## Introduction

Congrats on starting your data science journey! Machine learning is a powerful tool, and this tutorial will help you start using it, as well as introduce you to some important concepts.

### What will I learn?

By time you finish this tutorial, you will:

* Understand what a machine learning model is
* Build a machine learning model to predict house prices
* Learn how to evaluate and improve your model

### What do I need to know before I get started?

This tutorial will assume that you're familiar with some very basic R concepts, like functions and arguments. If you're brand new to programming, I recommend you work through this tutorial series (<https://www.kaggle.com/rtatman/getting-started-in-r-first-steps/> ), which doesn't assume any background, before getting started.

### What will I need to do?

As you complete this tutorial, there will be a number of exercises for you to complete. The exercises are in a separate notebook(https://www.kaggle.com/rtatman/welcome-to-data-science-in-r-workbook/), so you can keep all your work in one place.

***Ready? Let's get started! :D***

## Table of Contents

* How models work
* Starting your machine learning project
* Running your first model
* How do we know if our model is good
* Underfitting/overfitting and improving your model
* A different type of model: Random forests

## How models work

We make a lot of predictions in our everyday life. For example, I might predict that my phone will run out of battery during the day if I don't charge it the night before. Or I might predict that it will rain if the sky is overcast and I hear thunder. What prediction we make in a specific situation depends on our past experiences and conditions at the time we make our prediction. Machine learning models work a lot like people do: they build on examples they've been seen previously to predict what the outcome will be given a specific set of conditions.

For this lesson, we'll be looking at something a little more high-stakes than whether your phone will run out of battery before you get home: **can you figure out how much a house will sell for?**

We're going to be focusing on a specific type of machine learning model called decision trees. There are many different kinds of machine learning models, each with its own strengths and weaknesses, but decision trees are a good choice to start with because they're flexible, easy for a human to understand and form the basic building block for some very powerful models.

Let's start out with the simplest possible decision tree.

Uma imagem contendo texto

Descrição gerada automaticamente

It divides houses into only two categories. You predict the price of a new house by finding out which category it's in, and the prediction is the historical average price from that category.

This captures the relationship between house size and price. We use data to decide how to break the houses into two groups, and then again to determine the predicted price in each group. This step of capturing patterns from data is called **fitting** or **training** the model. The data used to **fit** the model is called the **training data**. After the model has been fit, you can apply it to new data **to predict** prices of additional homes.

**How does the tree know what splits to make?** There are several different approaches to building decision trees. The most common is to pick the feature and value that will split our data into sub-groups that are as homogenous as possible. So if you're trying to predict the weight of a set of birds made up of ostriches and hummingbirds, it makes sense to split the dataset based on the species of the birds since the weights of the ostriches are closer to each other than to the weights of the hummingbirds.

Assuming your decision tree works in a sensible way, which of the two trees shown here do you think you might get from **fitting** this especially simple decision tree?

Uma imagem contendo captura de tela

Descrição gerada automaticamente

The decision tree on the left (Decision Tree 1) probably makes more sense, because it captures the reality that houses with more bedrooms tend to sell at higher prices than houses with fewer bedrooms. The biggest shortcoming of this model is that it doesn't capture most factors affecting home price, like number of bathrooms, lot size, location, etc.

You can capture more factors using a tree that has more "splits." These are called "deeper" trees. A decision tree that also considers the total size of each house's lot might look like this:

Uma imagem contendo mapa, texto

Descrição gerada automaticamente

You predict the price of any house by tracing through the decision tree, always picking the path corresponding to that house's characteristics. The predicted price for the house is at the bottom of the tree. The point at the bottom where we make a prediction is called a leaf.

The splits and values at the leaves will be determined by the data, so it's time for you to check out the data you will be working with.

## Starting your machine learning project

You will build a simple model and then continually improve it as you go through the Machine Learning Track. It is easiest to keep one browser tab (or window) for the tutorials you are reading, and a separate browser window with the code you are writing. You will continue writing code in the same place even as you progress through the sequence of tutorials.

The starting point for your project is here (<https://www.kaggle.com/rtatman/welcome-to-data-science-in-r-workbook/> ). Open that link in a new tab. Then hit the "Fork Notebook" button towards the top of the screen.

Uma imagem contendo captura de tela

Descrição gerada automaticamente

## Working in Kaggle Notebooks

You will be coding in a "notebook" environment. These allow you to easily see your code and its output in one place. A couple tips on the Kaggle notebook environment:

1. It is composed of "cells." You will write code in the cells. Add a new cell by clicking on a cell, and then using the buttons in that look like this. Uma imagem contendo objeto, kit de primeiros socorros

   Descrição gerada automaticamente

The arrows indicate whether the new cell goes above or below your current location.

2) Execute the code in the current cell with the keyboard shortcut Control-Enter.

## Your Data

You will see examples predicting home prices using data from Melbourne, Australia. You will then write code to build a model predicting prices in the US state of Iowa. The Iowa data is pre-loaded in your workbook.

From the notebook tab where you are writing code, pull up descriptions of the **data fields** for your data by clicking on Input Files on the top left of the notebook:

Uma imagem contendo captura de tela

Descrição gerada automaticamente

The left sidebar shows your data is broken into three files. We will use a file called **train.csv**. But, don't worry about this for now.

Instead, scroll down to see a list of the types of information available in this dataset.

Once you've looked at the contents of your data, return to the coding interface by clicking on the \*Input Files\* link again, which now has a minus sign next to it.

## Read our data into R

The first thing you'll want to do is familiarize yourself with the data. You'll use the *Tidyverse* library for this. To use the library, we're going to have to load it into R.

Tidyverse: A collection of R packages built around the central idea that data should be formatted with each variable as a column, each observation as a row, and each type of observational unit as a separate table. You can find more information (<https://www.tidyverse.org/packages/>).

# load in packages we'll use

library(tidyverse) # utility functions

library(rpart) # for regression trees

library(randomForest) # for random forests

# read the data and store data in DataFrame titled melbourne\_data

melbourne\_data <- read\_csv("../input/melb\_data.csv")