
🔧 Project Title: `finCleanPy` – A Financial Data Cleaner Library

📌 Project Overview

`finCleanPy` is a Python package that provides tools to clean, preprocess, and validate raw financial datasets. It includes AI-enhanced capabilities like anomaly detection, auto-imputation, and feature engineering assistance for structured financial data.

🎯 Key Objectives

- Remove inconsistencies, missing values, and duplicates in financial datasets.
- Detect and correct outliers using AI.
- Normalize and encode data for analysis.
- Provide a user-friendly API for integration with pandas.
- Enable extendable custom cleaning pipelines.

🧱 Project Structure

```
finCleanPy/
│
├── fincleanpy/
│   ├── __init__.py
│   ├── core.py              # Core functions for data cleaning
│   ├── validators.py         # Schema and value validators
│   ├── imputers.py           # Smart AI-based imputers
│   ├── anomaly.py            # Anomaly detection models
│   ├── feature_engineering.py  # Optional feature enrichment
│   ├── utils.py             # Helper functions
│
├── examples/
│   ├── clean_stock_data.py
│   ├── clean_transaction_data.py
```

```
│
├── tests/
│   ├── test_core.py
│   ├── test_anomaly.py
│
├── setup.py
├── README.md
├── requirements.txt
└── pyproject.toml
```

🔄 Pipeline Design

1. Input: Financial Data

- Source: CSVs, APIs (like Yahoo Finance, Alpha Vantage), databases.
- Types: Stock prices, transaction logs, balance sheets, etc.

---

2. Data Profiling

- Summary statistics
- Missing values report
- Type detection

```
from fincleanpy import core
core.profile_data(df)
```

---

3. Data Cleaning Modules

a. Missing Value Handler

- Simple Imputation: Mean, Median, Forward-fill
- AI-Based Imputation: KNN, XGBoost, or Transformer-based model

```
from fincleanpy import imputers
df = imputers.smart_impute(df, method="xgboost")
```

b. Outlier Detection (AI Integrated)

- Isolation Forest
- AutoEncoder for anomaly scoring
- Z-Score and IQR based fallback

```
from fincleanpy import anomaly
df_clean = anomaly.remove_outliers(df, method="autoencoder")
```

c. Data Type Correction

- Detect and convert incorrect types (e.g., strings to float)
- Handle date parsing

d. Normalization & Scaling

- Min-Max, Z-Score, or log-transform for skewed data

e. Feature Engineering (Optional)

- Lag features
- Rolling averages
- Domain-specific metrics (e.g., Sharpe ratio, volatility)

---

4. Validation

- Schema checks: column names, types
- Logical checks: e.g., no negative values for volume

```
from fincleanpy import validators
validators.validate_schema(df, expected_schema)
```

---

5. Export Cleaned Data

- Output to CSV, Parquet, or DataFrame object
- Optionally serialize pipeline

🤖 AI Integration Points

Module	AI Model	Purpose
Missing Value Imputation	XGBoost, KNN	Context-aware imputation
Anomaly Detection	Isolation Forest, Autoencoder	Catch unusual financial behavior
Data Profiler	LLM or rule-based	Detect type mismatches, suggest column types
Feature Selector (optional)	RandomForest/SHAP	Rank relevant financial features

---

📦 Sample API Design

```
from fincleanpy import CleanPipeline

pipeline = CleanPipeline()
pipeline.add_step("profile")
pipeline.add_step("smart_impute", method="knn")
pipeline.add_step("remove_outliers", method="isolation_forest")
pipeline.add_step("normalize")

clean_df = pipeline.run(df)
```

🧪 Testing and Validation

- Unit tests for each module (`pytest`)
- Integration tests on example datasets

- Benchmark against open datasets (e.g., Yahoo Finance)

📊 Example Use Cases

- **Retail transaction data**: Clean logs of customer purchases
- **Stock market time series**: Remove outliers, handle missing OHLCV values
- **Banking records**: Clean large-volume CSVs for lending models

---

🛠️ Tools & Libraries

- **Data Handling**: `pandas`, `numpy`
- **AI Models**: `scikit-learn`, `xgboost`, `keras` (for AutoEncoders)
- **Validation**: `pandera`, `pydantic`
- **Packaging**: `setuptools`, `poetry`
- **Testing**: `pytest`

🚀 Extensions

- AutoML integration for data cleaning suggestions
- Web UI for drag-and-drop cleaning pipeline
- API deployment with FastAPI for SaaS-like usage
