Project PyFin- Design Doc

This design document describes the architecture of an open-source Python module designed for econometric analysis with a focus on model validation. The name of the package is "PyFin". This package is a practical tool for quantifying relationships among economic variables that provides robust methodologies to validate statistical models.

1 Module 1: Data Processing

Features:

1.1 Dataset import and export: **A feature to import and export data files using pandas.**

1.2 Dataset pre-processing: **Cleaning, handling missing values, outliers, and normalization.**

1.3 Dataset splitting: **Split data into training and testing sets**

2 Module 2: Econometric Models

Models/Features:

2.1 Linear Regression Models: Implements simple and multiple linear regression models. For model fitting, we will use the Ordinary Least Squares (OLS) and Generalized Least Squares (GLS) estimation techniques.

2.1.1 fit: For estimating the model parameters.

2.1.2 predict: For making predictions based on the model parameters.

2.1.3 summary: To print out a summary of the model results including coefficient estimates, standard errors, t-statistics, p-values, and R-squared.

2.2 Time Series Models: Implements Autoregressive Integrated Moving Average (ARIMA), Vector Autoregression (VAR), and Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models.

2.2.1 fit: To estimate model parameters.

2.2.2 predict: For making forecasts based on the model parameters.

2.2.3 summary: To provide a summary of the model results including coefficient estimates, standard errors, and various information criteria like AIC and BIC.

2.3 Panel Data Models: Implements pooled OLS, fixed effects, and random effects models.

2.3.1 fit: To estimate model parameters

2.3.2 predict: For making predictions based on the model parameters

2.3.3 summary: To provide a summary of the model results including coefficient estimates, standard errors, t-statistics, p-values, and various panel-specific test statistics like the Breusch-Pagan test for random effects and the Hausman test for fixed effects.

2.4 **Non-linear Models:** Implements Logit, Probit, and other non-linear models.

2.4.1

2.4.2 fit: To estimate model parameters.

2.4.3 predict: For making predictions based on the model parameters.

2.4.4 summary: To provide a summary of the model results including coefficient estimates, standard errors, t-statistics, p-values, and various goodness of fit measures like McFadden's R-squared.

2.5 **Cointegration Models and Error Correction Models (ECM):** These models are used to examine the long-run relationships between non-stationary variables.

2.5.1 fit: To estimate model parameters.

2.5.2 predict: For making predictions based on the model parameters.

2.5.3 summary: To provide a summary of the model results including coefficient estimates, standard errors, t-statistics, p-values, and cointegration test statistics.

2.6 Instrumental Variables: Implements Two-Stage Least Squares (2SLS) for addressing endogeneity problems.

2.6.1 fit: To estimate model parameters using 2SLS.

2.6.2 predict: For making predictions based on the model parameters.

2.6.3 summary: To provide a summary of the model results including coefficient estimates, standard errors, t-statistics, p-values, and tests for overidentification and endogeneity like the Sargan test and the Durbin-Wu-Hausman test.

2.7 Quantile Regression: This method investigates the impact of independent variables over the different quantiles of the dependent variable.

2.7.1 fit: To estimate model parameters.

2.7.2 predict: For making predictions based on the model parameters.

2.7.3 summary: To provide a summary of the model results including coefficient estimates, standard errors, t-statistics, p-values, and the pseudo R-squared for goodness of fit.

3 Module 3: Diagnostics and Model Validation

Features:

3.1 **Residual analysis:** Checking for assumptions of homoscedasticity, no autocorrelation, and normality. It will also include graphical residual analysis.

3.2 **Hypothesis Testing:** Implement tests for statistical significance (t-tests, F-tests, chi-square tests)

3.3 **Model Selection Metrics:** R-squared, AIC, BIC and other model selection metrics.

3.4 **Out of Sample Validation:** Implement hold-out and cross-validation methods for validating the models on unseen data.

3.5 **Stability Tests:** Implement common stability tests like CUSUM and CUSUM of squares.

3.6 **Unit Root Tests:** ADF, PP, KPSS tests for checking stationarity in time series data.

3.7 **Cointegration Tests:** Implement tests like the Johansen test for checking cointegration between variables.

4 Module 4: Model Comparison and Ensemble Learning

4.1 **Model comparison:** Comparison of various models based on various metrics.

4.2 **Ensemble methods:** Simple ensemble methods like model averaging.

5 Module 5: Visualization

5.1 **Graphical Analysis:** Plotting of data and econometric model results (e.g. line plots, scatter plots, etc.)

5.2 **Residual Plots:** Graphical representation of model residuals.

5.3 **Correlation Matrix:** Heatmap representation of correlations between different variables in the dataset.

6 Software Design

The design follows an Object-Oriented Programming approach. For each econometric model, a class will be implemented that includes methods for fitting the model to data, predicting new values, and validating the model.

7 Dependencies

The dependencies of the project are restricted to the following libraries:

numpy: for numerical computations

pandas: for data manipulation and handling

8 Usage and Documentation

Each method and function will have detailed docstrings explaining the usage. A separate README file will also be available with a complete guide on how to use the package.

Testing

Unit testing will be implemented using Python's built-in unittest framework. Each module, method, and function will have corresponding test cases.

Future Extensions

Machine learning models could be included in the future, allowing for modern, data-driven approaches to econometric analysis. Dependencies could be expanded to include libraries like scikit-learn and statsmodels to achieve this.