CHAPTER 2

End-to-End Machine Learning Project

Chapter 2 presents a full end-to-end machine learning project, where you act as a data scientist at a real estate company and build a model to predict California districts' median house prices from census features such as population and median income. The key idea is to follow a realistic workflow: understand the business goal, frame it as a supervised multiple regression problem, choose appropriate metrics, explore and prepare the data, train and tune models, and finally deploy and monitor the system.

The problem is framed as predicting a single numeric target (median house value) from multiple input features, so it is a univariate, multiple regression task within supervised learning. To evaluate models, the chapter introduces Root Mean Square Error (RMSE) as the main metric, which measures typical prediction error and penalizes large errors more strongly. Given m instances with features x_i, targets y_i, and predictions $\hat{y}_i = h(x_i)$, RMSE is defined as

$$RMSE(X, h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(h(xi) - yi)^2}$$

It also presents Mean Absolute Error (MAE) as an alternative,

$$MAE(X, h) = \frac{1}{m}\sum_{i=1}^{m}|h(xi) - yi|$$

Which is less sensitive to outliers.

You then move to working with real data: downloading the California housing dataset, creating a workspace, and doing an early train/test split to keep a clean test set that is never touched until final evaluation. The chapter emphasizes exploratory data analysis using summary statistics, histograms, geographical plots, and correlation matrices to understand distributions, detect anomalies, and see how features like median income and location correlate with house prices. These insights guide feature engineering, such as constructing new attributes (e.g., ratios or per-household measures) that help models capture meaningful patterns.

Data preparation is treated as a critical step: handling missing values (imputation or dropping), dealing with outliers, encoding categorical variables, and applying feature scaling (standardization or normalization) so that learning algorithms behave well. The chapter introduces transformation pipelines that chain these steps, ensuring the exact same preprocessing is applied to training, validation, test, and future production data. This makes the workflow reproducible and reduces bugs when the model is deployed.

Modeling starts with simple baselines and then explores more powerful algorithms like decision trees and Random Forests, evaluated using cross-validation rather than a single split to get more reliable performance estimates. Hyperparameter tuning is handled using grid search (systematically exploring a parameter grid) and randomized search (sampling combinations to cover large spaces more efficiently), always guided by metrics such as RMSE. The chapter also encourages analyzing error patterns—looking at which types of districts are mispredicted—to refine both features and model choices.

Finally, once a satisfactory model is found, it is evaluated on the untouched test set to estimate its true generalization performance, and the results are interpreted back in terms of the business objective (for example, how much better it is than manual expert rules).

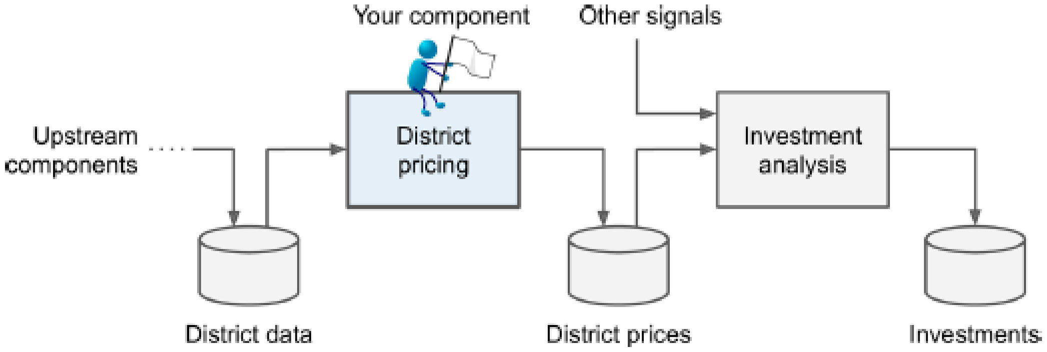

Figure 2 A Machine Learning Pipeline For Real Estate Investments

The model is then placed inside a broader data pipeline [Figure 2-2], which includes data collection, preprocessing, prediction, and a downstream investment decision component. The chapter closes by stressing deployment concerns: monitoring input data and performance for drift or failures in upstream components, logging predictions, and planning for periodic retraining so the end-to-end system remains accurate and useful over time