CoGrammar

Decision Trees (Part One)

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

Data Science Lecture Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(FBV: Mutual Respect.)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Open Classes. You can submit these questions here: **Open Class Questions**

Data Science Lecture Housekeeping cont.

- For all **non-academic questions**, please submit a query: **www.hyperiondev.com/support**
- Report a **safeguarding** incident: **www.hyperiondev.com/safeguardreporting**
- We would love your **feedback** on lectures: **Feedback on Lectures**

Lecture Objectives

- **Learn about tree-based methods for regression and classification.**
- **Tree-based methods to solve problems using a flow chart.**

Intro to Decision Trees

- Decision tree work by formulating simple rules that partition data into even smaller regions. Each partition can be thought of as a fork in the road, where a decision will be made.

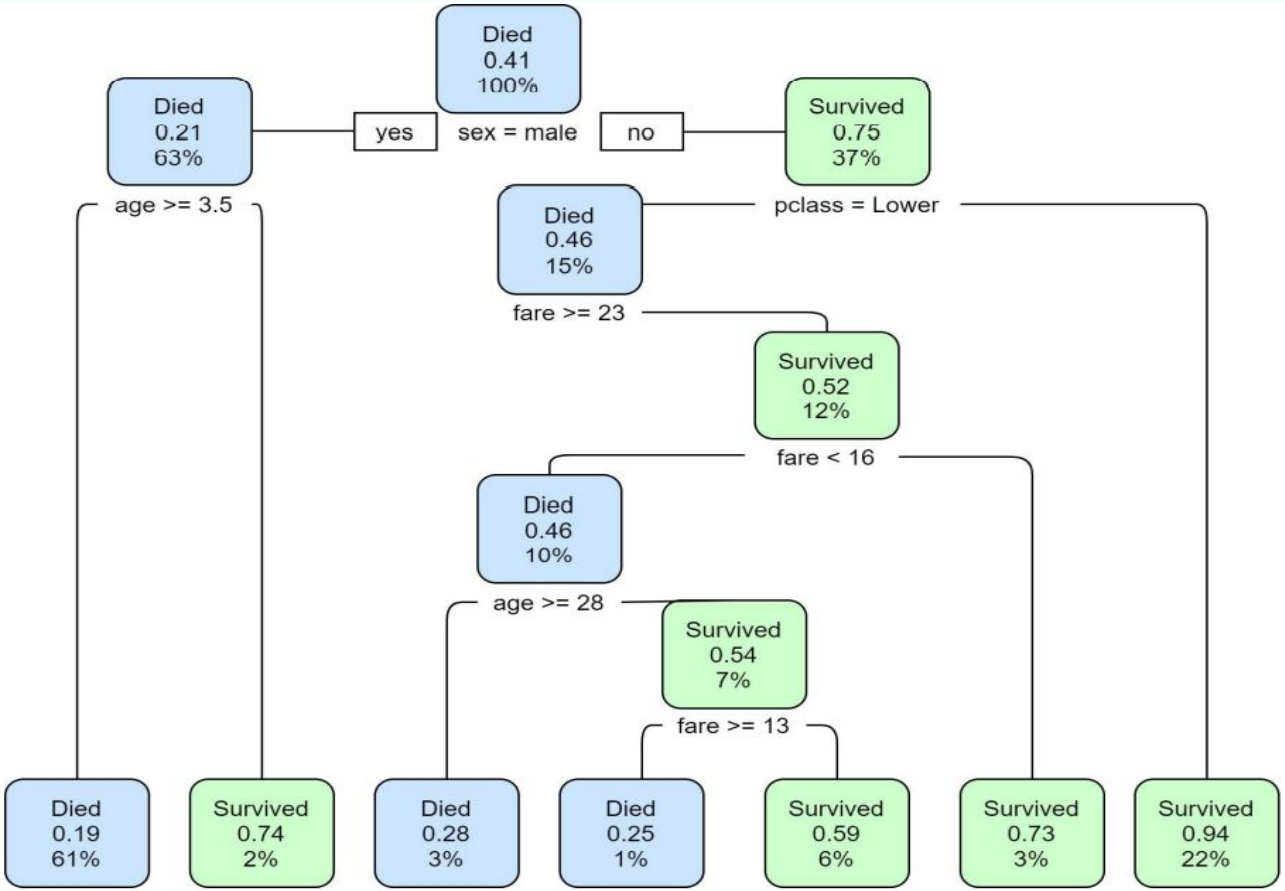
- The decision is made based on rules which are derived from learning experiences. Decision trees are among the most interpretable machine learning techniques based on how they resemble how humans make decisions.

- For example, to make dinner, one would first think if there are enough ingredients in the fridge to make a meal. If there aren't enough ingredients, we'll need to consider other options. Based on the time of day, you may decide to go to the grocery store or just decide on take-out instead.

Classification Trees

- Decision trees created for datasets with a categorical dependent variable are called classification trees. As an example, let's look at the Titanic dataset. A tree model of this dataset shows us the likelihood of different kinds of passengers surviving the sinking of the Titanic.

- The tree consists of nodes, and each node has a rule that determines whether an instance moves on to the left or right child node. At the end of each possible path is a leaf. In a classification tree, a leaf contains the predicted label for an instance with those input features.


Died
0.41
100%
sex = male
yes
no
Died
0.21
63%
Survived
0.75
37%
age >= 3.5
Died
0.46
15%
pclass = Lower
fare >= 23
Survived
0.52
12%
fare < 16
Died
0.46
10%
age >= 28
Survived
0.54
7%
fare >= 13
Died
0.19
61%
Survived
0.74
2%
Died
0.28
3%
Died
0.25
1%
Survived
0.59
6%
Survived
0.73
3%
Survived
0.94
22%

Classification Trees

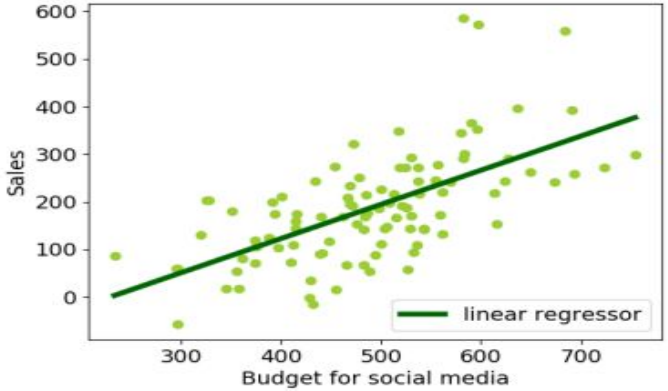
- This particular tree also shows at each node what the probability of survival is. Without any prior knowledge, a passenger's chance of survival was 41%. But if we know that an instance is male, survival is a lot less likely. This tree shows that the lowest chance of survival was for adult males.

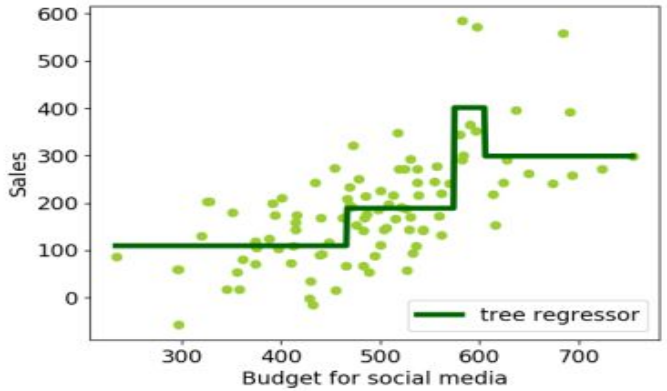
Regression Trees

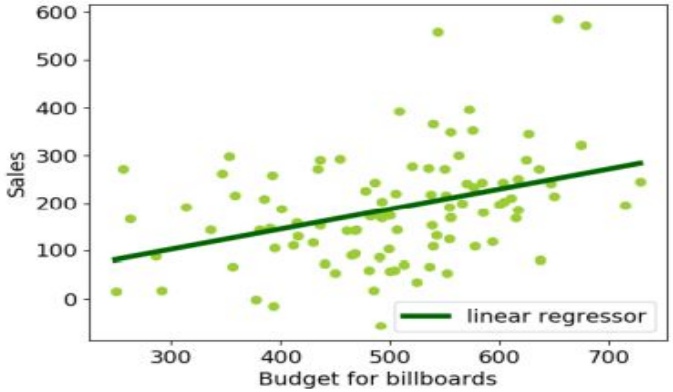
- ★ Decision trees can also be used for problems with a numerical dependent variable. A regression decision tree divides input features into regions and assigns categories to those regions.

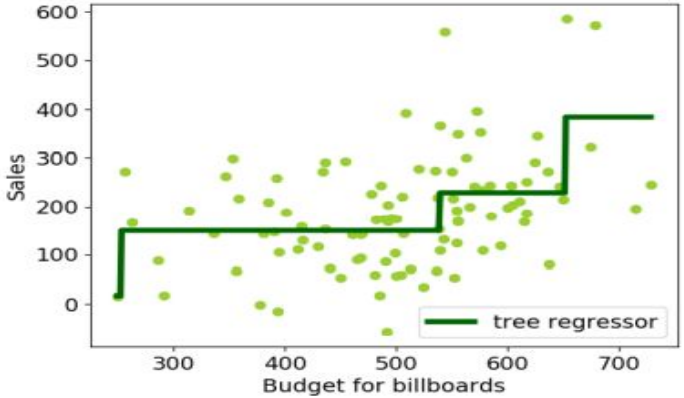
- ★ The regions and their labels are based on what best fits the data. Again, best fit here means the set of regions that minimises the distance between the predictions of the model and the observed values.

- ★ So while a linear regression approach to our advertising problem in the previous tasks gave different predictions for each unique value of x, a regression tree gives different predictions for regions of x.


Sales
Budget for social media
linear regressor


Sales
Budget for social media
tree regressor


Sales
Budget for billboards
linear regressor


Sales
Budget for billboards
tree regressor

Regression Trees

- You can imagine that this approach is more flexible. The regions could capture a steeper or less steep increase in sales if that fits the data better than a linear model. The regions can also be arbitrarily specific or broad. Trees can also be visualised easily to get information about the decisions the model makes, in case we want to change the parameters of the model.

Overfitting & Underfitting

When discussing trees it needs to be said that due to their flexibility, they are prone to something called “overfitting”. Overfitting is one of the biggest causes for poor performance of machine learning algorithms, together with its counterpart, underfitting.

★ Underfitting refers to a model that can neither model the training data nor generalise to new data. The training error is high because the model was not able to make good predictions based on the input features. A model may fail to fit the data because it is too simple to capture the patterns, but underfitting is more commonly caused by a problem with the task set-up (e.g. the features x are not a good predictor of y) or with the training data (e.g. there is too little training data to learn from, or it contains too many mistakes).

Overfitting & Underfitting

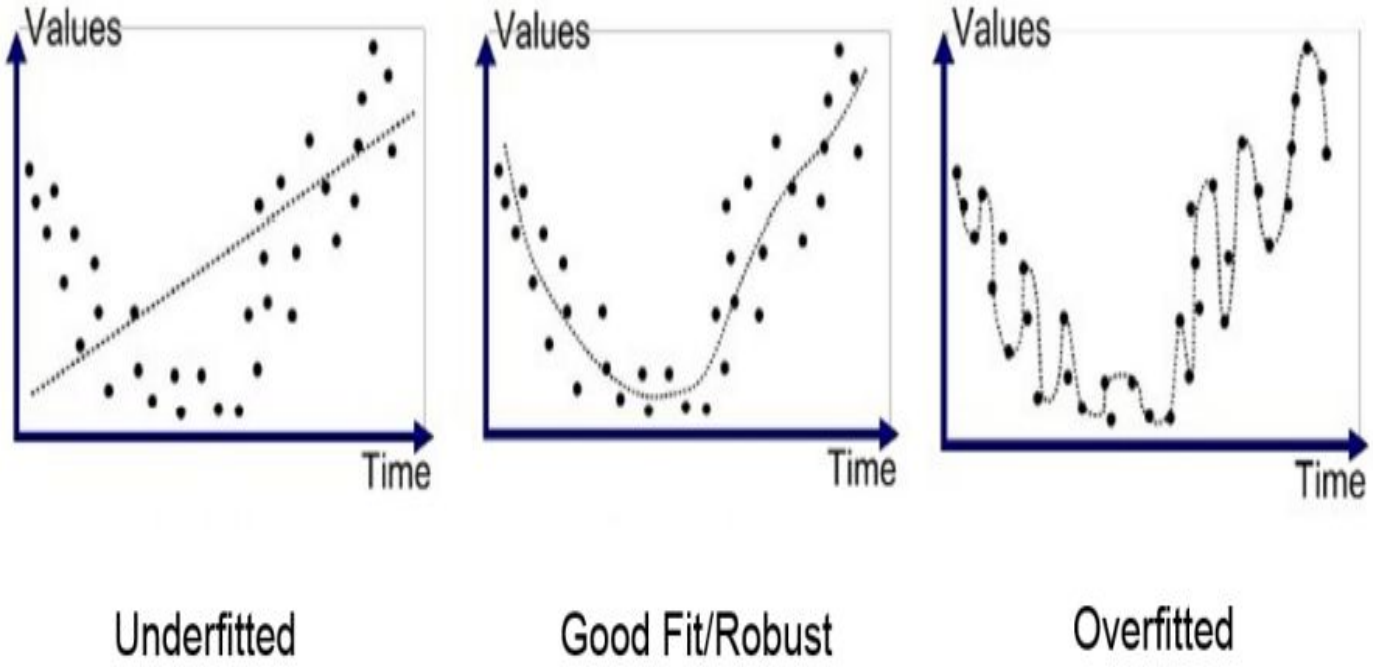
- **An underfitting model has low training error, but will not perform well on test data. However, that does not mean that a model with low training error is automatically going to do well on test data. A model with very low training error may suffer from overfitting: some of the rules the model learned are only applicable to training data and are not useful for solving the overall problem we want to solve. The model is too tailored and does not generalise well. It is important that the model is able to generalise beyond the training data, as this data is only a sample. If the model is overfitted, it will perform very well on training data, but it won't translate to good performance on test data.**

Diagnosing Over & Underfitting

- **What are the causes of overfitting and underfitting? There are many reasons, and it can be typically difficult to determine. Generally speaking, an overly-complex model will overfit, and an overly-simple model will underfit. Think of overfitting as simply “remembering” the training data piece by piece, without learning how to construct a rule.**

Let’s Breathe!

Overfitting & Underfitting



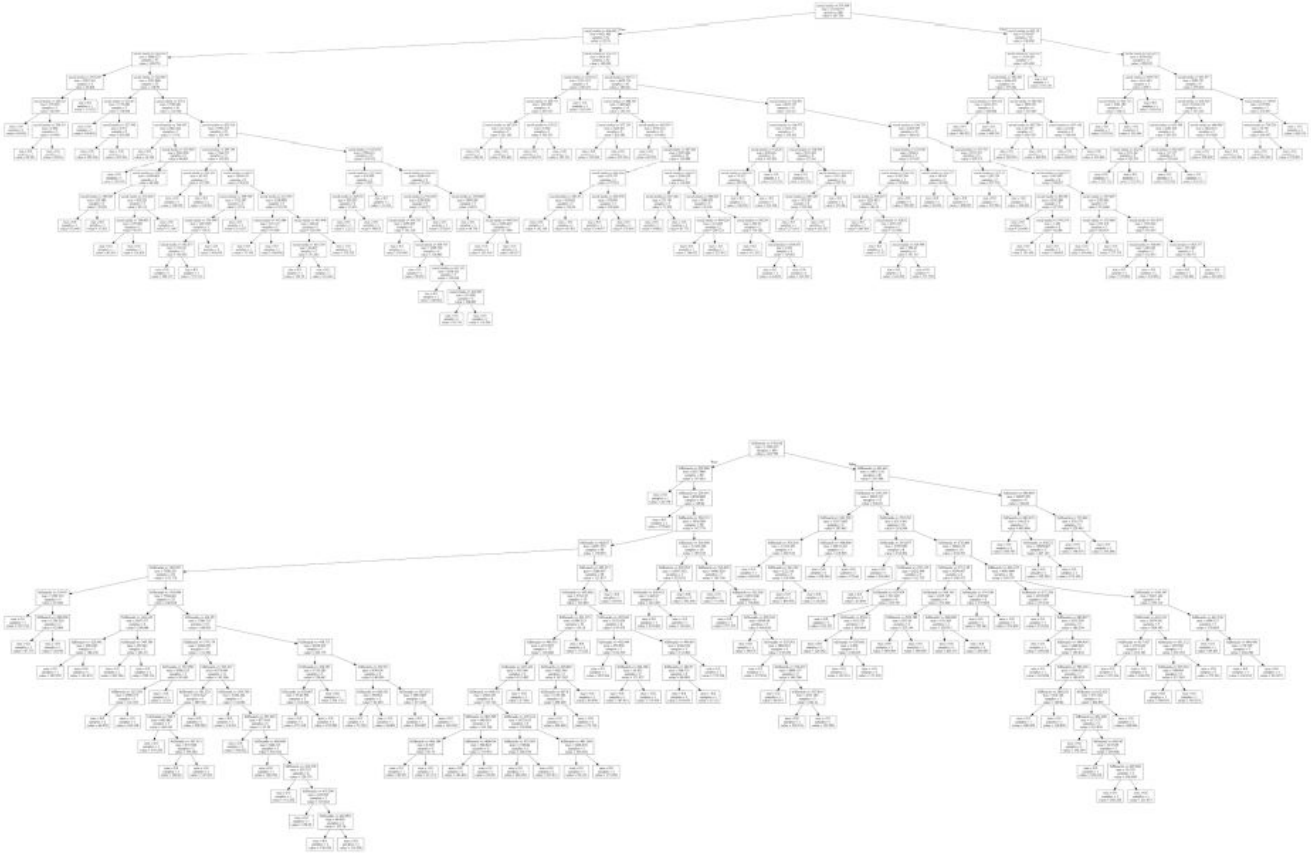
Diagnosing Over & Underfitting

- The first graph on the left is an incredibly simple model: it's just a straight line. There is no way that a straight line will be able to predict that data: hence you will see a high training error and a high testing error.
- The middle graph is probably the best fit for the data. I'm sure that you can picture more samples of this data being added, and the model being able to give a rough estimate of what that should be. Note that the training error will be lower than an underfit, but higher than an overfit model. However, the testing error will be lower than both - testing error is what is most important. Let this be a lesson: zero error is rarely a good thing, and lower training error doesn't always mean a better model.
- The last graph on the right shows a model that has overfit. The training error here will be incredibly low - practically zero! That being said, if more data were added, it's likely that the model will make a valuable prediction, as it doesn't seem to follow the overall trend of the data.

Pruning

- As mentioned before, decision trees are very flexible and, therefore, are likely to overfit training data. This problem can be addressed by “pruning” a tree after training in order to remove some of the detail it has picked up and make the model more abstract and more general.

- A strategy to prevent overfitting is thus to grow a very large tree without any restrictions first, which is then pruned to retain a more general subtree. If those trees were left to develop without restrictions, they would have become this detailed:



Development Data

- How do we determine the best way to prune a tree? We can take subtrees of an unpruned tree and compute the test error of predictions of that subtree. We can then select the subtree with the lowest test error rate. However, once we have done that we have fitted our pruning parameter to the test data.
- This means the test data is no longer completely unseen. Fitting to test data can be avoided by splitting the dataset into three parts instead of two before training: a training set, development set (‘dev set’), and test set.
- The development set is used to see whether the model seems to be generalising well to data that is not in the training set. This makes it possible to spot and try to remedy under- or overfitting. Only at the end do we test the model on totally unseen data. Another term for this third type of held-out data is the validation set.

CoGrammar

Q & A SECTION

Please use this time to ask any questions relating to the topic, should you have any.

CoGrammar

Thank you for joining us

1. Take regular breaks
2. Stay hydrated
3. Avoid prolonged screen time
4. Practise good posture
5. Get regular exercise

"With great power comes great responsibility"