Machine Learning

|Chapter 02 - End-To-End ML Project

Prof. Dr. Mostafa Elhosseini

https://youtube.com/drmelhosseini

Agenda

End-to-End Machine Learning Project

- Look at the big picture.
- Get the data.
- Explore and visualize the data to gain insights.
- Prepare the data for machine learning algorithms.
- Select a model and train it.
- Fine-tune your model.
- Present your solution.
- Launch, monitor, and maintain your system.

California census data

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8.3252	452600.0	NEAR BAY
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8.3014	358500.0	NEAR BAY
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7.2574	352100.0	NEAR BAY
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5.6431	341300.0	NEAR BAY
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3.8462	342200.0	NEAR BAY

Real-world Data

- When you are learning about Machine Learning it is best to actually experiment with real-world data, not just artificial datasets.
- Fortunately, there are thousands of open datasets to choose from, ranging across all sorts of domains

Open Datasets

Popular open data repositories:

- OpenML.org
- Kaggle.com
- PapersWithCode.com
- UC Irvine Machine Learning Repository
- Amazon’s AWS datasets
- TensorFlow datasets

Open Datasets

Meta portals (they list open data repositories):

- DataPortals.org
- OpenDataMonitor.eu

Other pages listing many popular open data repositories:

- Wikipedia's list of machine learning datasets
- Quora.com
- The datasets subreddit

Real estate company

- You are a **data scientist** at a real estate company
- Your role encompasses leveraging data **analytics and machine learning** techniques to glean insights from property data
  - California Housing Prices

California Housing Prices

- From the StatLib repository
- Based on data from the **1990 California census**.
- It is **not exactly recent** (a nice house in the Bay Area was still affordable at the time), but it has many qualities for learning, so we will pretend it is recent data.
- For **teaching purposes** I've added a categorical attribute and removed a few features.

California Housing Prices

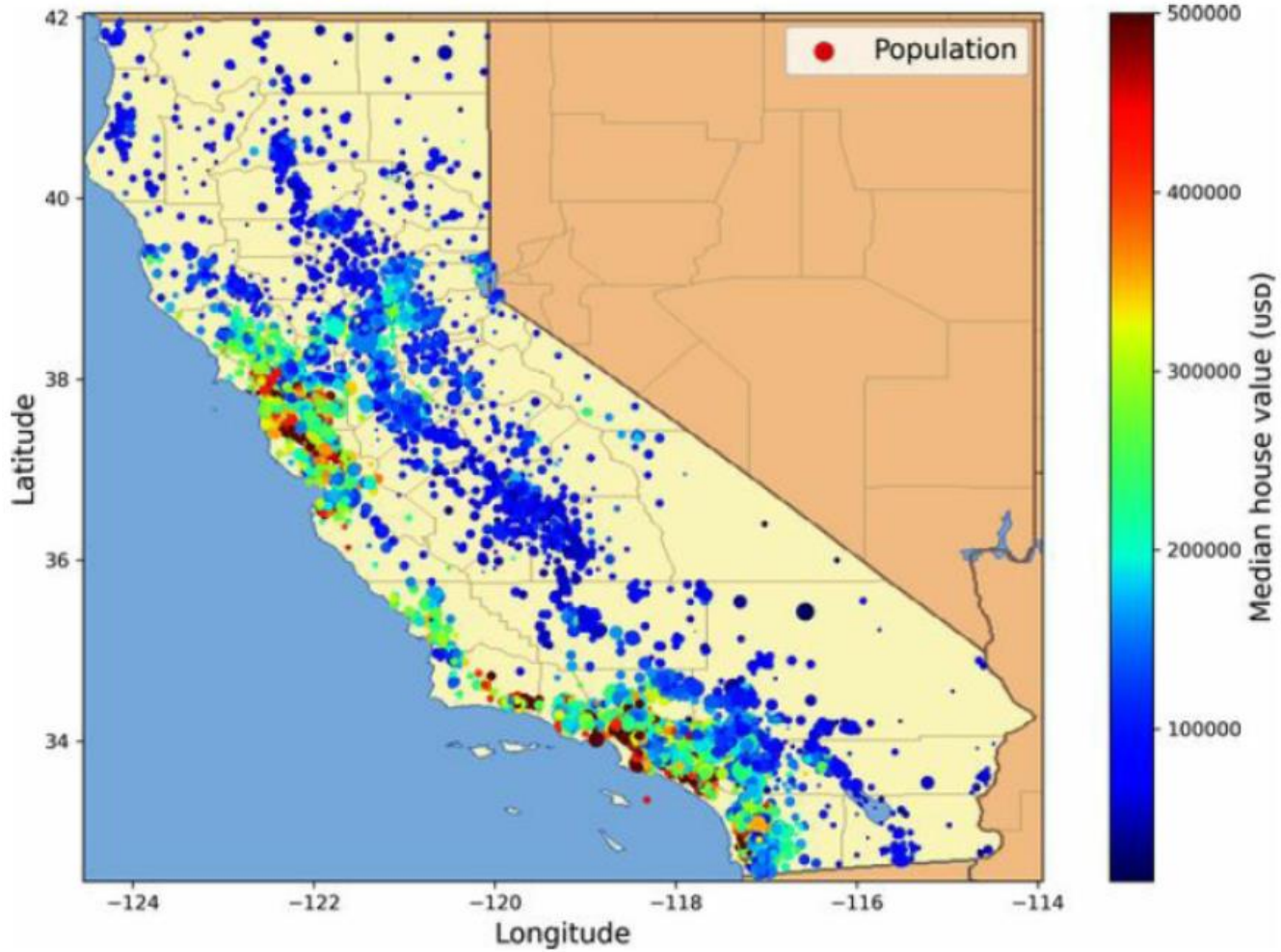

Figure 2-1. California housing prices

California Housing Prices

- Look at the big picture
- Frame the problem
- Get the data
- Discover and visualize the data to gain insights
- Prepare the data for machine learning algorithms

Look at the big picture

- Use California census data to **build a model** of housing prices in the state
- This data includes **metrics (Variables)** such as
  - The population,
  - Median income, and
  - Median housing price for each **block group** in California.

Look at the big picture

- **Block groups** are the smallest geographical unit for which the US Census Bureau publishes sample data
  - # home / block: 200 : 1200
  - # Residents / Home: 2.5 : 3
  - # Residents / Block: 600: 3000
- Call it "**district**" for short.

Look at the big picture

Main Objective:

- Your model should learn from this data and be able to predict the median housing price in any district, given all the other metrics.

Frame the Problem

- What exactly is the business objective?
  - Building a model is probably not the end goal.
  - How does the company expect to use and benefit from this model?
  - This is important because it will determine how you frame the problem
- What algorithms you will select?
- What performance measure you will use to evaluate your model
- How much effort you should spend tweaking it

Frame the Problem

- your **model's output** (a prediction of a district's median housing price) will be **fed to** another Machine Learning system
- This **downstream system** will determine whether it is worth investing in a given area or not

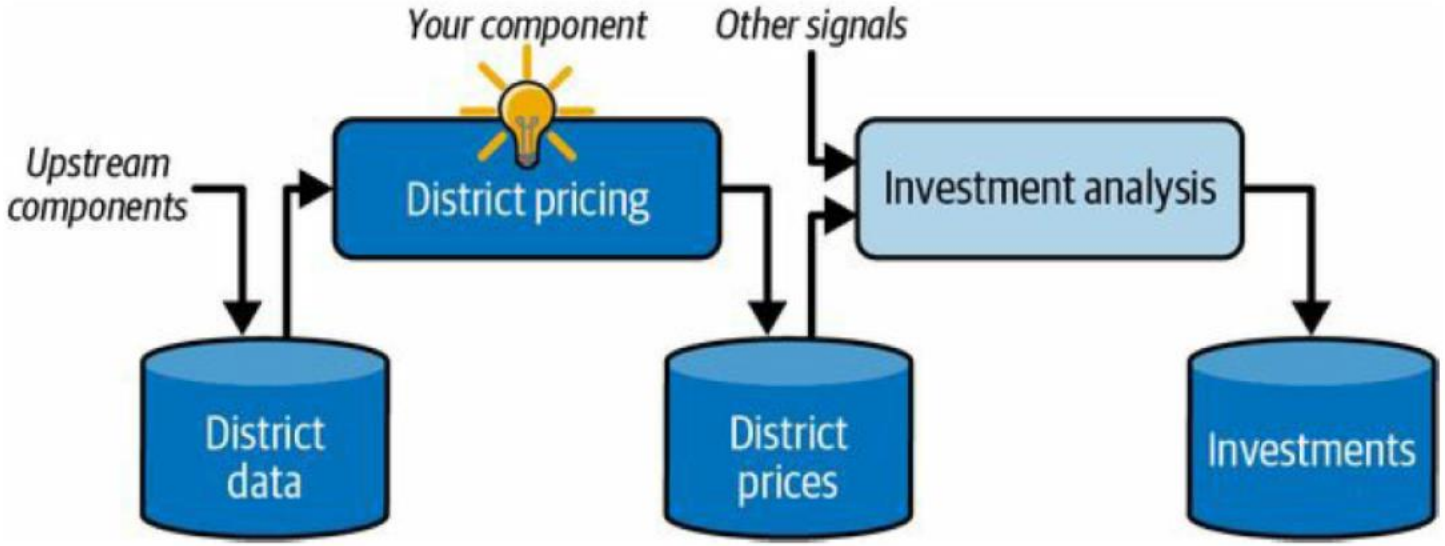

Figure 2-2. A machine learning pipeline for real estate investments

Frame the Problem

Pipeline

- Pipeline refers to a **series of automated processes or steps** that data goes through from its raw form to a usable output, typically including steps like data collection, cleaning, feature extraction, modeling, and evaluation
- Components typically run **asynchronously**. Each component pulls in a large amount of data, processes it, and spits out the result in another data store.
- The interface between components is simply the **data store**
- With libraries like Scikit-Learn, a pipeline is a **specific tool** that helps in chaining multiple preprocessing steps and models together into a single coherent workflow.

Frame the Problem

Key Characteristics of Pipelines:

- The system simple to grasp (with the help of a data flow graph), and
- **Automation**: Once set up, pipelines automate the transition from raw data to predictions, ensuring consistent application of all steps and reducing manual error.
- Different teams can focus on different components.

Frame the Problem

Key Characteristics of Pipelines:

- **Simplicity and Clarity**: By bundling the entire data flow into a single entity, pipelines make the code cleaner and easier to understand.
- **Reproducibility**: They ensure that the exact same data processing and modeling steps are consistently applied, enhancing the reproducibility of results across different stages of the model lifecycle.
- **Efficiency**: Pipelines can sometimes optimize the workflow by avoiding redundant computations, leading to more efficient processing.
- Moreover, if a component breaks down, the downstream components can often continue to run normally (at least for a while) by just using the last output from the broken component - This makes the architecture quite robust.

Frame the Problem

- What the current solution looks like (if any).
  - A reference performance,
  - Insights on how to solve the problem.
- Manually by experts: They use complex rules to come up with an estimate.
- This is costly and time-consuming, and their estimates are not great; their typical error rate is about 15%.

Frame the Problem

You are now ready to start designing your system.

- First, you need to **frame the problem**: is it supervised, unsupervised, or Reinforcement Learning?
- Is it a classification task, a regression task, or something else?
- Should you use batch learning or online learning techniques?

Frame the Problem

- Supervised learning task
- Regression task
- Multiple regression problem (multiple features)
- Multivariate regression is where there is more than 1 dependent variable

Frame the Problem

- **Batch learning**: there is no continuous flow of data coming in the system, there is no particular need to adjust to changing data rapidly, and the data is small enough to fit in memory

- If the data was huge, you could either split your batch learning work across multiple servers (using the **MapReduce technique**), or you could use an online learning technique instead.

Frame the Problem

Select a Performance Measure

- A typical performance measure for regression problems is the **root mean square error** (RMSE).
- It measures the **standard deviation** of the errors the system makes in its predictions
- $RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2}$
  - m is the number of instances
  - h is your system prediction function, also called a hypothesis
- Mean Absolute Error, also called Average Absolute Deviation
  - $MAE = \frac{1}{m}\sum_{i=1}^{m}\left|h(x^{(i)}) - y^{(i)}\right|$

Frame the Problem

Select a Performance Measure

- Both the RMSE and the MAE are ways to measure the distance between two vectors: the vector of predictions and the vector of target values
  - MAE corresponds to ℓ_1 norm, noted $\| \quad \|_1$, also called Manhattan norm
  - RMSE corresponds Euclidean norm, or to ℓ_2 norm, noted $\| \quad \|_2$ or just $\| \quad \|$
  - $\| \quad \|_k$ norm of a vector containing n elements

Frame the Problem

Select a Performance Measure

- if there are many **outlier** districts. In that case, you may consider using the mean absolute error
- The **higher the norm index**, the more it focuses on large values and neglects small ones. This is why the RMSE is more sensitive to outliers than the MAE.

For MAE:

$$\text{MAE} = \frac{1}{4}(|10-12| + |8-9| + |100-103| + |5-4|) = \frac{1}{4}(2+1+3+1) = 1.75$$

For RMSE:

$$\text{RMSE} = \sqrt{\frac{1}{4}((10-12)^2 + (8-9)^2 + (100-103)^2 + (5-4)^2)} =$$
$$\sqrt{\frac{1}{4}(4+1+9+1)} = \sqrt{3.75} \approx 1.936$$

Get the Data

- **Open source** and **available online** as **Jupyter notebooks**,
- Jupyter notebooks are interactive documents containing text, images, and executable code snippets (Python in our case).
- **Google Colab**: Free service that lets you run any Jupyter notebook directly online, without having to install anything on your machine.
- If you want to use another online platform (e.g., **Kaggle**)
- First, open a web browser and visit https://homl.info/colab3: this will lead you to Google Colab, and it will display the list of Jupyter notebooks for this book