Kaggle Machine Learning: https://www.kaggle.com/dansbecker/how-models-work

- How learning models work
- How they are used
- We'll start with a model called the Decision Tree. There are fancier models that give more accurate predictions. But decision trees are easy to understand, and they are the basic building block for some of the best models in data science.
- Fitting or training the model is step capture patterns from data
- 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.

Using Pandas to get familiar with your data

- Import data : read_csv
- Look at data from pandas data frame attributes: describe(), head(), tail()
- Watch out for missing value

Selecting Data for Modeling

- Start by picking a few variables using our intuition
- Statistical techniques to automatically prioritize variables
- Use columns attribute to look at columns of DataFrame
- Simplify by using dropna(axis=0) to drop column with missing values
- There are several approaches for selecting a subset of data using Panda
    - Focus on two ways:
        - Dot notation, which we use to select the "prediction target"
        - Selecting with a column list, which we use to select the "features"

Selecting prediction target

- Dot notation, single column stored in Series; similar to DataFrame with only single column.
- Use dot notation to select column we want to predict

Choosing Features

- The columns that are inputted into our model (and later used to make predictions) are called "features."
- Sometimes, you will use all columns except the target as features. Other times you'll be better off with fewer features.
- We select multiple features by providing a list of column names inside brackets. Each item in that list should be a string (with quotes).

Building your model

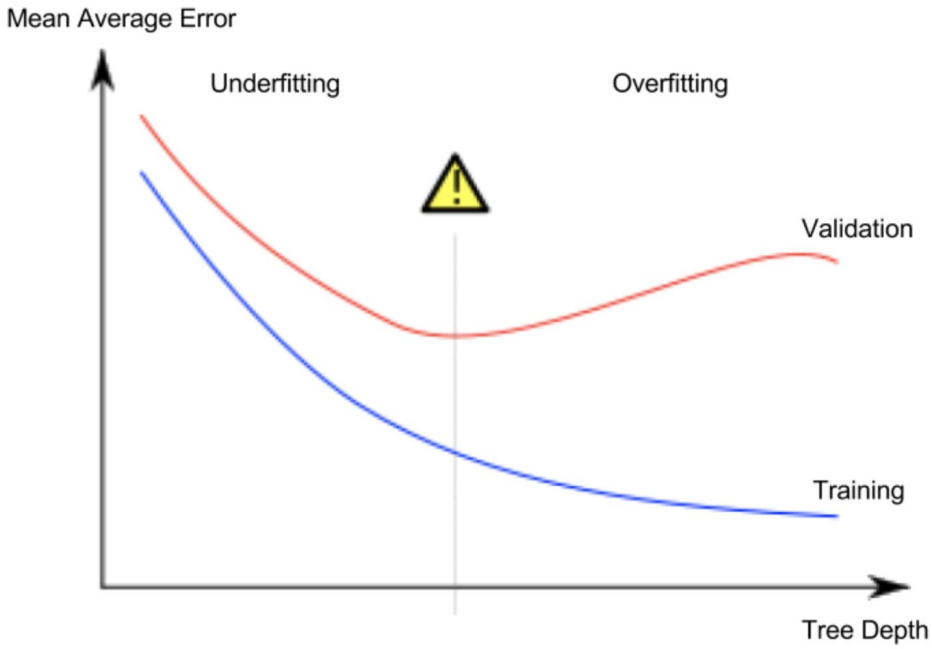
- Scikit-learn is easily the most popular library for modeling the types of data typically stored in DataFrames.
- The steps to building and using a model are:
    - **Define:** What type of model will it be? A decision tree? Some other type of model? Some other parameters of the model type are specified too.
    - **Fit:** Capture patterns from provided data. This is the heart of modeling.
    - **Predict:** Just what it sounds like
    - **Evaluate**: Determine how accurate the model's predictions are.
- Many machine learning models allow some randomness in model training. Specifying a number for random_state ensures you get the same results in each run.
- `DecisionTreeRegressor is one of the algorithms provided by the sklearn`

Model Validation

- the relevant measure of model quality is predictive accuracy. In other words, will the model's predictions be close to what actually happens.
- Don’t make validation against training data
- How ?
    - Summarize predicted value to a single metric
    - There are many metrics for summarizing model quality, but we'll start with one called **Mean Absolute Error** (also called **MAE**)
    - With the MAE metric, we take the absolute value of each error. This converts each error to a positive number. We then take the average of those absolute errors. This is our measure of model quality.
    - `from sklearn.metrics import mean_absolute_error`
- Validate with train data , “In-Sample” score:
    - Since models' practical value comes from making predictions on new data, we measure performance on data that wasn't used to build the model. The most straightforward way to do this is to exclude some data from the model-building process, and then use those to test the model's accuracy on data it hasn't seen before. This data is called **validation data**.
    - `from sklearn.model_selection import train_test_split`

Optimizing the model

- Try out different models to see which provide the best predictions
- Be aware of underfitting and overfitting



- With DecisionTreeRegressor, we can use *max_leaf_nodes* to prevent underfitting and overfitting

```
Max leaf nodes: 5               Mean Absolute Error:  347380
Max leaf nodes: 50              Mean Absolute Error:  258171
Max leaf nodes: 500             Mean Absolute Error:  243495
Max leaf nodes: 5000            Mean Absolute Error:  254983
```

- When max leaf nodes 500 is the right options because when max leaf nodes is 5000, the error is increased, so that's the sign of overfitting.

Here's the takeaway: Models can suffer from either:

- **Overfitting:** capturing spurious patterns that won't recur in the future, leading to less accurate predictions, or
- **Underfitting:** failing to capture relevant patterns, again leading to less accurate predictions.

We use **validation** data, which isn't used in model training, to measure a candidate model's accuracy. This lets us try many candidate models and keep the best one.