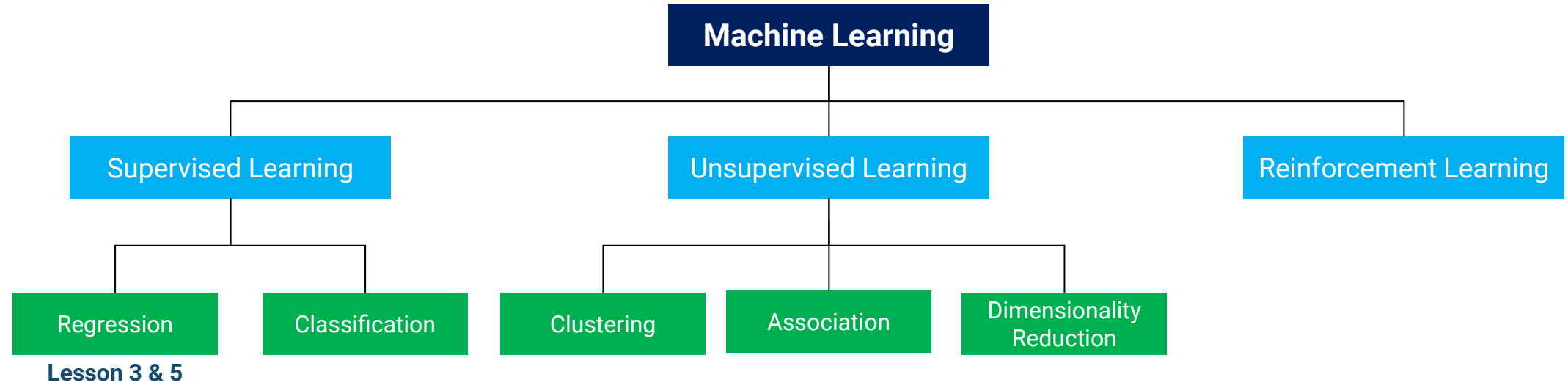
AI200: APPLIED MACHINE LEARNING

DECISION TREE (REGRESSION TREE)

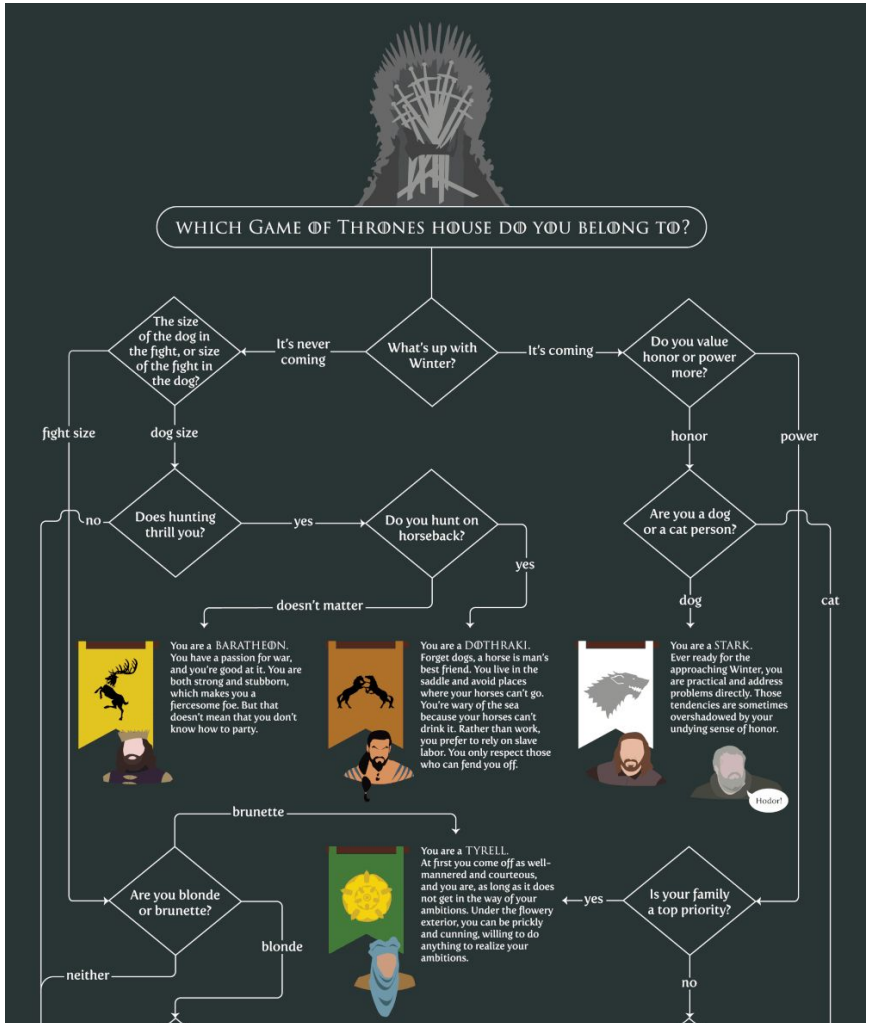
OVERVIEW & LITERATURE OF MACHINE LEARNING: TYPES OF MODELS

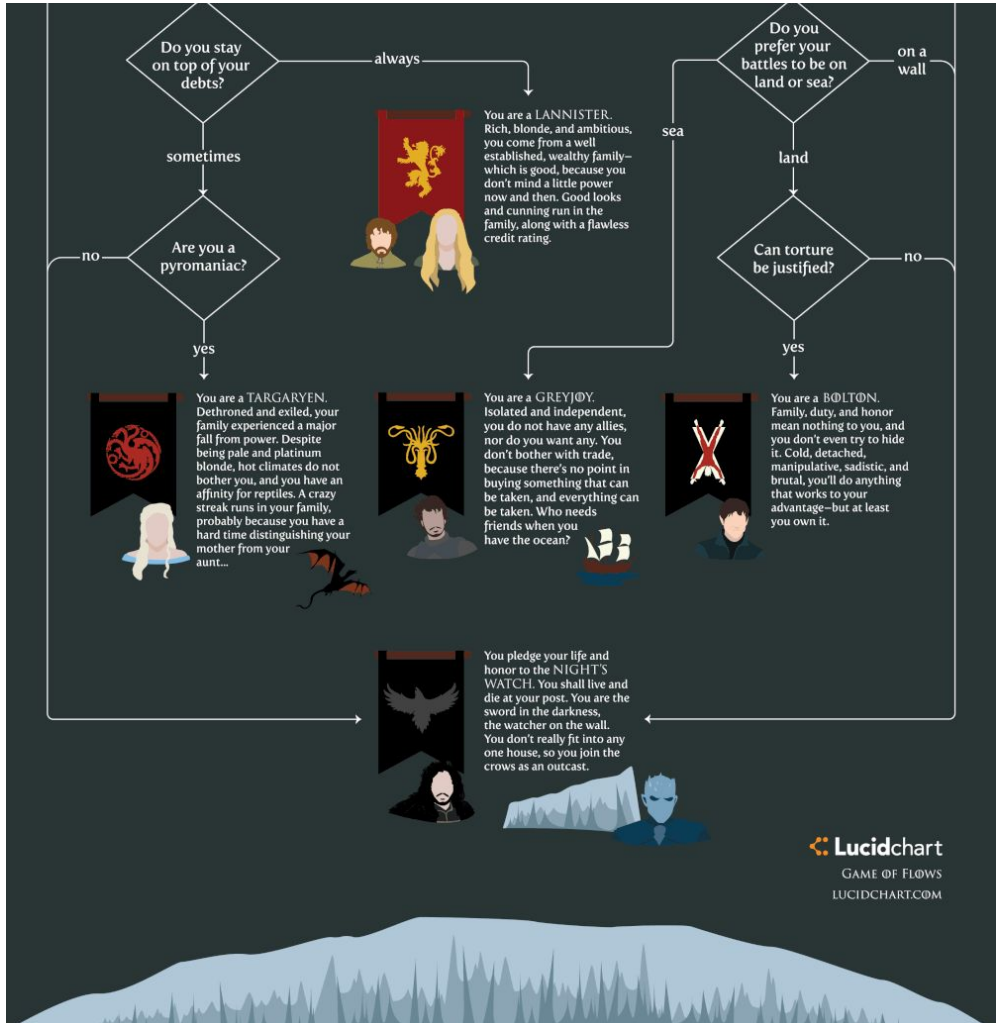


1. Spectrum of supervision					
2. Types of problems	Regression **Lesson 3 & 5**	Classification	Clustering	Association	Dimensionality Reduction
3. Types of algorithm / models	Linear regression Decision Tree Random Forest XGBoost CatBoost	Logistic Regression KNN SVM Decision Tree Random Forest XGBoost CatBoost	K-Means Hierarchical Clustering Hidden Markov Models Gaussian Mixture	Apriori Algorithm FP-Growth Algorithm Eclat Algorithm	Principal Component Analysis (PCA) Linear Discriminant Analysis (LDA) t-SNE
4. Examples of applications	What will be the price of S&P on Monday?	Does the patient have cancer or not?	Which customers are similar to one another?	What product should Amazon cross-sell to customer A?	

WHAT IS DECISION TREE: LAYMAN INTUITION

- Have you played this before? This flowchart closely resembles the intuition behind decision trees.

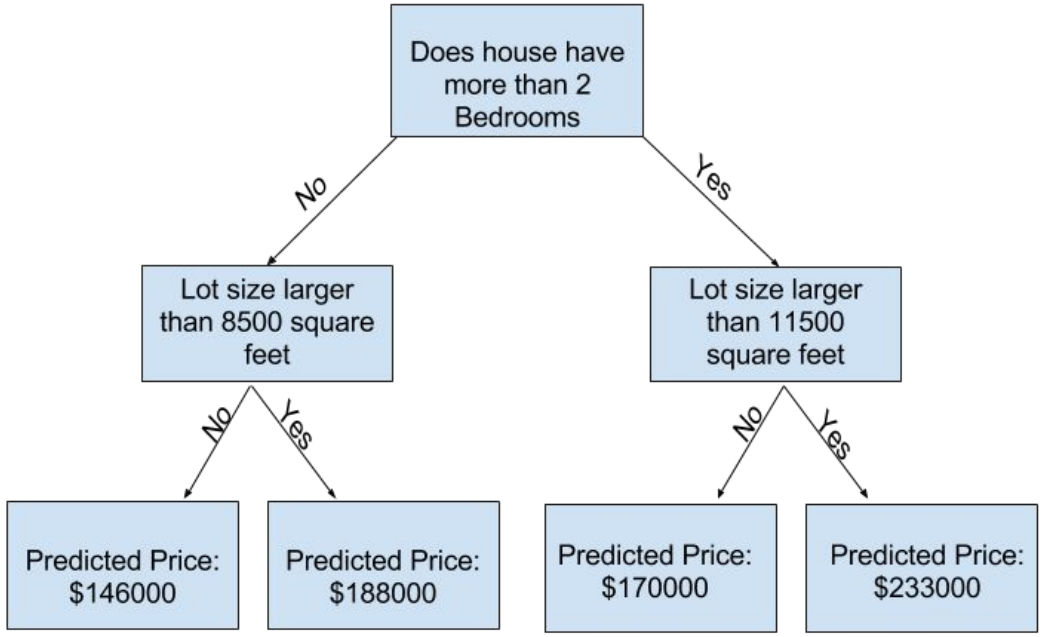




WHAT IS DECISION TREE: LAYMAN INTUITION

- The idea behind decision tree is very simple:
  - Imagine your cousin has made millions of dollars speculating on real estate, and he wants to partner with you because of your expertise in data science. He'll supply the money, and you'll **supply models that predict how much various houses are worth**.
  - You ask him how he predicted real estate values in the past and he says it is just intuition.
  - But more questioning reveals that he's identified price patterns from houses he has seen in the past, and he uses those patterns to make predictions for new houses he is considering to speculate on.

  - When you build a decision tree, it'll also look very much like your cousin's intuition, only far more consistent and transparent!

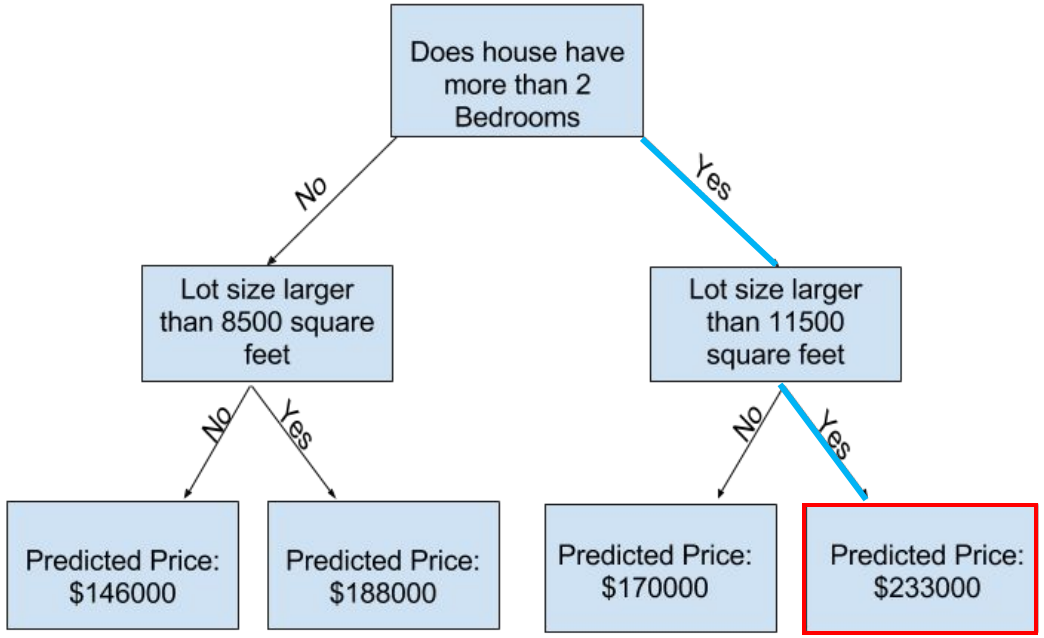


As you progress, you will begin to fall in love with Decision Trees. Not only are they easy to understand, they are the basic building block for some of the best models in data science (which will be covered subsequently).

WHAT IS DECISION TREE: LAYMAN INTUITION

- The idea behind decision tree is very simple:
  - Imagine your cousin has made millions of dollars speculating on real estate, and he wants to partner with you because of your expertise in data science. He'll supply the money, and you'll **supply models that predict how much various houses are worth**.
  - You ask him how he predicted real estate values in the past and he says it is just intuition.
  - But more questioning reveals that he's identified price patterns from houses he has seen in the past, and he uses those patterns to make predictions for new houses he is considering to speculate on.

  - When you build a decision tree, it'll also look very much like your cousin's intuition, only far more consistent and transparent!



Features: 3 bedroom, 12000 sq ft
Prediction: $233000

As you progress, you will begin to fall in love with Decision Trees. Not only are they easy to understand, they are the basic building block for some of the best models in data science (which will be covered subsequently).

DECISION TREE

MECHANISM BEHIND MODEL

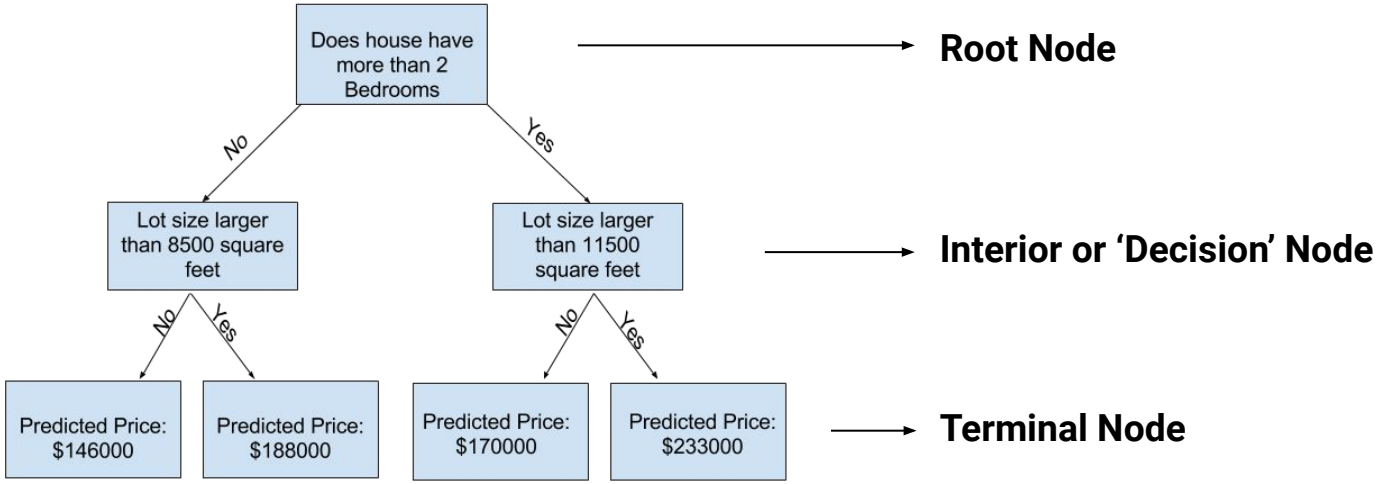
MECHANISM BEHIND MODEL: PARTITIONING

- Every tree-based model is built top-down from a root node. At every ***decision node***, data is partitioned into subsets containing similar instances. Each partition is also known as a “decision split”.

- Another way to look at this is, each decision splits is chosen to minimize Residual Sum of Squares (RSS) or mean squared error, which are measures of “deviation” between the prediction and observed data

$$\text{RSS} = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

- Let’s seek to understand the mechanism from a graphical point of view.



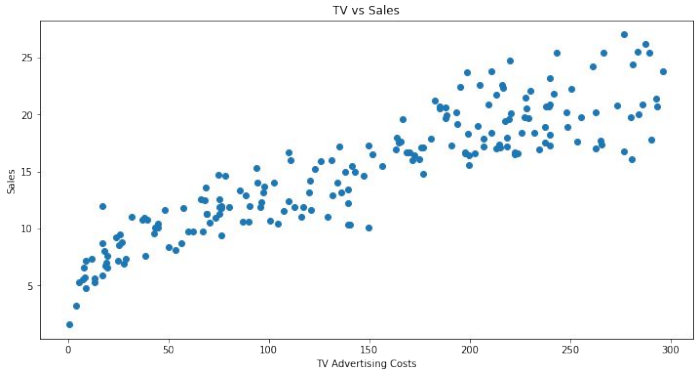
MECHANISM BEHIND MODEL: PARTITIONING

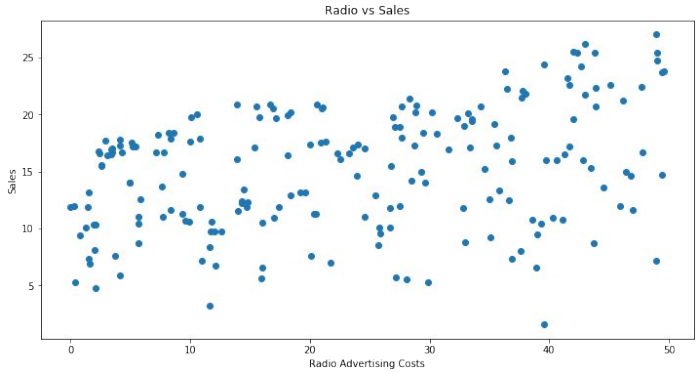
- To illustrate the mechanism of Decision Trees, we use the same dataset example as the one from Linear Regression.
- We would like to predict how spending in different advertising mediums affects sales of a company

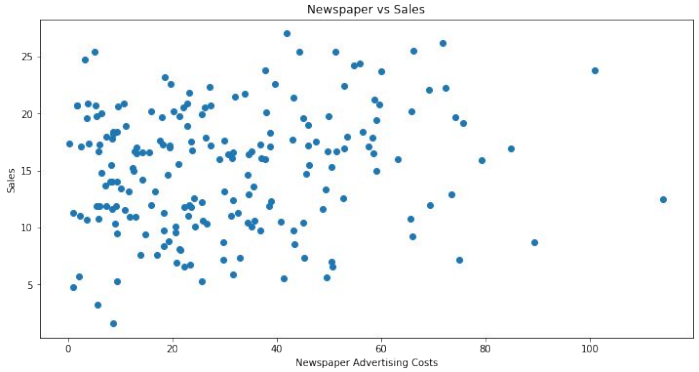
Features **Outcome**

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

200 rows × 4 columns





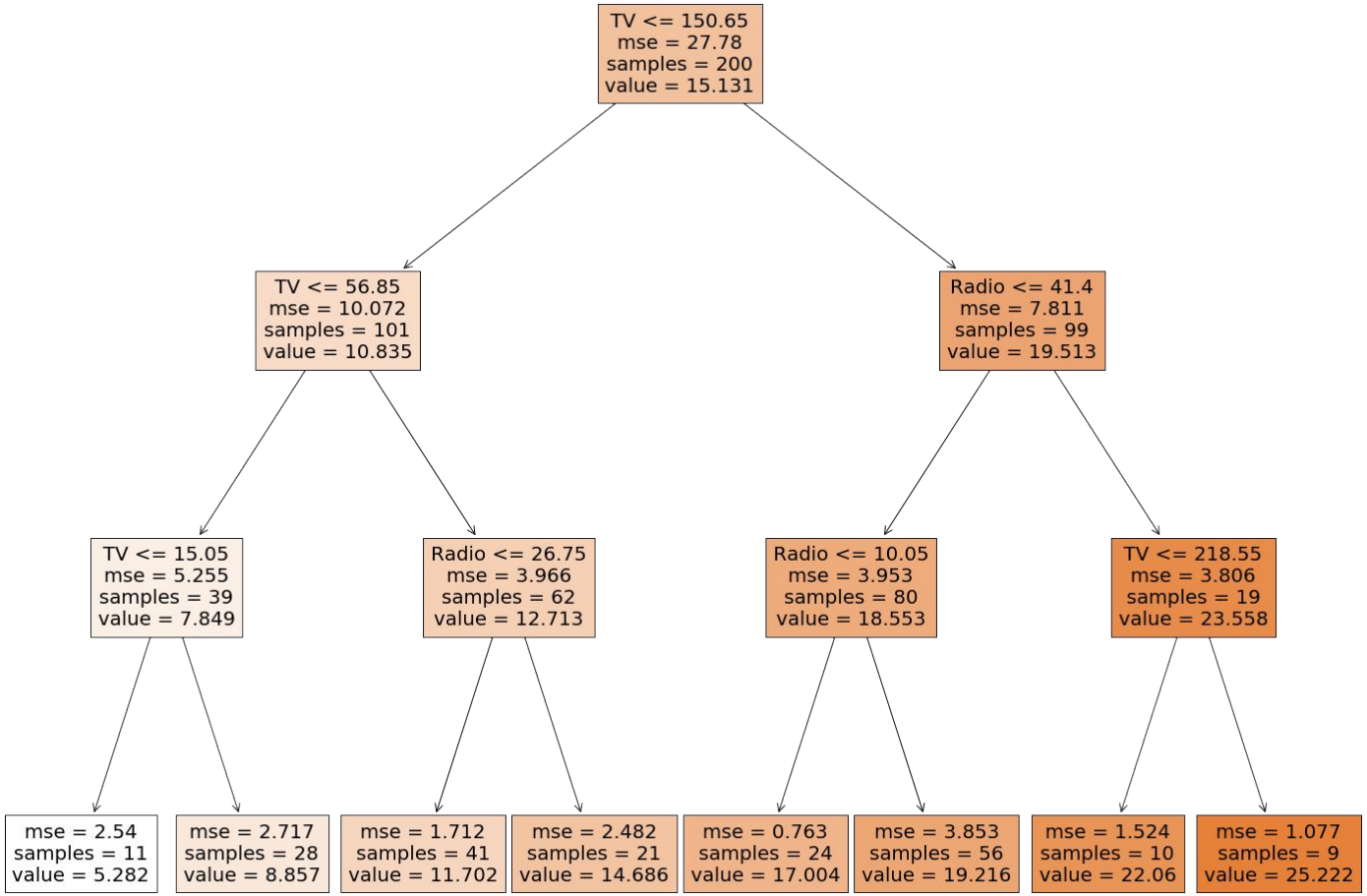


MECHANISM BEHIND MODEL: PARTITIONING

- Let's create an example decision tree with the advertising dataset, using only the features '**TV**' and '**Radio**' to predict **Sales**
- Using the final generated decision tree, we will show with step-by-step illustration how this decision tree regressor was constructed

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...	...	...	...	...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

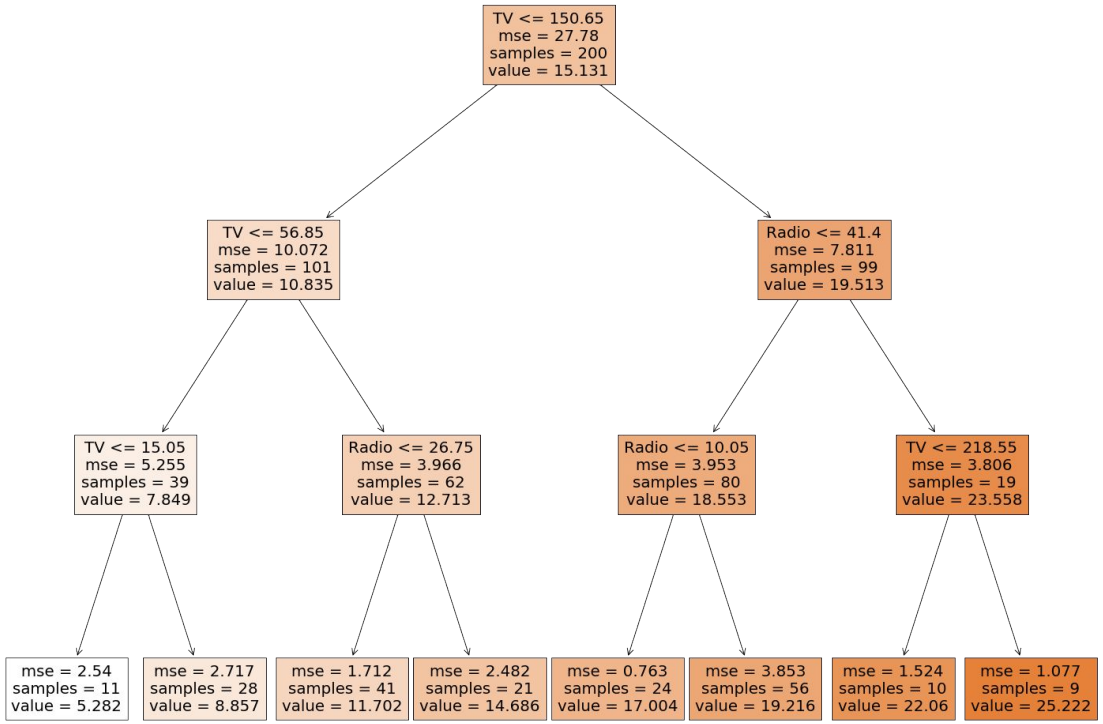
200 rows × 4 columns



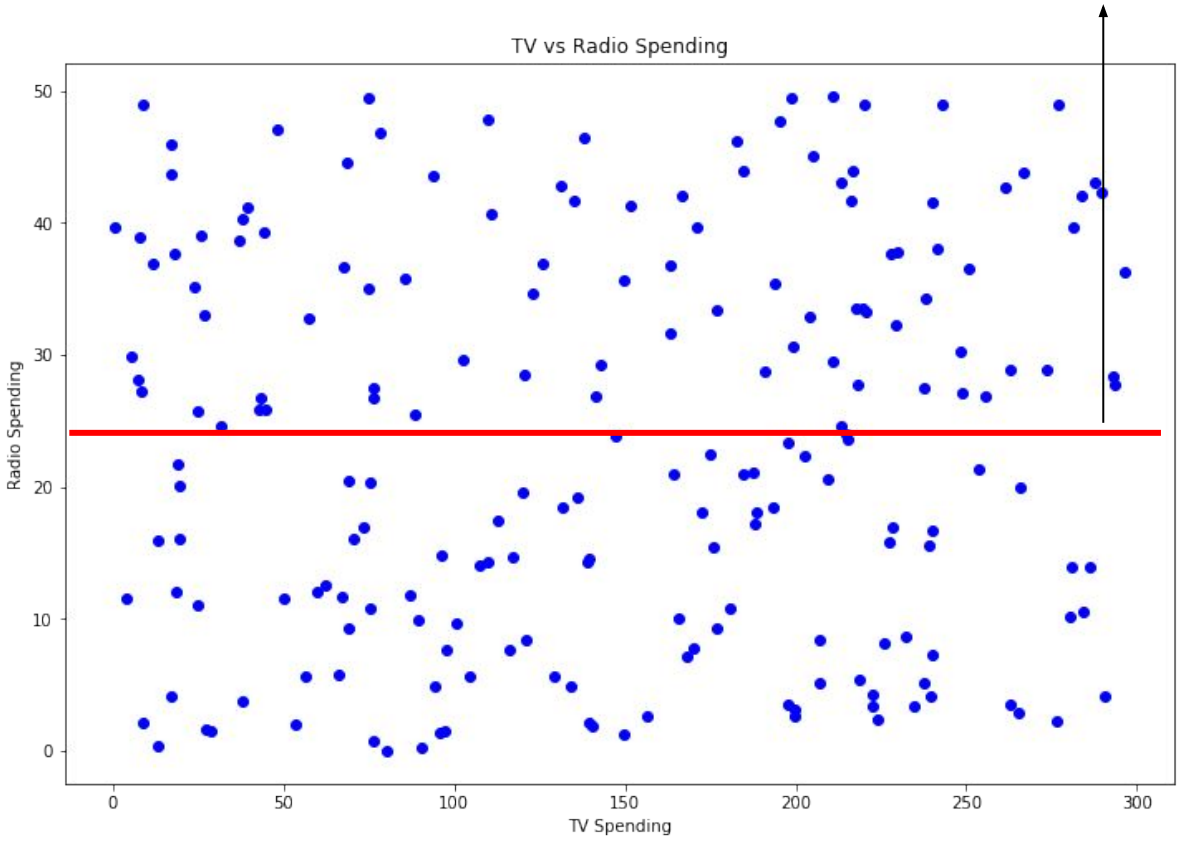
MECHANISM BEHIND MODEL: PARTITIONING

- **Step 1**: At the very initial split, the decision tree algorithm tries various features & values and calculate the RSS for each value.
  It will selects the feature & value that gives the lowest RSS

$$\text{RSS} = \sum_{i=1}^{n}(y_i - f(x_i))^2$$



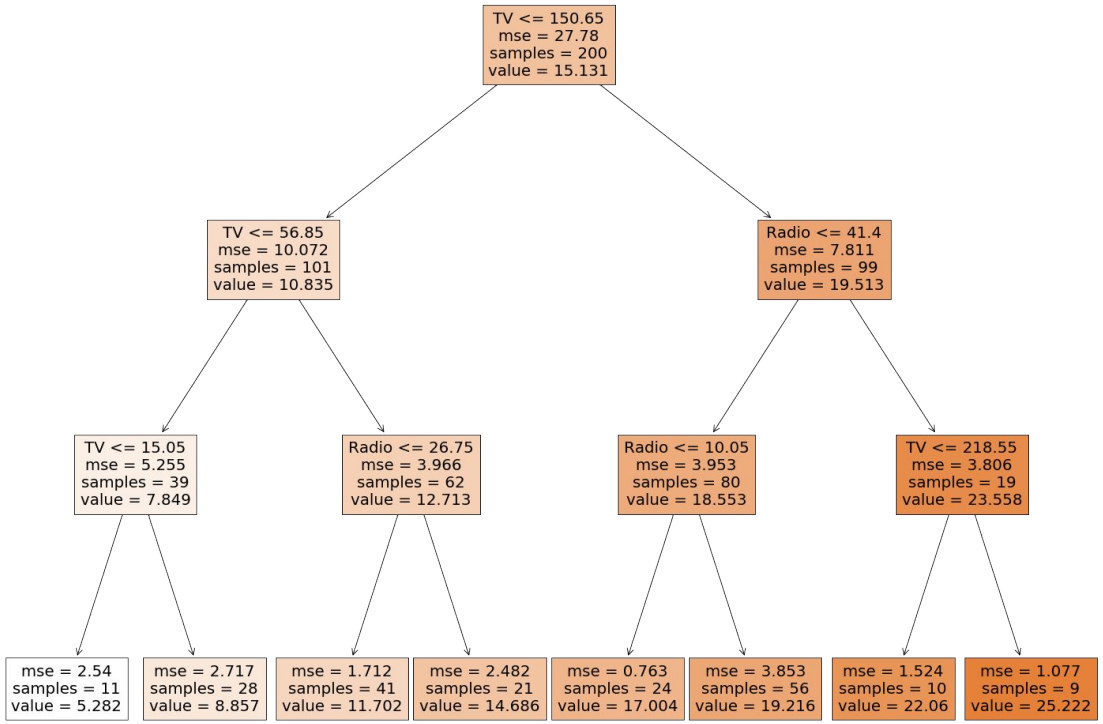
Calculate RSS for this split

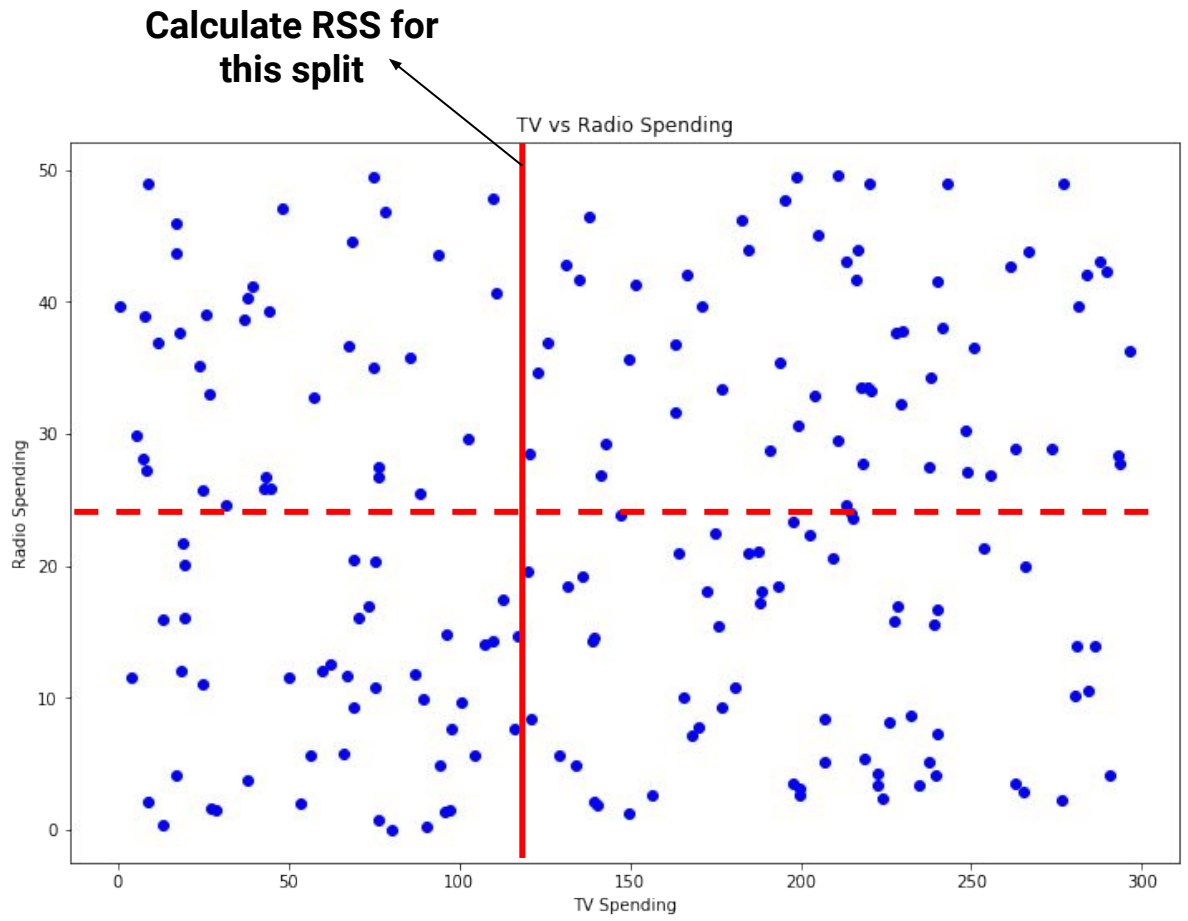


MECHANISM BEHIND MODEL: PARTITIONING

- **Step 1**: At the very initial split, the decision tree algorithm tries various features & values and calculate the RSS for each value.
  It will selects the feature & value that gives the lowest RSS

$$\text{RSS} = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

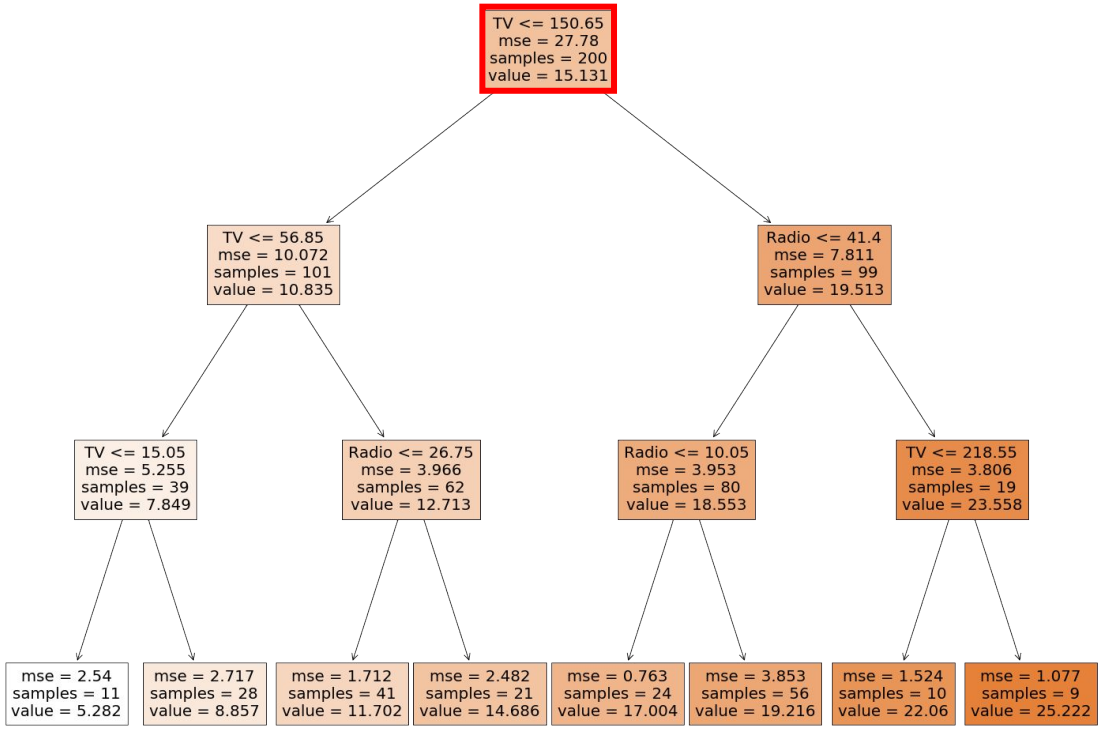




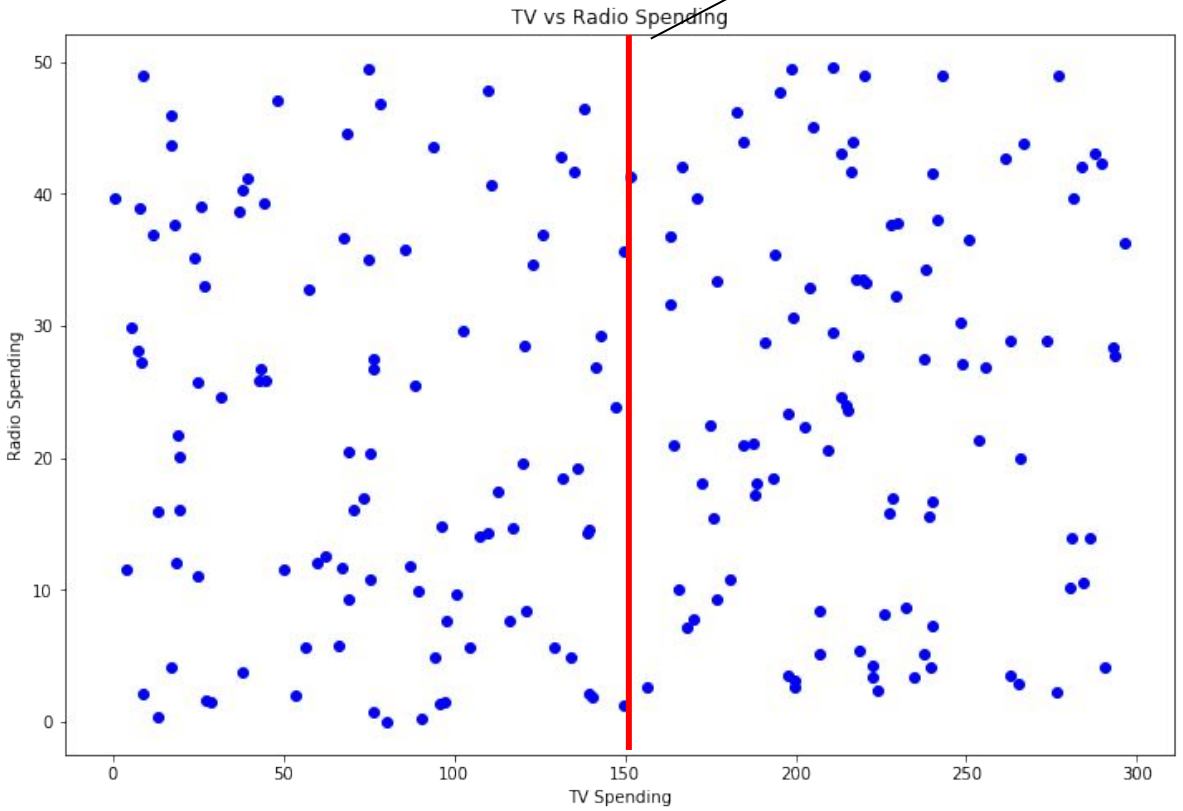
MECHANISM BEHIND MODEL: PARTITIONING

- **Step 1**: At the very initial split, the decision tree algorithm tries various features & values and calculate the RSS for each value.
  It will selects the feature & value that gives the lowest RSS

$$\text{RSS} = \sum_{i=1}^{n}(y_i - f(x_i))^2$$



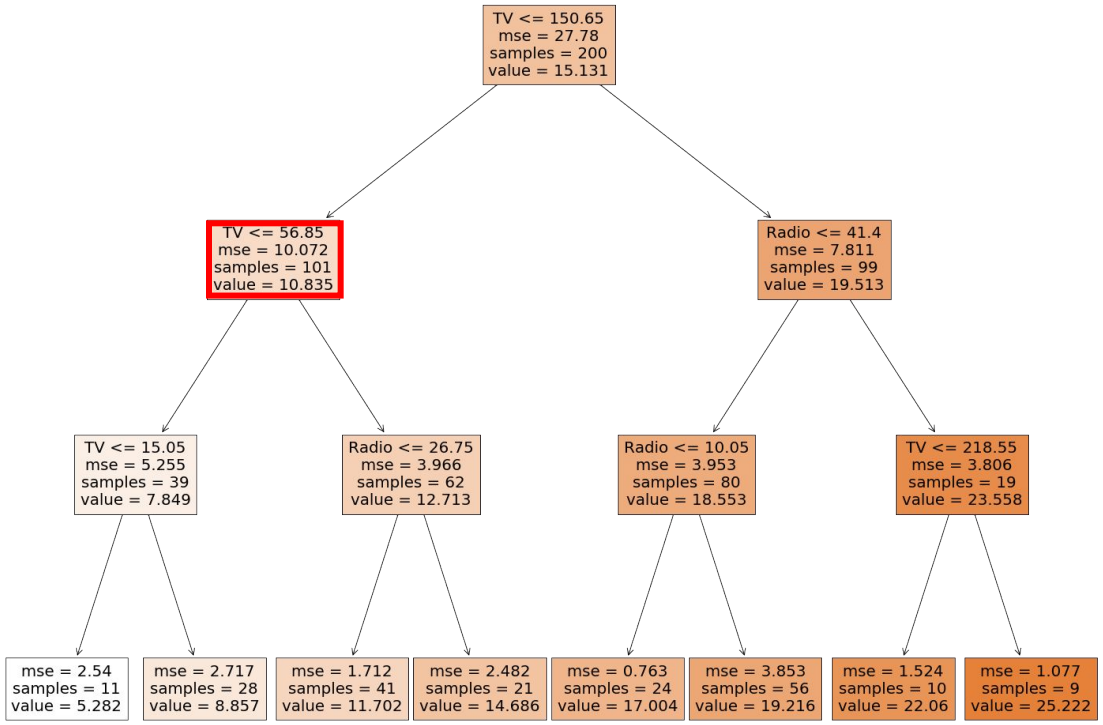
This is the split with the lowest RSS: 27.78



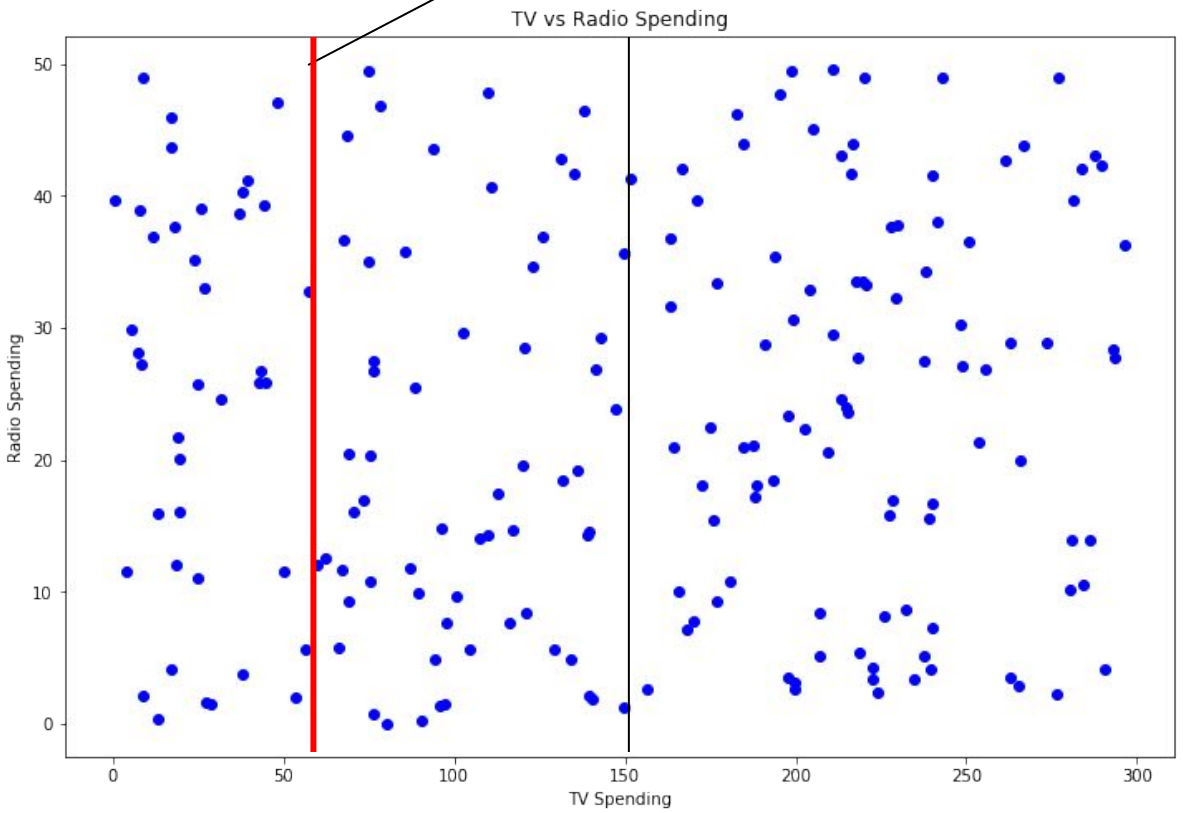
MECHANISM BEHIND MODEL: PARTITIONING

- **Step 2**: Building on the previous split, the decision tree algorithm repeats the process from Step 1 to select the feature & value that gives the next lowest RSS.

$$\text{RSS} = \sum_{i=1}^{n}(y_i - f(x_i))^2$$



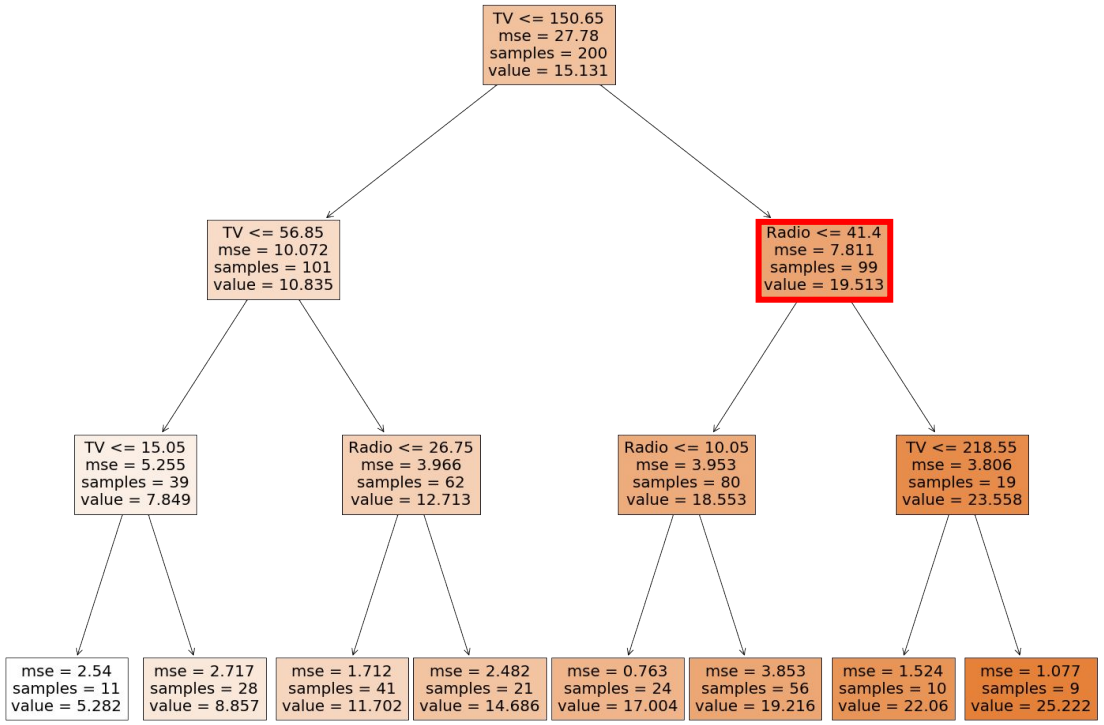
This is the split with the next lowest RSS: 10.835



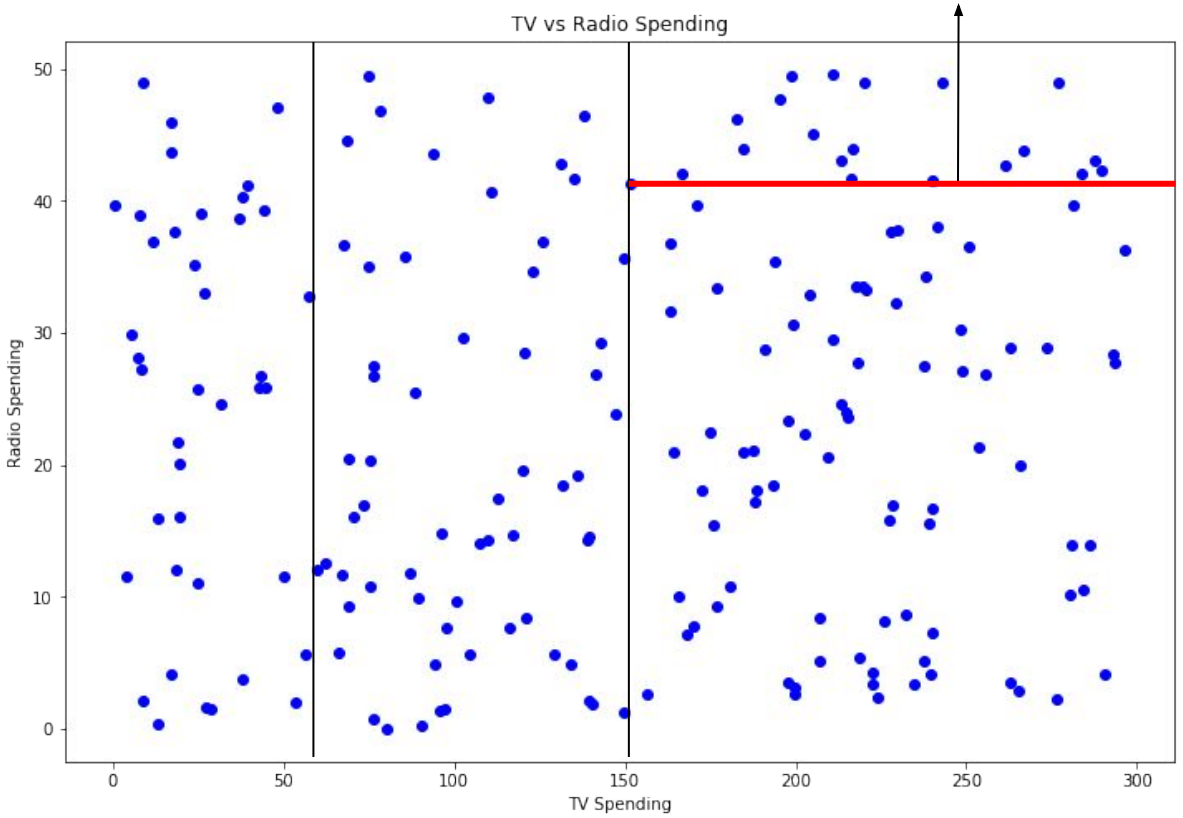
MECHANISM BEHIND MODEL: PARTITIONING

- **Step 3**: Repeat **step 2** to continue splitting until the splits no longer generate improvement in the RSS.

$$\text{RSS} = \sum_{i=1}^{n}(y_i - f(x_i))^2$$



Next split, RSS: 7.811

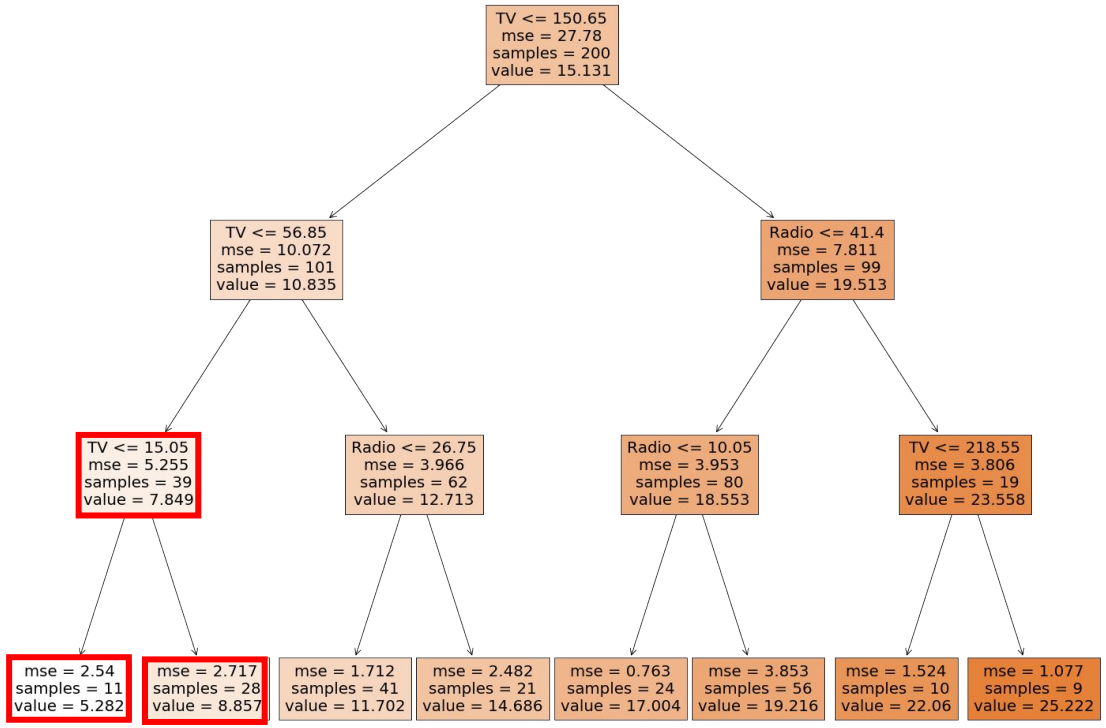


MECHANISM BEHIND MODEL: PARTITIONING

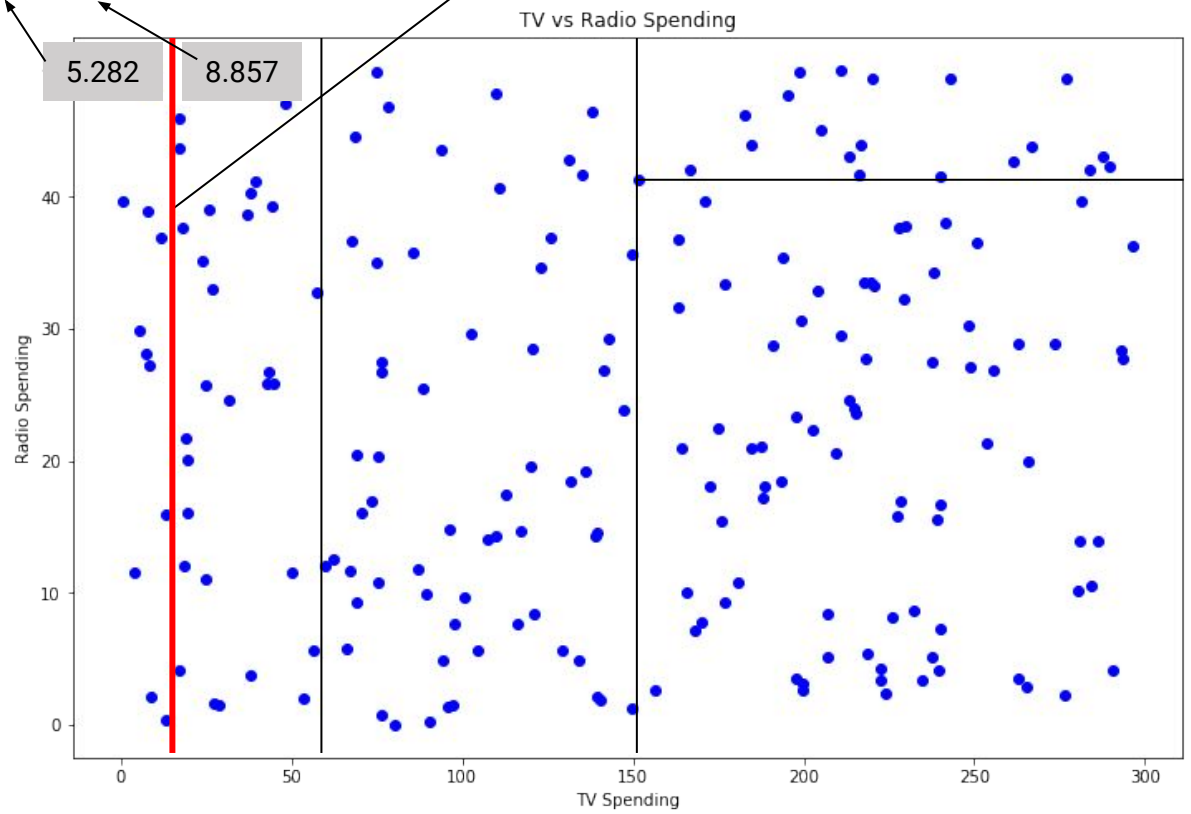
- **Step 3**: Repeat **step 2** to continue splitting until the splits no longer generate improvement in the RSS.

$$\text{RSS} = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

Populate prediction value of zone since it is a terminal node, which is the **average of sales** of all data points in the zone



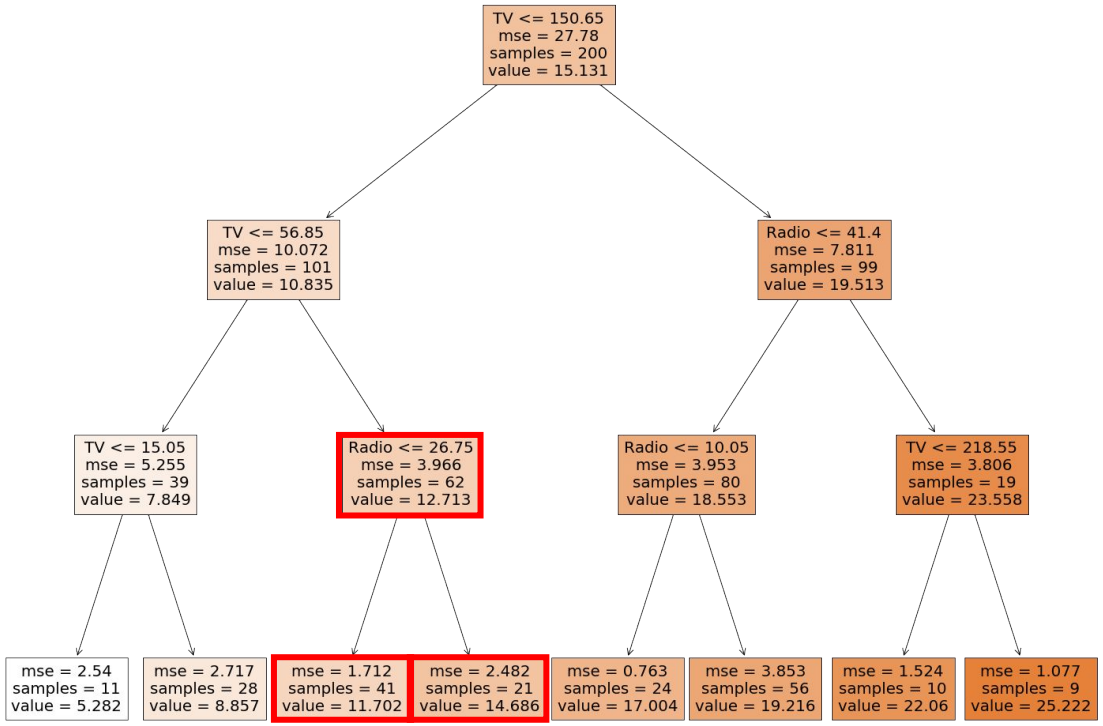
Next split, RSS: 5.255



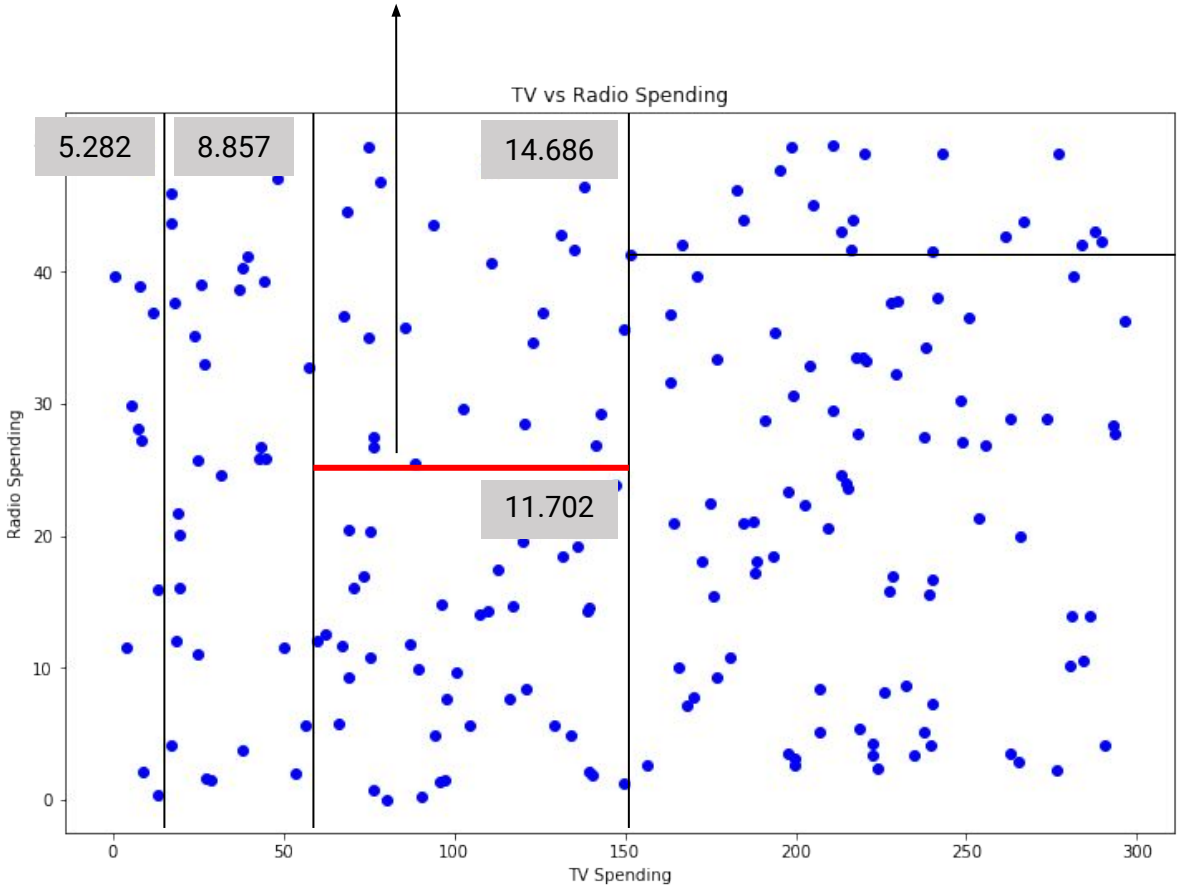
MECHANISM BEHIND MODEL: PARTITIONING

- **Step 3**: Repeat **step 2** to continue splitting until the splits no longer generate improvement in the RSS.

$$\text{RSS} = \sum_{i=1}^{n}(y_i - f(x_i))^2$$



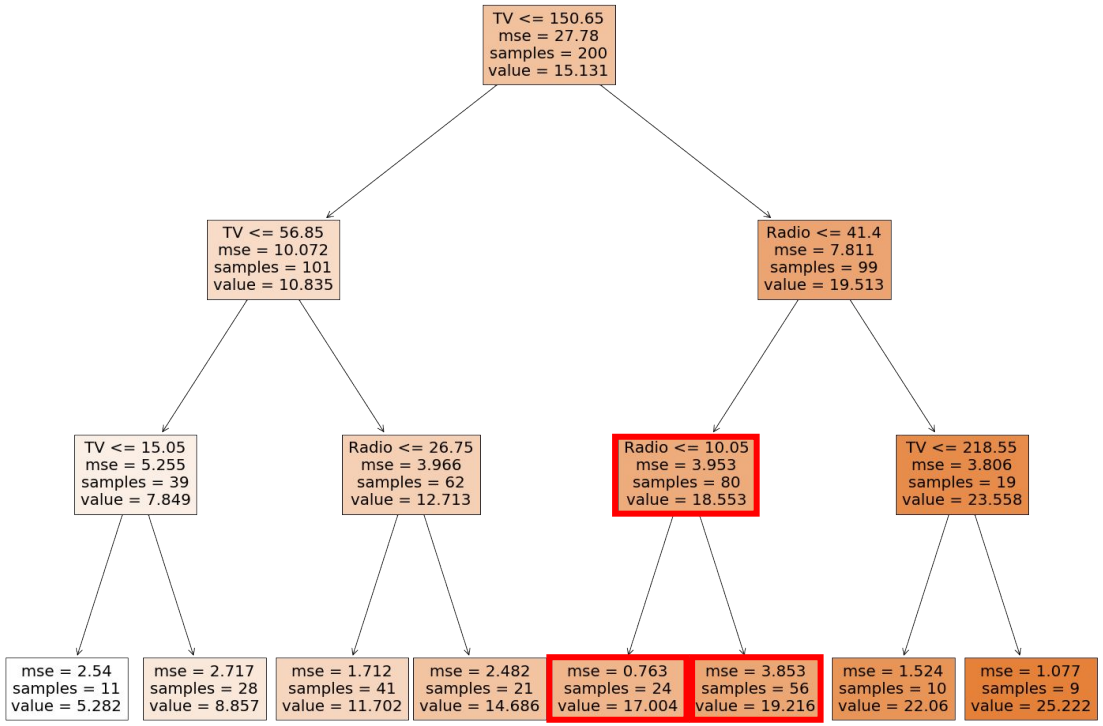
Next split, RSS: 3.966



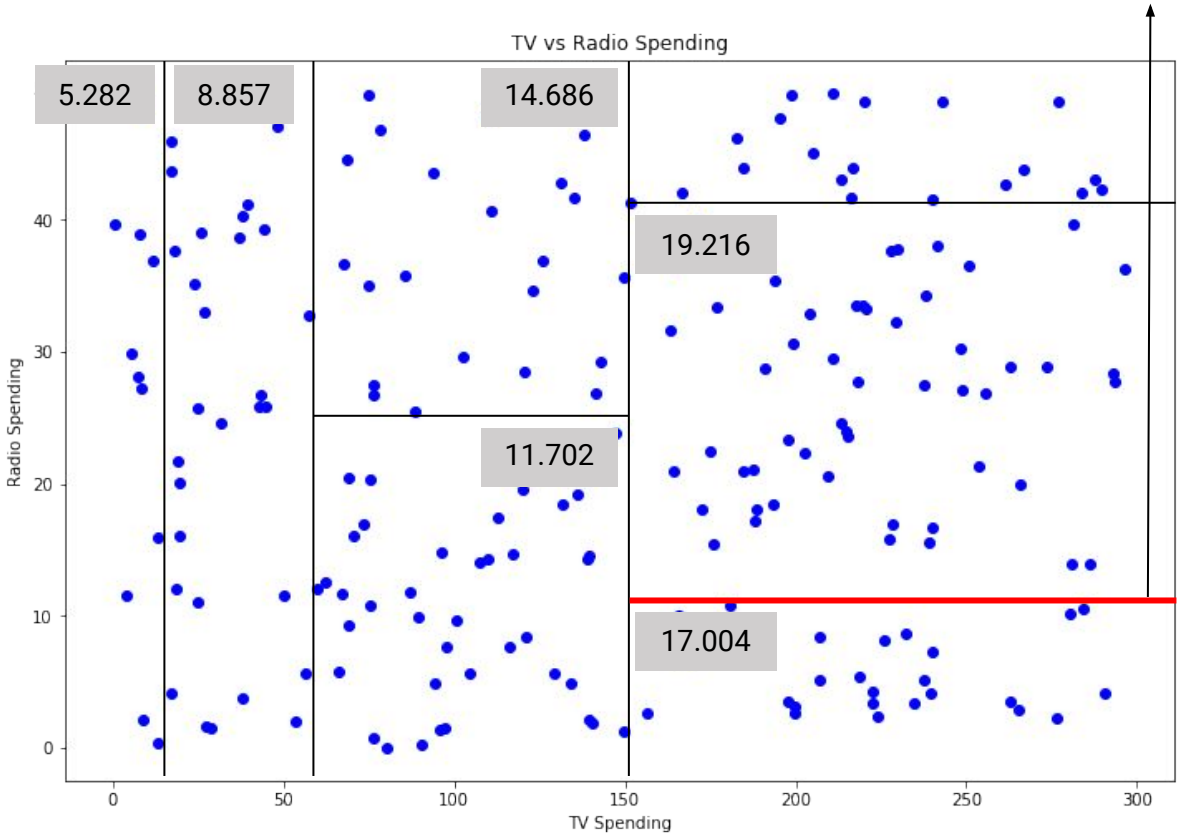
MECHANISM BEHIND MODEL: PARTITIONING

- **Step 3**: Repeat **step 2** to continue splitting until the splits no longer generate improvement in the RSS.

$$\text{RSS} = \sum_{i=1}^{n}(y_i - f(x_i))^2$$

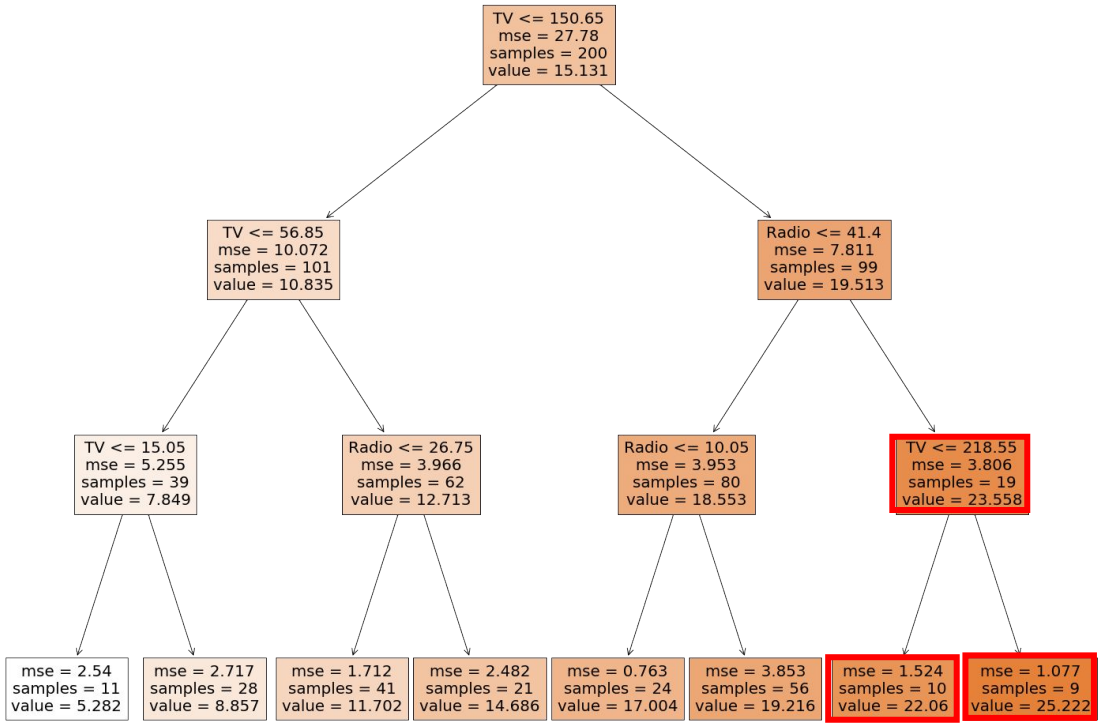


Next split, RSS: 3.953

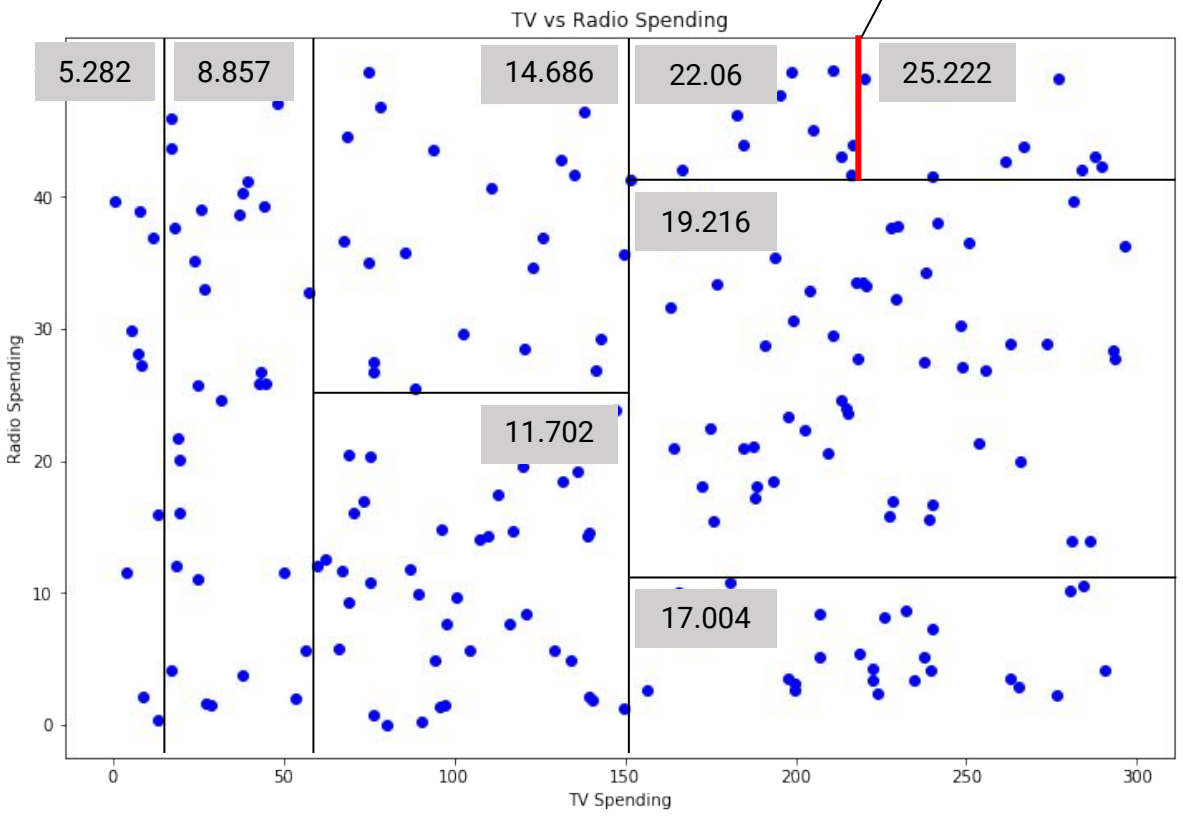


MECHANISM BEHIND MODEL: PARTITIONING

- **Step 3**: Repeat **step 2** to continue splitting until the splits no longer generates improvement in the RSS.
- **Step 4**: From here on, the algorithm deems that any further splits would not result in any significant improvements in RSS, and hence it terminates the process for further splits

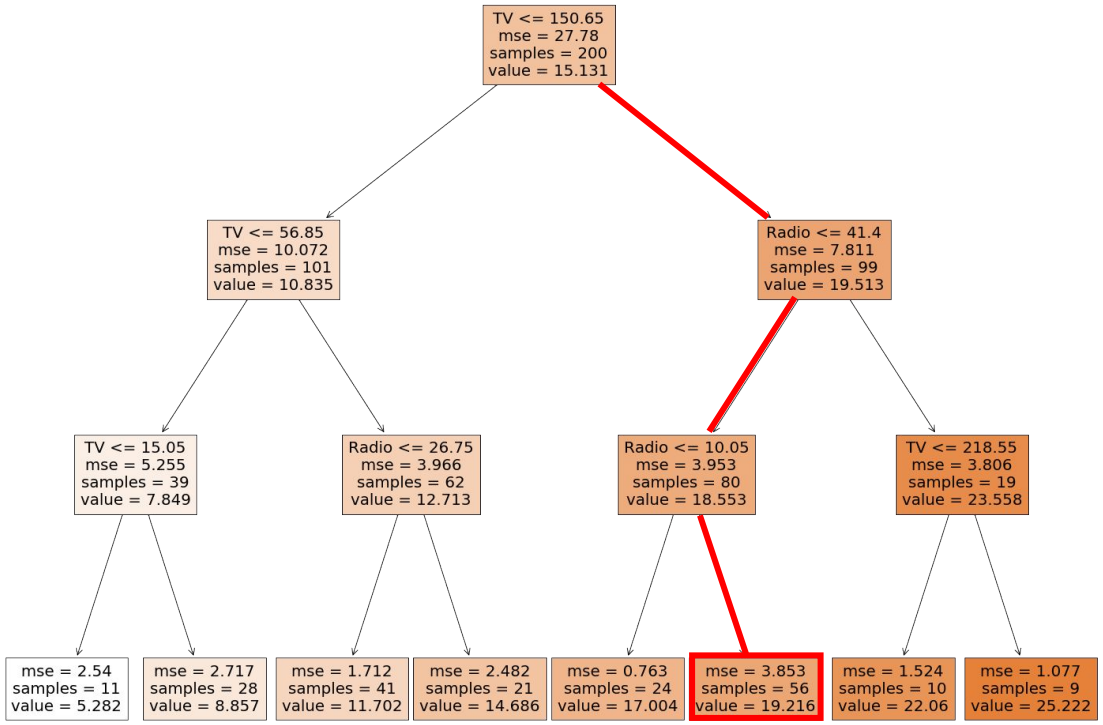


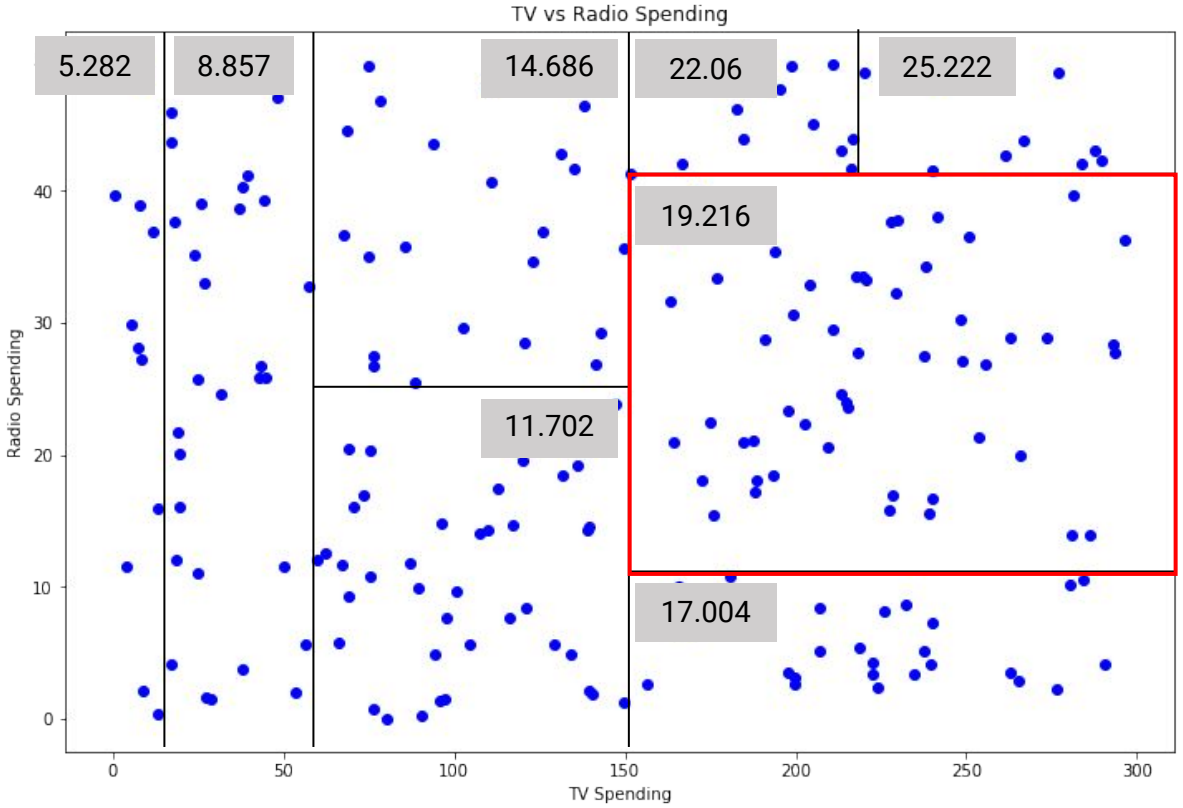
Last split, RSS: 3.806



MECHANISM BEHIND MODEL: PARTITIONING

- Let's say we want to use the newly constructed tree to do some prediction:
  - **TV = 200, Radio = 40**
  - **Sales = 19.216**





DECISION TREE

PRUNING DECISION TREES TO IMPROVE PERFORMANCE

DECISION TREE PRUNING: IMPORTANCE OF PRUNING

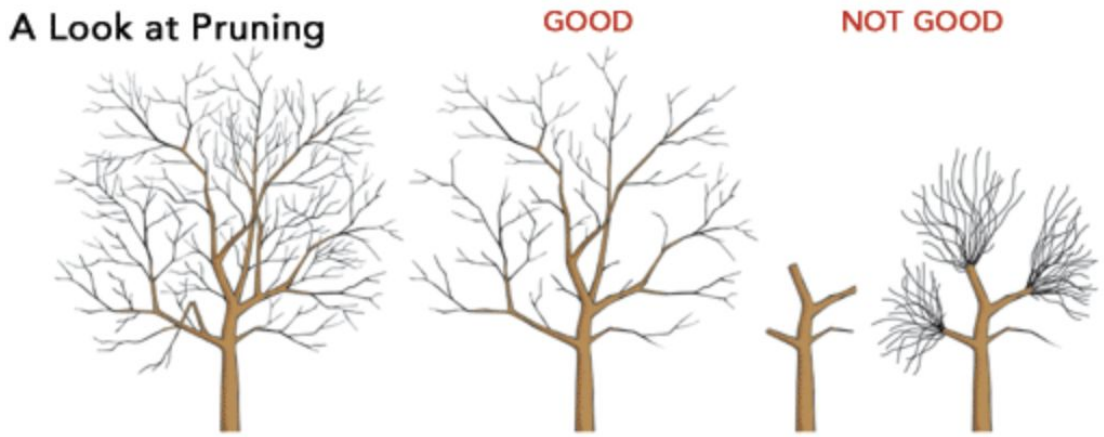
- Just like it is easy to plant a tree, it is very easy to build a decision tree model with Sklearn. It only takes a few lines of code!
- But this is hardly the end of the story for both tree planting (or building a decision tree model)
- For a tree to look beautiful, **it needs to be pruned**. In the same vein, we need to prune our decision tree as well

Trees needs to be pruned, else they are no different from weeds

DECISION TREE PRUNING: IMPORTANCE OF PRUNING

- If you have a tree that is not well-fitted, you will be faced with issues:
  - A tree that is *too large* risks **over-fitting the training data** and poorly generalizing to new samples.
  - A *small tree* might not capture important structural information from the training dataset about the sample space.

- There are several parameters that we have to determine when generating a decision tree, including:
  - **Depth:** How many layers should our tree have?
  - **Number of Leafs:** How many terminal nodes should our tree have?



DECISION TREE PRUNING: PRUNING IN SKLEARN

- In sklearn, we can make use of the following parameters to “prune” the tree (to ensure it does not overfit or underfit):
  - Maximum depth of tree
  - Maximum number of leaf/terminal nodes
  - Minimum samples (or data points) in leaf/terminal nodes
- The following code snippet illustrates how easy it is to perform decision tree pruning with Sklearn (more on this later when we go through the in-class Jupyter notebook):

```
# Create a decision tree regressor object of depth=8
regressor = DecisionTreeRegressor(random_state=1234, max_depth=8, min_samples_split=50)
```

- There are several systematic techniques for pruning decisions trees, such as Cost Complexity Pruning aka **Weakest Link Pruning**, which you can learn more about from the video below:
  - https://www.youtube.com/watch?v=D0efHEJsfHo
- In Lesson 5 on Ensemble Learning, we cover more advanced applications of decision trees, which removes the need for pruning individual decision trees to make powerful predictions.

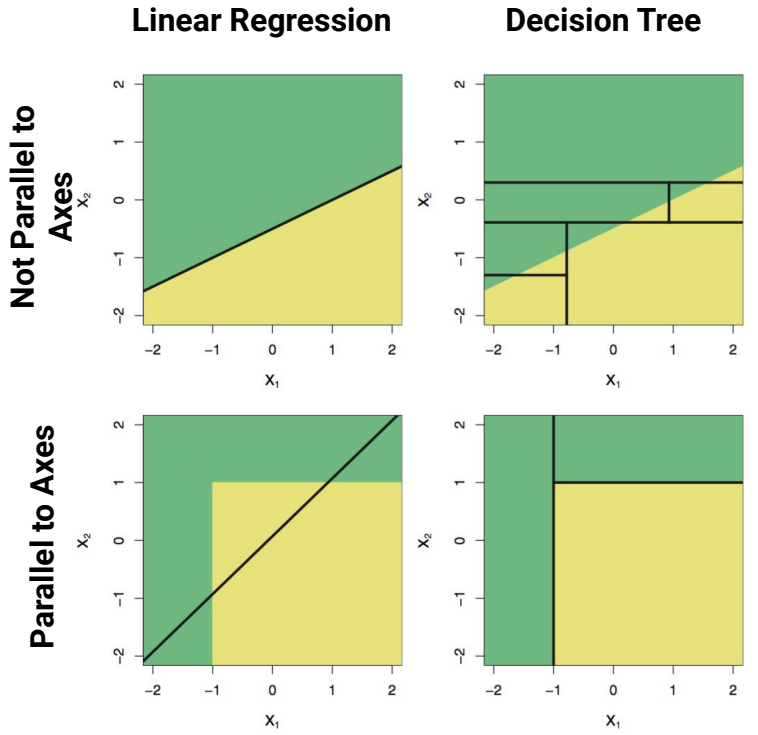
DECISION TREE

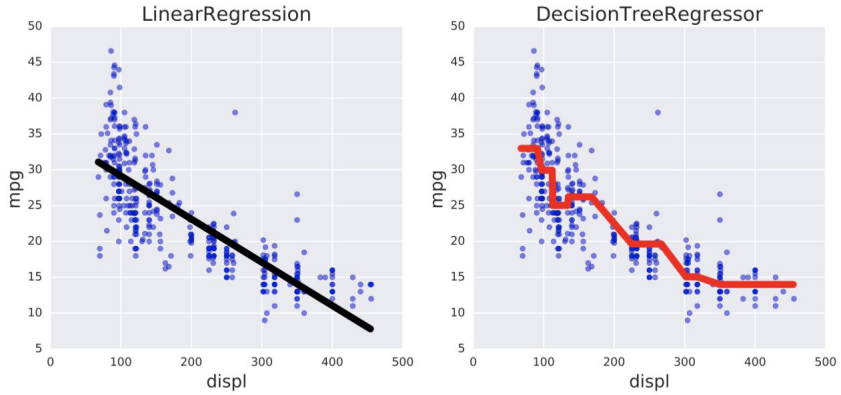
ADDITIONAL CONSIDERATIONS

ADDITIONAL CONSIDERATIONS

- **Strengths**: Decision Trees are
  - Highly interpretable
  - Require minimal parameter tuning
  - Works on discrete & continuous variables out of the box

- **Weakness**: However they are
  - Prone to overfitting
  - Tends to be weaker when the relationship between the features and outcome are linear but not parallel to the axes





ADDITIONAL CONSIDERATIONS

- We seldom use the vanilla decision tree. To improve on decision trees, we can use models / techniques such as:
  - Bagging & Boosting techniques
  - Random Forest model
  - XGBoost model
  - CatBoost model

- *In fact, one of the most cutting-edge and powerful models today is **a variant of the Decision Tree**. Since its introduction, this algorithm has not only been credited with winning numerous Kaggle competitions but also for being the driving force under the hood for several cutting-edge industry applications!*

- We will cover this model in detail, along with above techniques & models, in Lesson 5 on Ensemble Learning.

- Lastly, take note that decision trees can also be used for classification problems, which we will cover in the next class (Lesson 4).

Photo by Jared Subia on Unsplash

XGBoost Algorithm: Long May She Reign!

The new queen of Machine Learning algorithms taking over the world...