Introduction to Tree Methods

PIERIAN DATA
www.pieriandata.com

Tree Methods

Let’s start off with a thought experiment to give some motivation behind using a decision tree method.

Tree Methods

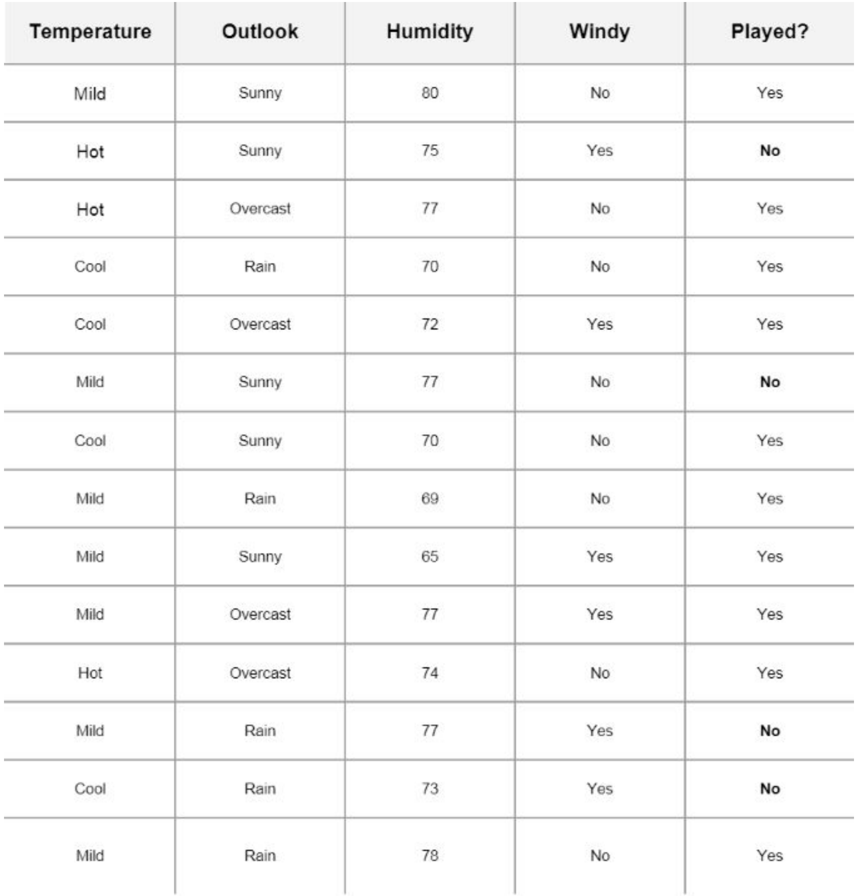
Imagine that I play Tennis every Saturday and I always invite a friend to come with me.

Sometimes my friend shows up, sometimes not.

For him it depends on a variety of factors, such as: weather, temperature, humidity, wind etc..

I start keeping track of these features and whether or not he showed up to play with me.

Tree Methods

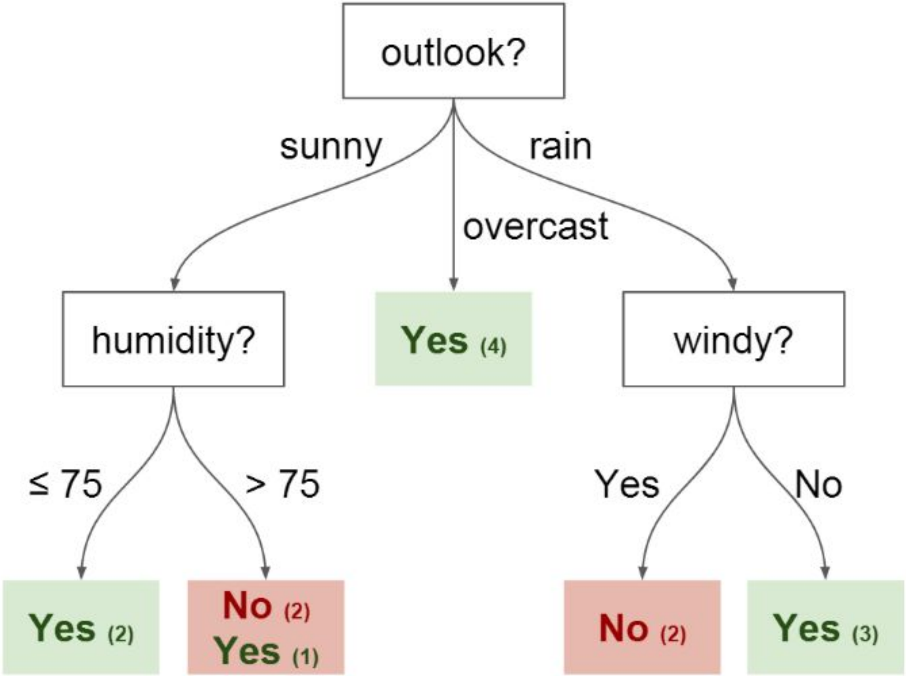


Temperature	Outlook	Humidity	Windy	Played?
Mild	Sunny	80	No	Yes
Hot	Sunny	75	Yes	**No**
Hot	Overcast	77	No	Yes
Cool	Rain	70	No	Yes
Cool	Overcast	72	Yes	Yes
Mild	Sunny	77	No	**No**
Cool	Sunny	70	No	Yes
Mild	Rain	69	No	Yes
Mild	Sunny	65	Yes	Yes
Mild	Overcast	77	Yes	Yes
Hot	Overcast	74	No	Yes
Mild	Rain	77	Yes	**No**
Cool	Rain	73	Yes	**No**
Mild	Rain	78	No	Yes

Tree Methods

I want to use this data to predict whether or not he will show up to play.

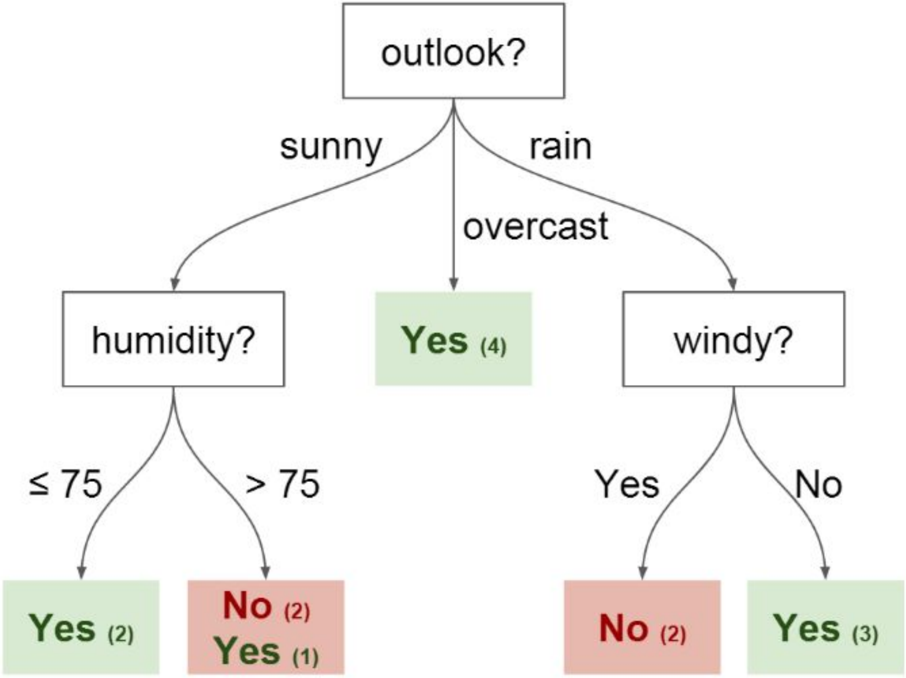
An intuitive way to do this is through a Decision Tree



Tree Methods

In this tree we have:

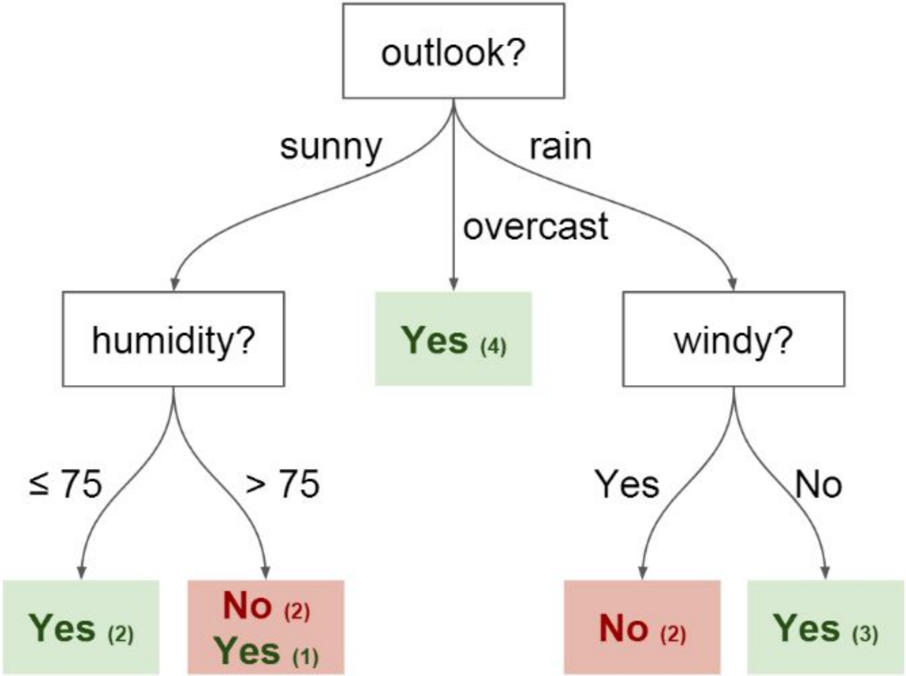
- Nodes
  - Split for the value of a certain attribute
- Edges
  - Outcome of a split to next node



Tree Methods

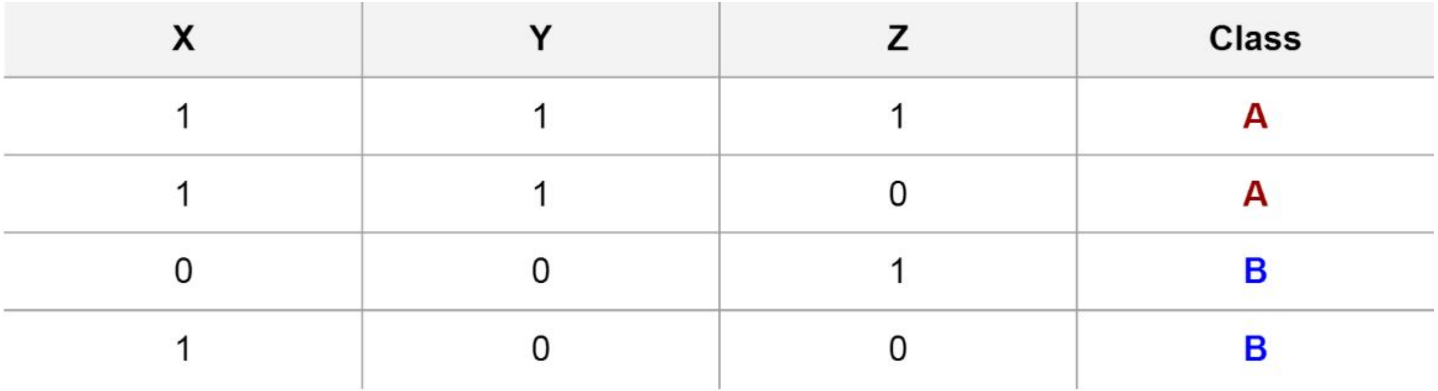
In this tree we have:

- Root
  - The node that performs the first split
- Leaves
  - Terminal nodes that predict the outcome



Intuition Behind Splits

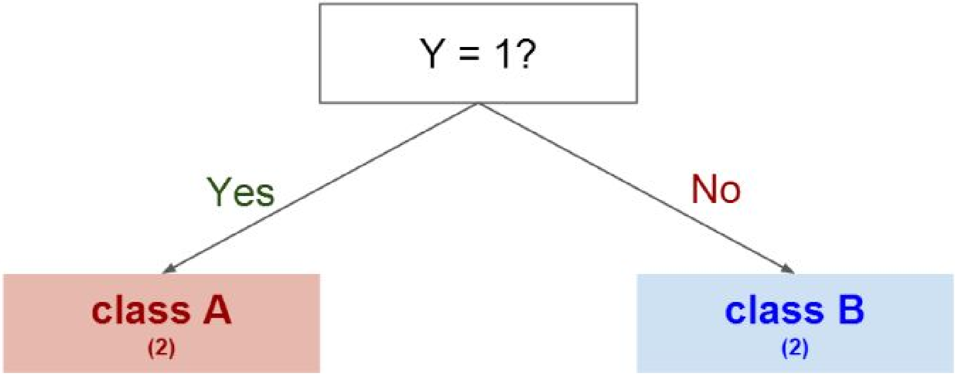
Imaginary Data with 3 features (X,Y, and Z) with two possible classes.



X	Y	Z	Class
1	1	1	A
1	1	0	A
0	0	1	B
1	0	0	B

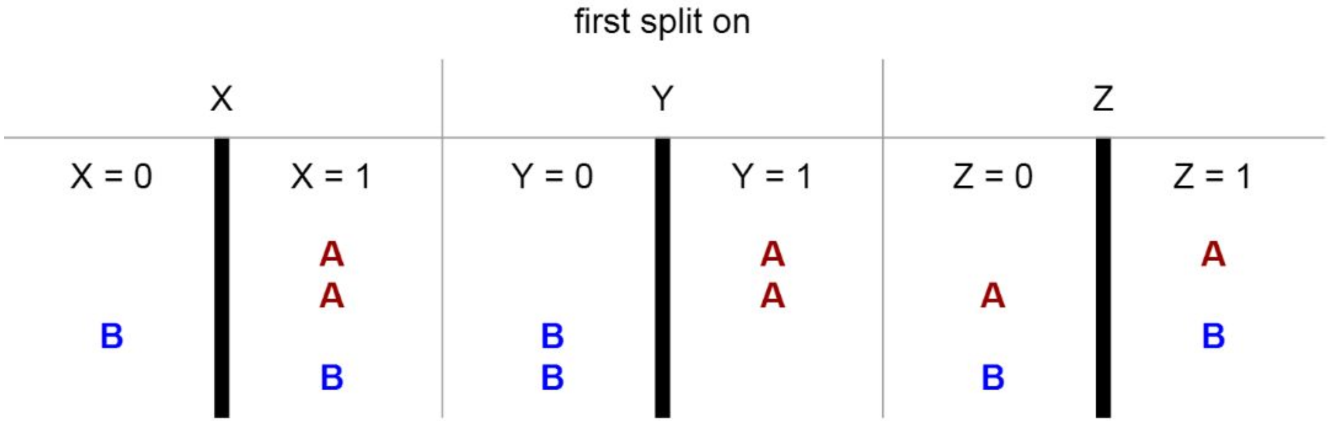
Intuition Behind Splits

Splitting on Y gives us a clear separation between classes



Intuition Behind Splits

We could have also tried splitting on other features first:

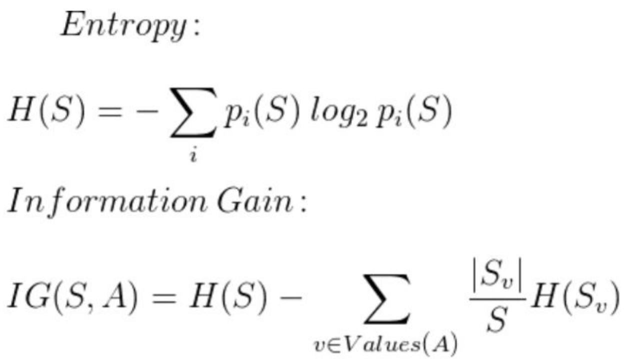


first split on

X		Y		Z	
X = 0	X = 1	Y = 0	Y = 1	Z = 0	Z = 1
	A		A		A
	A		A	A	
B		B			B
	B	B		B	

Intuition Behind Splits

Entropy and Information Gain are the Mathematical Methods of choosing the best split. Refer to reading assignment.



$Entropy:$

$$H(S) = -\sum_{i} p_i(S)\, log_2\, p_i(S)$$

$Information\, Gain:$

$$IG(S,A) = H(S) - \sum_{v \in Values(A)} \frac{|S_v|}{S} H(S_v)$$

Random Forests

To improve performance, we can use many trees with a random sample of features chosen as the split.

- A new random sample of features is chosen for **every single tree at every single split**.
- For **classification**, m is typically chosen to be the square root of p.

Random Forests

What's the point?

- Suppose there is **one very strong feature** in the data set. When using “bagged” trees, most of the trees will use that feature as the top split, resulting in an ensemble of similar trees that are **highly correlated**.

Random Forests

What's the point?

- Averaging highly correlated quantities does not significantly reduce variance.
- By randomly leaving out candidate features from each split, **Random Forests "decorrelates" the trees**, such that the averaging process can reduce the variance of the resulting model.

MACHINE LEARNING

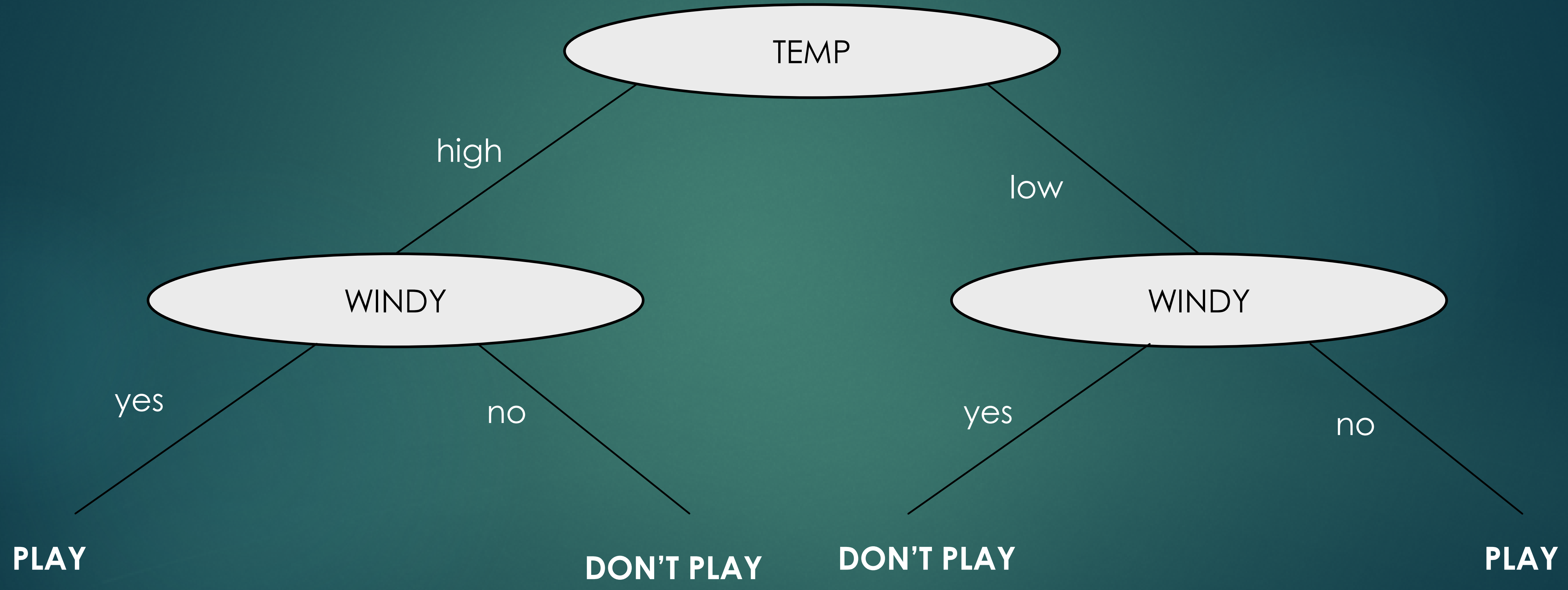
DECISION TREES

Decision trees

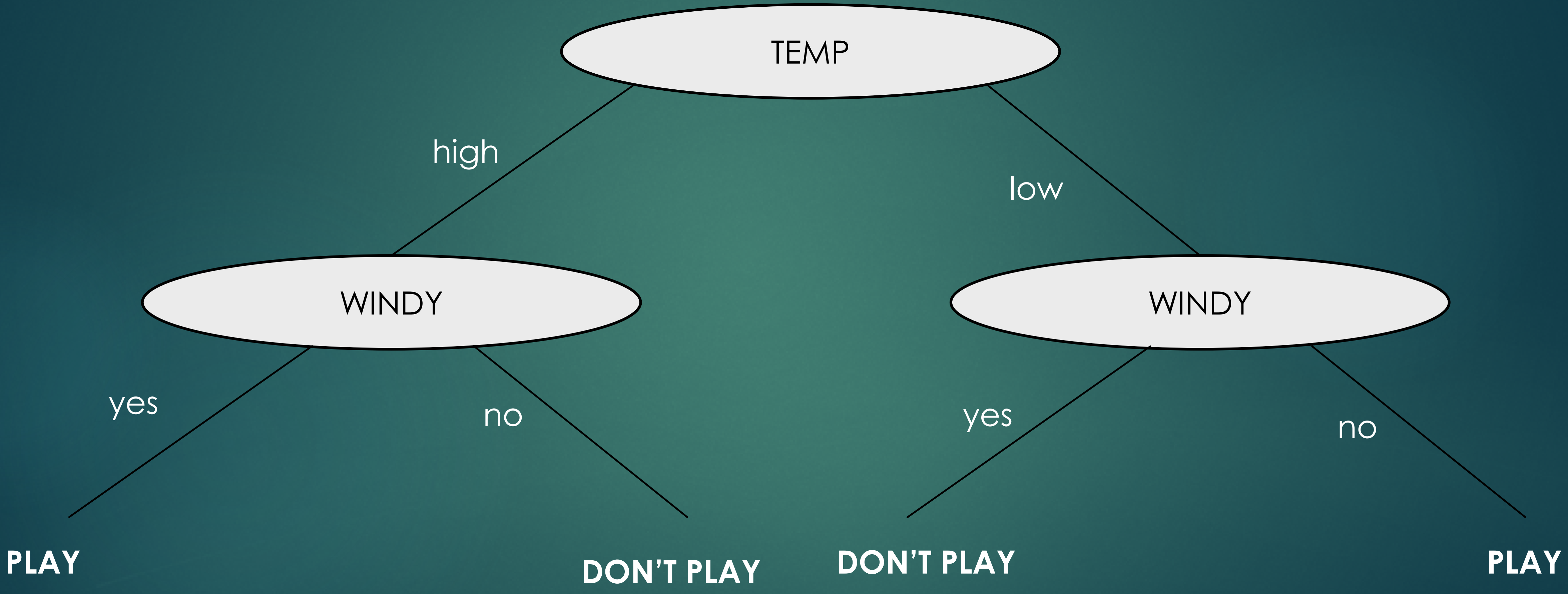
- Supervised learning technique for classification or regression problems
- We create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features
- Not as effective as the best supervised machine learning techniques
- Boosting and random forests → improve the performance
- Has several advantages + disadvantages

TEMPERATURE	HUMIDITY	WINDY	PLAYING TENNIS
hot	high	true	no
mild	low	false	yes
⋮	⋮	⋮	⋮

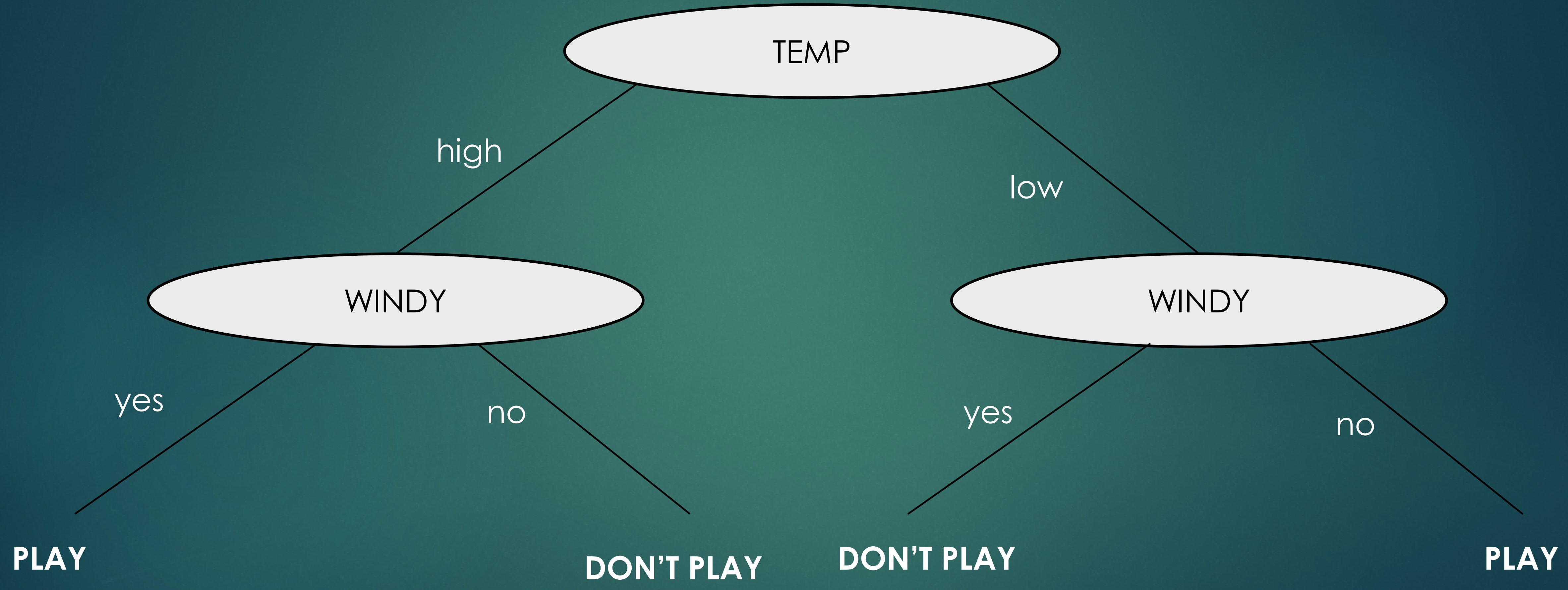
Constructing the tree !!!



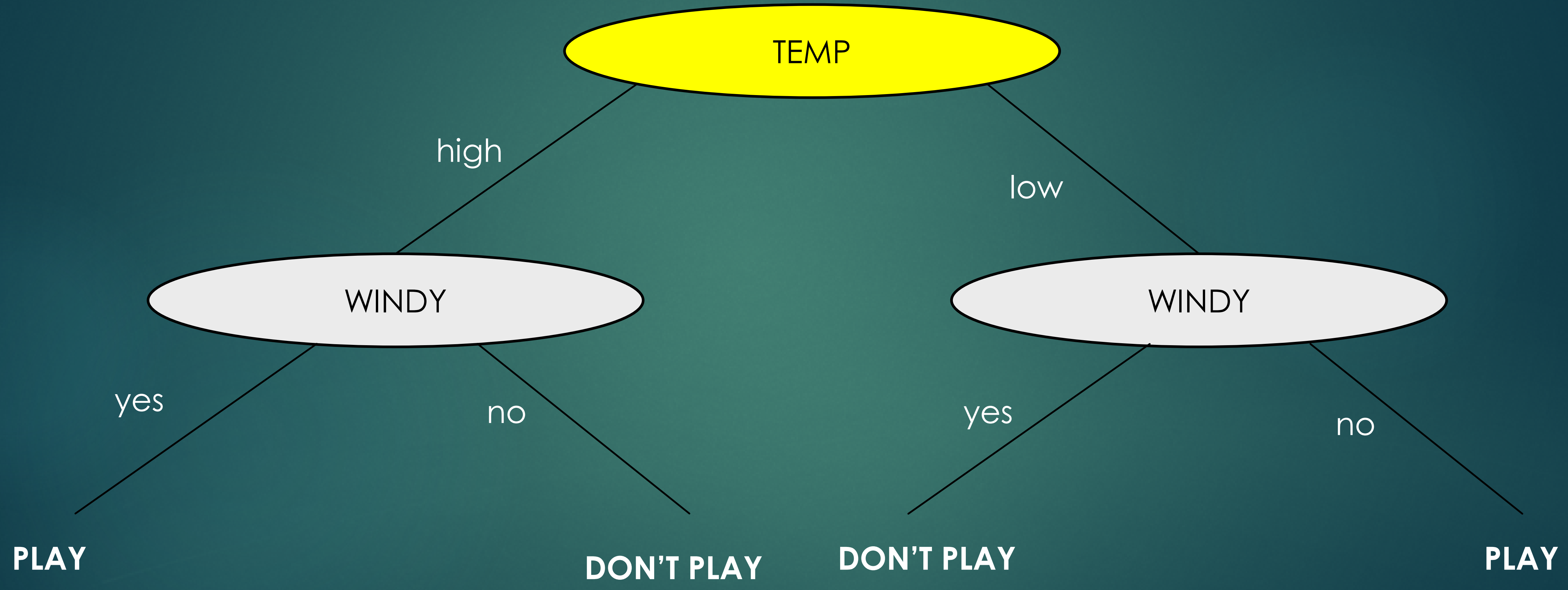
When making decision → we just have to traverse the tree according to the features !!



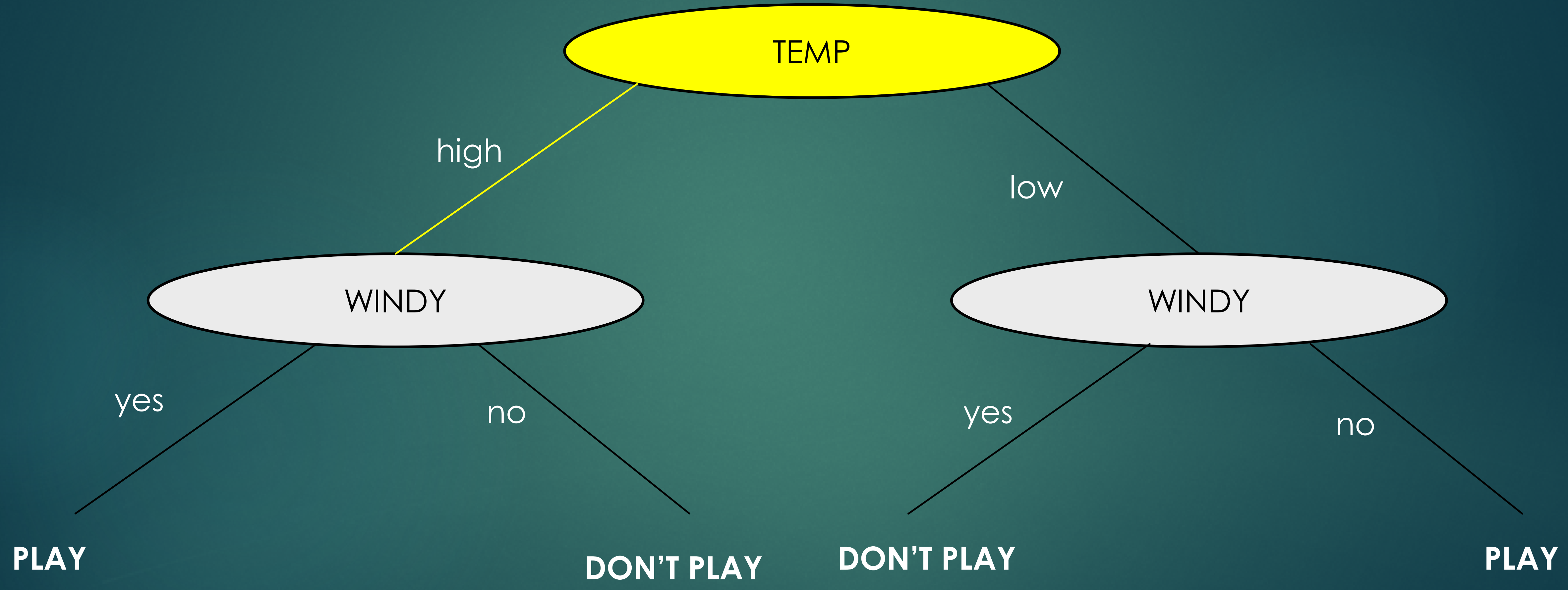
Temp: high
Windy: yes
Do we play tennis?



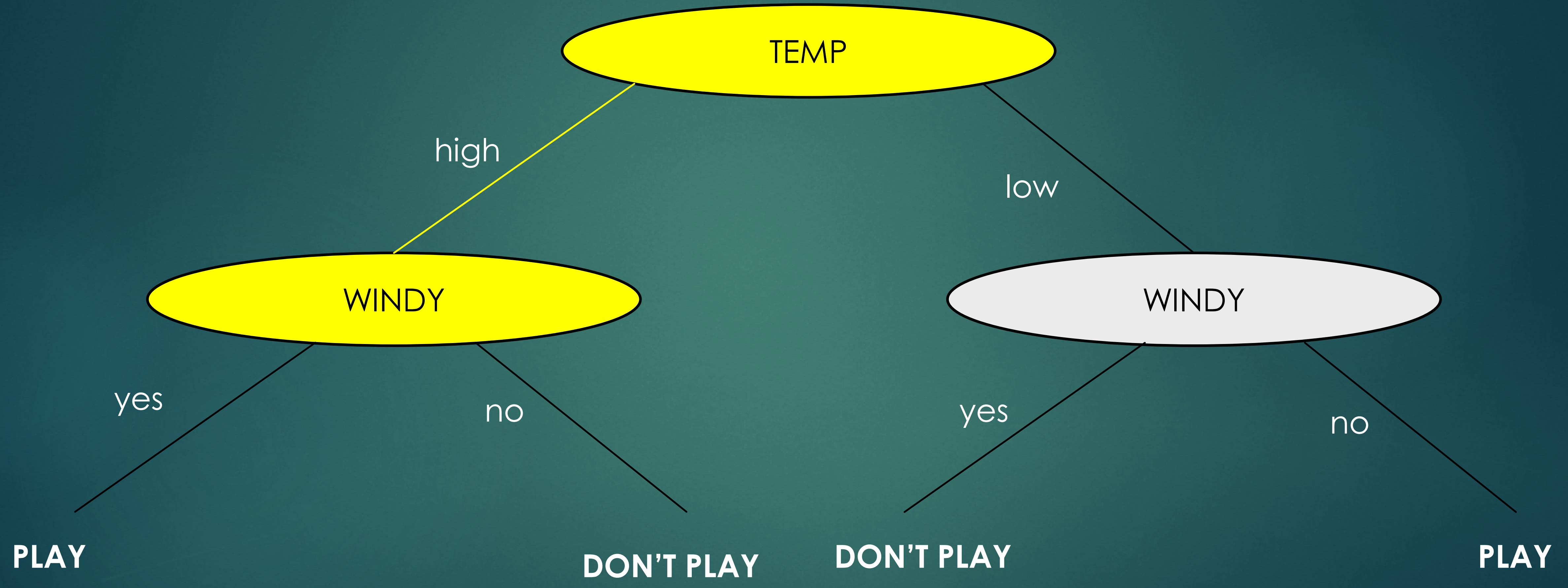
Temp: high
Windy: yes
Do we play tennis?



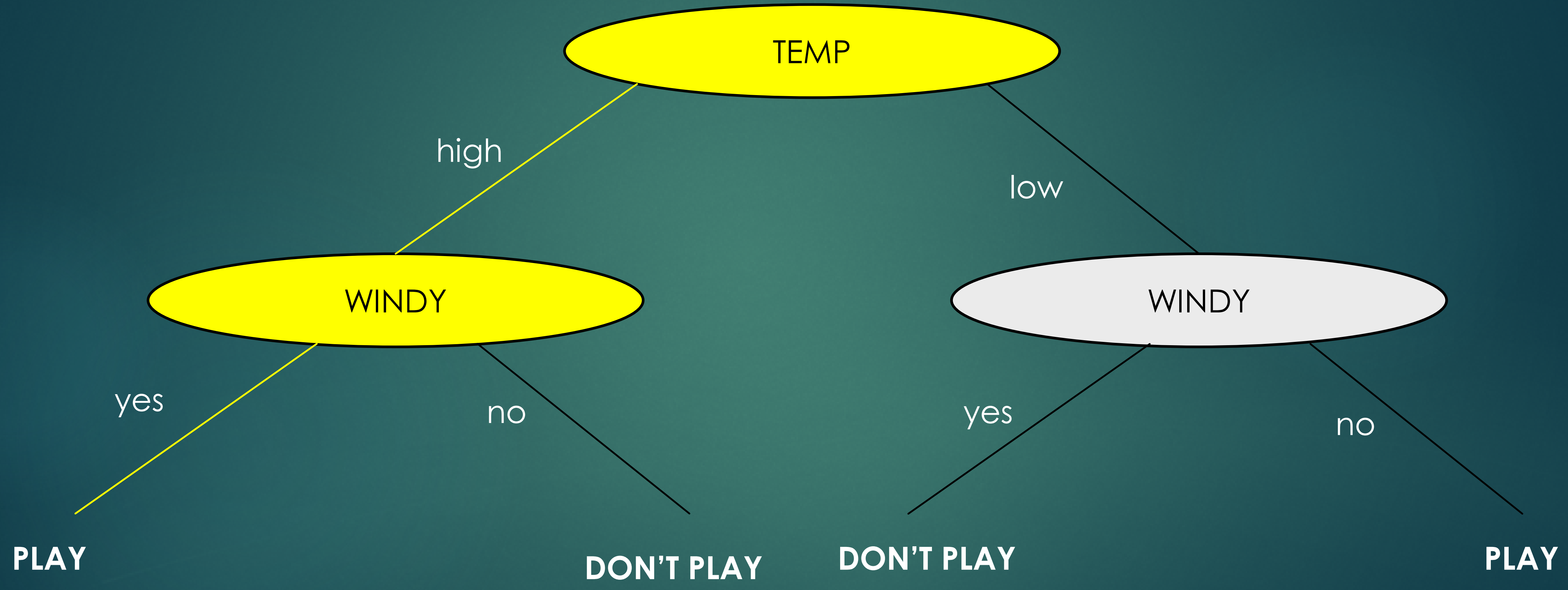
Temp: high
Windy: yes
Do we play tennis?



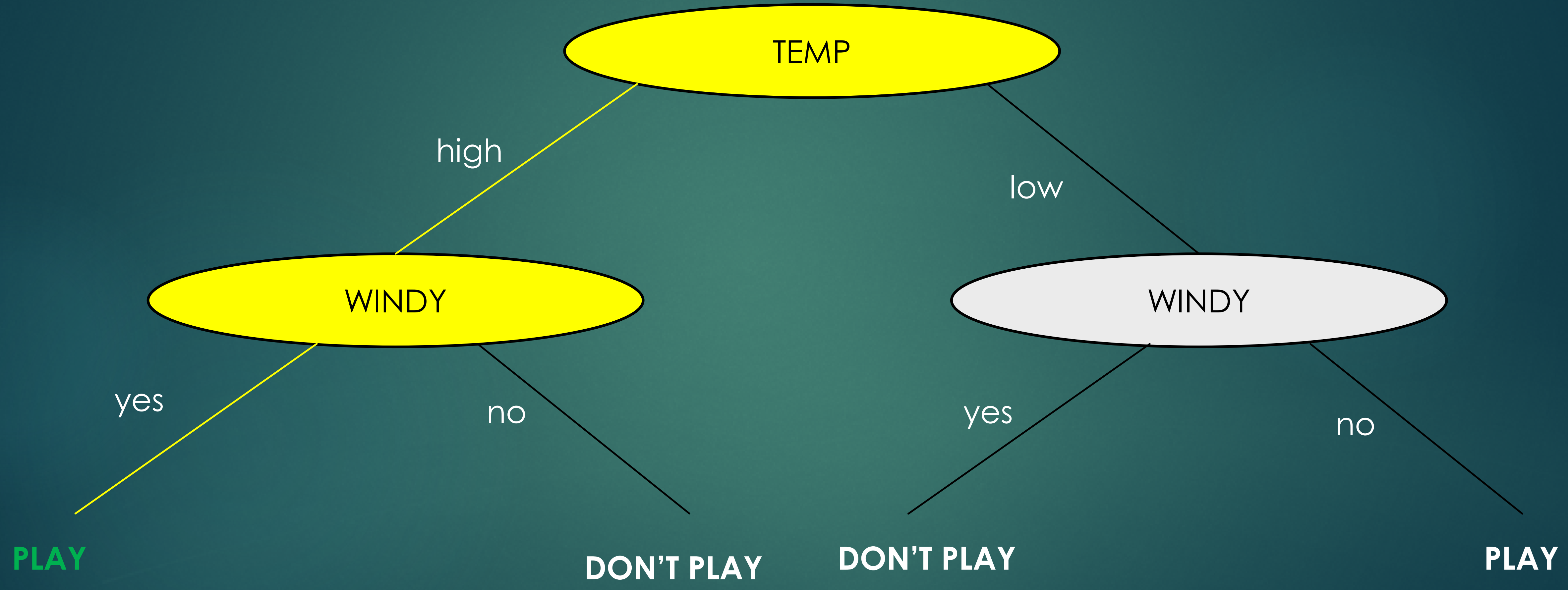
Temp: high
Windy: yes
Do we play tennis?



Temp: high
Windy: yes
Do we play tennis?



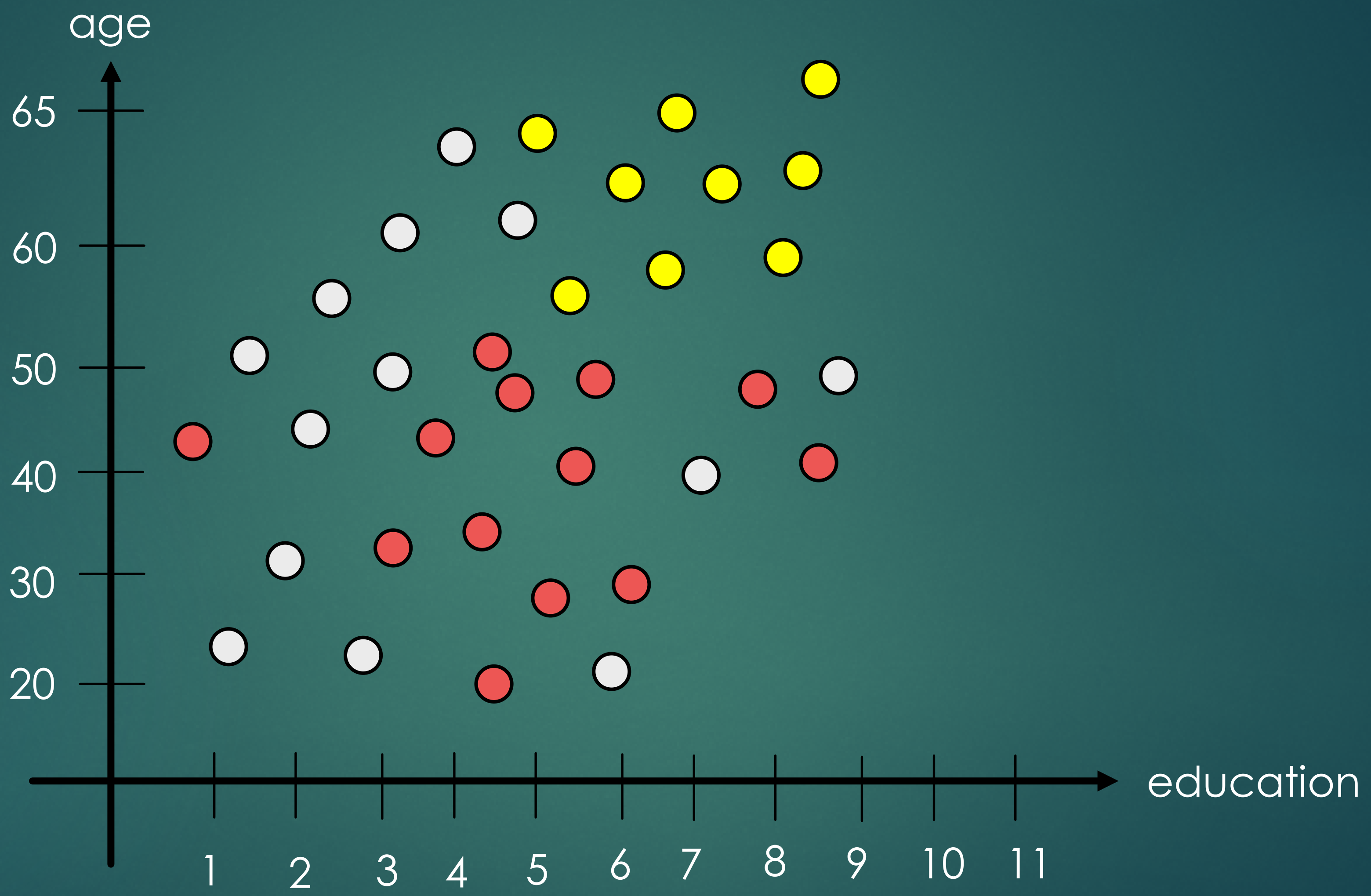
Temp: high
Windy: yes
Do we play tennis?

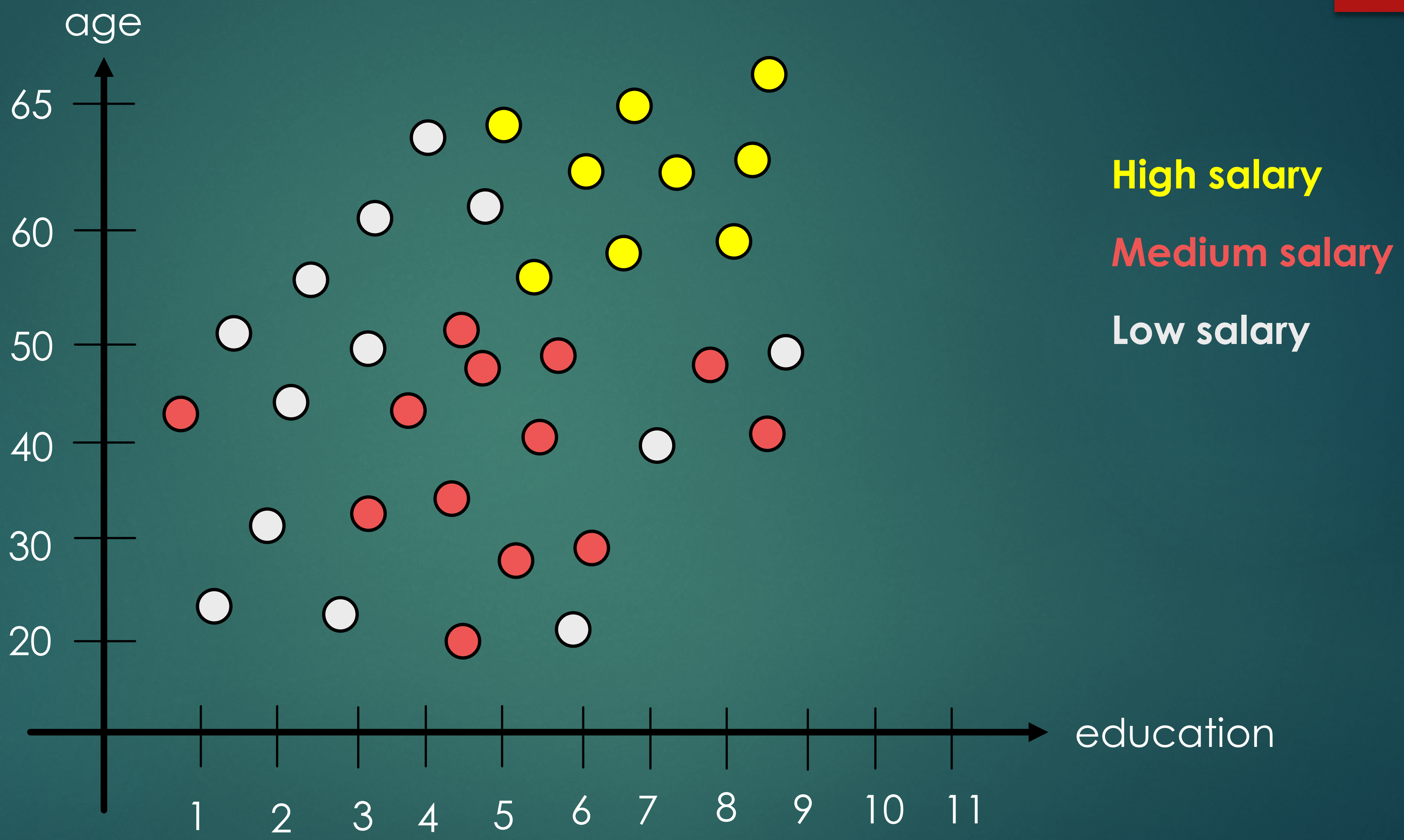


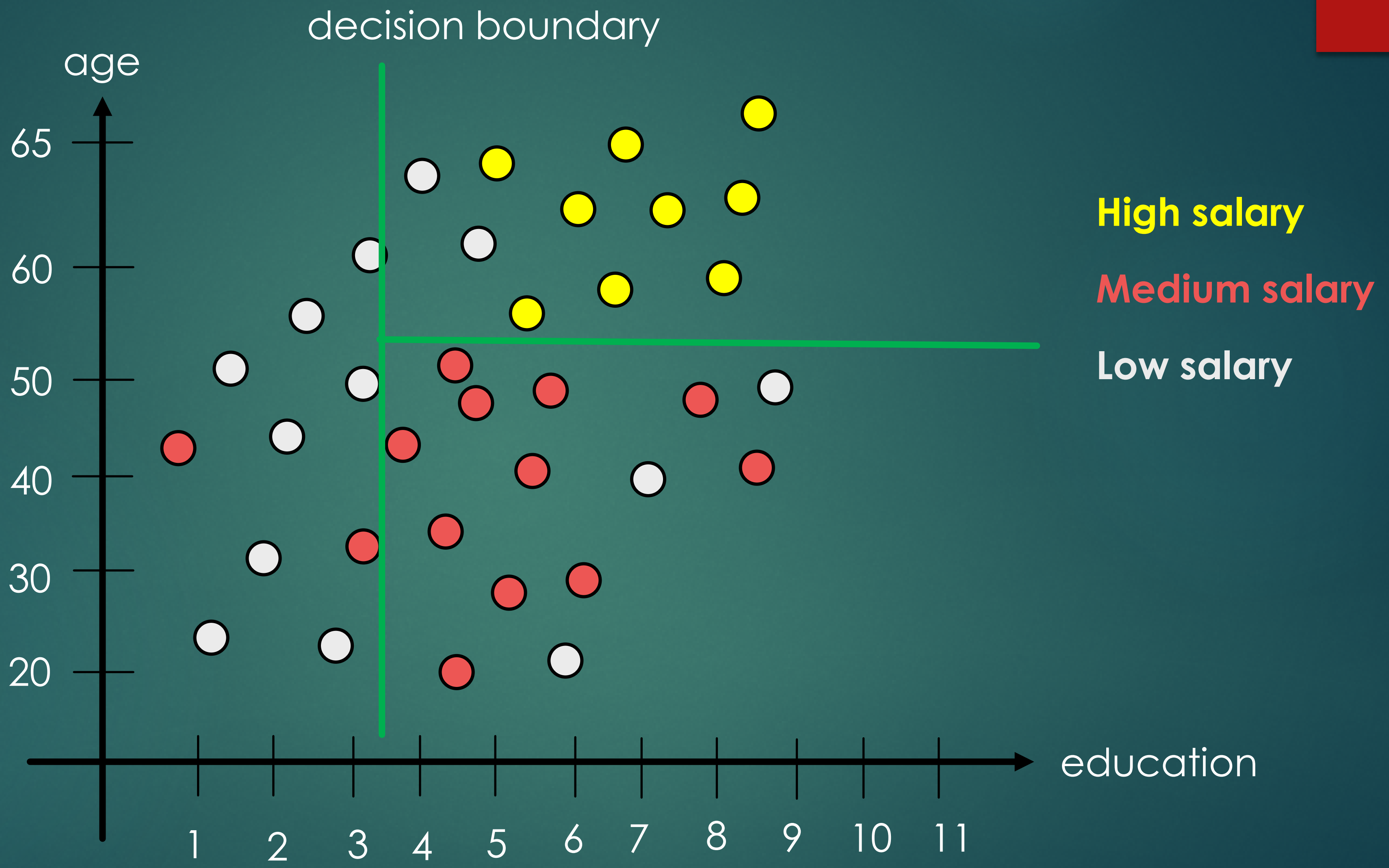
Regression

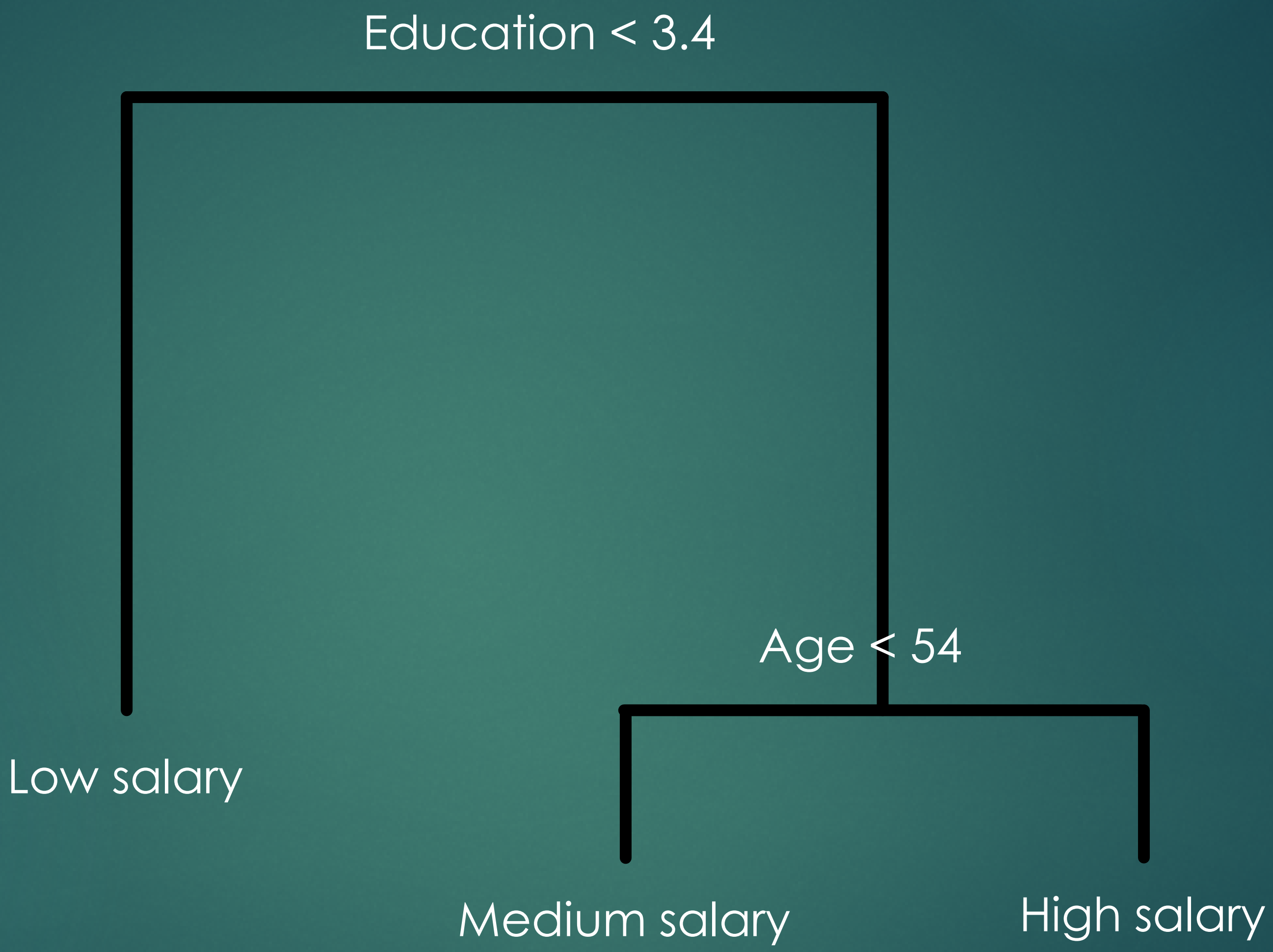
Regression with decision trees

- We assume that salary is the function of age and education
- This is a typical regression problem: we know the age and education and want to make prediction to the salary


age
65
60
50
40
30
20
education
1
2
3
4
5
6
7
8
9
10
11


age
65
60
50
40
30
20
education
1
2
3
4
5
6
7
8
9
10
11
High salary
Medium salary
Low salary


decision boundary
age
65
60
50
40
30
20
1
2
3
4
5
6
7
8
9
10
11
education
High salary
Medium salary
Low salary



Education < 3.4

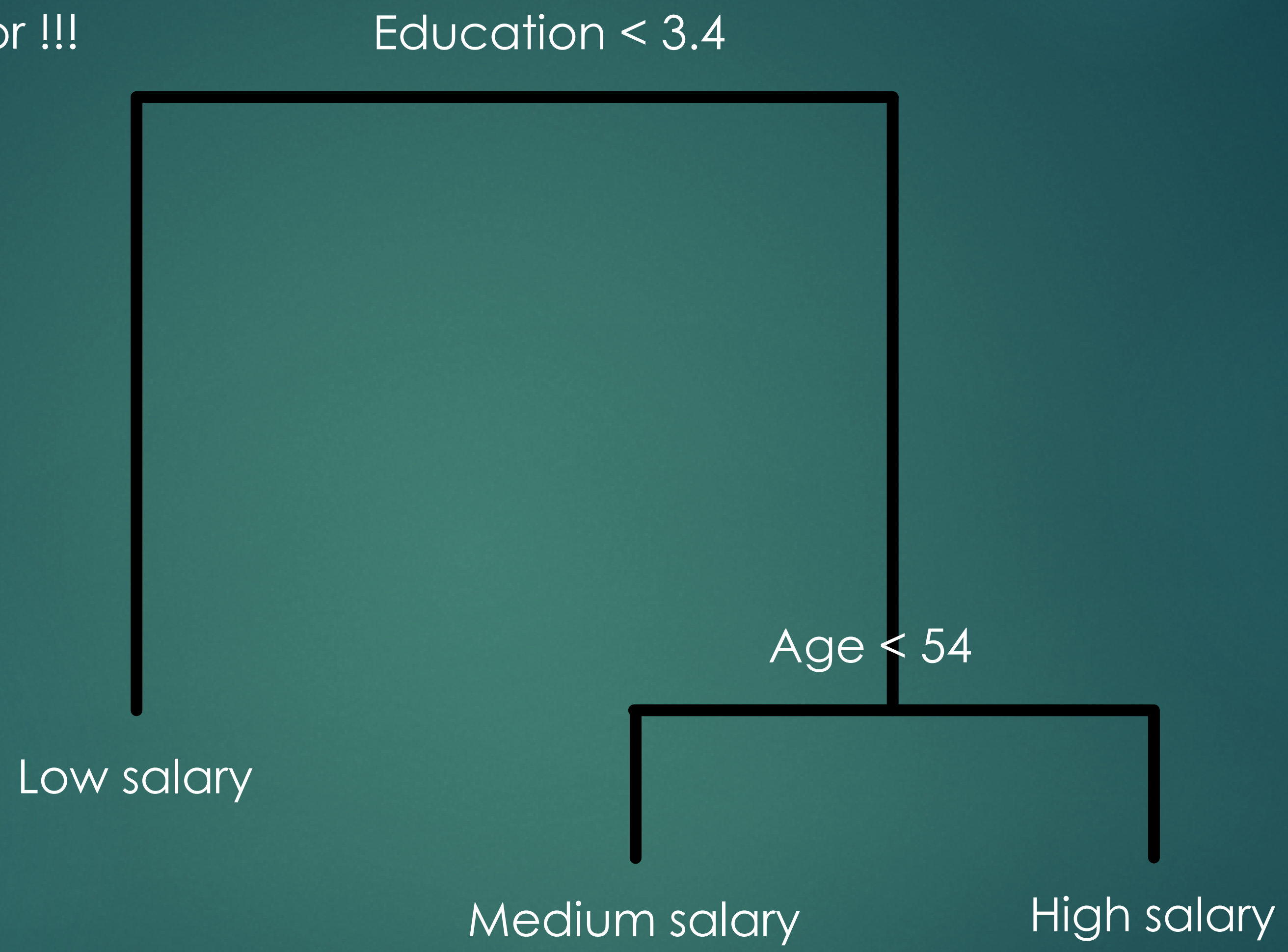
Age < 54

Low salary

Medium salary

High salary

The root node is the
most important factor !!!



Advantages

- Simple to understand and to interpret + trees can be visualized
- No need for data preparations such as normalization or dummy variables
- Logarithmic **O(logN)** running time

Distadvantages

- Decision-tree learners can create over-complex trees that do not generalize to the data well
- This is the problem of overfitting // pruning somethime helps
- Decision trees can be unstable because small variations in the data might result in a completely different tree being generated
- The problem of learning an optimal decision tree is known to be **NP-complete** !!!
- Practical decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm

MACHINE LEARNING

PRUNING TREES AND BAGGING

bias: error from misclassifications in the learning algorithm

High bias → the algorithm misses the relevant relationships between features and target outputs !!!

ERROR DUE TO MODEL MISMATCH

variance: error from sensitivity to small changes in the training set

High variance → can cause overfitting

VARIATION DUE TO TRAINING SAMPLE AND RANDOMIAZTION

Bias / variance tradeoff

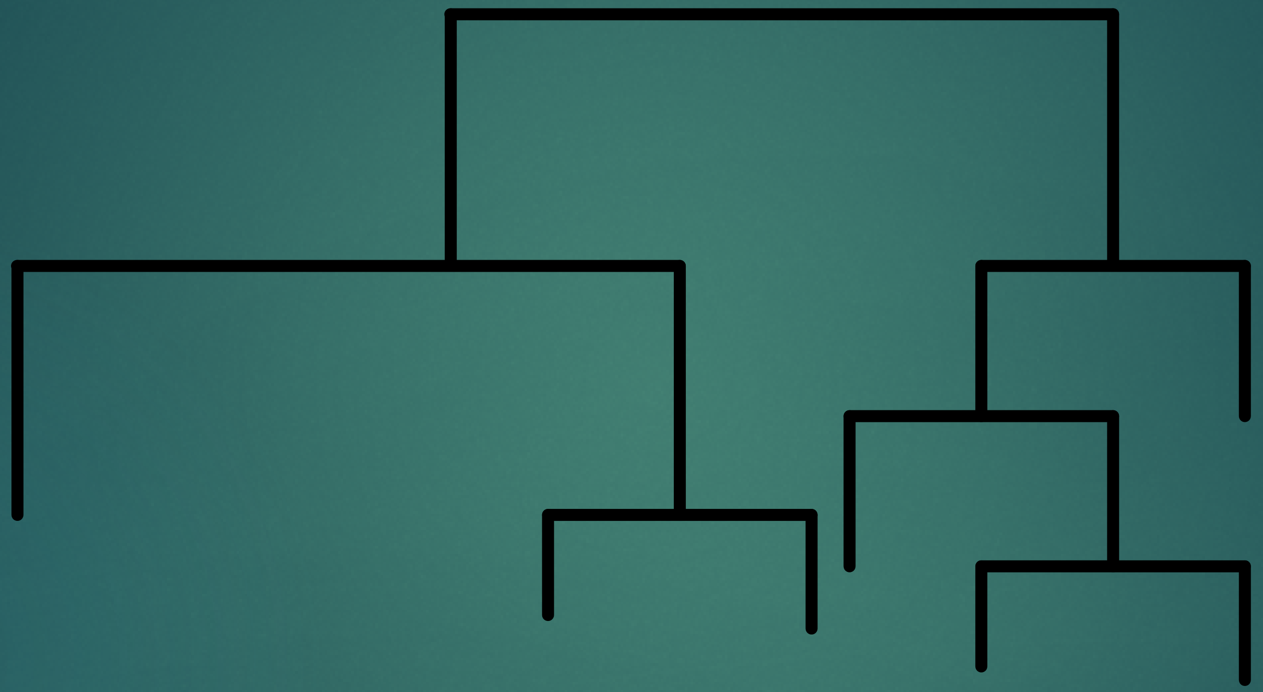
~ we are not able to optimize both bias and variance at the same time

low bias → high variance
low variance → high bias

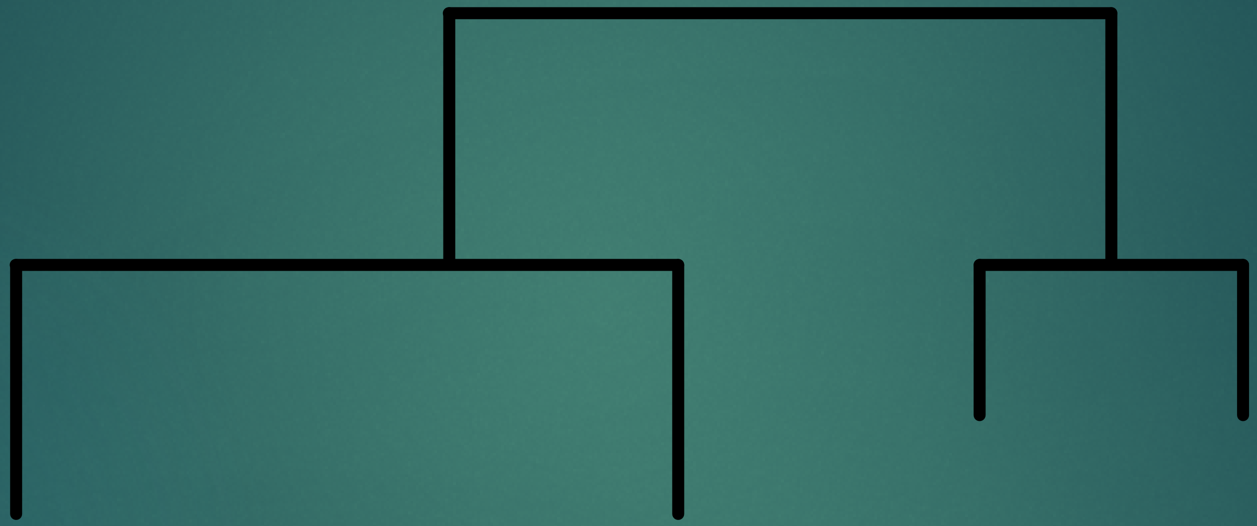
Pruning

- Usually decision trees are likely to overfit the data leading to poor test performance
- Smaller tree + fewer splits → better predictor at the cost of a little bias
- Better solution: grow a large tree and then prune it back to a smaller subtree
- „weakest link pruning”

The large tree before pruning !!!



After pruning: will not overfit the data !!!



Bagging

- Bagging = bootstrap aggregation
- Reduce the variance of a learning algorithm
- If we have a set of **n** independent varibles $\mathbf{x}_1$, $\mathbf{x}_2$, ... , $\mathbf{x}_n$ each with variance **V** → the variance of the mean **X** (the mean of the $\mathbf{x}_1$, $\mathbf{x}_2$ ... $\mathbf{x}_n$ variables) is $\frac{V}{n}$!!!
- So we can reduce the variance by averaging a set of observations
- Good idea: have multiple training sets and construct a decision tree (without pruning) on every single training sets !!!
- Problem → we do not have several training sets

Bagging

- We should take repeated samples from the single data set + construct trees + average all the predictions in the end
- **THIS IS BAGGING**
- Pruning → variance decreases but we have some bias ... here we can reduce the variance without extra bias
- Regression problem: we take the average
- Classification problem: we take the majority vote

MACHINE LEARNING

RANDOM FORESTS

Random forests

- Better than bagging: this algorithm decorrelates the single decision trees that has been constructed
- This reduces the variance even more when averaging the trees
- Similar to bagging: we keep constructing decision trees on the training data
- **BUT** on every split in the tree, a random selection of features / predictors is chosen from the full feature set
- The number of features considered at a given split is approximately equal to the square root of the total number of features !!!

Random forests

- **Why is it good?**
- If one or a few features are very strong predictors for the response variable (target output), these features will be selected in many of the decision trees → they will become correlated
- Huge advantage → at some point the variance stops decreasing no matter how many more trees we add to our random forest + it is not going to produce overfitting !!!

MACHINE LEARNING

BOOSTING

Boosting

- It can be used for classification and regression too
- Helps to reduce variance and bias !!!
- Bagging: creates multiple copies of the original data → constructs several decision trees on the copies → combining all the trees to make predictions
- **THESE TREES ARE INDEPENDENT FROM EACH OTHER !!!**
- Boosting: here the decision trees are grown sequentially → each tree is grown using information from previously grown trees
- **THESE TREES ARE NOT INDEPENDENT FROM EACH OTHER !!!**

Boosting

- Can a set of weak learners create a single strong learner?
- Yes, we can turn a weak learner into a strong learner !!!
- Fit a large decision tree to the data → overfitting
- The boosting algorithm learns slowly instead
- By fitting small trees we slowly improve the final result in cases when it does not perform well

Parameters

- The number of trees: random forests are not able to overfit. Boosting can overfit if the number of trees is too large

  Cross validation → we can get the optimal number of trees

- Shrinkage parameter: determines the learning rate for boosting. When this parameter is very small → we should have a lot of trees !!!