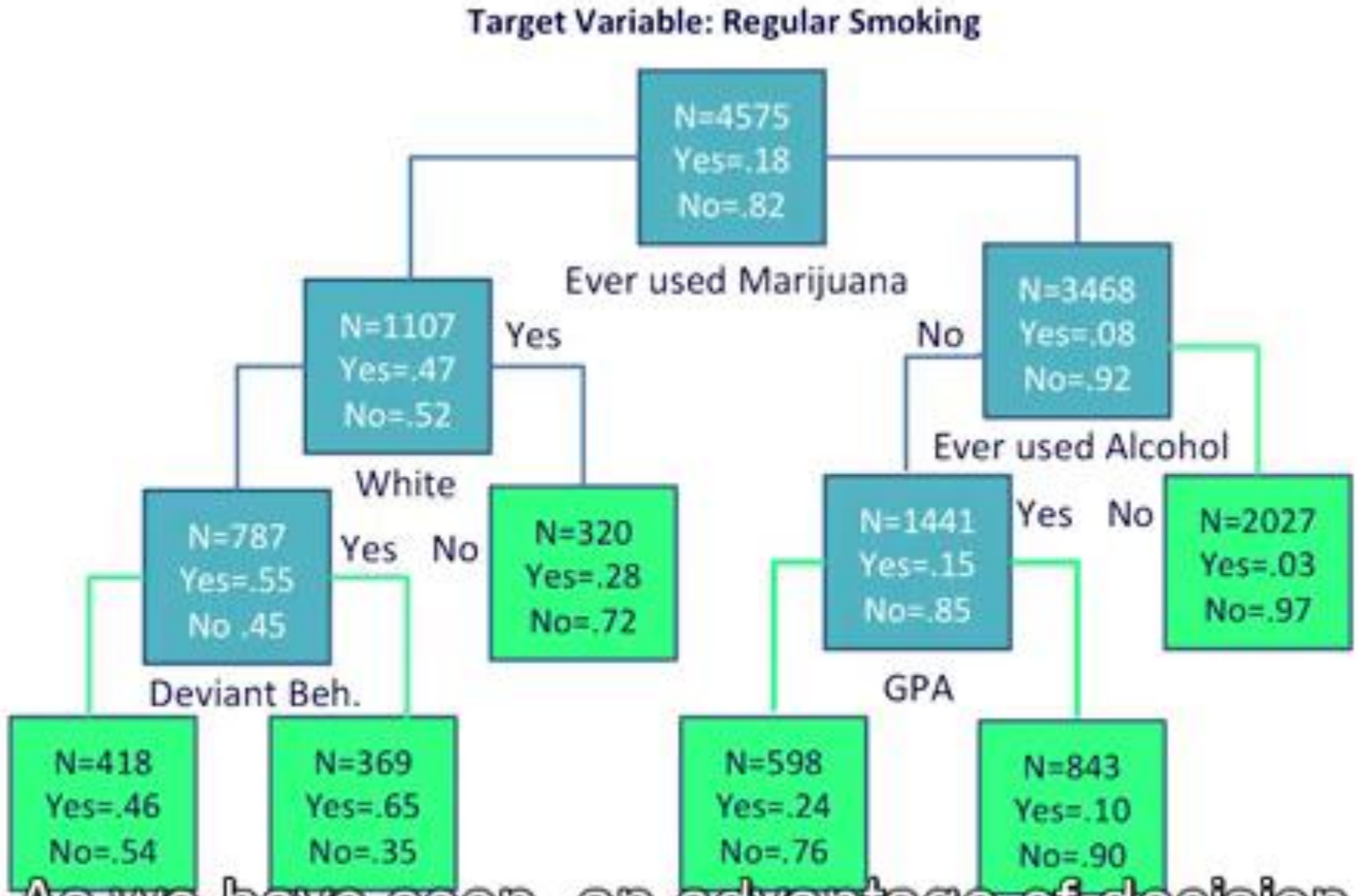
WESLEYAN
UNIVERSITY

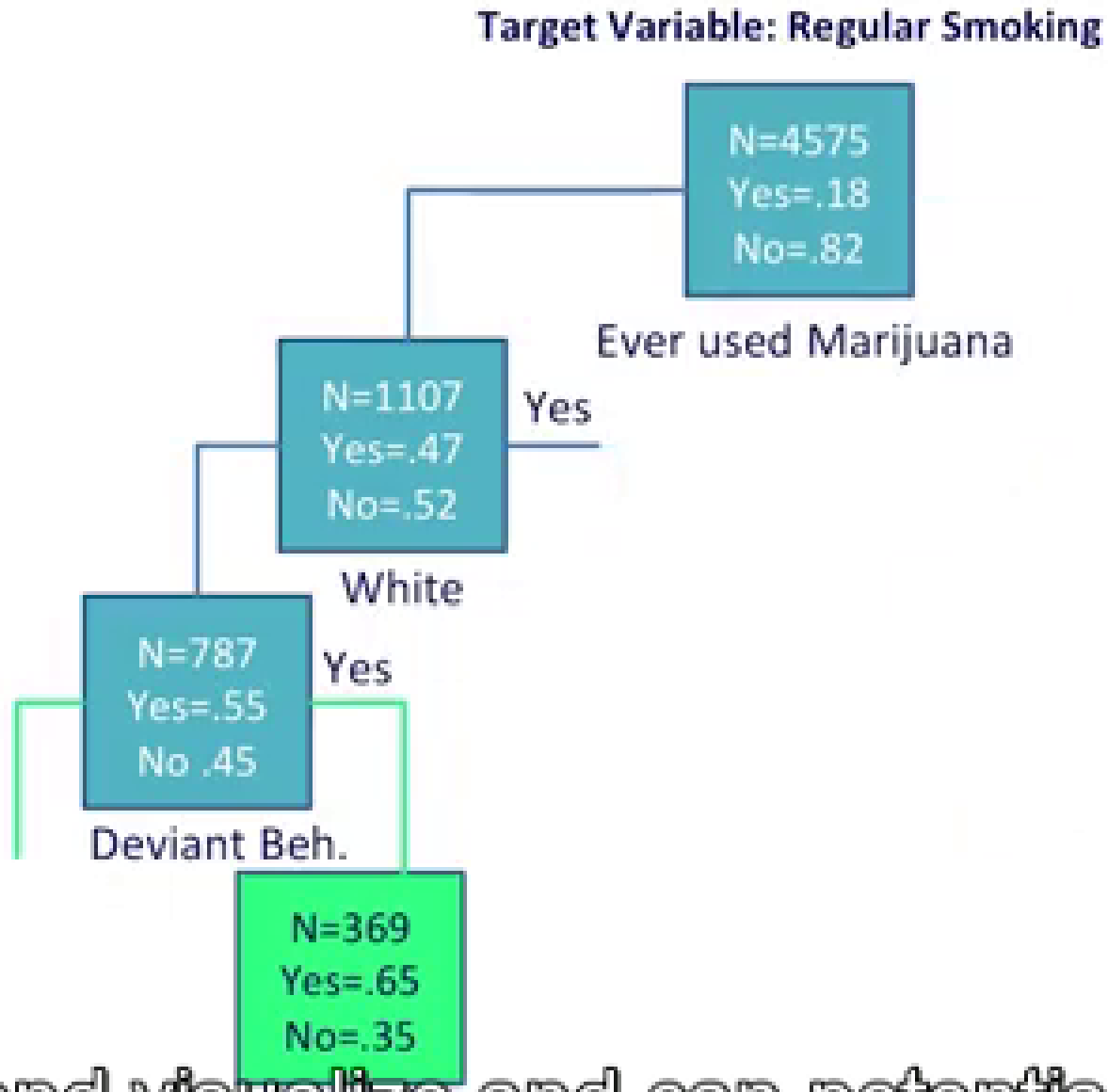
Random Forests

What is a Random Forest and how is it "grown"?

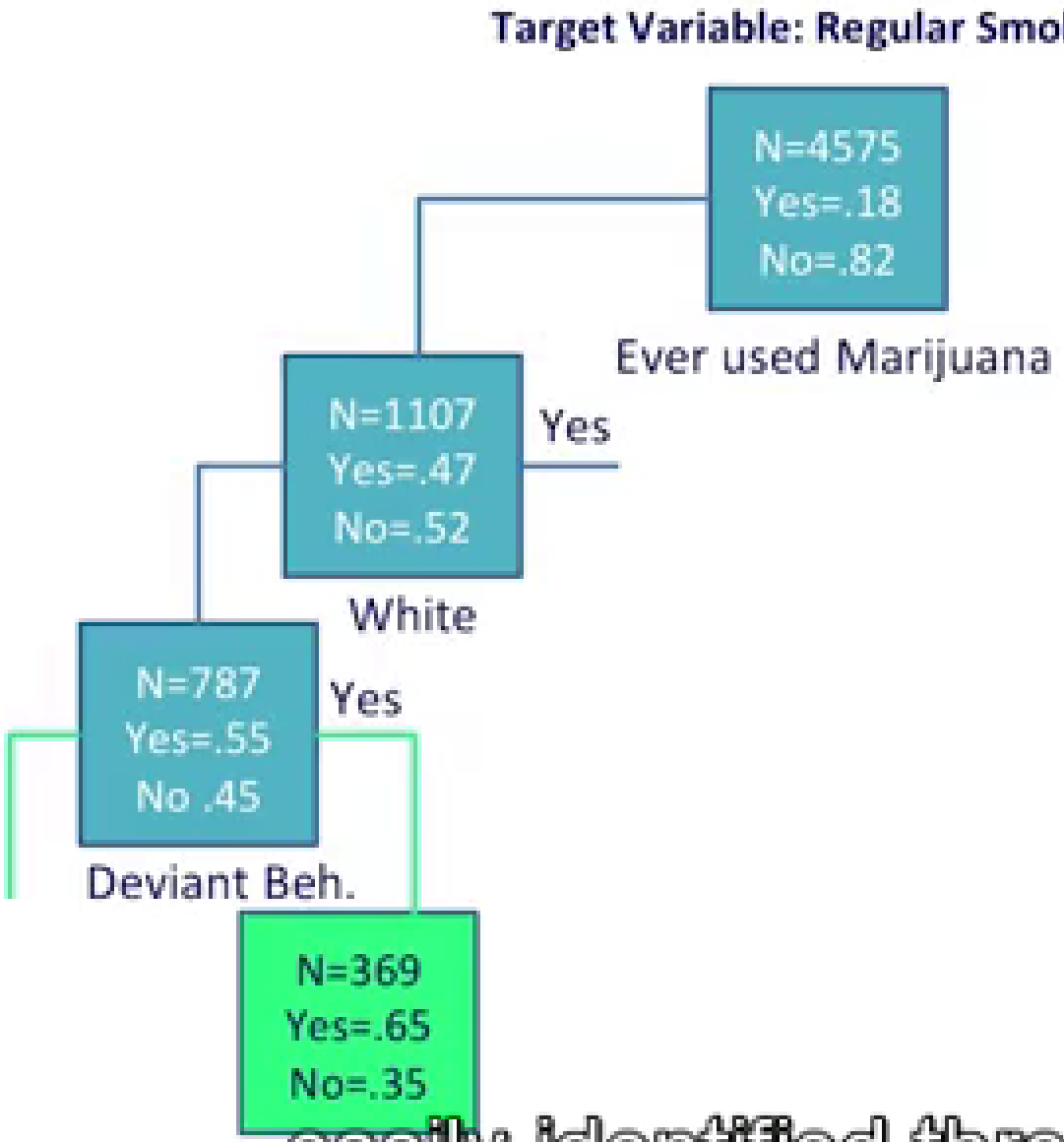
with Professor Lisa Dierker



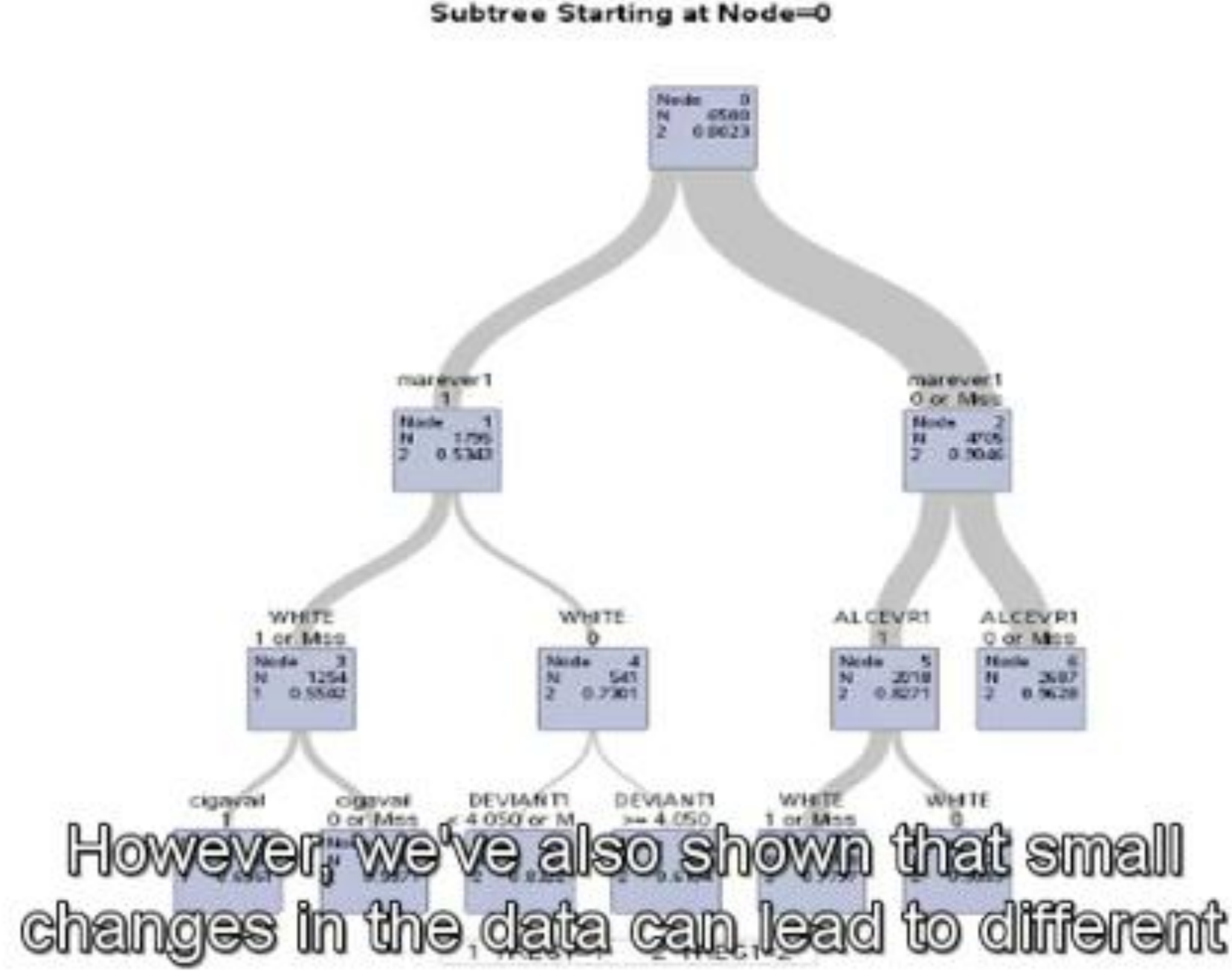
As we have seen, an advantage of decision trees is that they're easy to interpret



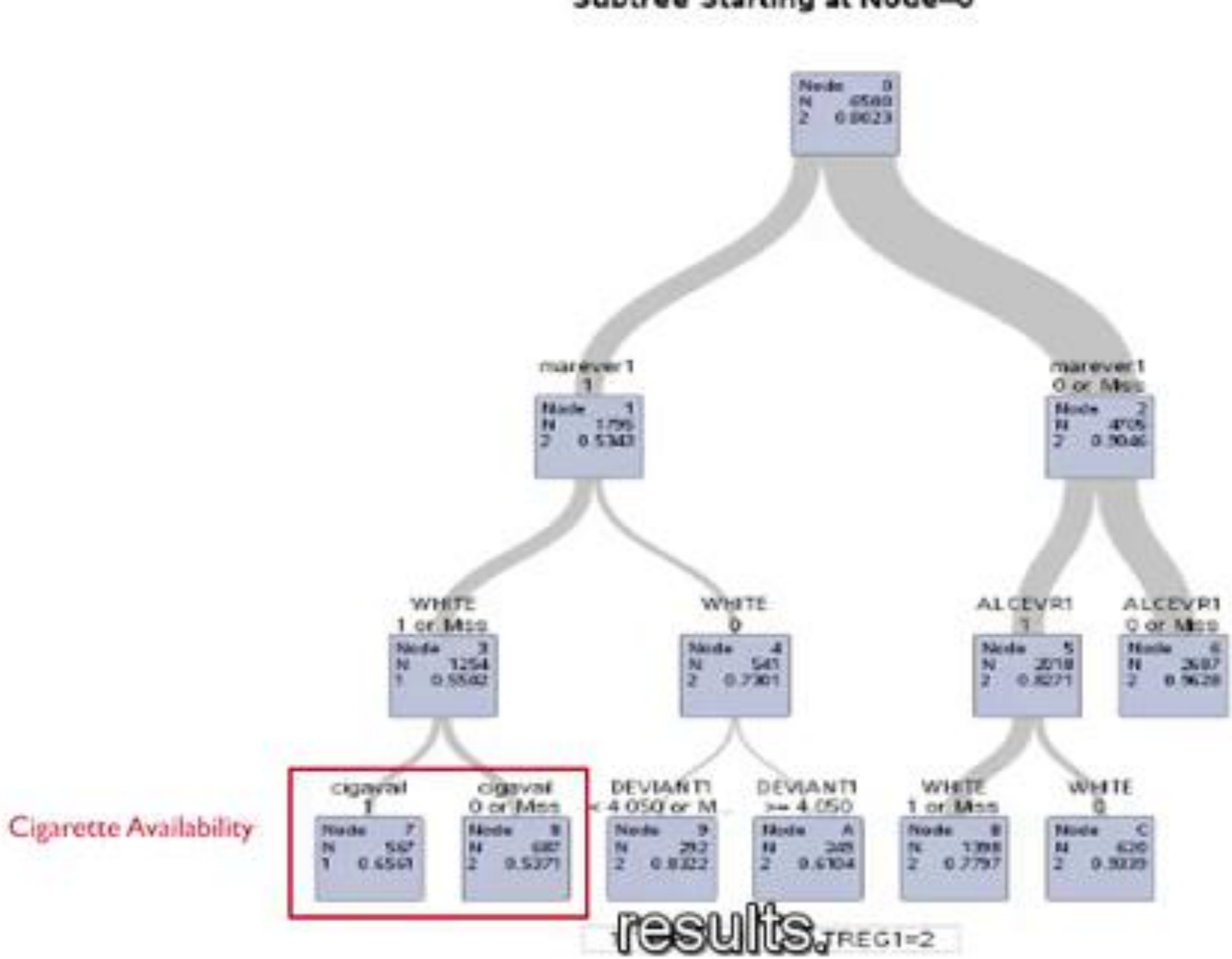
and visualize and can potentially uncover patterns in our data that can not be



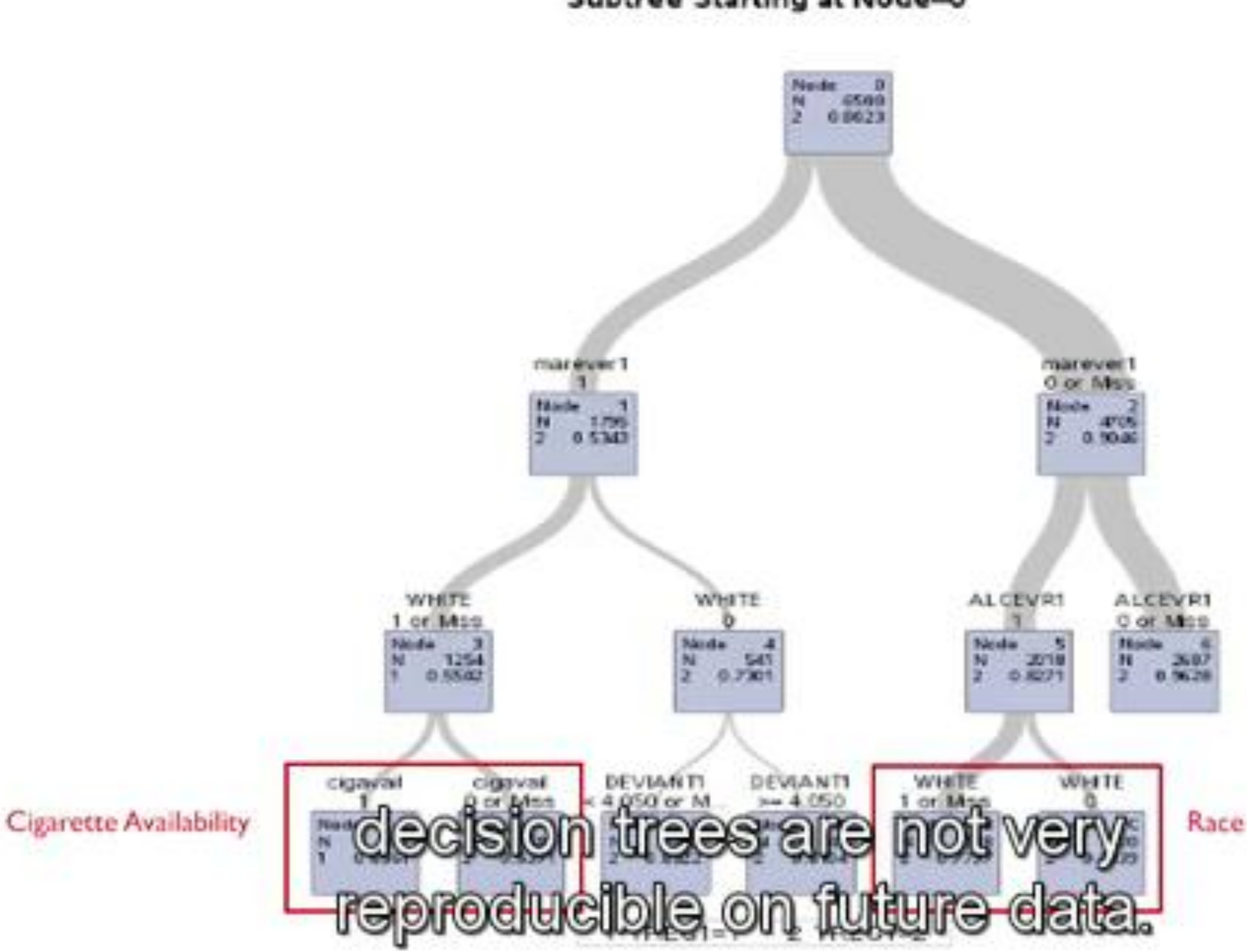
easily identified through traditional regression methods.

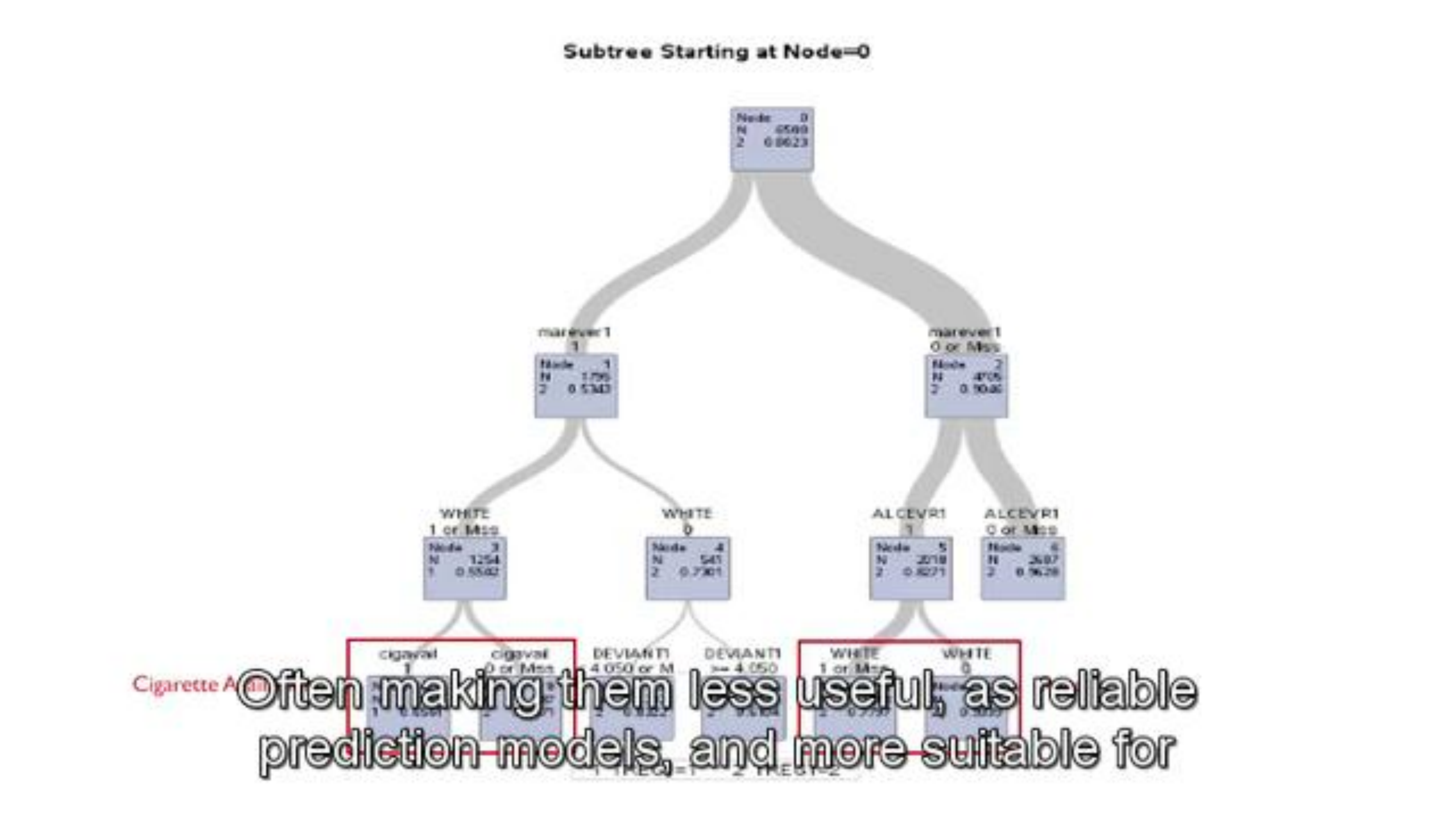


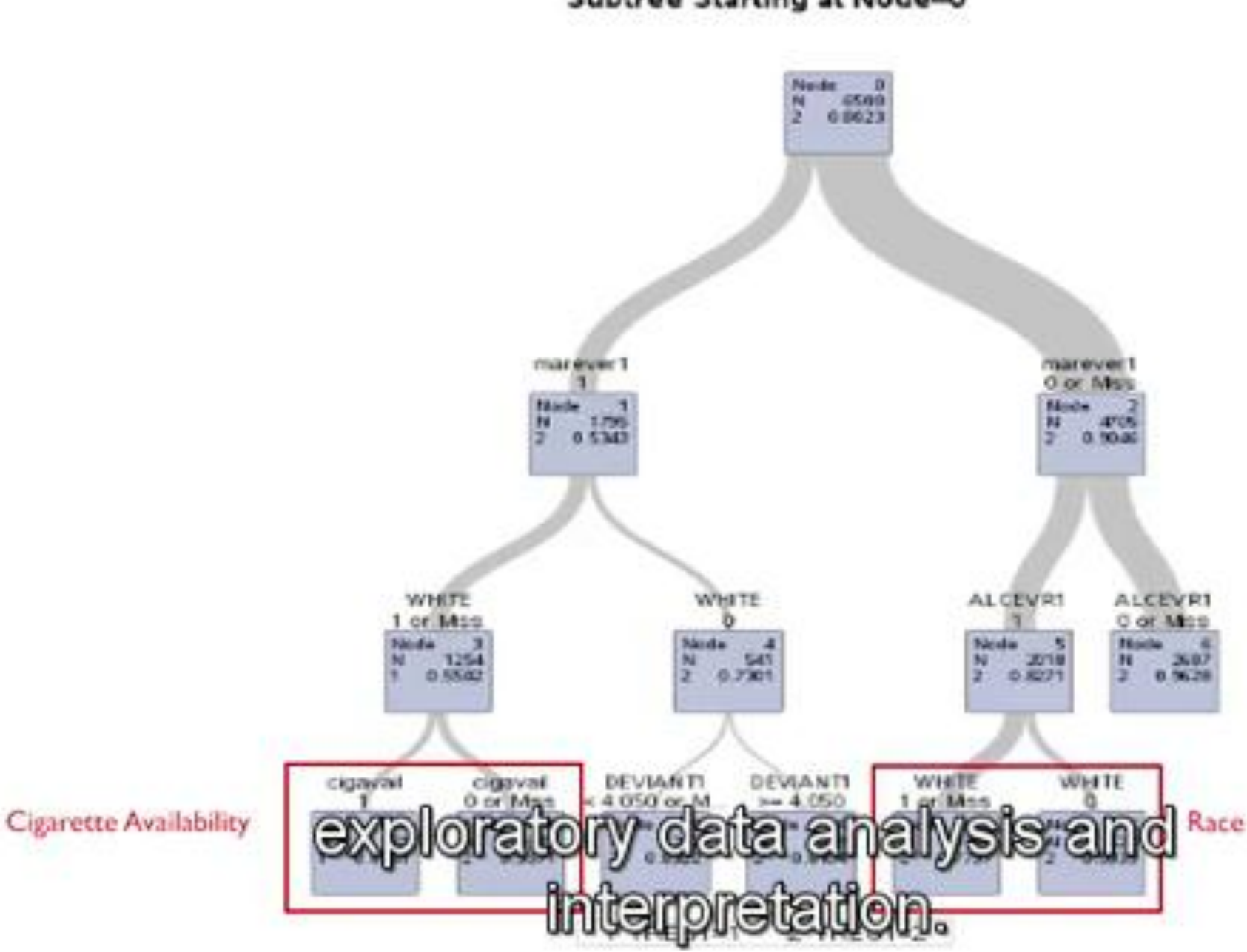
However, we've also shown that small changes in the data can lead to different



Cigarette Availability


Subtree Starting at Node=0
marever1
1
marever1
0 or Miss
WHITE
1 or Miss
WHITE
0
ALCEVR1
1
ALCEVR1
0 or Miss
cigaval
1
cigaval
0 or Miss
DEVIANT1
< 4.050 or M
DEVIANT1
>= 4.050
WHITE
1 or Miss
WHITE
0
Cigarette Availability
Race
decision trees are not very
reproducible on future data.


Subtree Starting at Node=0
Node 0
N 6569
2 0.8623
marever1
1
Node 1
N 1795
2 0.5343
marever1
0 or Miss
Node 2
N 4705
2 0.9936
WHITE
1 or Miss
Node 3
N 1254
1 0.5502
WHITE
0
Node 4
N 541
2 0.7301
ALCEVR1
1
Node 5
N 2019
2 0.8271
ALCEVR1
0 or Miss
Node 6
N 3687
2 0.9628
cigaval
1
cigaval
0 or Miss
DEVIANT1
4.050 or M
DEVIANT1
>= 4.050
WHITE
1 or Miss
WHITE
0
Cigarette A
Often making them less useful, as reliable
prediction models, and more suitable for



exploratory data analysis and interpretation.

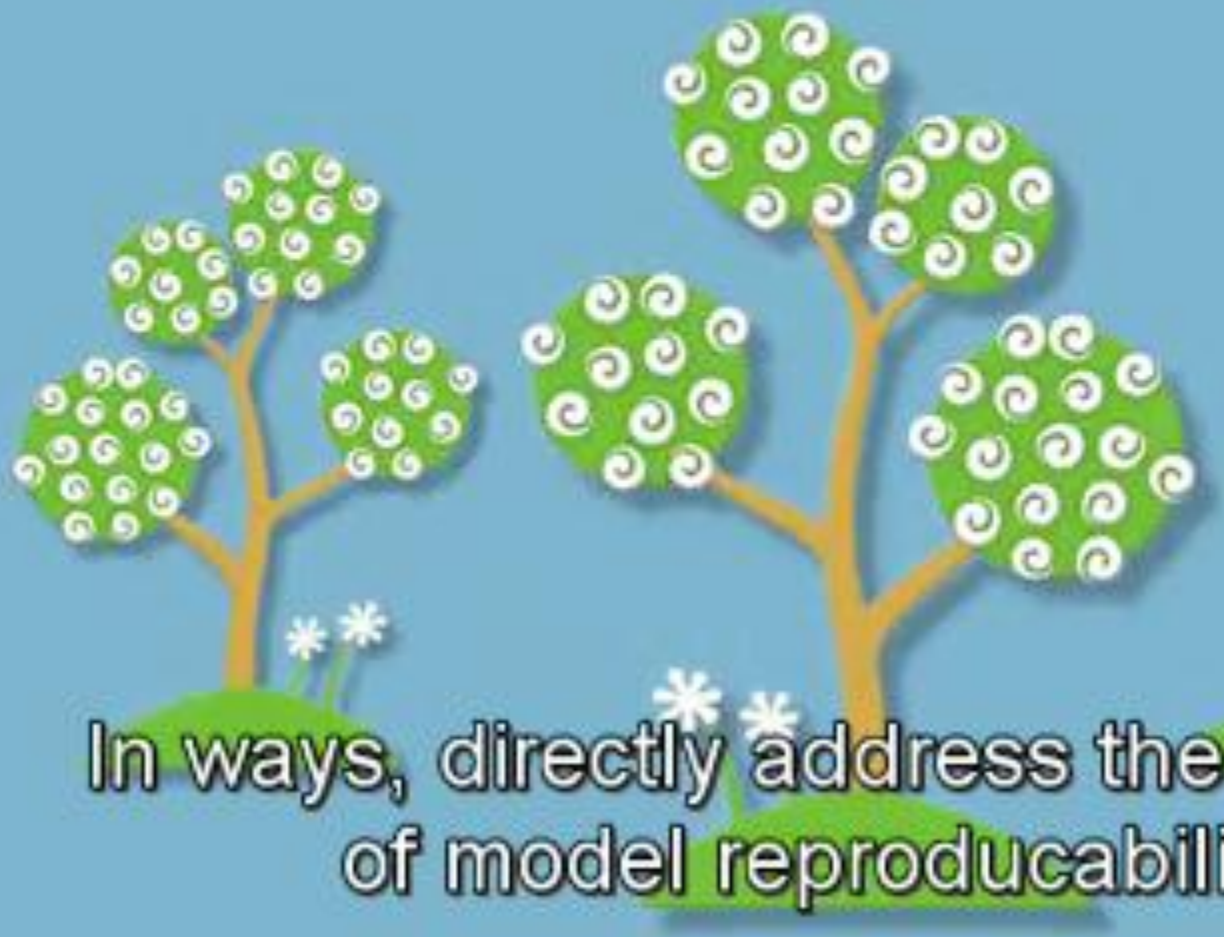
Random Forests

This data mining algorithm is based on decision trees, but

Random Forests

proceeds by growing many trees,
that is a decision tree forest.

Random Forests

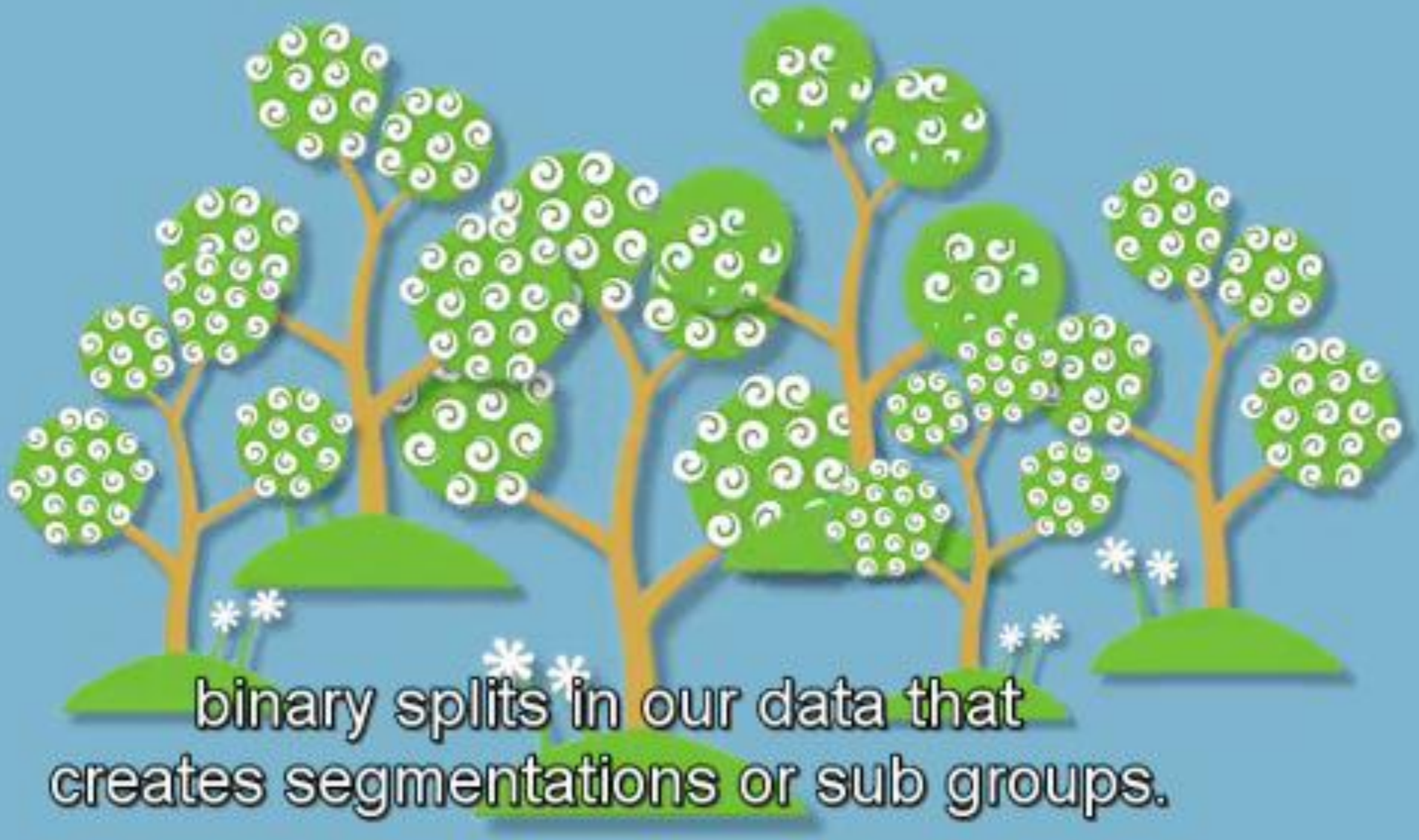


In ways, directly address the problem
of model reproducability.

Random Forests

Like decision trees,
Random Forests allow us to make

Random Forests



binary splits in our data that
creates segmentations or sub groups.

Random Forests

By applying a series of simple rules or criteria over and

Random Forests

over again, which choose variables
that best predict our target variable.

Random Forests

Splits on only ONE variable in a node

Variable with largest association with Target among candidate explanatory variables

ONLY among those variables that were randomly selected

variables that have been randomly selected to be tested for that node.

Random Forests

First a subset of explanatory variables is selected at random

Next the node is split with the BEST variable of the subset

After this node is split, a new list of eligible variables is selected at random

Random Forests

This continues until the tree is fully grown

Ideally, there will be only one observation in each termnial node

Random Forests

Eligible variable set will be different from node to node

Important variables eventually make it to the tree

Their relative success in predicting the target variable will get them more "votes"

Each Tree is Grown by:

A subset of the explanatory variables at each node

AND

A random subset of the sample for each tree in the forest

Bagging

This process of selecting a random sample of observations is known as Bagging.

Bagged and Unbagged Data

Importantly, each tree is growing
on a different randomly selected

Bagged and Unbagged Data

Bagged and Unbagged Data

available to test
the accuracy of each tree.

Bagged

Out of Bag

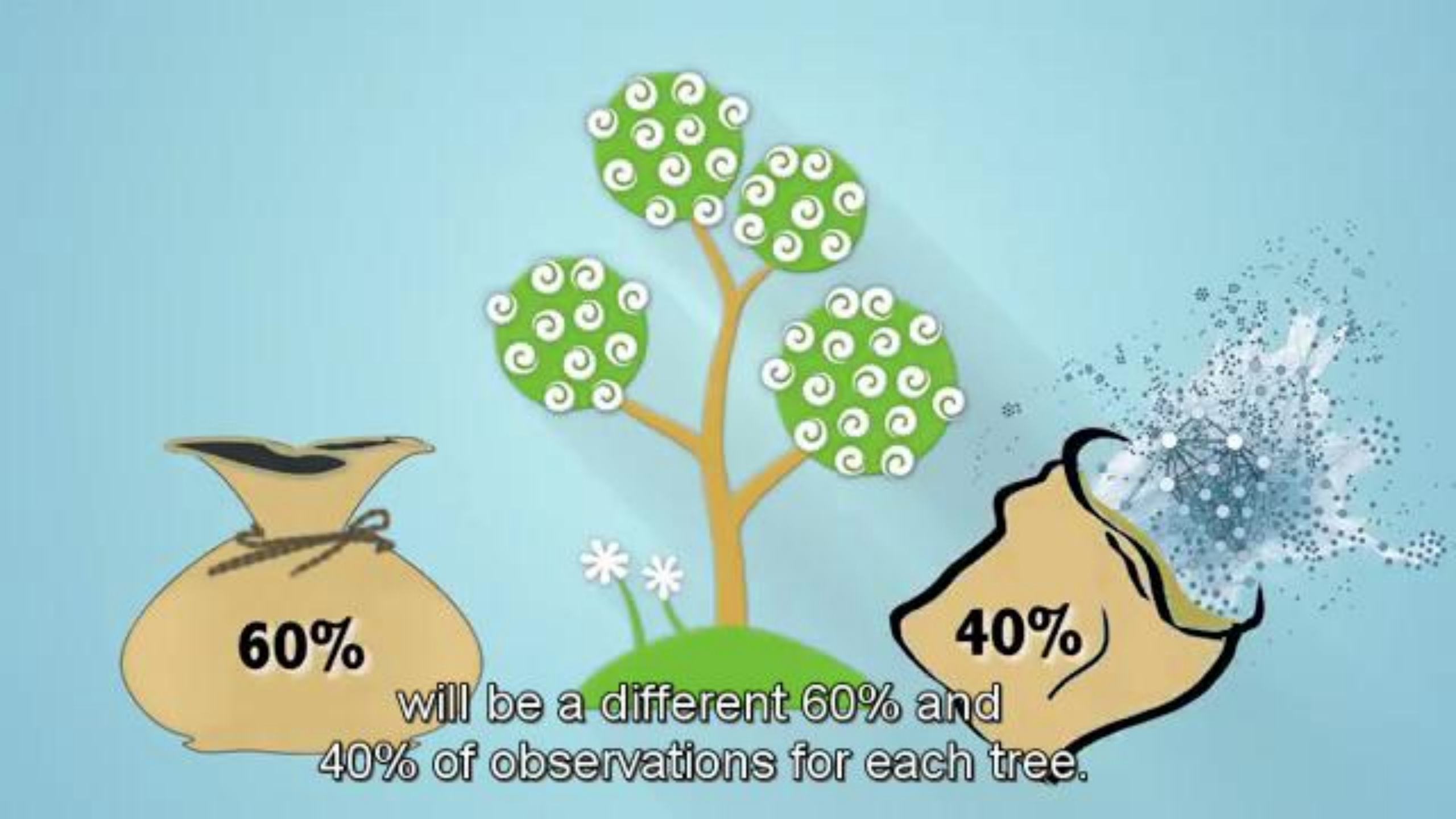
Bagging Process

For each tree, the Bagging Process
selects about 60% of the original sample,

Bagging Process

while the resulting tree is tested
against the remaining 40% of the sample.


Thus, the randomly selected bag data and
out of bag data,


60%
40%
will be a different 60% and
40% of observations for each tree.

Important!

I want to mention the most important thing
to know when interrupting the results of

Important!

Trees generated are not themselves interpreted

They are used collectively to rank the importance of variables in predicting the target of interest

[MUSIC]

Module 2
Lesson 2 - Building a Random Forest with Python

Target Variable: TREG1 (ever smoked regularly 1=yes and 2=no)

Explanatory Variables - Categorical

BIO_SEX (1=male, 2=female)
HISPANIC (1=yes, 0=no)
WHITE (1=yes, 0=no)
BLACK (1=yes, 0=no)
NAMERICAN (1=yes, 0=no)
ASIAN (1=yes, 0=no)
ALCEVR1 (ever drank alcohol 1=yes 0=no)
MAREVER1 (ever smoked marijuana 1=yes 0=no)
COCEVER1 (ever used cocaine 1=yes 0=no)
INHEVER1 (ever used inhalants 1=yes 0=no)
CIGAVAIL (cigarettes available in the home 1=yes 0=no)
PASSIST (either parent on public assistance 1=yes 0=no)
EXPEL1 (ever expelled from school 1=yes 0=no)

CIGAVAIL (cigarettes available in the home 1=yes 0=no)
PASSIST (either parent on public assistance 1=yes 0=no)
EXPEL1 (ever expelled from school 1=yes 0=no)

Explanatory Variables - Quantitative

AGE
ALCPROB1 (alcohol problems 0 to 6)
DEVIANT1 (deviant behavior scale)
VIOL1 (violent behavior scale)
DEP1 (depression scale)
ESTEEM1 (self esteem scale)
PARPRES (parental presence scale)
PARACTV (parent activities scale)
FAMCONCT (family connectedness scale)
SCHCONN1 (school connectedness scale)
GPA1 (Grade Point Average - 4.0 scale)

CIGAVAIL (cigarettes available in the home 1=yes 0=no)
PASSIST (either parent on public assistance 1=yes 0=no)
EXPEL1 (ever expelled from school 1=yes 0=no)

Have you ever smoked cigarettes regularly (1/day for 30 days)?

ALCPROB1 (alcohol problems 0 to 6)
DEVIANT1 (deviant behavior scale)
VIOL1 (violent behavior scale)
DEP1 (depression scale)
ESTEEM1 (self esteem scale)
PARPRES (parental presence scale)
PARACTV (parent activities scale)
FAMCONCT (family connectedness scale)
SCHCONN1 (school connectedness scale)
GPA1 (Grade Point Average - 4.0 scale)

```
pred_train, pred_test, tar_train, tar_test  = train_test_split(predictors, targets, test_size=.4)

pred_train.shape
pred_test.shape
tar_train.shape
tar_test.shape

#build model on training data
from sklearn.ensemble import RandomForestClassifier
```

```
Out[16]:
array([[1435,   82],
       [ 207.  106]])
```

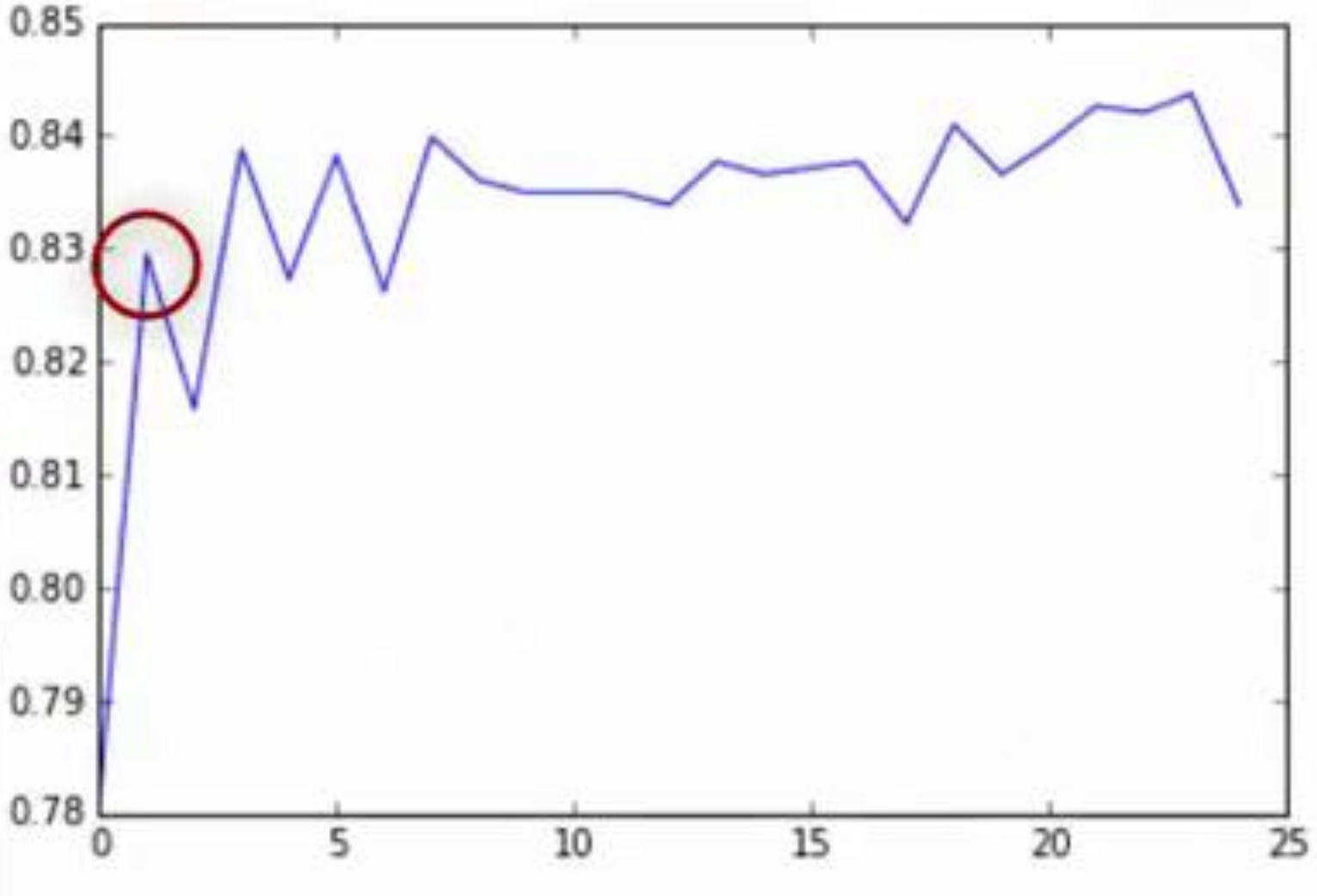
```
In [17]: sklearn.metrics.accuracy_score(tar_test, predictions)
    ...:
Out[17]: 0.8420765027322404
```

```
sklearn.metrics.
sklearn.metrics.

# fit an Extra Trees model to the data
model = ExtraTreesClassifier()
model.fit(pred_train,tar_train)
# display the relative importance of each attribute
print(model.feature_importances_)


"""
```

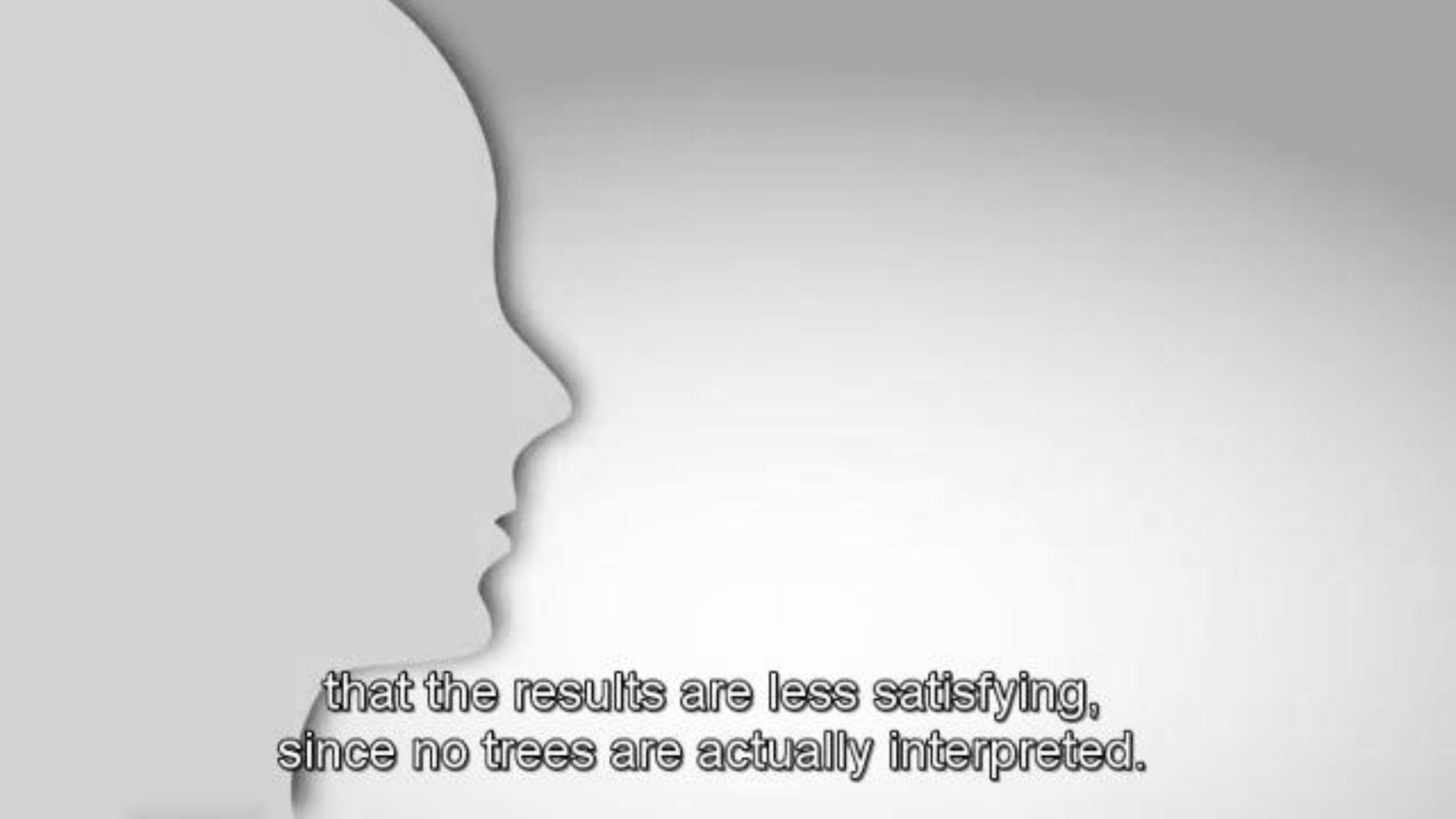
```
sklearn.metrics.confusion_matrix(tar_test,predictions)
sklearn.metrics.accuracy_score(tar_test, predictions)


# fit an Extra Trees model to the data
model = ExtraTreesClassifier()
model.fit(pred_train,tar_train)
# display the relative importance of each attribute
print(model.f
...:
[ 0.02600278  0.01396994  0.02567394  0.02442814  0.00774378  0.00644213
  0.05456547  0.0496497   0.04160439  0.13894647  0.01411572  0.0161918
  0.02505653  0.05602452  0.05461356  0.0456791   0.01462392  0.07791334
  0.05771935  0.07464764  0.01431716  0.05608567  0.05622758  0.04775738]

"""
Impact of tre

Running a different number of trees and see the effect
 of that on the accuracy of the prediction
"""

trees=range(25)
accuracy=np.zeros(25)

for idx in range(len(trees)):
    classifier=RandomForestClassifier(n_estimators=idx + 1)
    classifier=classifier.fit(pred_train,tar_train)
    predictions=classifier.predict(pred_test)
    accuracy[idx]=sklearn.metrics.accuracy_score(tar_test, predictions)

```

```
sklearn.metrics.confusion_matrix(tar_test,predictions)
sklearn.metrics.accuracy_score(tar_test, predictions)


# fit an Extra Trees model to the data
model = ExtraTreesClassifier()
model.fit(pred_train,tar_train)
# display the relative importance of each attribute
print(model.f
...:
[ 0.02600278  0.01396994  0.02567394  0.02442814  0.00774378  0.00644213
  0.05456547  0.0496497   0.04160439  0.13894647  0.01411572  0.0161918
  0.02505653  0.05602452  0.05461356  0.0456791   0.01462392  0.07791334
  0.05771935  0.07464764  0.01431716  0.05608567  0.05622758  0.04775738]

"""
Impact of tr

Running a dif
 of that on t
"""

predictors = data_clean[['BIO_SEX','HISPANIC','WHITE','BLACK','NAMERICAN','ASIAN','age',
'ALCEVR1','ALCPROBS1','marever1','cocever1','inhever1','cigavail','DEP1','ESTEEM1','VIOL1',
'PASSIST','DEVIANT1','SCHCONN1','GPA1','EXPEL1','FAMCONCT','PARACTV','PARPRES']]

trees=range(25)
accuracy=np.zeros(25)

for idx in range(len(trees)):
    classifier=RandomForestClassifier(n_estimators=idx + 1)
    classifier=classifier.fit(pred_train,tar_train)
    predictions=classifier.predict(pred_test)
    accuracy[idx]=sklearn.metrics.accuracy_score(tar_test, predictions)
```

```
print(model.feature_importances_)


"""
Impact of tree size on prediction accuracy

Running a different number of trees and see the effect
 of that on the accuracy of th
"""

trees=range(25)
accuracy=np.zeros(25)

for idx in range(len(trees)):
    classifier=RandomForestClas
    classifier=classifier.fit(p
    predictions=classifier.pred
    accuracy[idx]=sklearn.metri

plt.cla()
plt.plot(trees, accuracy)
```



In my opinion, the main weakness
of random forests is simply


that the results are less satisfying,
since no trees are actually interpreted.

Instead, the forest of trees is used to

rank the importance of variables
in predicting the target.

Thus we get a sense of the most
important predictive variables but

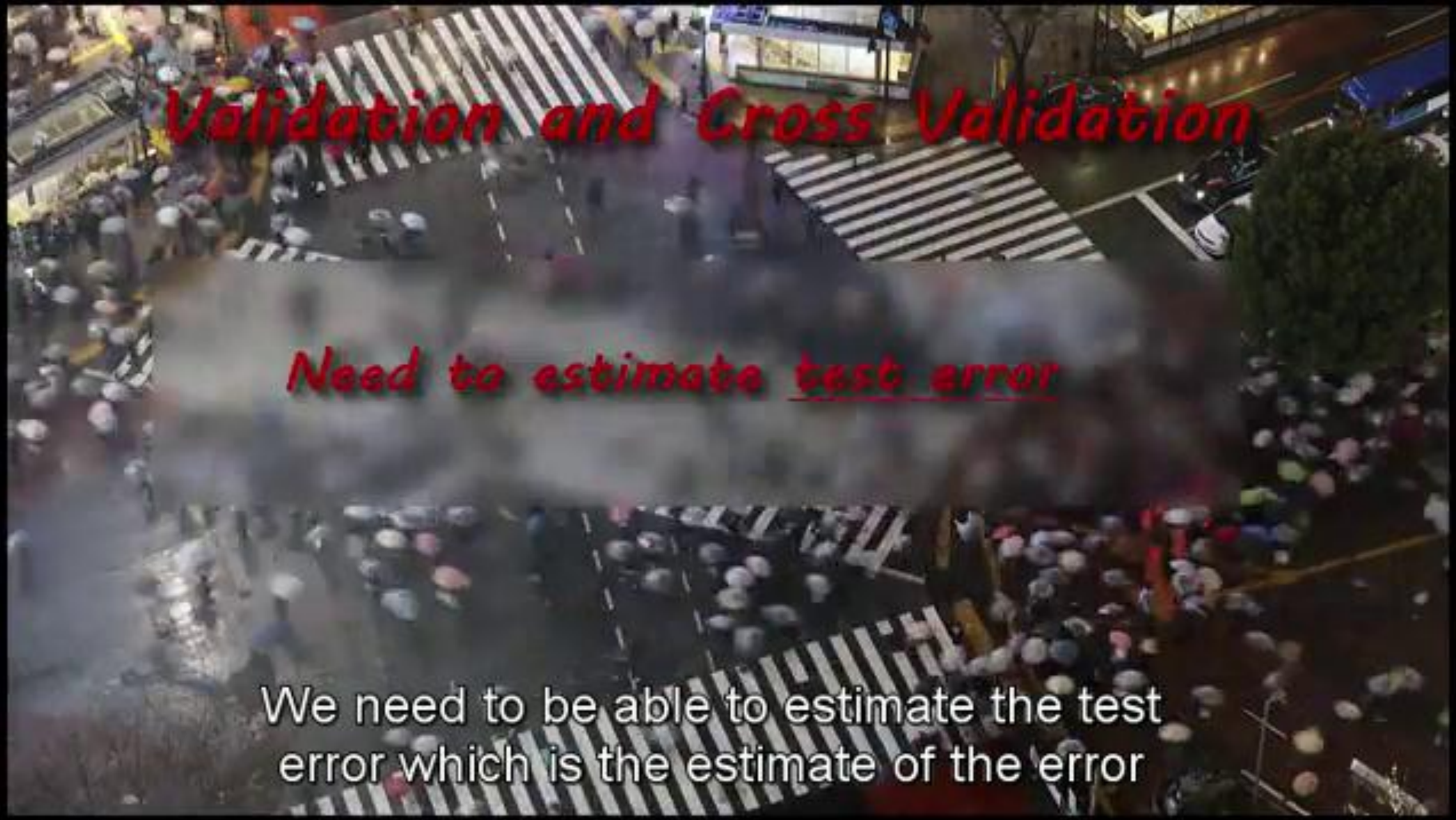
not necessarily their
relationships to one another.

Module 2
Lesson 4 - Validation and Cross-Validation


Validation and Cross Validation
Validation and cross-validation are
critical in the machine learning process.


Validation and Cross Validation
Training set model estimation capitalizes on random, sample specific patterns and associations
As we noted in
the Buy Experience Tradeoff video,

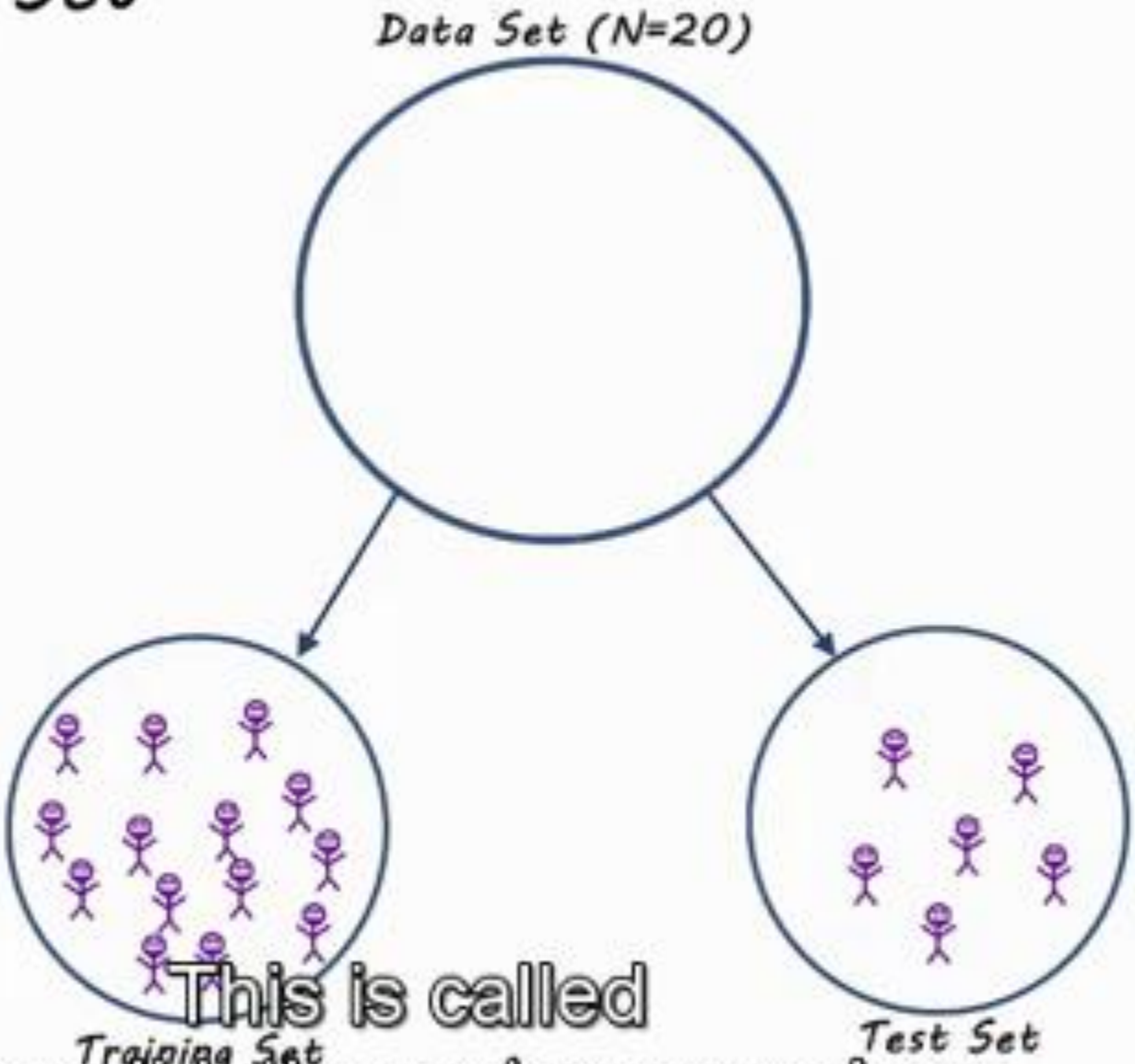

Validation and Cross Validation
How do we know which model is the best model??


Validation and Cross Validation
Need to estimate test error
We need to be able to estimate the test error which is the estimate of the error

Validation Set

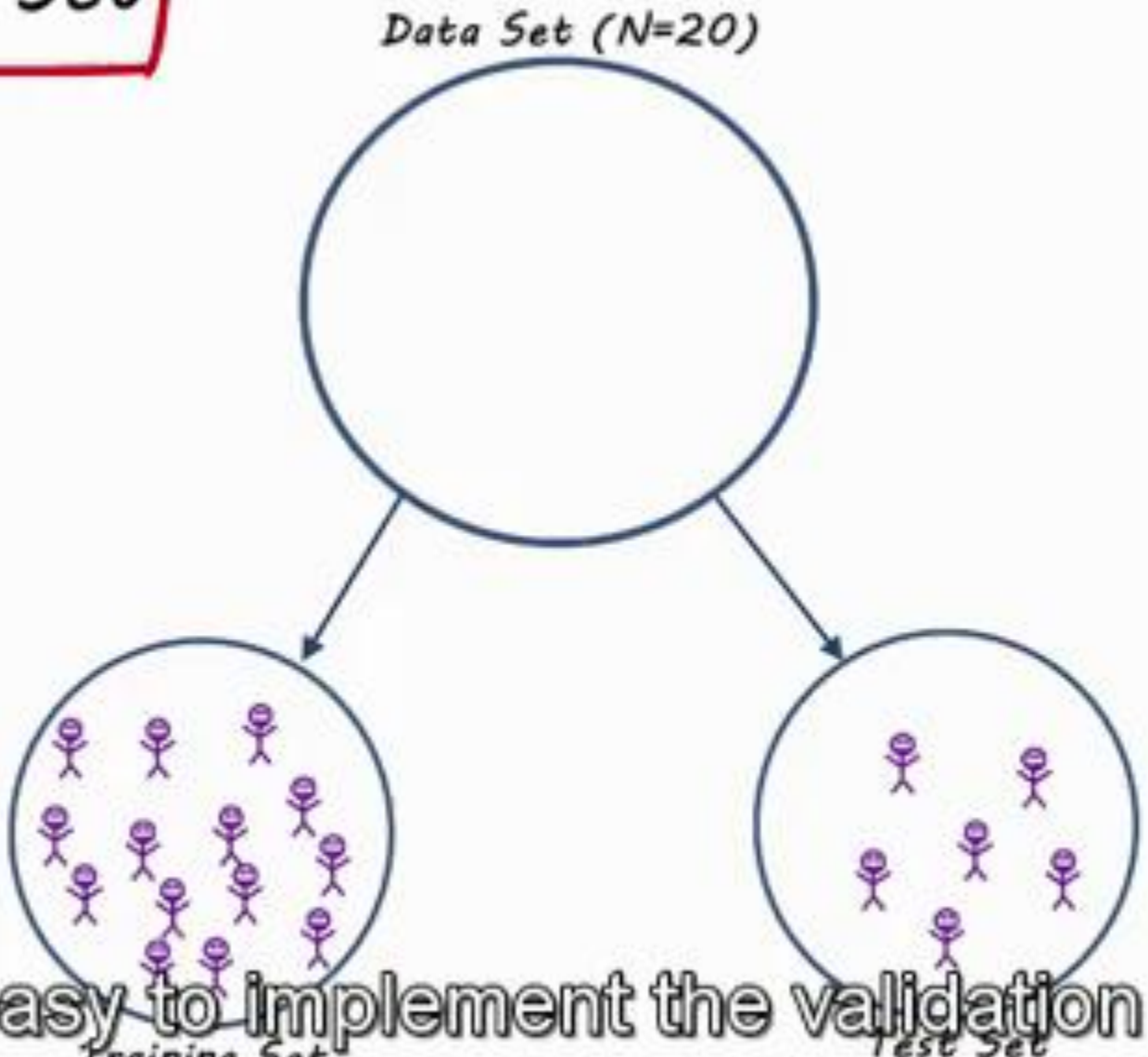
One way to do this is to randomly split the data into training and test or

Validation Set



This is called
the validation set approach,

Validation Set

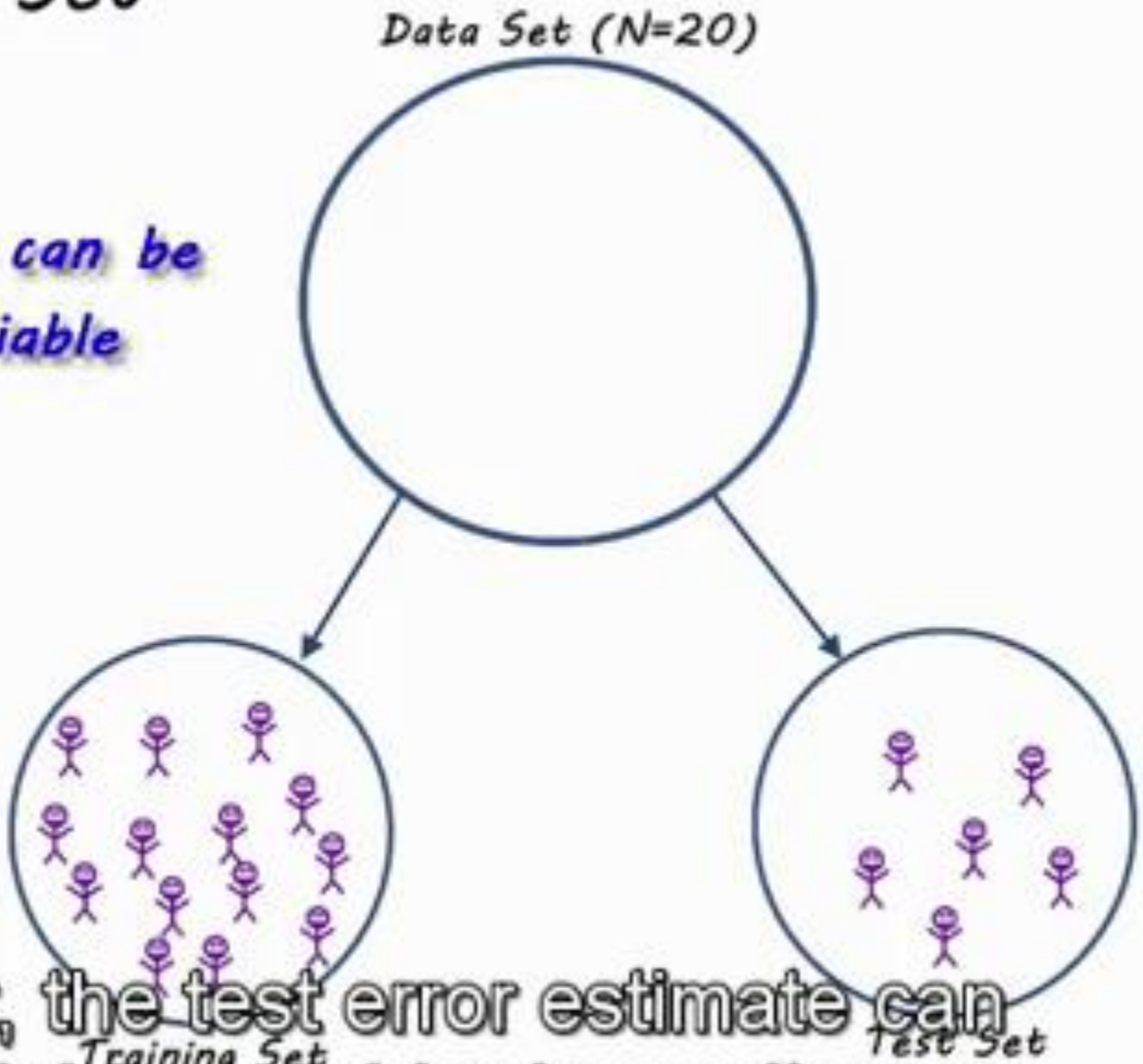


while easy to implement the validation set approach has a couple of drawbacks.

Validation Set

Drawbacks:

- Test error can be highly variable

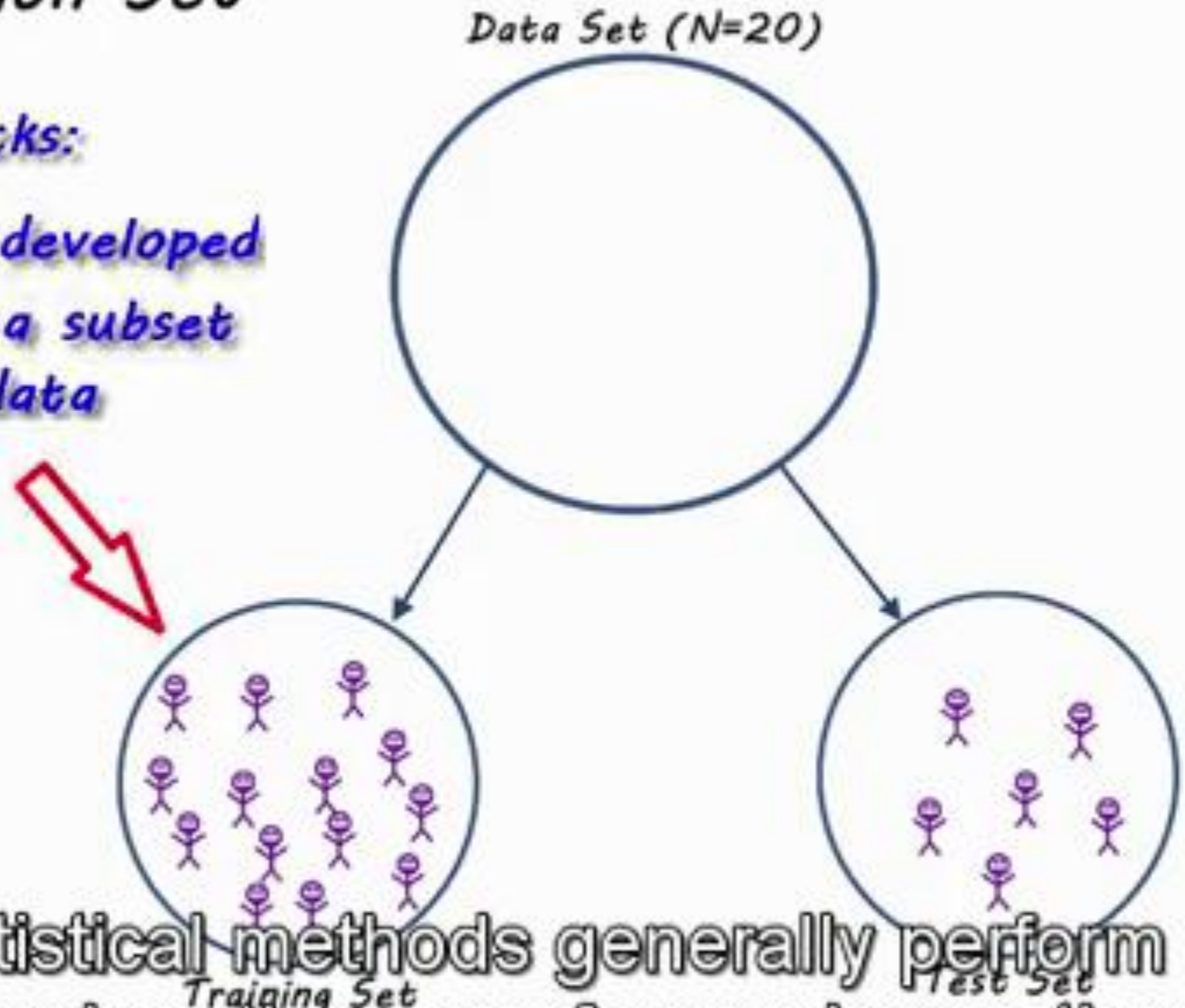


First, the test error estimate can be highly variable depending on

Validation Set

Drawbacks:

- Model developed on only a subset of the data

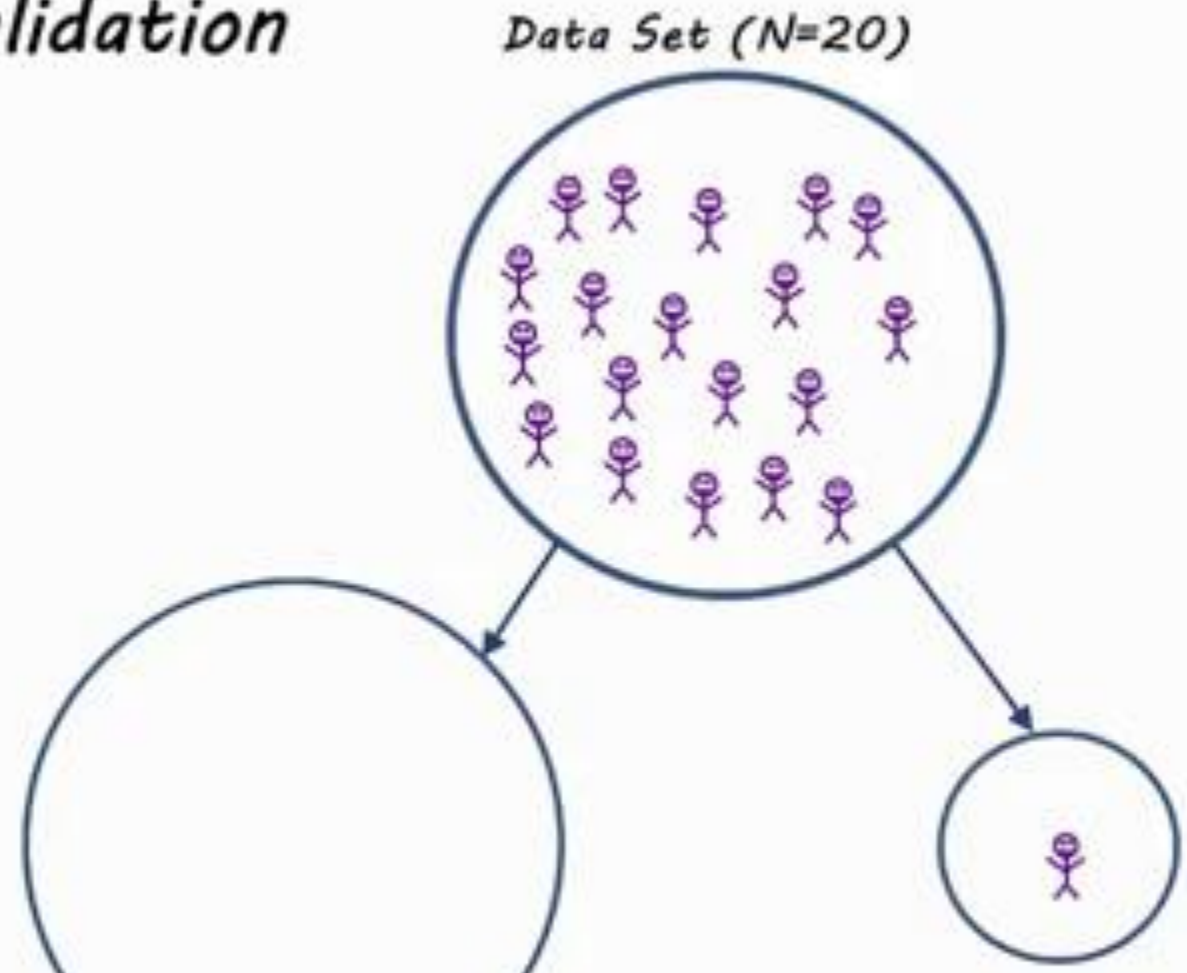


Statistical methods generally perform worse when there are fewer observations,

Cross Validation

- Goal is to define a data set to "test" the model during the training phase

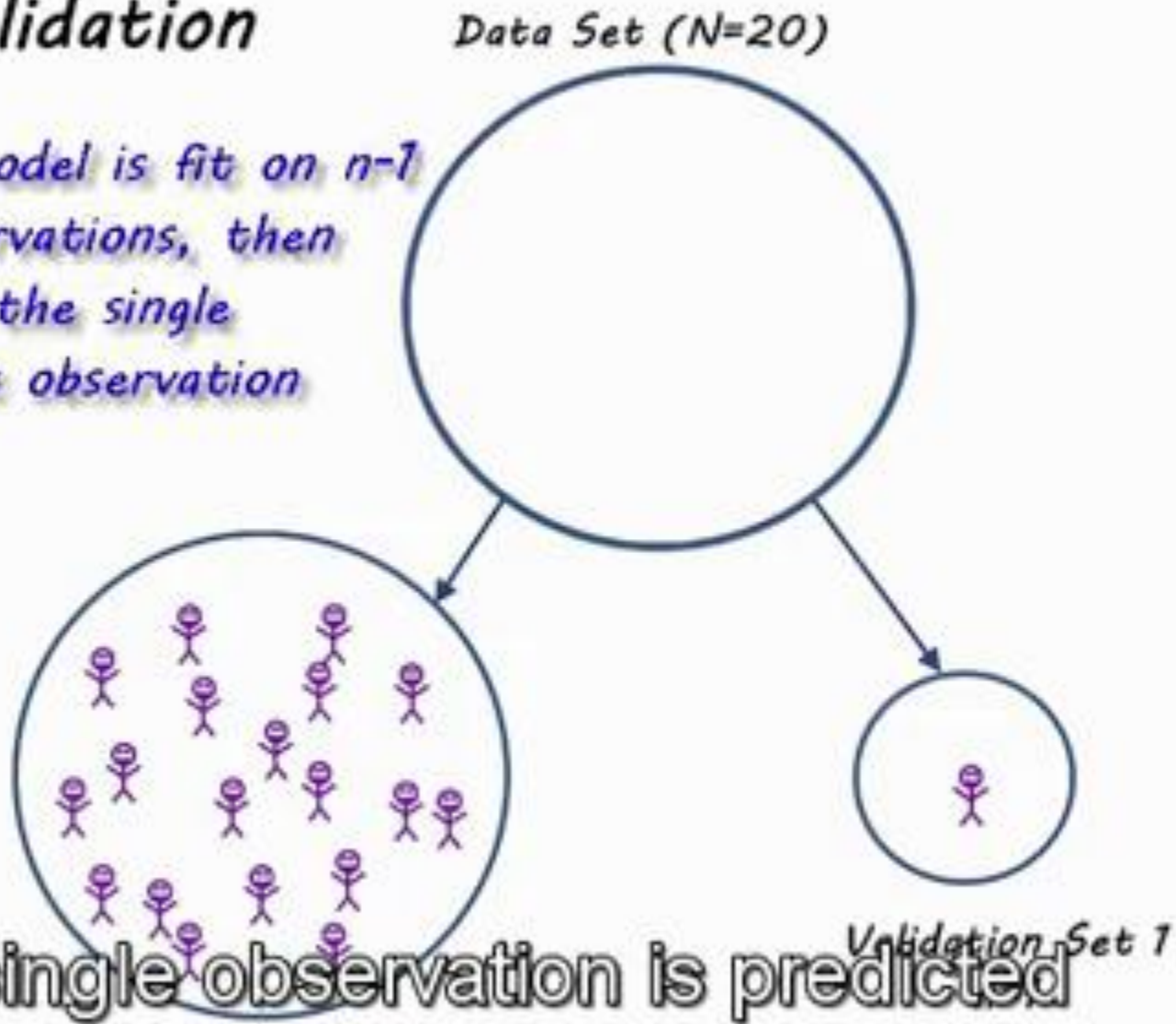
Leave One Out Cross Validation



holds out one observation from the training set for validation.

Leave One Out Cross Validation

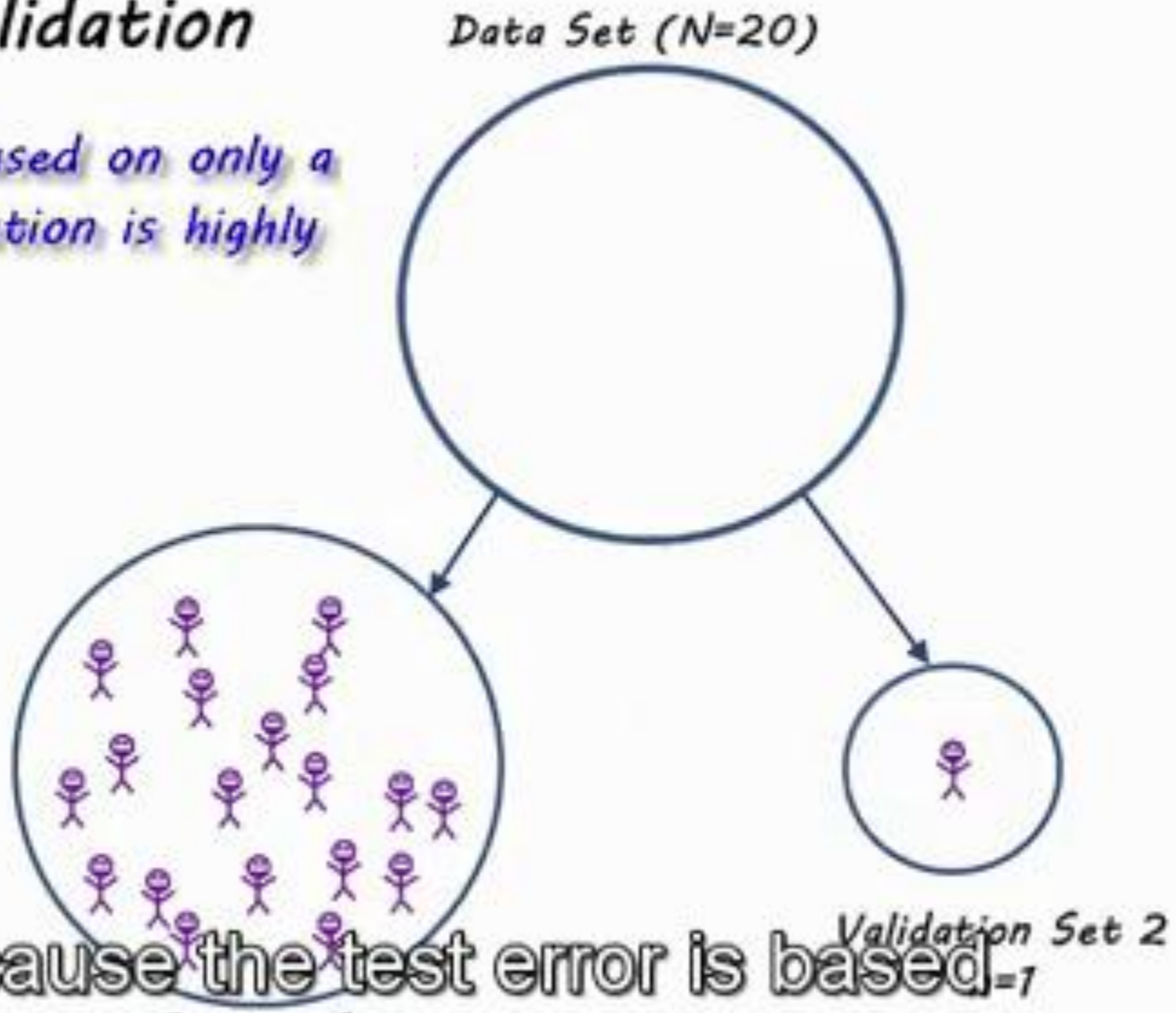
- Statistical model is fit on n-1 training observations, then validated on the single validation set observation



the single observation is predicted based on the values of the predictors.

Leave One Out Cross Validation

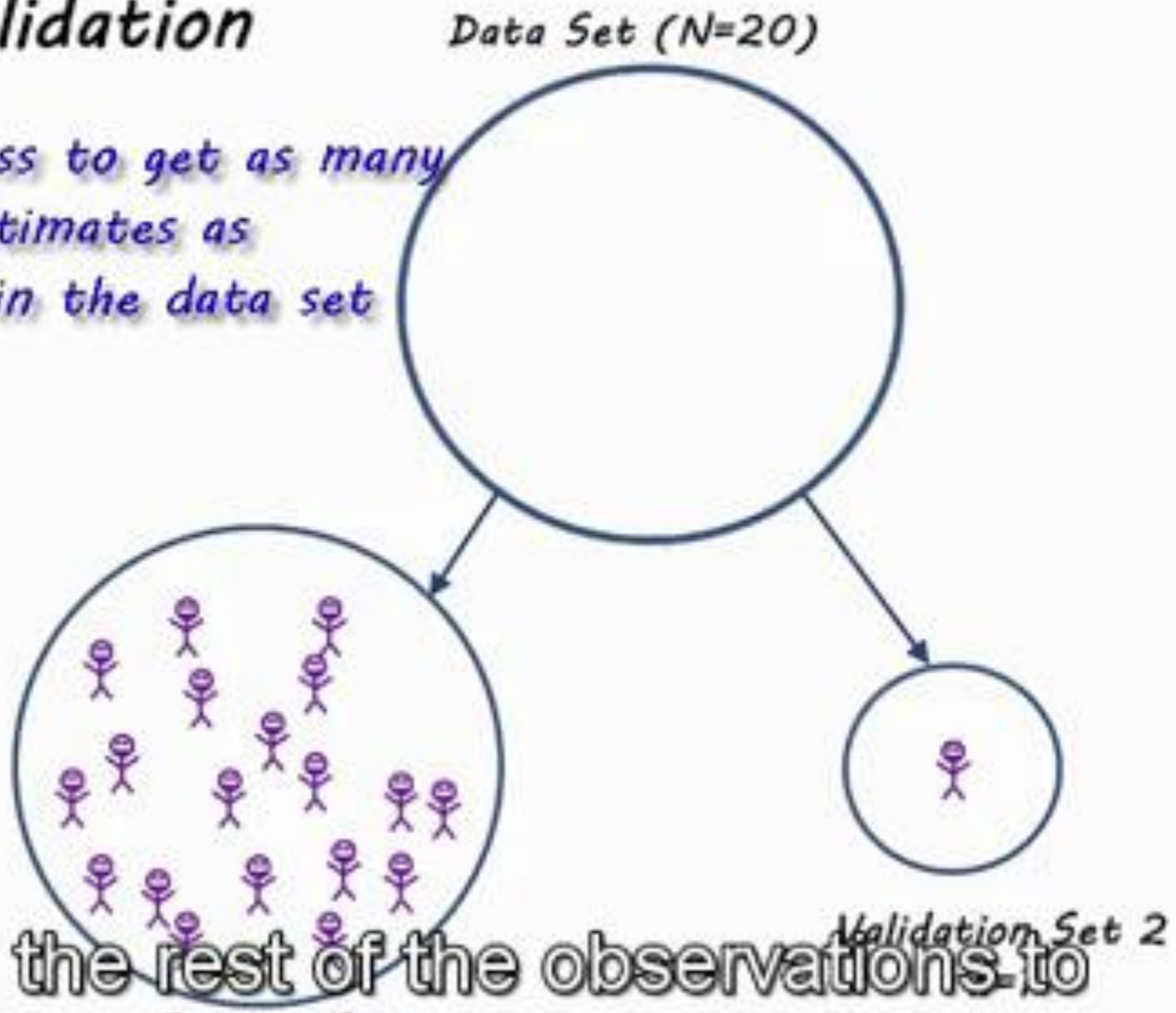
- Test error based on only a single observation is highly variable



Because the test error is based on only a single observation,

Leave One Out Cross Validation

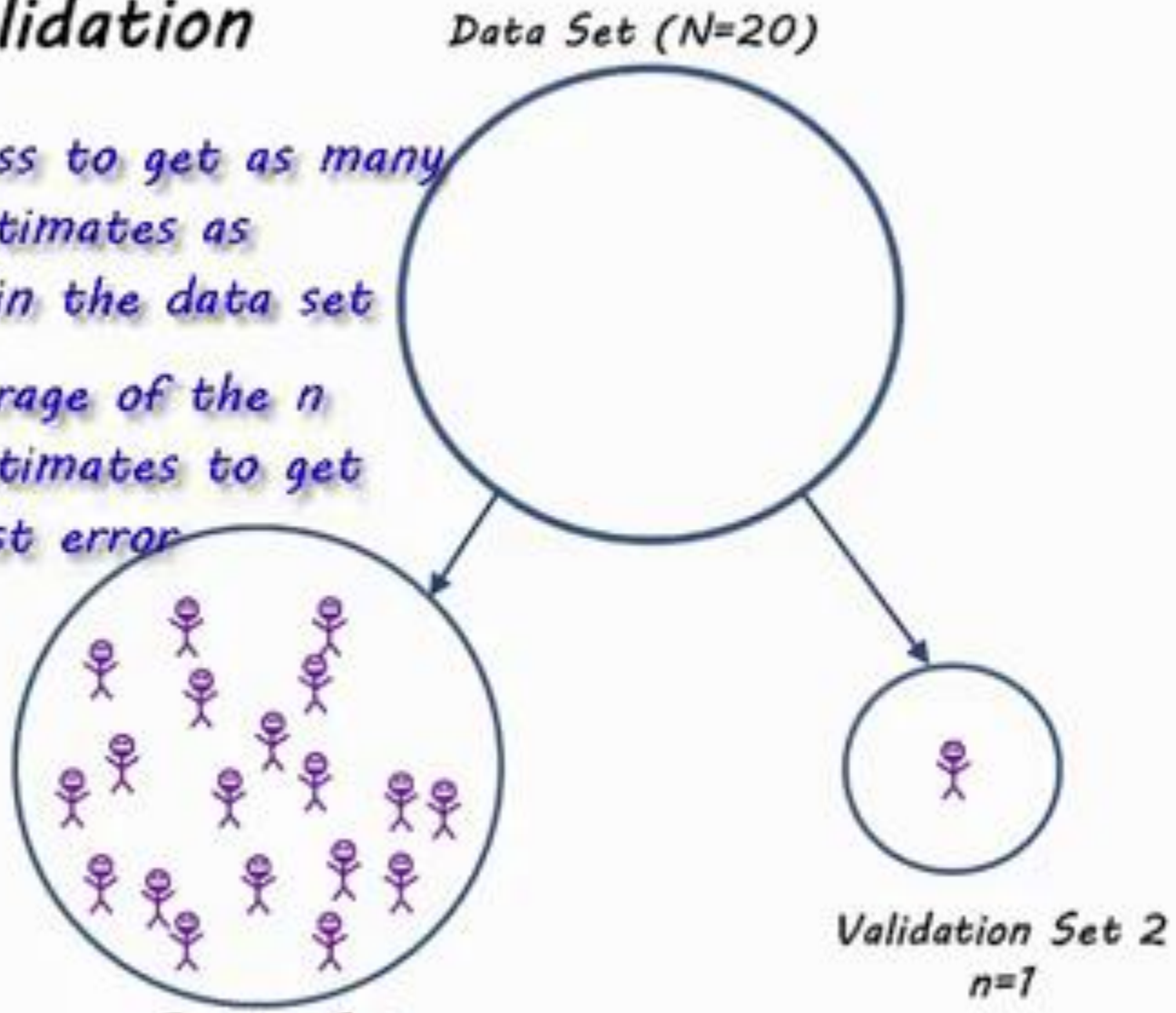
- Repeat process to get as many test error estimates as observations in the data set



using the rest of the observations to train the model, then we will end up

Leave One Out Cross Validation

- Repeat process to get as many test error estimates as observations in the data set
- Compute average of the n test error estimates to get an overall test error estimate



to get an overall test error estimate.

Leave One Out Cross Validation (LOOCV)

Advantages

1. Less bias in regression coefficients
2. Parameter estimates don't vary across training samples

Leave One Out Cross Validation (LOOCV)

Advantages

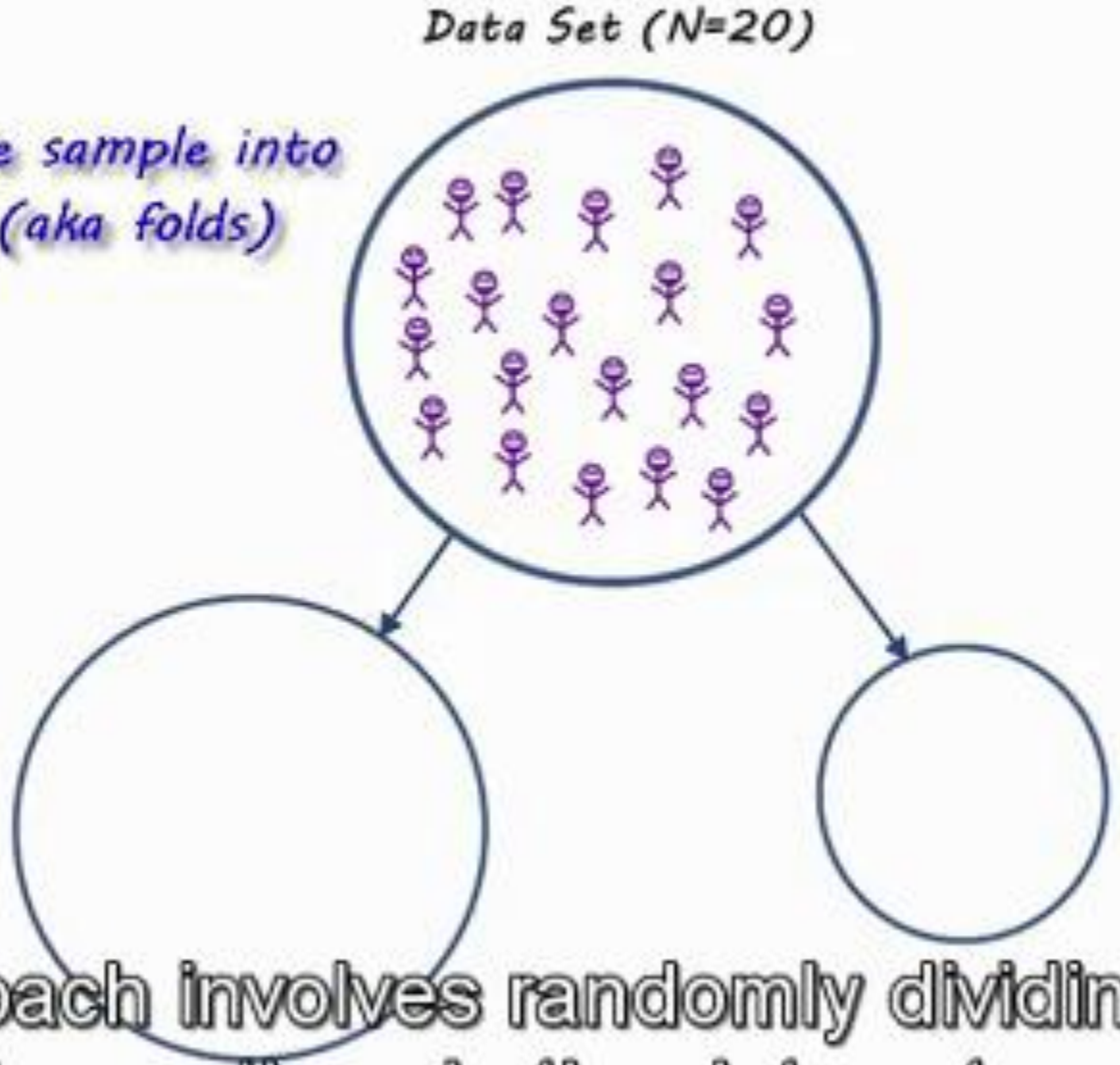
1. Less bias in regression coefficients
2. Parameter estimates don't vary across training samples

Disadvantages

Time-consuming and computationally intensive, especially with large data sets

K-fold Cross Validation

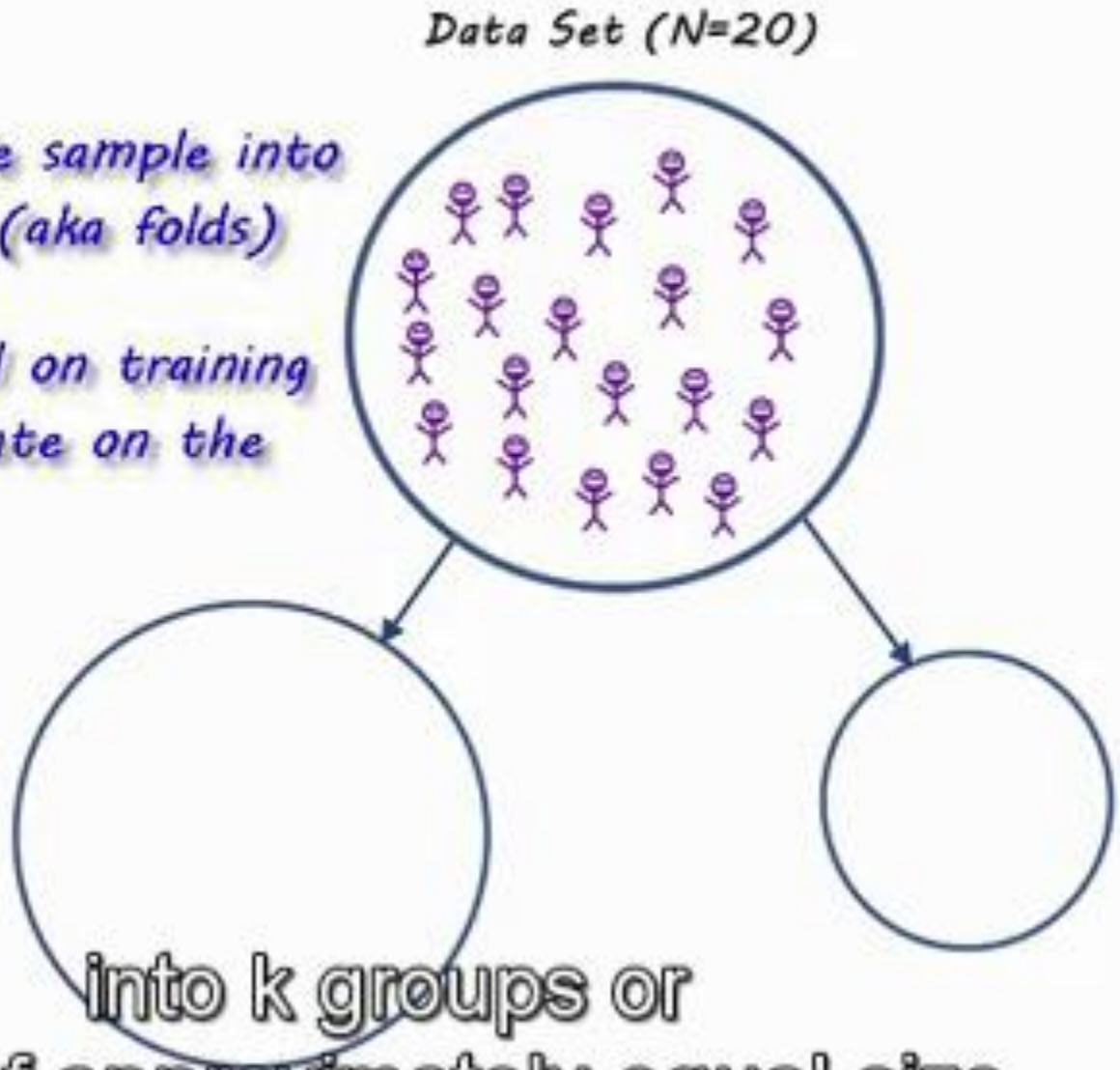
- Randomly divide sample into k equal groups (aka folds)



This approach involves randomly dividing the observations in the data set

K-fold Cross Validation

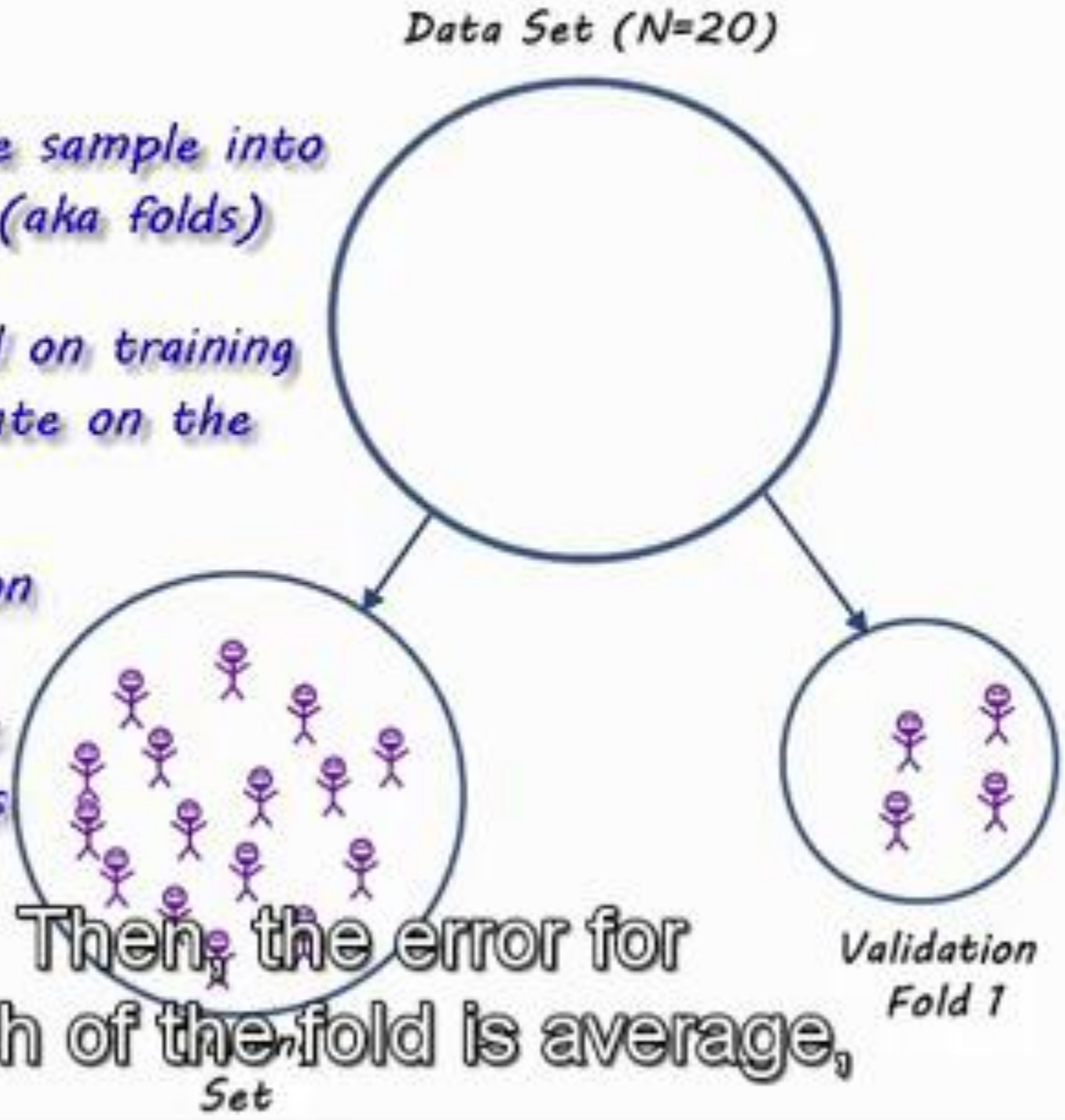
- Randomly divide sample into k equal groups (aka folds)
- Estimate model on training set, then validate on the validation fold



into k groups or
folds of approximately equal size.

K-fold Cross Validation

- Randomly divide sample into k equal groups (aka folds)
- Estimate model on training set, then validate on the validation fold
- Repeat process on remaining folds, and average test error across folds

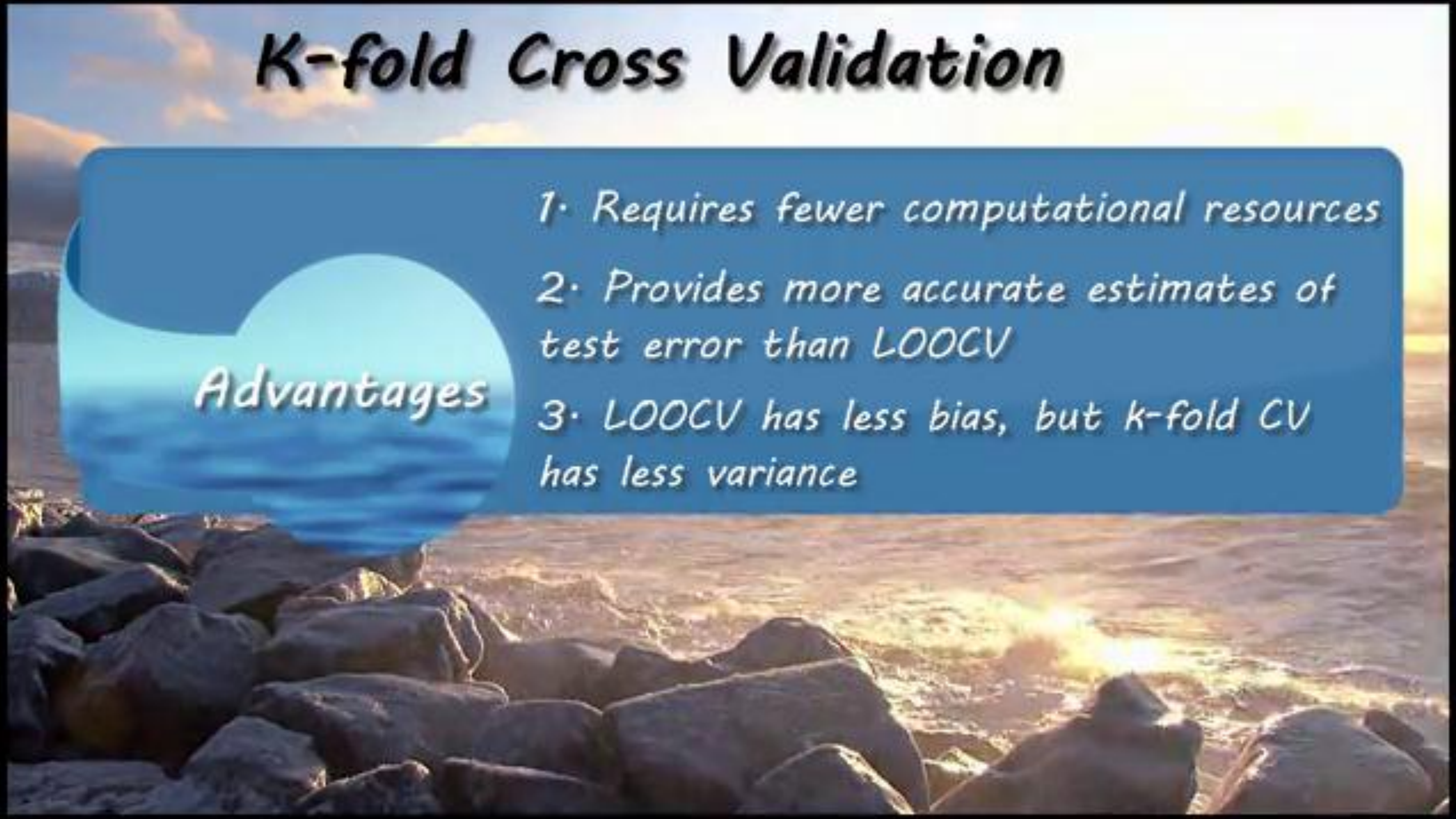


Then, the error for each of the fold is average,

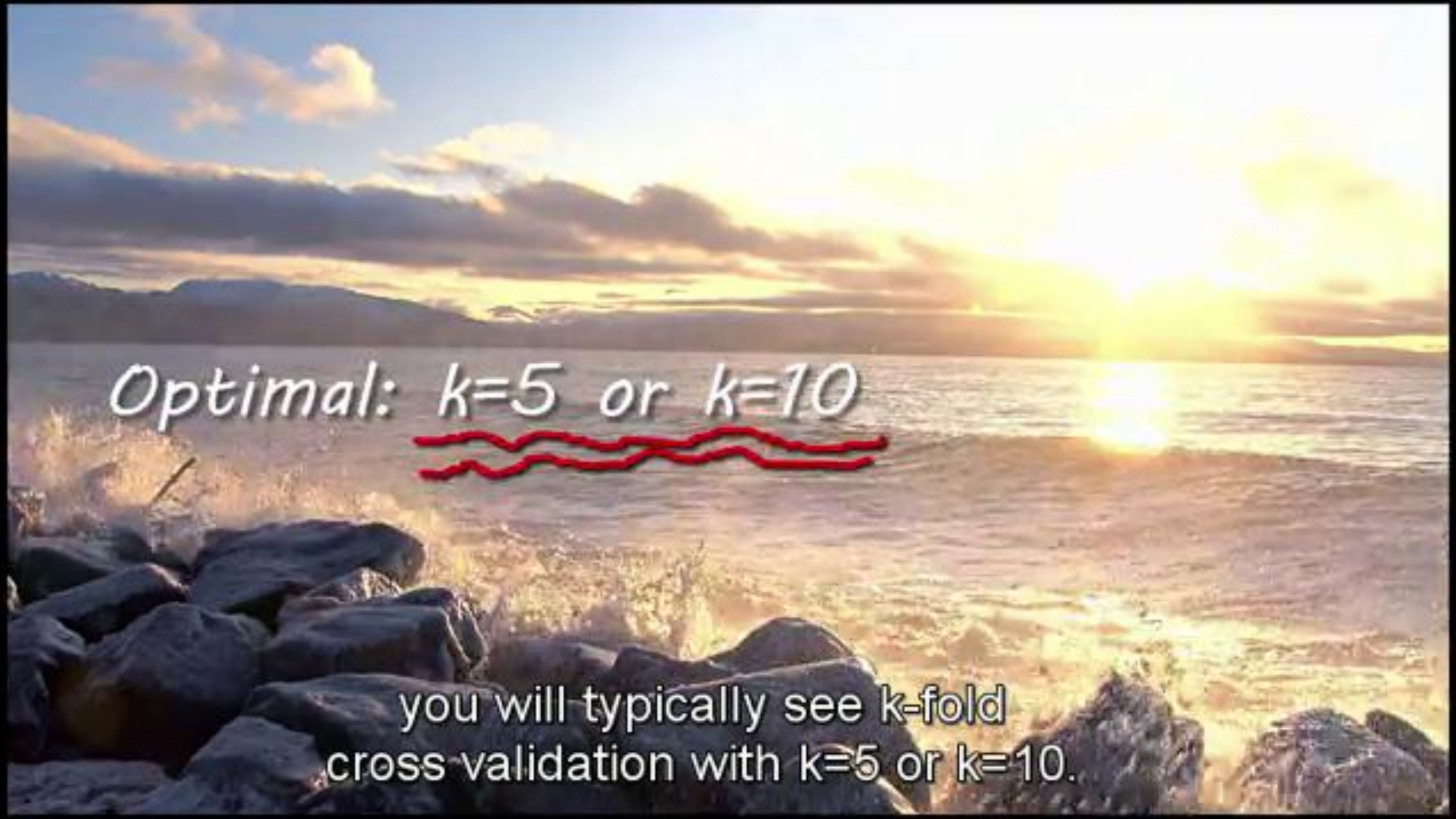
K-fold Cross Validation

Advantages

1. Requires fewer computational resources
2. Provides more accurate estimates of test error than LOOCV
3. LOOCV has less bias, but k-fold CV has less variance



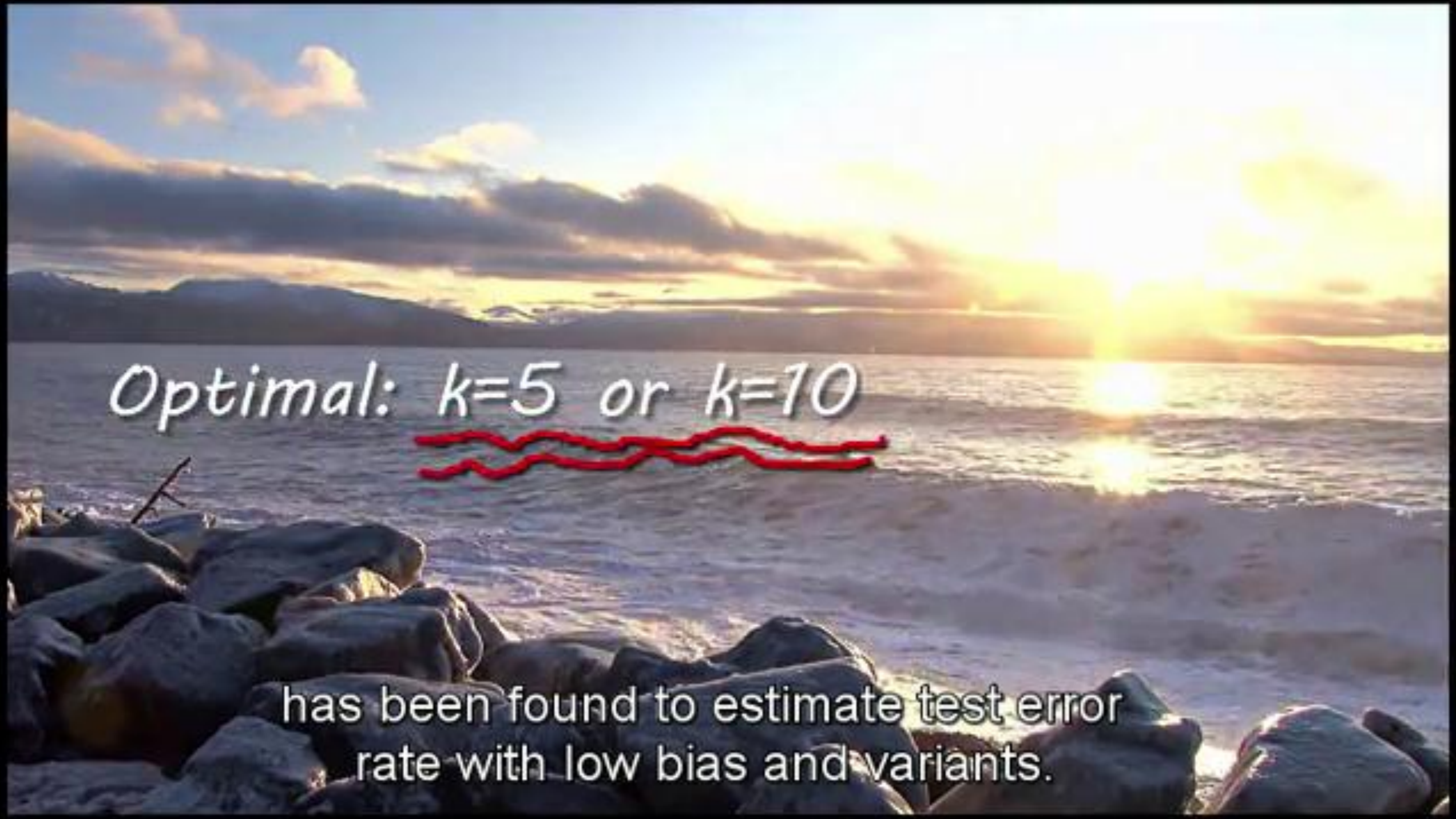

The number of folds can vary but


Optimal: k=5 or k=10
you will typically see k-fold
cross validation with k=5 or k=10.


Optimal: k=5 or k=10
There is a bias variant trade off
associated with the choice of how many


Optimal: k=5 or k=10
folds to specify in
k-fold cross validation.


Optimal: k=5 or k=10
Using k=5, or k=10,


Optimal: k=5 or k=10
has been found to estimate test error
rate with low bias and variants.