

Multivariate Splits

The background of the image is a light gray gradient. It is populated with various 3D geometric shapes in a muted sage green color. These shapes include a small cube in the top left, a large sphere on the left, a rectangular frame in the upper center, a sphere in the upper right, a wavy line in the top right, a cone on the right, a small cube on the bottom right, a cone on the bottom left, a large ring-like structure in the bottom center, and several other spheres and ellipsoids scattered throughout. The shapes are rendered with soft shadows, giving them a three-dimensional appearance.



ROOM 5 Memembers


- | | |
|----------------------|-----------|
| 1. CHHON CHAINA | e20200934 |
| 2. VINLAY ANUSAR | e20201068 |
| 3. Ek Vong Panharith | e20200877 |
| 4. CHEA MAKARA | e20201131 |
| 5. HON RATANA | e20201053 |
| 6. KHENG DALISH | e20200909 |
| 7. SENG LAY | e20200872 |
| 8. Thou chanamakara | e20200227 |
| 9. SRENG Seangleng | e20200840 |
| 10. NOEM KOEMHAK | e20200808 |
- 

Table of Contents

- 1. What are Multivariate Splits?**
- 2. Illustration and Example**
- 3. Implementing multivariate splits**
- 4. CART: finds multivariate splits based on a
linear comb. of attrs**
- 5. Conclusion**

1. What is Multivariate Splits?

+ Multivariate Splits is a type of split in a decision tree that tests multiple variables at the same. In contrast of Univariate Splits which tests only single variable. Multivariate splits can be more powerful because they can capture more complex relationships between variables.

+ By incorporating multiple variable combinations, decision trees can capture more complex relationships and interactions among the features, leading to improved predictive power. With multivariate splits, instead of considering a single feature at a time, the decision tree algorithm evaluates combinations of multiple features to create more nuanced and informative splits.

Example of Illustration

```
## -----
## multivariate regression forests
## - comparison of Mahalanobis splitting to standard splitting rule
## - lipids (all real values) used as the multivariate y
## - genes, genotype and diet used as the x features
## -----
library(randomForestSRC)
## load the data
data(nutrigenomic)

## parse into y and x data
ydta <- nutrigenomic$lipids
xdta <- data.frame(nutrigenomic$genes,
                  diet = nutrigenomic$diet,
                  genotype = nutrigenomic$genotype)

## mahalanobis splitting
obj <- rfsrc(get.mv.formula(colnames(ydta)),
            data.frame(ydta, xdta),
            importance=TRUE, nsplit = 10, splitrule = "mahalanobis")

## default composite (independence) splitting
obj2 <- rfsrc(get.mv.formula(colnames(ydta)),
             data.frame(ydta, xdta),
             importance=TRUE, nsplit = 10)

## compare standardized VIMP for top 25 variables
imp <- data.frame(mahalanobis = rowMeans(get.mv.vimp(obj, standardize = TRUE)),
                 default      = rowMeans(get.mv.vimp(obj2, standardize = TRUE)))
print(100 * imp[order(imp["mahalanobis"], decreasing = TRUE)[1:25], ])

>          mahalanobis      default
> diet          4.6368819 19.674080838
> CYP3A11       2.7681390  2.540102290
> PMNCT         1.8182919   2.815791465
```

Reference

3. Martin PG, Guillou H, Lasserre F, Déjean S, Lan A, Pascussi J-M, et al. Novel aspects of PPAR α -mediated regulation of lipid and xenobiotic metabolism revealed through a nutrigenomic study. Hepatology. 2007;45:767-77



For example, suppose we have a dataset for predicting loan default, and we have features such as income, debt-to-income ratio, and credit score. Instead of considering these variables separately, we can create combinations like "income <= \$40,000 and debt-to-income ratio <= 0.35" and credit score >= 650" to partition the data more effectively.

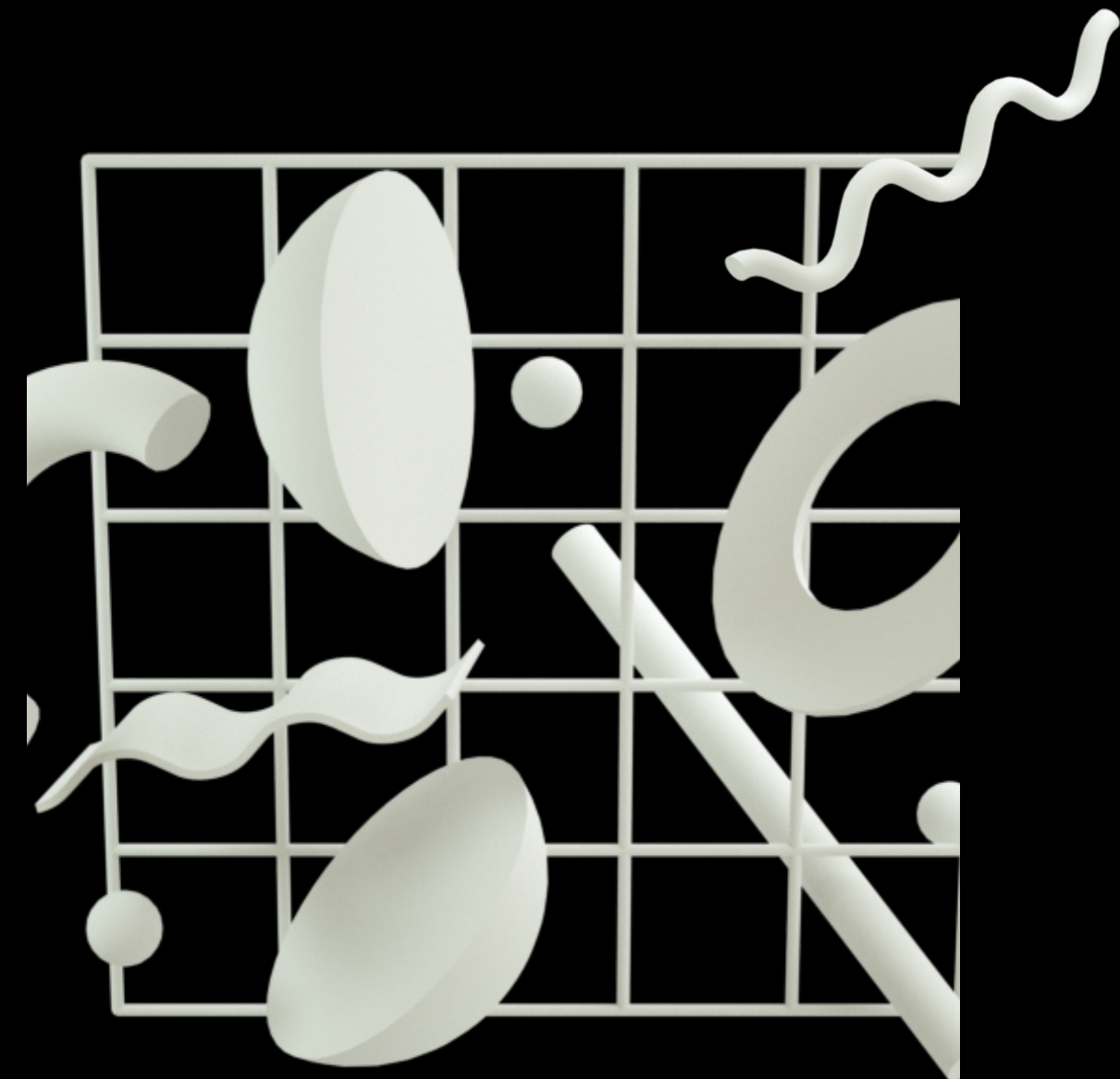
Income	Debt-to-Income Ratio	Credit Score
\$40,000	<=0.35	>=650
\$60,000	>0.50	<600
\$35,000	<=0.42	<550
\$80,000	<=0.30	>=700
\$45,000	<=0.25	>=720
\$55,000	>0.60	<580

Linear Combination = 2 * income - 3 * debt-to-income ratio + 0.5 * credit score

3. Implementing multivariate splits

Implementing multivariate splits in decision trees typically involves modifying the splitting criteria and incorporating combinations of multiple variables. Here's a general approach to implementing multivariate splits:

- 1. Data Preprocessing**
- 2. Define the Target Variable**
- 3. Select Splitting Criteria**
- 4. Attribute Selection**
- 5. Multivariate Splitting**
- 6. Build the Tree**
- 7. Handle Overfitting**
- 8. Evaluate and Validate**



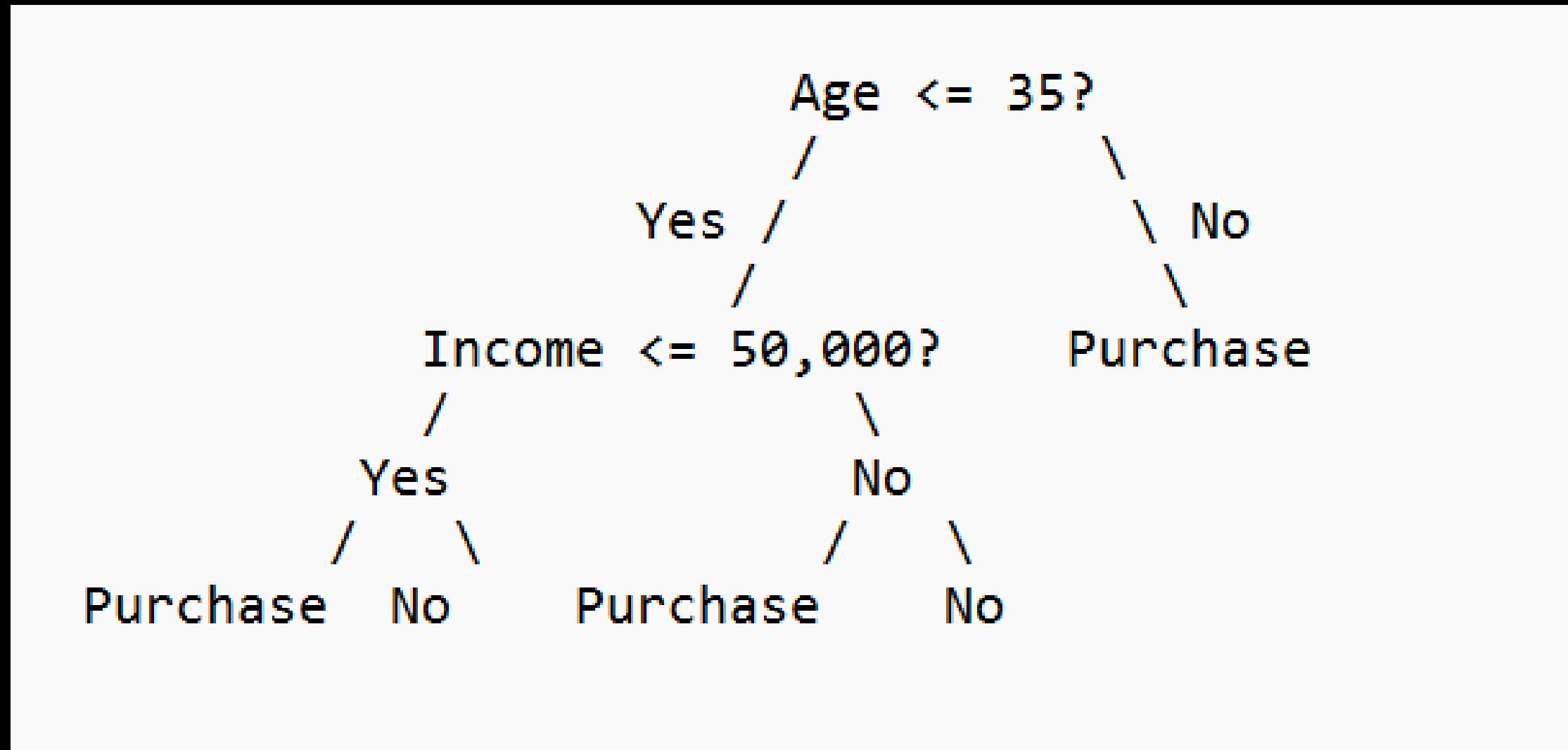
4. CART: finds multivariate splits based on a linear comb. of attrs

The CART (Classification and Regression Trees) algorithm is a decision tree algorithm that can be used to find multivariate splits based on a linear combination of attributes. CART is widely used for both classification and regression tasks.

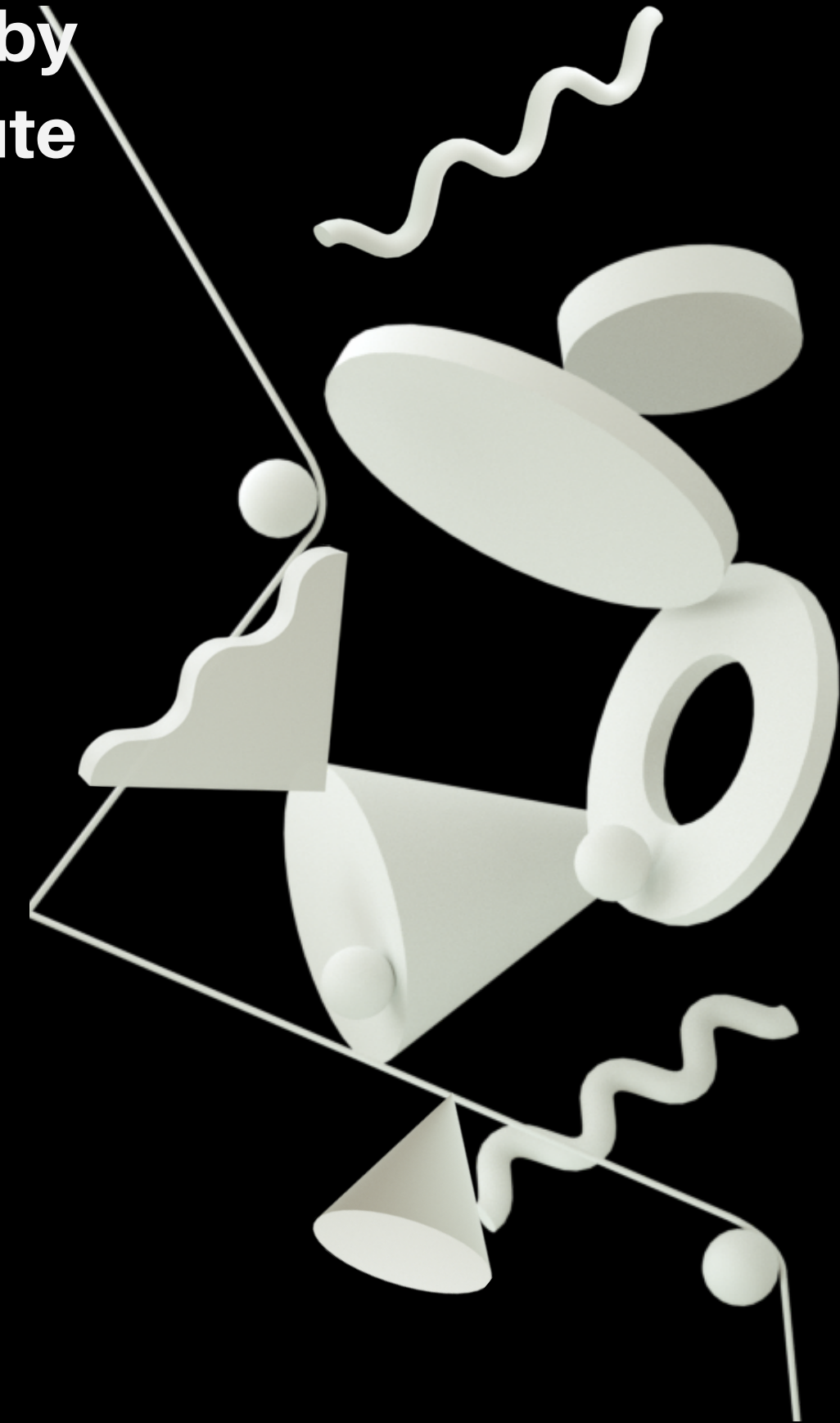
Here's an example to illustrate how the CART algorithm performs univariate splits instead of multivariate splits based on a linear combination of attributes.

Age	Income	Purchase
25	40,000	1
30	60,000	0
35	55,000	1
40	70,000	1
45	50,000	0
50	80,000	1

By using the CART algorithm, the decision tree would be built by recursively performing univariate splits. Let's consider the attribute "Age" as the first splitting attribute.



In this example, the CART algorithm performs univariate splits, evaluating each attribute independently and selecting the best split based on a chosen criterion at each node. It does not consider linear combinations of attributes for multivariate splits.



5. Conclusion



In conclusion, multivariate splits refer to partitioning a dataset based on combinations of multiple variables. Instead of considering individual variables independently, multivariate splits allow for capturing relationships and interactions among multiple features simultaneously. This can lead to more informative and nuanced splits in decision tree algorithms and other machine learning models. The purpose of CART is to recursively build a decision tree by selecting the best attribute and threshold at each node to minimize impurity or maximize information gain. The algorithm evaluates all possible thresholds for each attribute and selects the one that optimizes the splitting criterion.





Thank you

