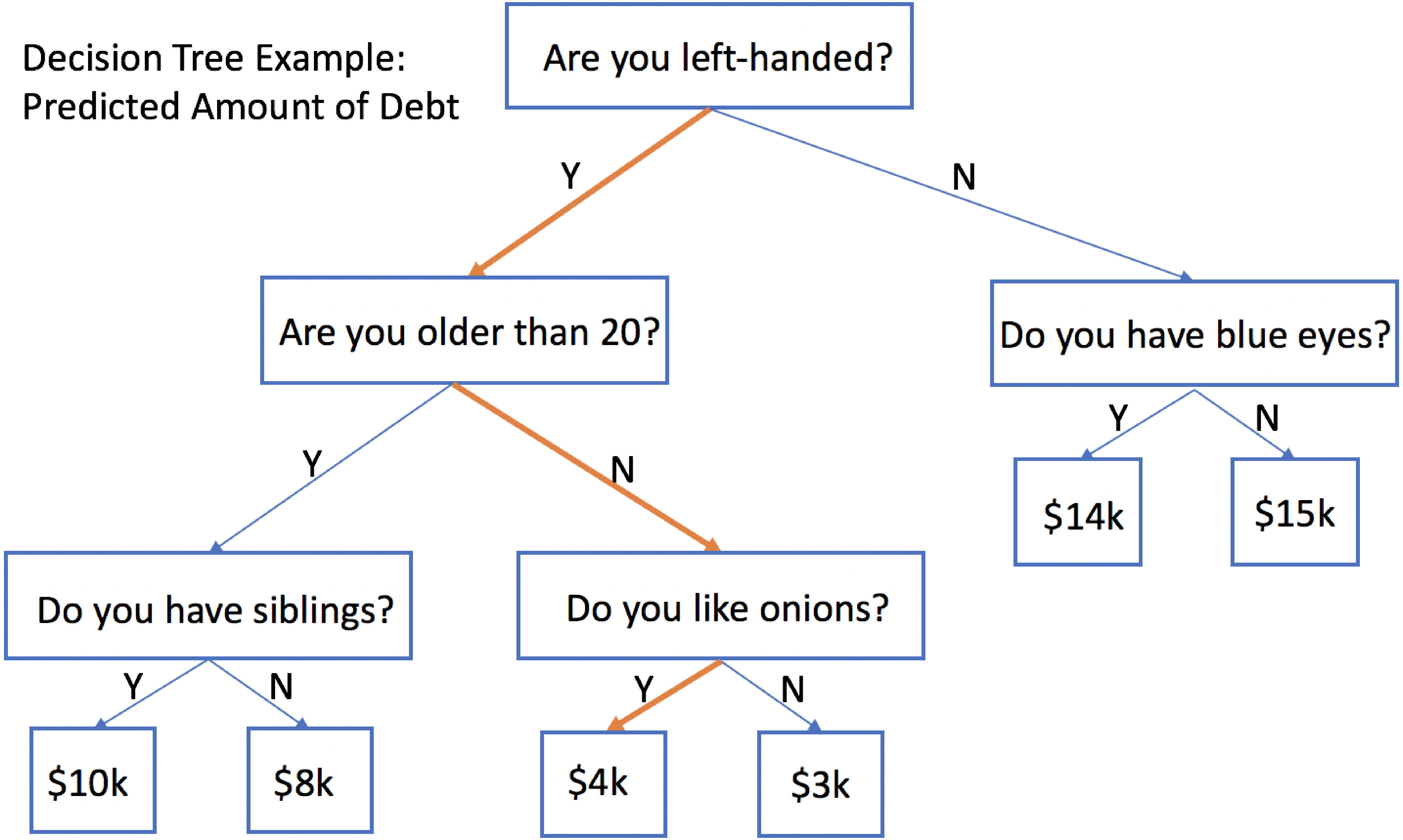
Kasey Jones
Data Scientist

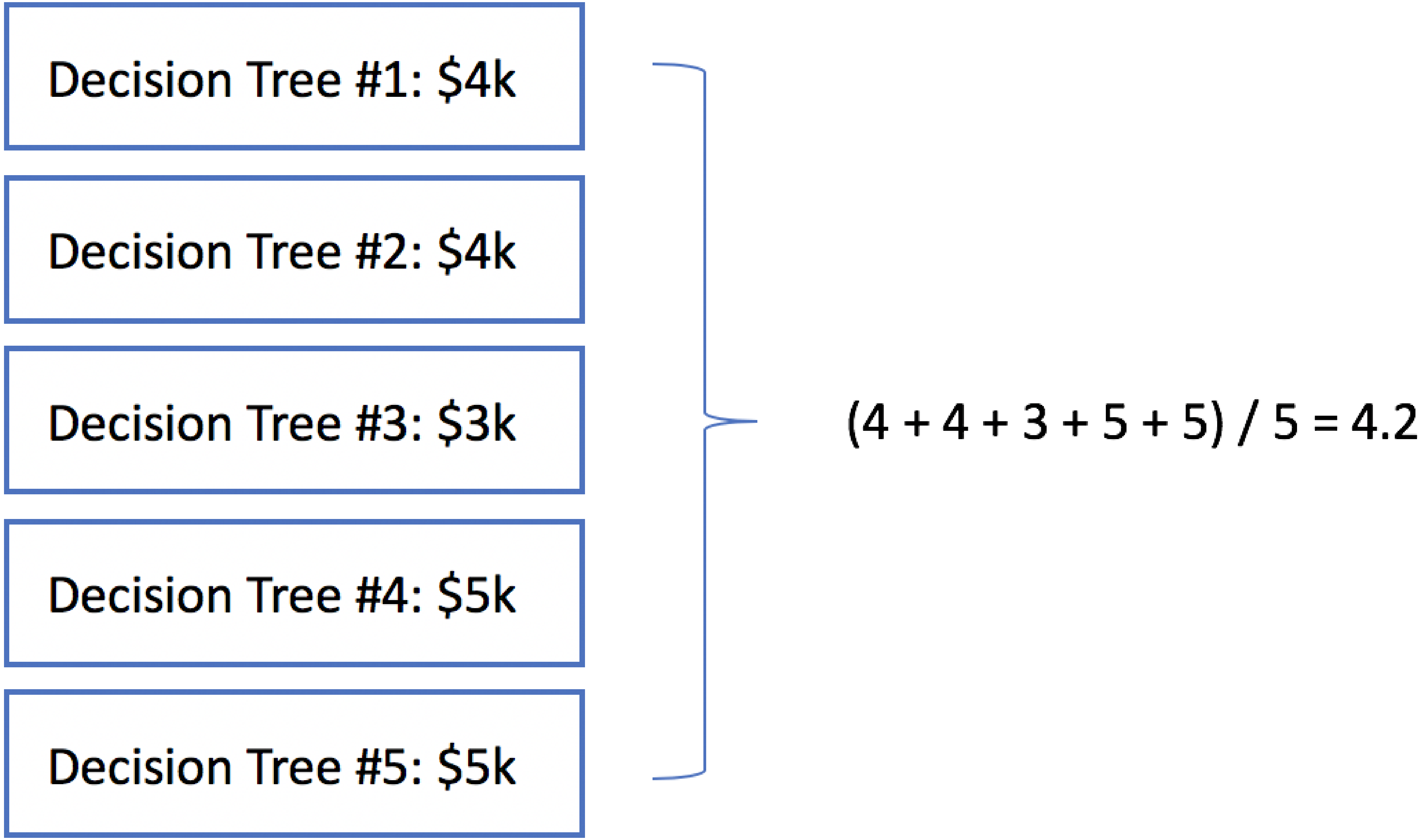
Random forests in scikit-learn

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import RandomForestClassifier
```

```
rfr = RandomForestRegressor(random_state=1111)
rfc = RandomForestClassifier(random_state=1111)
```

Decision Tree Example:
Predicted Amount of Debt





Decision Tree #1: $4k

Decision Tree #2: $4k

Decision Tree #3: $3k

Decision Tree #4: $5k

Decision Tree #5: $5k

$$(4 + 4 + 3 + 5 + 5) / 5 = 4.2$$

Random forest parameters

`n_estimators` : the number of trees in the forest

`max_depth` : the maximum depth of the trees

`random_state` : random seed

```
from sklearn.ensemble import RandomForestRegressor
rfr = RandomForestRegressor(n_estimators=50, max_depth=10)
```

```
rfr = RandomForestRegressor(random_state=1111)
rfr.n_estimators = 50
rfr.max_depth = 10
```

Feature importance

Print how important each column is to the model

```
for i, item in enumerate(rfr.feature_importances_):
    print("{0:s}: {1:.2f}".format(X.columns[i], item))
```

```
weight: 0.50
height: 0.39
left_handed: 0.72
union_preference: 0.05
eye_color: 0.03
```

Let's begin

MODEL VALIDATION IN PYTHON

Classification models

MODEL VALIDATION IN PYTHON

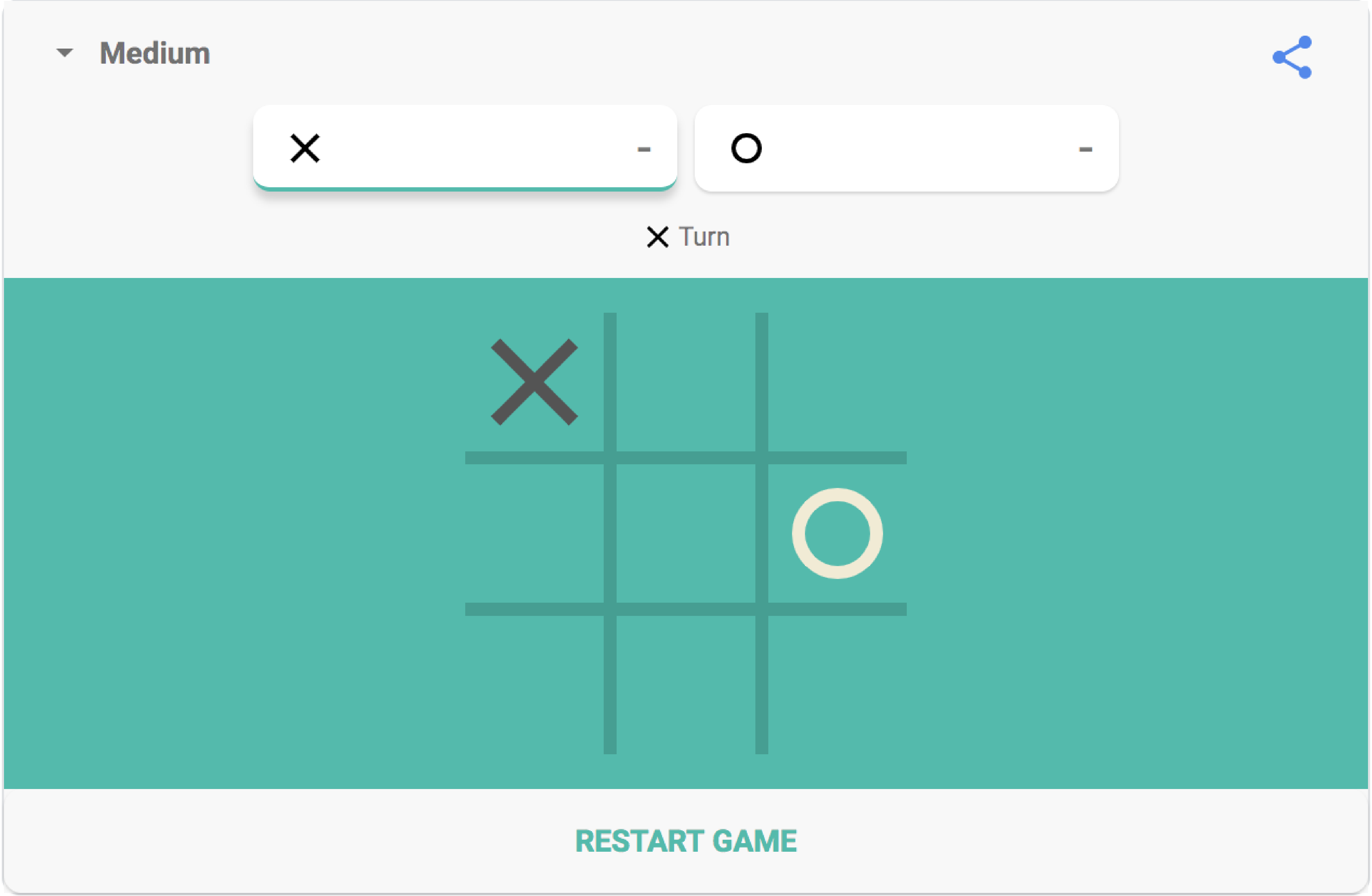
Kasey Jones
Data Scientist

Classification models

- Categorical Responses:
  - Newborn's hair color
  - Winner of a basketball game
  - Genre of the next song on the radio

The Tic-Tac-Toe dataset

...	Bottom-Left	Bottom-Middle	Bottom-Right	Class
...	X	O	O	positive
...	O	X	O	positive
...	O	O	X	positive
...	X	X	O	negative
...	...	...	...	...


Medium
X -
O -
X Turn
RESTART GAME

Using .predict() for classification

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(random_state=1111)
rfc.fit(X_train, y_train)
rfc.predict(X_test)
```

```
array([1, 1, 1, 1, 0, 1, ...])
```

```
pd.Series(rfc.predict(X_test)).value_counts()
```

```
1    627
0    331
```

Predicting probabilities

```
rfc.predict_proba(X_test)
```

```
array([[0. , 1. ],
       [0.1, 0.9],
       [0.1, 0.9],
       ...])
```

```python
rfc = RandomForestClassifier(random_state=1111)
rfc.get_params()
```

```
{'bootstrap': True,
 'class_weight': None,
 'criterion': 'gini',
 ...}
```

```python
rfc.fit(X_train, y_train)
rfc.score(X_test, y_test)
```

```
0.8989
```

Let's classify Tic-Tac-Toe end-game scenarios

MODEL VALIDATION IN PYTHON