Intro to Machine Learning

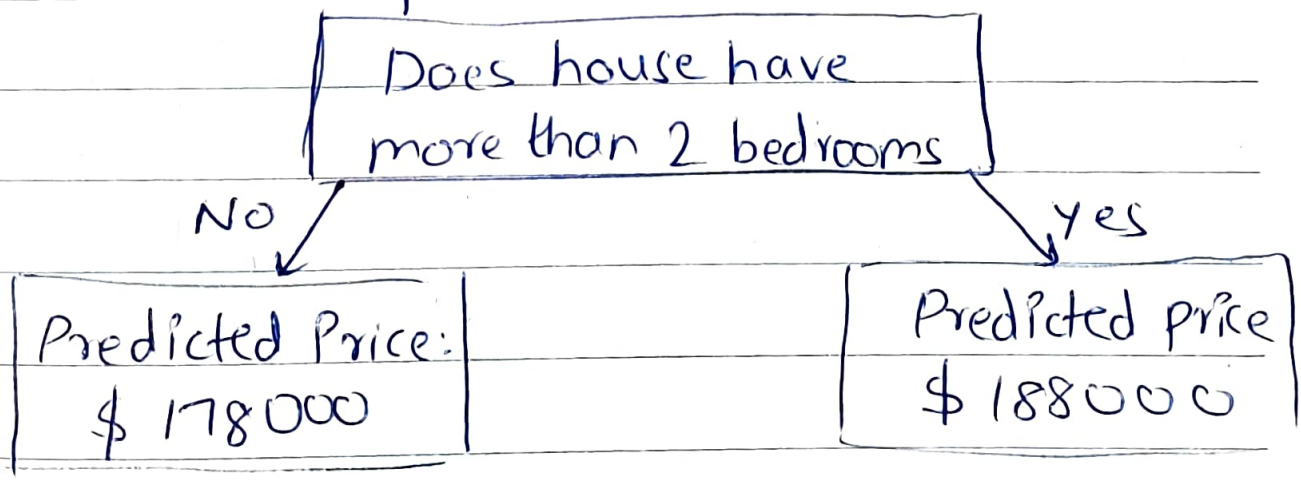
* **Machine Learning**: Training a piece of software, called a model, to make useful predictions or generate content from data

- How Machine Learning Models work?

→ **Decision Tree**: They are the basic building blocks for some of the best models in Data Science. They are easy to understand.

Eg:-

Sample Decision Tree



We use data to decide how to break houses into 2 groups & 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.

Decision Tree can be expanded which has more "splits". These are called as "deeper" trees.
The point at the bottom where we make a prediction is called a 'leaf'.
The splits & values at the leaves will be determined by the data.

* Using Pandas to get familiar with Data

We'll use Panda's library to familiarize yourself with data in machine learning project.

→ Pandas is the primary tool data scientists use for exploring & manipulating data. (Abbreviated as pd)

```
# import pandas as pd
```

→ Imp part of Pandas library is Data Frame. DataFrame holds type of data you might think of as a table, similar to sheet in Excel; or table in SQL database.

* Loading Data:

```
import pandas as pd
file_path = '____.csv'          # Path of file to read
home_data = pd.read_csv(file_path)
~~step1.check()~~               # call line below with no orgument to check that you have loaded data correctly
```

* Review Data:

```
home_data.describe()            # shows data in forms of rows & columns
```

- Selecting Data for Modeling:

If your dataset has too many variables it is difficult to understand it. Hence to choose variables/columns, we'll need to see a list of all columns in dataset. This is done with columns property of DataFrame.

Eg: readed_file.columns

File name

→ Ways to select a subset of your Data:

1. Dot Notation, which we use to select 'prediction target'
2. Selecting with a column list, which we use to select the "features".

① Dot Notation:

You can pull out a variable with dot-notation. This single column is stored in Series, which is like a DataFrame with only single column of data

By convection, prediction target is called y.

Eg:- y = file_name . price

↓ File name

→ required column

② Choosing Features:

The columns that are inputted into our model (& later used to make predictions) are called "features".

By convection this data is called 'X'

③ Building Model:

→ We'll use Scikit-learn library to create models. Also called sklearn

→ Steps to build & using a model:

- Define
- Fit
- Predict
- Evaluate.

Eg:- from sklearn.tree import DecisionTreeRegressor

```
me_model = DecisionTreeRegressor(ran_state=1)
# Define model
me_model.fit(X, y)
# Fit model
```

- Model Validation:

→ To measure quality of your model. Measuring model quality is the key to improving your models.

→ The relevant measure of model quality is predictive accuracy. In other words, will the model's predictions be close to what actually happens.

We need to summarize model quality into an understandable way. ie into a single metric. There are many metrics for summarizing model quality. we'll start with Mean Absolute Error (MAE)

Error = Actual — Predicted.

with MAE metric, we take absolute value for each error. this converts each error to a positive number. Then we take avg of those absolute errors.

dropna() method removes the rows that contains null values.

first define a model then,

```
from sklearn.metrices import mean_absoluk_error
predicted_val = me_model.predict(X)
mean_absoluk_error(y, predicted_val)
```

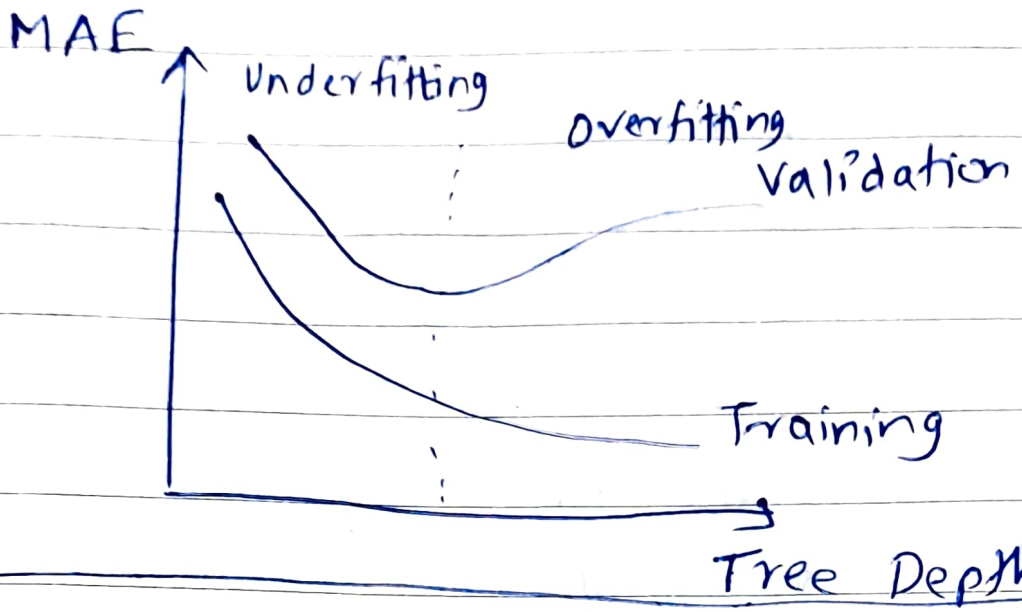
- The measure we just computed can be called an "in-sample" score. We used a single sample for both building model & evaluting it. This is bad since model's practical value come from making predictions on new data, we measure performance on data that wasn't used to build the model. The way to do this is to exclude some data from

model building process & then use those to test model's accuracy. This data is called validation data.

The scikit-learn library has a fxn train_test_split to break up data into 2 pieces. We'll use some data as training data & other data as validation data to cal mean_absolute_error.

- Overfitting:- It is a phenomenon where model matches the training data almost perfectly, but does poorly in validation & other new data.

- Underfitting: When a model fails to capture imp distinctions & patterns in the data so it performs poorly even in training data, that is called underfitting.



Since we care on accuracy of new data, we want to find the sweet spot b/w underfitting & overfitting. Visually we want low point of validation curve in fig above.

get_mae() is a fxn for calculating MAE.

We can use a for loop to compare the accuracy of diff model values for diff leafs.

Random Forests

(Using a more sophisticated machine learning algorithm)

→ Uses many trees, & makes a prediction by averaging the predictions of each component tree.

→ Better predictive accuracy than a single decision tree & works well with default parameters.

We can build a model using the class 'RandomForestRegressor'

```
# from sklearn.ensemble import RandomForestRegressor
```