Intro to Machine Learning with Python

Based on Kaggle Learn

V. Alavi

“Machine intelligence is the last invention that humanity will ever need to make.”
~Nick Bostrom

How Models Work

Introduction

We'll start with an overview of how machine learning models work and how they are used.

This may feel basic if you've done statistical modeling or machine learning before. Don't worry, we will progress to building powerful models soon.

Scenario

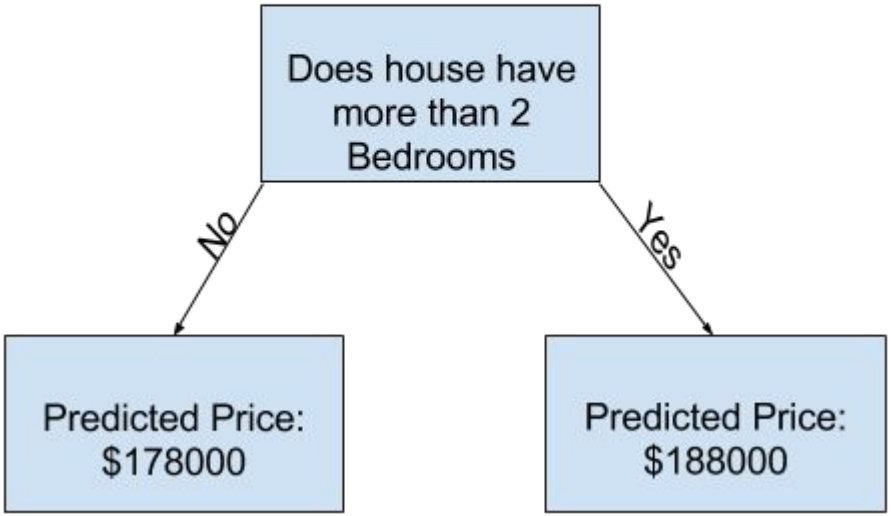
Your cousin has made millions of dollars speculating on real estate. He's offered to become business partners with you because of your interest in data science. He'll supply the money, and you'll supply models that predict how much various houses are worth.

Decision Tree

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.

For simplicity, we'll start with the simplest possible decision tree.

Sample Decision Tree



It divides houses into only two categories.

Sample Decision Tree

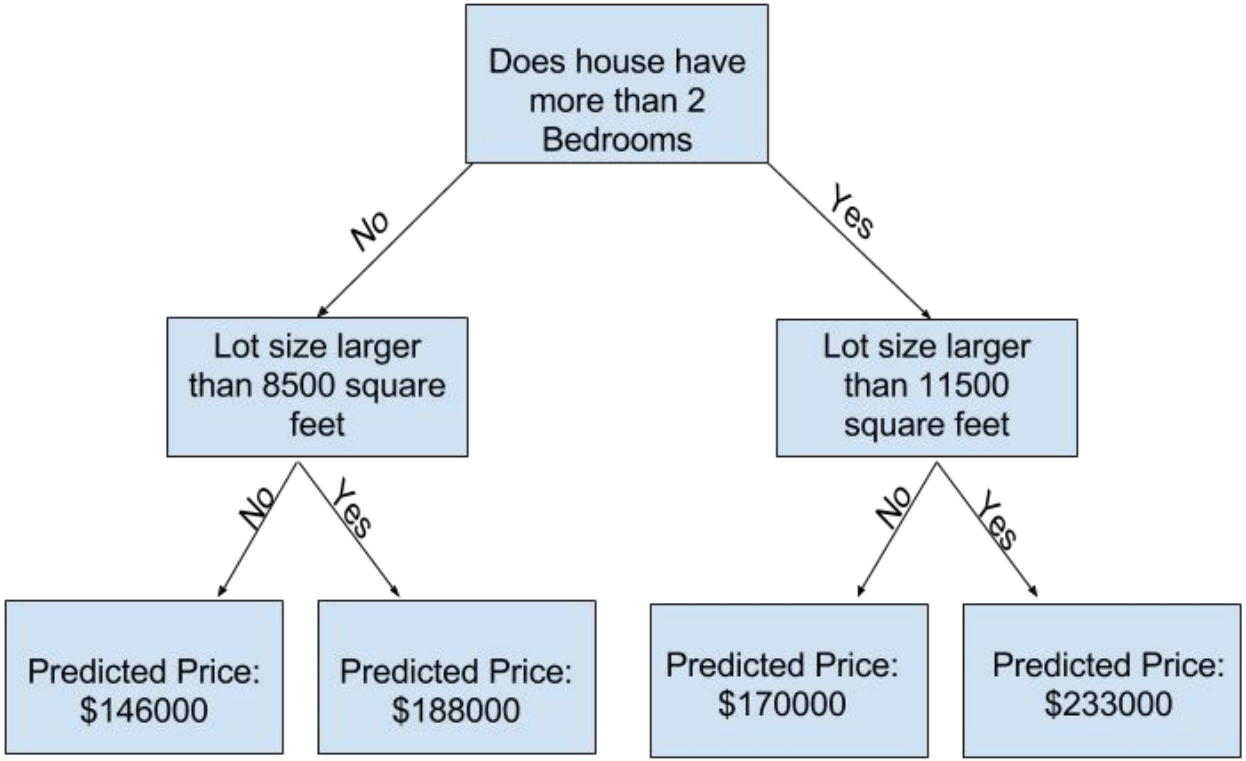
We use data to decide how to break the houses into two groups, and then again to determine the predicted price in each group. This step of capturing patterns from data is called **fitting** or **training** the model. The data used to **fit** the model is called the **training data**.

After the model has been fit, you can apply it to new data to predict prices of additional homes.

Improving the Decision Tree

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

Conclusion

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 (**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.

Basic Data Exploration

Using Pandas to Get Familiar With Your Data

The first step in any machine learning project is familiarize yourself with the data. You'll use the Pandas library for this. Pandas is the primary tool data scientists use for exploring and manipulating data.

Pandas

The most important part of the Pandas library is the DataFrame. A DataFrame holds the type of data you might think of as a table. This is similar to a sheet in Excel, or a table in a SQL database.

Pandas has powerful methods for most things you'll want to do with this type of data.

Your First Machine Learning Model

Selecting Data for Modeling

Your dataset had too many variables to wrap your head around, or even to print out nicely. How can you pare down this overwhelming amount of data to something you can understand?

Selecting a Subset Data

There are many ways to select a subset of your data, but we will focus on two approaches for now.

- **Dot notation**, which we use to select the **"prediction target"**
- Selecting with a `column list`, which we use to select the **"features"**

Selecting The Prediction Target

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

We'll use the dot notation to select the column we want to predict, which is called the **prediction target**. By convention, the prediction target is called **y**.

Choosing "Features"

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

In our case, those would be the columns used to determine the home price. Sometimes, you will use all columns except the target as features. Other times you'll be better off with fewer features.

By convention, this data is called **X**.

Building Your Model

You will use the scikit-learn library to create your models.

When coding, this library is written as sklearn, as you will see in the sample code. Scikit-learn is easily the most popular library for modeling the types of data typically stored in DataFrames.

Steps to Building & Using a Model

- **Define:** What type of model will it be? A decision tree? Some other type of model?
- **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.

Model Validation

What is Model Validation

You've built a model. But how good is it?

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

Summarizing Model Quality Metrics

There are many metrics for summarizing model quality, but we'll start with one called **Mean Absolute Error** (also called **MAE**).

The Problem with "In-Sample" Scores

The measure we just computed can be called an "in-sample" score. We used a single "sample" of houses for both building the model and evaluating it.

We need 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.**

Random Forests

Experimenting With Different Models

Now that you have a reliable way to measure model accuracy, you can experiment with alternative models and see which gives the best predictions. But what alternatives do you have for models?

Underfitting and Overfitting in Decision Trees

Decision trees leave you with a difficult decision. A deep tree with lots of leaves will overfit because each prediction is coming from historical data from only the few houses at its leaf. But a shallow tree with few leaves will perform poorly because it fails to capture as many distinctions in the raw data.

Random Forests

The random forest uses many trees, and it makes a prediction by averaging the predictions of each component tree. It generally has much better predictive accuracy than a single decision tree and it works well with default parameters. If you keep modeling, you can learn more models with even better performance, but many of those are sensitive to getting the right parameters.

Takeaways

- kaggle.com/learn

Thanks