Random Forest Algorithm

Andrew Wang

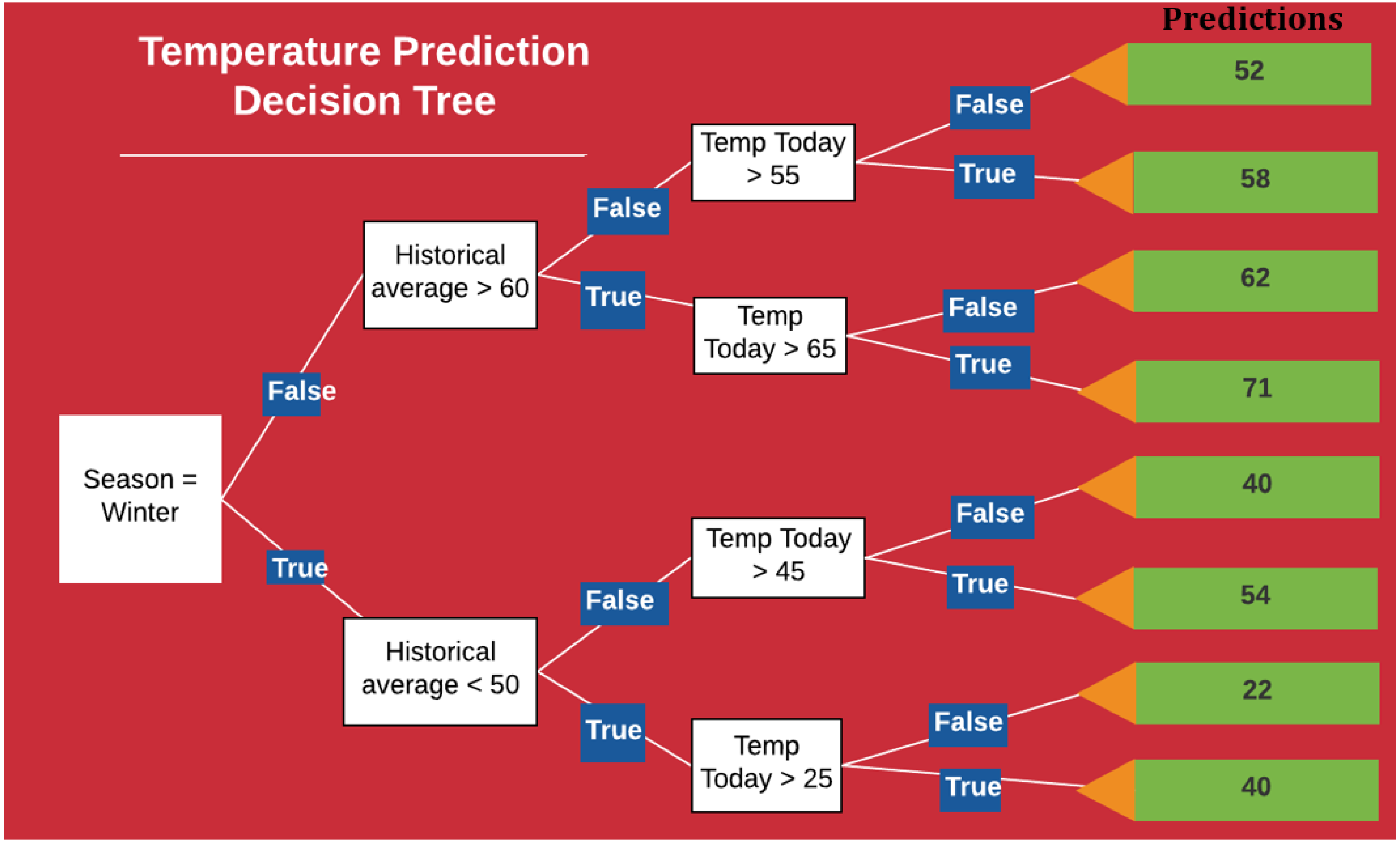
Background Content

- Random forest comprised of multiple decision trees -> first understand what decision tree is, then understand rand forest
- What is decision tree + what it looks like
  - Why/how it works
  - Idea of entropy + information gain + gini index
- Random forest classifier
  - What is it + what does it look like
  - How does it work?
  - Why "random" and why "forest" in random forest classifier name?
- Misc
  - Difference between random forest regression vs classification

What is a decision tree?

- Flowchart-like tree that is used to model how outputs are predicted from inputs
  - Branches/edges represent result (Ex: True/False) of nodes
  - Nodes represent either
    - Conditions (decision nodes)
    - OR results (end/leaf nodes)
- Models decisions and ALL possible results
- Supervised learning algorithm
- Works for both continuous and categorical output

Visualization of decision tree w/ discrete output



https://williamkoehrsen.medium.com/random-forest-simple-explanation-377895a60d2d

Examples of discrete vs continuous output in decision trees

- **“Discrete output example:** A weather prediction model that predicts whether or not there’ll be rain in a particular day.”
- **“Continuous output example:** A profit prediction model that states the probable profit that can be generated from the sale of a product.”

Source: https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/?ref=rp

Decision tree implementation details from geeksforgeeks in Python

Continuous output example: A profit prediction model that states the probable profit that can be generated from the sale of a product.

Here, continuous values are predicted with the help of a decision tree regression model.

Let's see the Step-by-Step implementation –

- **Step 1:** Import the required libraries.

```python
# import numpy package for arrays and stuff
import numpy as np

# import matplotlib.pyplot for plotting our result
import matplotlib.pyplot as plt

# import pandas for importing csv files
import pandas as pd
```

https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/?ref=rp

Decision tree implementation details from geeksforgeeks in Python

- **Step 2:** Initialize and print the Dataset.

https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/?ref=rp

```
# import dataset
# dataset = pd.read_csv('Data.csv')
# alternatively open up .csv file to read data

dataset = np.array(
[['Asset Flip', 100, 1000],
['Text Based', 500, 3000],
['Visual Novel', 1500, 5000],
['2D Pixel Art', 3500, 8000],
['2D Vector Art', 5000, 6500],
['Strategy', 6000, 7000],
['First Person Shooter', 8000, 15000],
['Simulator', 9500, 20000],
['Racing', 12000, 21000],
['RPG', 14000, 25000],
['Sandbox', 15500, 27000],
['Open-World', 16500, 30000],
['MMOFPS', 25000, 52000],
['MMORPG', 30000, 80000]
])

# print the dataset
print(dataset)
```

```
# print the dataset
print(dataset)
```

```
[['Asset Flip' '100' '1000']
 ['Text Based' '500' '3000']
 ['Visual Novel' '1500' '5000']
 ['2D Pixel Art' '3500' '8000']
 ['2D Vector Art' '5000' '6500']
 ['Strategy' '6000' '7000']
 ['First Person Shooter' '8000' '15000']
 ['Simulator' '9500' '20000']
 ['Racing' '12000' '21000']
 ['RPG' '14000' '25000']
 ['Sandbox' '15500' '27000']
 ['Open-World' '16500' '30000']
 ['MMOFPS' '25000' '52000']
 ['MMORPG' '30000' '80000']]
```

Decision tree implementation details from geeksforgeeks in Python

- **Step 3:** Select all the rows and column 1 from dataset to "X".

```
# select all rows by : and column 1
# by 1:2 representing features
X = dataset[:, 1:2].astype(int)

# print X
print(X)
```

```
[[  100]
 [  500]
 [ 1500]
 [ 3500]
 [ 5000]
 [ 6000]
 [ 8000]
 [ 9500]
 [12000]
 [14000]
 [15500]
 [16500]
 [25000]
 [30000]]
```

https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/?ref=rp

Decision tree implementation details from geeksforgeeks in Python

- **Step 4:** Select all of the rows and column 2 from dataset to “y”.

```
# select all rows by : and column 2
# by 2 to Y representing labels
y = dataset[:, 2].astype(int)

# print y
print(y)
```

```
[ 1000  3000  5000  8000  6500  7000 15000 20000 21000 25000 27000 30000
 52000 80000]
```

https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/?ref=rp

Decision tree implementation details from geeksforgeeks in Python

- **Step 5:** Fit decision tree regressor to the dataset

```
# import the regressor
from sklearn.tree import DecisionTreeRegressor

# create a regressor object
regressor = DecisionTreeRegressor(random_state = 0)

# fit the regressor with X and Y data
regressor.fit(X, y)
```

```
DecisionTreeRegressor(criterion='mse', max_depth=None, max_features=None,
           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,
           presort=False, random_state=0, splitter='best')
```

https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/?ref=rp

Decision tree implementation details from geeksforgeeks in Python

- **Step 6:** Predicting a new value

```python
# predicting a new value

# test the output by changing values, like 3750
y_pred = regressor.predict(3750)

# print the predicted price
print("Predicted price: % d\n"% y_pred)
```

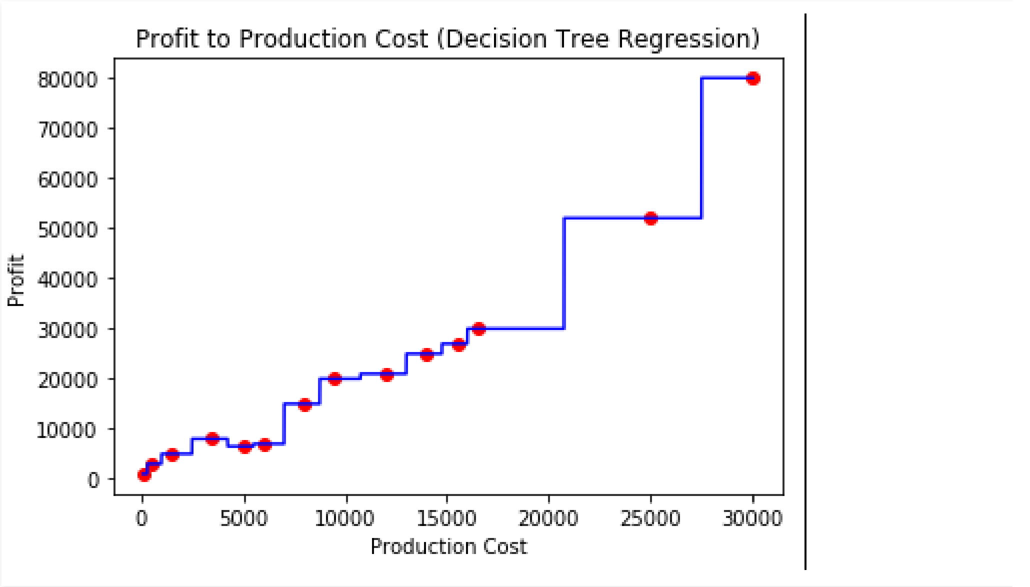
```
Predicted price: 8000
```

https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/?ref=rp

Decision tree implementation details from geeksforgeeks in Python

- **Step 7:** Visualising the result

```
# arange for creating a range of values
# from min value of X to max value of X
# with a difference of 0.01 between two
# consecutive values
X_grid = np.arange(min(X), max(X), 0.01)

# reshape for reshaping the data into
# a len(X_grid)*1 array, i.e. to make
# a column out of the X_grid values
X_grid = X_grid.reshape((len(X_grid), 1))

# scatter plot for original data
plt.scatter(X, y, color = 'red')

# plot predicted data
plt.plot(X_grid, regressor.predict(X_grid), color = 'blue')

# specify title
plt.title('Profit to Production Cost (Decision Tree Regression)')

# specify X axis label
plt.xlabel('Production Cost')

# specify Y axis label
plt.ylabel('Profit')

# show the plot
plt.show()
```



https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/?ref=rp

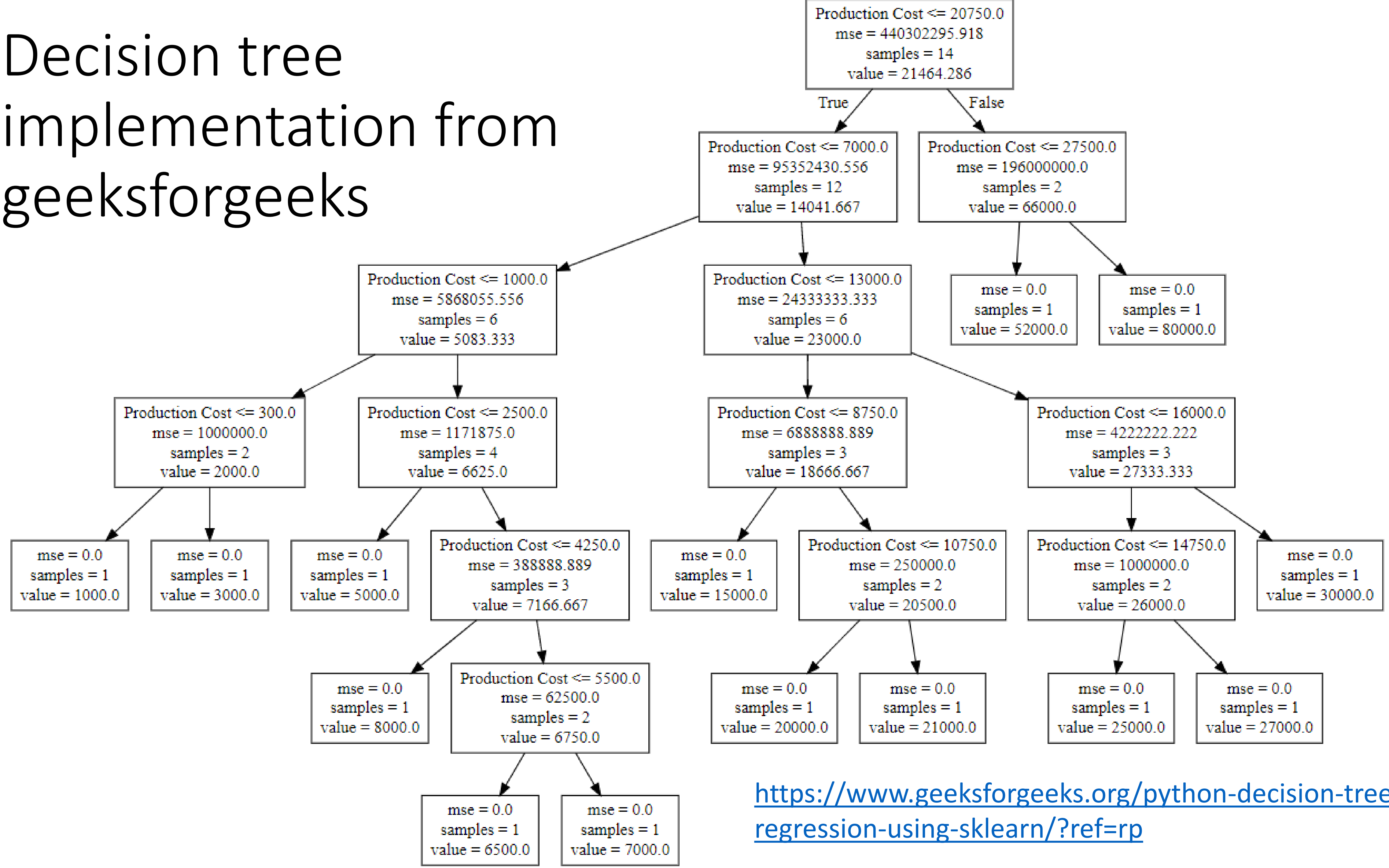
Decision tree implementation details from geeksforgeeks in Python

- **Step 8:** The tree is finally exported and shown in the TREE STRUCTURE below, visualized using http://www.webgraphviz.com/ by copying the data from the 'tree.dot' file.

```
# import export_graphviz
from sklearn.tree import export_graphviz

# export the decision tree to a tree.dot file
# for visualizing the plot easily anywhere
export_graphviz(regressor, out_file ='tree.dot',
               feature_names =['Production Cost'])
```

https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/?ref=rp

Decision tree implementation from geeksforgeeks



https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/?ref=rp

Ok… great… but how does a decision tree really work?

- How to determine which node/attribute cutoff is root?
  - How to determine best cutoff for data (Ex: production cost <= 7000 in previous slide)?
  - Decision trees need to be able to identify + quantify best “cutoffs” in data
- Recall:
  - Each internal node corresponds to an attribute
  - Each leaf node corresponds to class label
- Decision trees need to know which attributes to be considered as root node at each level of tree

https://dataaspirant.com/how-decision-tree-algorithm-works/

Overview of attribute selection

- Popular attribute selection measures:
  - Information gain (attributes assumed to be categorical)
  - Gini index (attributes assumed to be continuous)

https://dataaspirant.com/how-decision-tree-algorithm-works/

Impurity Criterion

Gini Index

$$I_G = 1 - \sum_{j=1}^{c} p_j^2$$

p_j: proportion of the samples that belongs to class c for a particular node

Entropy

$$I_H = - \sum_{j=1}^{c} p_j log_2(p_j)$$

p_j: proportion of the samples that belongs to class c for a particular node.

*This is the the definition of entropy for all non-empty classes (p ≠ 0). The entropy is 0 if all samples at a node belong to the same class.

https://www.quora.com/What-is-difference-between-Gini-Impurity-and-Entropy-in-Decision-Tree

Using info gain to quantify how good "cutoffs" are

Example of using information gain as criterion

- Entropy – randomness or uncertainty of random variable X
- Assume binary classification problem (2 classes, + and -)
  - If all examples are + or all – then entropy = 0 (low)
  - If ½ of examples are + and ½ are – then entropy = 1 (high)
- Calculate entropy measure for each “attribute” -> calculate info gain
  - Information gain – expected reduction in entropy due to sorting on the “attribute” selected

Example of using info gain as criterion

- Predictors = columns A, B, C, D = attributes
- Target variable = Column E = class labels

<u>Data</u>

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

Let’s choose some random values/thresholds to categorize each attribute:

A	**B**	**C**	**D**
>= 5	>= 3.0	>= 4.2	>= 1.4
< 5	< 3.0	< 4.2	< 1.4

https://dataaspirant.com/how-decision-tree-algorithm-works/

Example of using info gain as criterion

- To calculate info gain for attribute:
  - 1. Calculate entropy of target
  - 2. Calculate entropy for attribute
  - 3. Calculate info gain = Entropy of target – Entropy of attribute

Calculating entropy of target

$$H(X) = \mathbb{E}_X[I(x)] = -\sum_{x \in \mathbb{X}} p(x) \log p(x).$$

<u>Data</u>

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

The entropy of Target: We have 8 records with negative class and 8 records with positive class. So, we can directly estimate the entropy of target as 1.

Variable E	
Positive	Negative
8	8

Calculating entropy using formula:

E(8,8) = -1*((p(+ve)*log(p(+ve)) + (p(-ve)*log(p(-ve)))

$= -1*((8/16)*\log_2(8/16)) + (8/16) * \log_2(8/16))$

= 1

Calculating info gain for attribute A

Data

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

$$H(X) = \mathbb{E}_X[I(x)] = -\sum_{x \in \mathbb{X}} p(x) \log p(x).$$

Information gain for Var A

Var A has value >=5 for 12 records out of 16 and 4 records with value <5 value.

- For Var A >= 5 & class == positive: 5/12
- For Var A >= 5 & class == negative: 7/12
  - Entropy(5,7) = -1 * ((5/12)*log2(5/12) + (7/12)*log2(7/12)) = 0.9799
- For Var A <5 & class == positive: 3/4
- For Var A <5 & class == negative: 1/4
  - Entropy(3,1) = -1 * ((3/4)*log2(3/4) + (1/4)*log2(1/4)) = 0.81128

Entropy(Target, A) = P(>=5) * E(5,7) + P(<5) * E(3,1)
= (12/16) * 0.9799 + (4/16) * 0.81128 = 0.937745

$\text{Information Gain(IG)} = \text{E(Target)} - \text{E(Target,A)} = 1- 0.9337745 = 0.062255$

Calculating info gain for attribute B

Data

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

$$H(X) = \mathbb{E}_X[I(x)] = -\sum_{x \in \mathbb{X}} p(x) \log p(x).$$

Information gain for Var B

Var B has value >=3 for 12 records out of 16 and 4 records with value <5 value.

- For Var B >= 3 & class == positive: 8/12
- For Var B >= 3 & class == negative: 4/12
  - Entropy(8,4) = -1 * ((8/12)*log2(8/12) + (4/12)*log2(4/12)) = 0.39054
- For VarB <3 & class == positive: 0/4
- For Var B <3 & class == negative: 4/4
  - Entropy(0,4) = -1 * ((0/4)*log2(0/4) + (4/4)*log2(4/4)) = 0

Entropy(Target, B) = P(>=3) * E(8,4) + P(<3) * E(0,4)
= (12/16) * 0.39054 + (4/16) * 0 = 0.292905

Information Gain(IG) = E(Target) - E(Target,B) = 1- 0.292905= 0.707095

Calculating info gain for attribute C

Data

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

$$H(X) = \mathbb{E}_X[I(x)] = -\sum_{x \in \mathbb{X}} p(x) \log p(x).$$

Information gain for Var C

Var C has value >=4.2 for 6 records out of 16 and 10 records with value <4.2 value.

- For Var C >= 4.2 & class == positive: 0/6
- For Var C >= 4.2 & class == negative: 6/6
  - Entropy(0,6) = 0
- For VarC < 4.2 & class == positive: 8/10
- For Var C < 4.2 & class == negative: 2/10
  - Entropy(8,2) = 0.72193

Entropy(Target, C) = P(>=4.2) * E(0,6) + P(< 4.2) * E(8,2)
= (6/16) * 0 + (10/16) * 0.72193 = 0.4512

Information Gain(IG) = E(Target) - E(Target,C) = 1- 0.4512= 0.5488

Calculating info gain for attribute D

$$H(X) = \mathbb{E}_X[I(x)] = -\sum_{x \in \mathbb{X}} p(x) \log p(x).$$

Data

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

Information gain for Var D

Var D has value >=1.4 for 5 records out of 16 and 11 records with value <5 value.

- For Var D >= 1.4 & class == positive: 0/5
- For Var D >= 1.4 & class == negative: 5/5
  - Entropy(0,5) = 0
- For Var D < 1.4 & class == positive: 8/11
- For Var D < 14 & class == negative: 3/11
  - Entropy(8,3) = -1 * ((8/11)*log2(8/11) + (3/11)*log2(3/11)) = 0.84532

Entropy(Target, D) = P(>=1.4) * E(0,5) + P(< 1.4) * E(8,3)
= 5/16 * 0 + (11/16) * 0.84532 = 0.5811575

Information Gain(IG) = E(Target) - E(Target,D) = 1- 0.5811575 = 0.41189

Summary of calculations

		Target: Positive	Target: Negative
A	>= 5.0	5	7
	<5	3	1
Information Gain of A = 0.062255			

		Target: Positive	Target: Negative
B	>= 3.0	8	4
	< 3.0	0	4
Information Gain of B= 0.7070795			

		Target: Positive	Target: Negative
C	>= 4.2	0	6
	< 4.2	8	2
Information Gain of C= 0.5488			

		Target: Positive	Target: Negative
D	>= 1.4	0	5
	< 1.4	8	3
Information Gain of D= 0.41189			

Constructing decision tree

- Now we know info gain from choosing current cutoffs (previous slide), we can build decision tree
- How to construct tree?
  - More info gain -> Better/higher node
  - Entropy == 0 -> Leaf node
  - Entropy > 0 -> Node needs further splitting

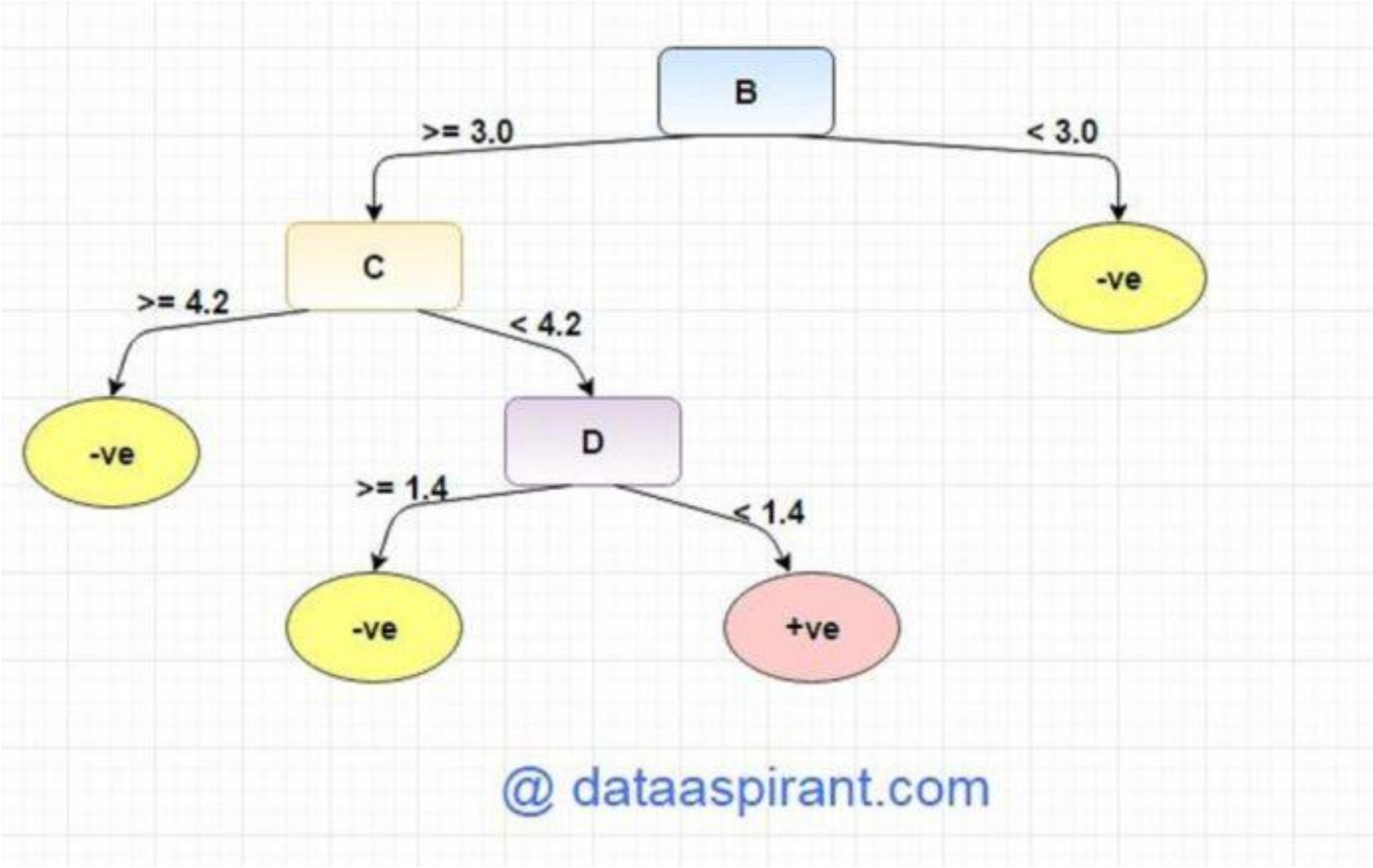
Constructing decision tree

		Target	
		Positive	Negative
A	>= 5.0	5	7
	<5	3	1
Information Gain of A = 0.062255			

		Target	
		Positive	Negative
B	>= 3.0	8	4
	< 3.0	0	4
Information Gain of B= 0.7070795			

		Target	
		Positive	Negative
C	>= 4.2	0	6
	< 4.2	8	2
Information Gain of C= 0.5488			

		Target	
		Positive	Negative
D	>= 1.4	0	5
	< 1.4	8	3
Information Gain of D= 0.41189			



Using gini index to quantify how good “cutoffs” are

Example of using gini index as criterion

- Gini index – metric that measures how often a randomly chosen element is correctly identified
  - Means attributes with lower gini index are preferred

Gini Formula

$$I_G = 1 - \sum_{j=1}^{c} p_j^2$$

Example of using gini index as criterion

Let's assume previous set of data and random choice of thresholds

Data

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

Let's choose some random values/thresholds to categorize each attribute:

A	**B**	**C**	**D**
>= 5	>= 3.0	>= 4.2	>= 1.4
< 5	< 3.0	< 4.2	< 1.4

https://dataaspirant.com/how-decision-tree-algorithm-works/

Calculating gini index for variable A

Data

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

Gini Index for Var A

Var A has value >=5 for 12 records out of 16 and 4 records with value <5 value.

- For Var A >= 5 & class == positive: 5/12
- For Var A >= 5 & class == negative: 7/12
    - gini(5,7) = 1- ((5/12)2 + (7/12)2) = 0.4860
- For Var A <5 & class == positive: 3/4
- For Var A <5 & class == negative: 1/4
    - gini(3,1) = 1- ((3/4)2 + (1/4)2) = 0.375

By adding weight and sum each of the gini indices:

$$\mathrm{gini}(\mathrm{Target},\ A) = (12/16) * (0.486) + (4/16) * (0.375) = 0.45825$$

Calculating gini index for variable B

<u>Data</u>

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

Gini Index for Var B

Var B has value >=3 for 12 records out of 16 and 4 records with value <5 value.

- For Var B >= 3 & class == positive: 8/12
- For Var B >= 3 & class == negative: 4/12
    - gini(8,4) = 1- ((8/12)2 + (4/12)2) = 0.446
- For Var B <3 & class == positive: 0/4
- For Var B <3 & class == negative: 4/4
    - gin(0,4) = 1- ((0/4)2 + (4/4)2) = 0

$\mathrm{gini}(\mathrm{Target},\ \mathrm{B}) = (12/16)\ ^{*}\ 0.446 + (4/16)\ ^{*}\ 0 = 0.3345$

Calculating gini index for variable C

Data

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

Gini Index for Var C

Var C has value >=4.2 for 6 records out of 16 and 10 records with value <4.2 value.

- For Var C >= 4.2 & class == positive: 0/6
- For Var C >= 4.2 & class == negative: 6/6
    - gini(0,6) = 1- ((0/8)2 + (6/6)2) = 0
- For Var C < 4.2& class == positive: 8/10
- For Var C < 4.2 & class == negative: 2/10
    - gin(8,2) = 1- ((8/10)2 + (2/10)2) = 0.32

$\mathrm{gini(Target,\ C)} = (6/16)\ *\ 0 +\ (10/16)\ *\ 0.32 = 0.2$

Calculating gini index for variable D

<u>Data</u>

	A	B	C	D	E
1	4.8	3.4	1.9	0.2	positive
2	5	3	1.6	0.2	positive
3	5	3.4	1.6	0.4	positive
4	5.2	3.5	1.5	0.2	positive
5	5.2	3.4	1.4	0.2	positive
6	4.7	3.2	1.6	0.2	positive
7	4.8	3.1	1.6	0.2	positive
8	5.4	3.4	1.5	0.4	positive
9	7	3.2	4.7	1.4	negative
10	6.4	3.2	4.5	1.5	negative
11	6.9	3.1	4.9	1.5	negative
12	5.5	2.3	4	1.3	negative
13	6.5	2.8	4.6	1.5	negative
14	5.7	2.8	4.5	1.3	negative
15	6.3	3.3	4.7	1.6	negative
16	4.9	2.4	3.3	1	negative

Gini Index for Var D

Var D has value >=1.4 for 5 records out of 16 and 11 records with value <1.4 value.

- For Var D >= 1.4 & class == positive: 0/5
- For Var D >= 1.4 & class == negative: 5/5
  - gini(0,5) = 1- ((0/5)2 + (5/5)2) = 0
- For Var D < 1.4 & class == positive: 8/11
- For Var D < 1.4 & class == negative: 3/11
  - gin(8,3) = 1- ((8/11)2 + (3/11)2) = 0.397

$$\mathrm{gini}(\mathrm{Target},\ \mathrm{D}) = (5/16) * 0 + (11/16) * 0.397 = 0.273$$

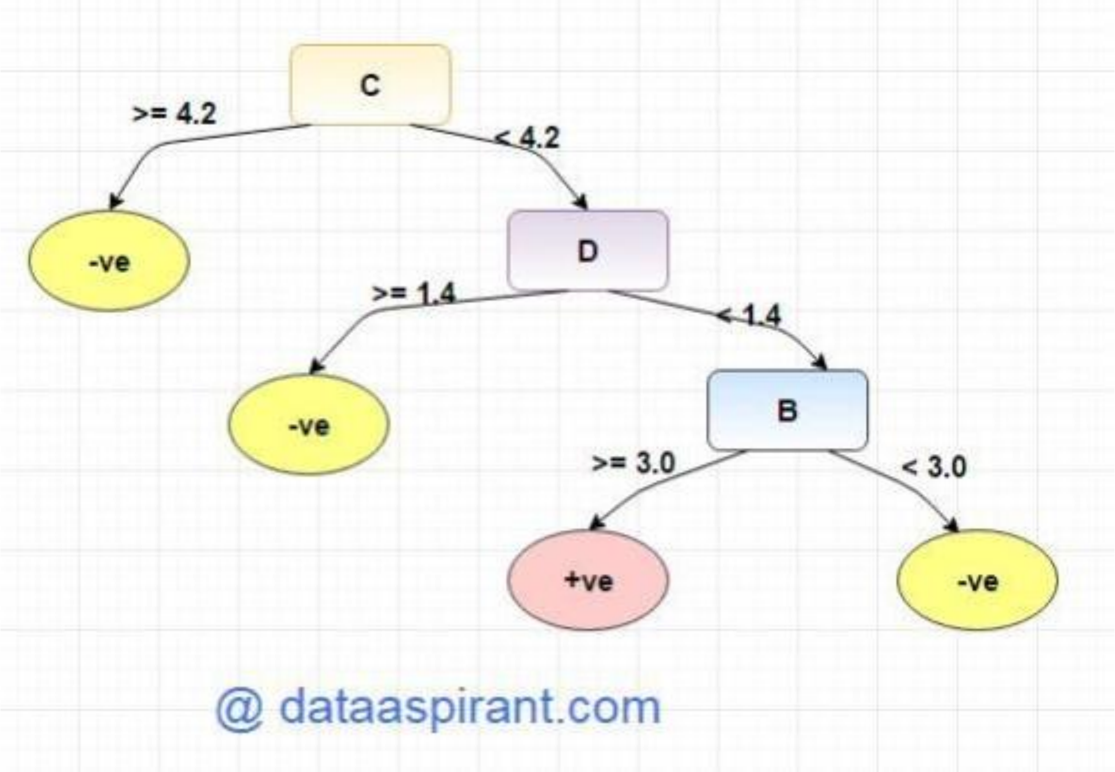
Constructing decision tree from gini index

		wTarget	
		Positive	Negative
A	>= 5.0	5	7
	<5	3	1
Ginin Index of A = 0.45825			

		Target	
		Positive	Negative
B	>= 3.0	8	4
	< 3.0	0	4
Gini Index of B= 0.3345			

		Target	
		Positive	Negative
C	>= 4.2	0	6
	< 4.2	8	2
Gini Index of C= 0.2			

		Target	
		Positive	Negative
D	>= 1.4	0	5
	< 1.4	8	3
Gini Index of D= 0.273			



Remember: Lower gini index = better/more desirable

Doing research to predict something by yourself is hard… but doing research with a team of people is easier

One decision tree -> accuracy not great…but multiple decision trees -> higher accuracy

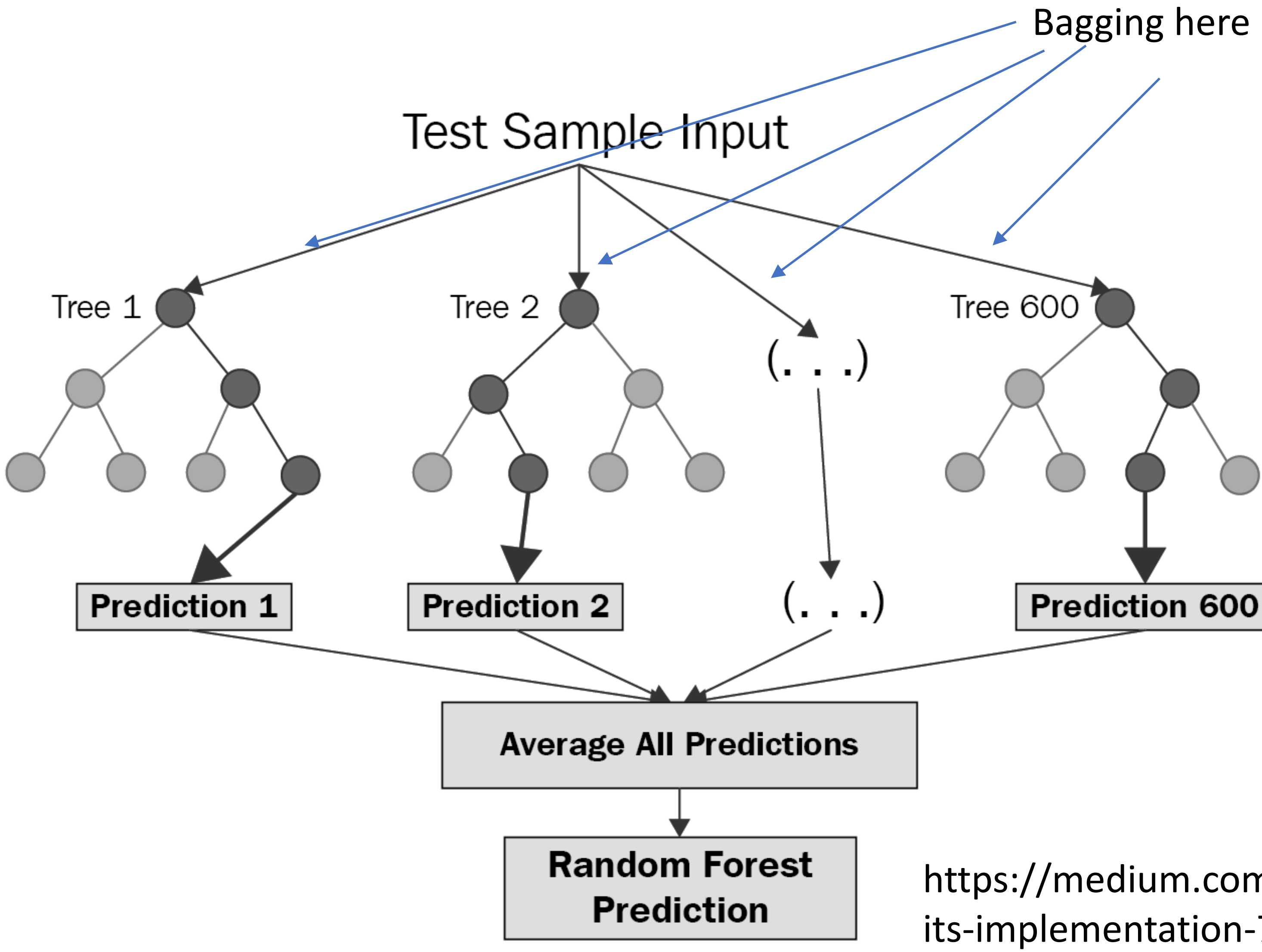
Thus -> use random forests

What’s a random forest?

- TLDR: Classification technique that combines/averages results from multiple decision trees where input data for each decision tree comes from random sampling with replacement (bootstrap aggregation/bagging)
  - Why bagging? -> generates data sets that have low variance
  - Why combine results from multiple decision trees? Improves accuracy

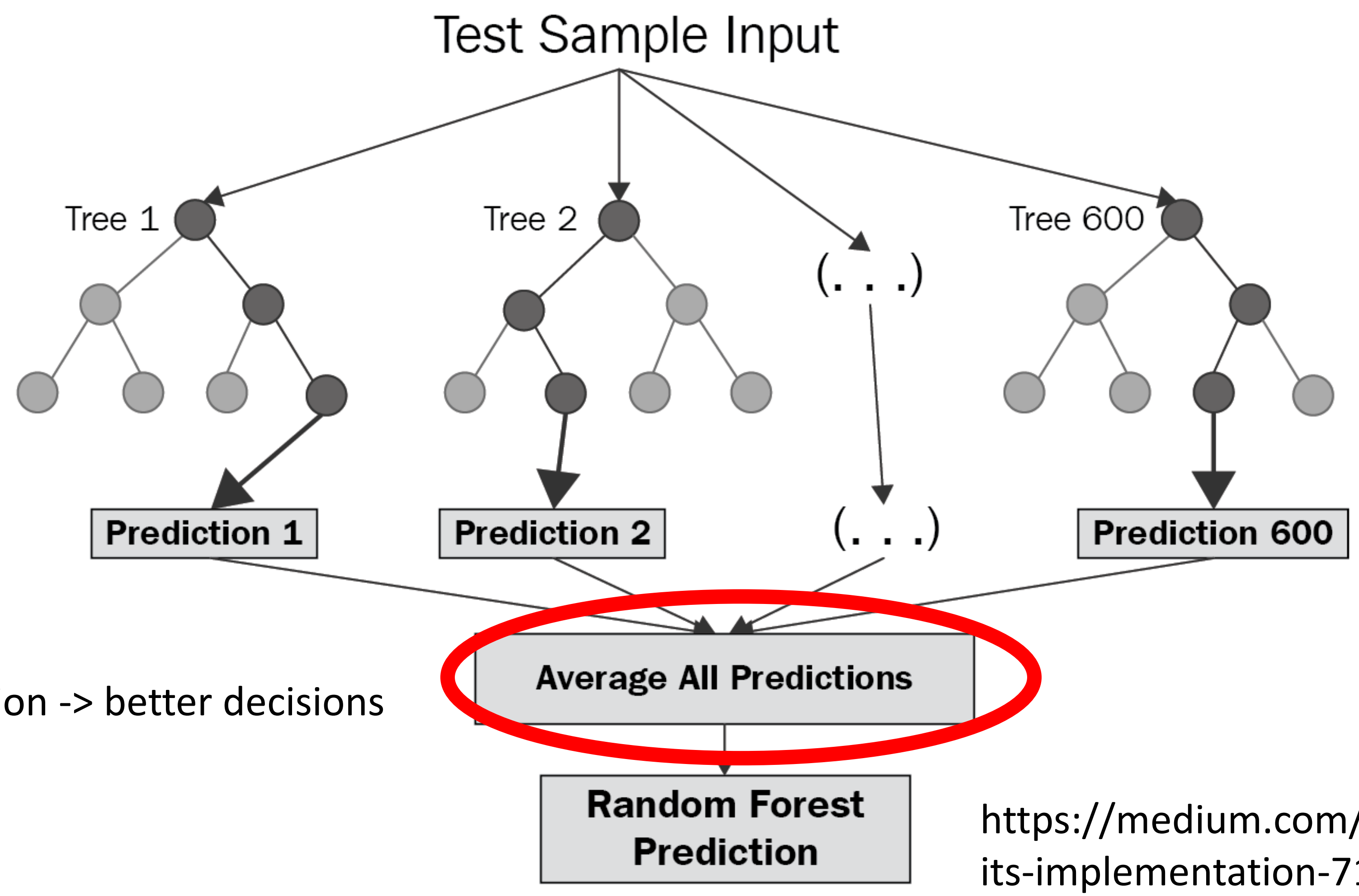
- “A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting” (https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html)
  - Meta-estimator – combines results of multiple predictions

Visualization of random forest



https://medium.com/swlh/random-forest-and-its-implementation-71824ced454f

How does random forest work?



Avg of more information -> better decisions

https://medium.com/swlh/random-forest-and-its-implementation-71824ced454f

Code

- pIC50 prediction (drug potency prediction) using scikit-learn

References for content

1. https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/?ref=rp
2. https://dataaspirant.com/how-decision-tree-algorithm-works/
3. https://www.quora.com/What-is-difference-between-Gini-Impurity-and-Entropy-in-Decision-Tree
4. https://dataaspirant.com/random-forest-algorithm-machine-learing/
5. https://www.geeksforgeeks.org/random-forest-regression-in-python/
6. https://medium.com/swlh/random-forest-and-its-implementation-71824ced454f
7. https://williamkoehrsen.medium.com/random-forest-simple-explanation-377895a60d2d
8. https://towardsdatascience.com/random-forest-in-python-24d0893d51c0

References for code

- https://www.youtube.com/watch?v=wGaGm0sj04M&list=PLtqF5YXg7GLlQJUv9XJ3RWdd5VYGwBHrP&index=4
- https://github.com/dataprofessor/code/blob/master/python/CDD_ML_Part_4_Acetylcholinesterase_Regression_Random_Forest.ipynb
- https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
- https://builtin.com/data-science/random-forest-algorithm
- https://www.collaborativedrug.com/what-is-pic50-2/#:~:text=Simply%20stated%2C%20pIC50%20is%20the,is%20a%20pIC50%20of%209.