Heading	Steps	Python Libraries	Definition and Use of Library	Real-World Project Example
Problem Definition	Define problem, expected outcomes, measure success.	N/A	No specific library; this is a business/research problem-solving phase.	**Predicting house prices:** Define objective—predict house prices based on features like size, location, etc.
Data Collection	Gather data (CSV, JSON, Excel) from databases, APIs, etc.	pandas, requests, beautifulsoup4, selenium	pandas: Data manipulation; requests: Access APIs; beautifulsoup4: Scraping web data.	**Collecting house data:** Scrape housing websites or use an API for property prices, size, and location.
Data Exploration	Understand structure, stats, visualize, detect outliers.	pandas, matplotlib, seaborn	pandas: Data exploration; matplotlib & seaborn: Visualization tools.	**Exploring house data:** Visualize relationships like size vs. price, detect outliers, explore missing values.
Data Cleaning	Handle missing data, treat outliers, normalize/encode.	pandas, scikit-learn	pandas: Handle missing/outlier data; scikit-learn: Transformations like normalization, encoding.	**Cleaning house data:** Remove outliers (e.g., extreme prices), impute missing values, and normalize numerical features.
Feature Engineering	Select/create features, reduce dimensions using PCA.	scikit-learn, pandas	scikit-learn: Feature selection and dimensionality reduction (PCA).	**Feature creation in housing:** Add new features like price per square foot, or reduce dimensions using PCA.
Data Splitting	Split into training/testing (e.g., 70-30).	scikit-learn	scikit-learn: train_test_split() function to divide data.	**Train/test split for house prices:** Split data into training (70%) and testing (30%) to validate predictions.
Model Selection	Choose algorithm (e.g., Linear Regression, SVM, K-Means).	scikit-learn, tensorflow, keras	scikit-learn: Classical models (e.g., Linear Regression, SVM); tensorflow/keras: Neural networks.	**Choosing a model for house price prediction:** Test models like Linear Regression, Random Forests, or Neural Networks.
Model Training	Train model, tune hyperparameters, cross-validation.	scikit-learn, tensorflow, keras	scikit-learn/tensorflow/keras: Used for training models, hyperparameter tuning with Grid Search, Random Search, or Bayesian.	**Training a regression model:** Train a Random Forest model to predict house prices based on the dataset.
Model Evaluation	Test on unseen data, evaluate using metrics like accuracy, RMSE.	scikit-learn	scikit-learn: Provides metrics like precision, recall, RMSE to evaluate model performance.	**Evaluating house price prediction:** Check model accuracy by comparing predicted prices with actual prices (RMSE score).
Model Tuning	Fine-tune, use ensemble techniques like boosting/bagging, hyper parameter tuning	cross valdation	scikit-learn: Hyperparameter tuning.	**Tuning for house price prediction:** Use boosting (e.g., XGBoost / hyperparameter tuning) for better accuracy in price prediction.
Model Deployment	Deploy using Flask, FastAPI, or cloud (AWS, GCP).	Flask, FastAPI, AWS SDK	Flask/FastAPI: Frameworks for building APIs; AWS SDK: For cloud deployment.	**Deploying house price prediction model:** Create an API using Flask to predict house prices in
Model Monitoring	Monitor performance, retrain with new data.	MLFlow	Monitor model performance; pandas: Process new data for retraining.	**Monitoring house price prediction:** Monitor model accuracy over time and retrain when needed with updated market data.
Communication & Reporting	Present insights with reports, dashboards, visualizations.	matplotlib, seaborn, PowerBI	matplotlib/seaborn: Visualization libraries; PowerBI: Business analytics for reporting.	**Report house price predictions:** Create a dashboard to display market trends, predicted vs. actual prices.
Continuous Improvement	Perform error analysis, update model with feedback.	scikit-learn, pandas	pandas/scikit-learn: Analyze errors and continuously improve the model with feedback.	**Improving house price prediction:** Regularly refine the model based on feedback, correcting errors in price predictions.