Configurable Data Science Framework

A **real-world use case** for this project would be **predicting house prices based on real estate data**.

Scenario: Real Estate Price Prediction

You're a data scientist working for a real estate company that wants to predict house prices based on various factors such as location, number of bedrooms, square footage, and more. Your goal is to build a machine learning model that accurately estimates the price of a house given its attributes.

How This Code Helps

The provided framework automates the following **five steps** in the data science workflow:

1. Problem Definition

You configure the `config.yaml` to define the problem:

```yaml
problem_definition:
  name: "House Price Prediction"
  description: "Predict the price of houses based on features like size, location, and number of rooms."
```

✅ Clearly states the objective of the analysis.

---

2. Data Collection

The company provides a dataset (`housing.csv`) with house listings, which contains:

ID	Location	Bedrooms	Bathrooms	Sqft	Price
1	New York	3	2	1500	450000

ID	Location	Bedrooms	Bathrooms	Sqft	Price
2	Chicago	2	1	1000	250000
3	Miami	4	3	2200	600000

The **data_collection** module reads this CSV file into a Pandas DataFrame.

```yaml
data_collection:
  source: "csv"
  file_path: "data/housing.csv"
```

✅ Loads data automatically from CSV files.

---

3. Data Preparation

Raw data might have missing values or be in different scales. The **data_preparation** module:
✅ **Fills missing values** (e.g., if some houses are missing sqft values, they are replaced by the mean).
✅ **Normalizes numeric columns** to avoid bias (e.g., house size might dominate the impact of location).

Configurable via:

```yaml
data_preparation:
  missing_values: "mean"
  normalize: true
```

---

4. Data Analysis

Before building the model, you analyze the data:
✅ **Correlation Heatmap**: Shows which variables impact house prices the most (e.g., larger

houses = higher prices).

✅ **Feature Selection**: Helps determine which attributes should be included in the model.

```yaml
data_analysis:
  correlation_threshold: 0.5
```

The **data_analysis** module visualizes the relationships, allowing you to filter out unnecessary columns.

---

5. Model Building & Evaluation

Now, we train a **Linear Regression model** to predict house prices.

```yaml
model_building:
  model_type: "linear_regression"
  train_test_split: 0.8
  metrics: ["rmse", "r2"]
```

✅ **Splits data** into training (80%) and testing (20%).

✅ **Trains a machine learning model** based on features.

✅ **Evaluates performance** (RMSE, R^2 Score).

Expected Results

After running `python main.py`, the model outputs:

```sh
Problem: House Price Prediction
Description: Predict the price of houses based on features like size, location, and
number of rooms.
```

```
Data Loaded: 500 rows
Missing values filled using mean strategy
Data normalized
Strong correlations detected with Price: Sqft (0.85), Bedrooms (0.7)

Model: Linear Regression
RMSE: 25000
R² Score: 0.89
```

Why This is Useful in Real Life?

1. **For Real Estate Agents**: Helps them price houses accurately based on data.
2. **For Home Buyers**: Predicts fair market value before making an offer.
3. **For Investors**: Identifies undervalued properties based on trends.
4. **For Governments**: Analyzes housing market trends for policy-making.

This framework **automates everything** from loading data to model training and evaluation, making it reusable for different industries beyond real estate.

Would you like enhancements, such as adding more models (e.g., RandomForest, Neural Networks)? 🚀