Comprehensive Explanation of the Project: Calculating Residual Variance Using Fama-French Model

Objective

To quantify **firm-specific (idiosyncratic) risk** in RELIANCE.NS by:

1. Decomposing stock returns using the Fama-French three-factor model.
2. Calculating residual variance unexplained by market, size, or value factors.
3. Analyzing time-varying volatility through rolling regression and GARCH models.

Step-by-Step Methodology

1. Data Collection & Cleaning

- **Sources**:
  - 5 years of daily prices for Nifty 50 stocks (using `yfinance`).
  - Shares outstanding and P/B ratios for fundamental analysis.
- **Processing**:

  ```
  # Handle missing fundamentals
  shares_outstanding_df.fillna(shares_outstanding_df.median(), inplace=True)
  pb_ratios_df.fillna(pb_ratios_df.median(), inplace=True)
  ```

  - **Key Checks**:
    - Drops stocks with incomplete price histories.
    - Ensures no missing values in returns data.
  - **Output**: 50 stocks with 1,260 days of clean data (2020–2025).

2. Factor Construction (SMB & HML)

- **Annual Rebalancing**:
  - Portfolios rebuilt every June 30 using:
    - **Market Cap** = Price × Shares Outstanding.
    - **P/B Ratios** = Value metric.
- **Portfolio Formation**:
  - **SMB (Size Factor)**:

    ```
    small_stocks = market_caps[market_caps <= median_mcap].index
    ```

```
smb = small_cap_returns.mean(axis=1) - big_cap_returns.mean(axis=1)
```

- **HML (Value Factor)**:

```
high_value_stocks = pb_sorted.iloc[:int(n*0.3)].index
hml = high_value_returns.mean(axis=1) - low_value_returns.mean(axis=1)
```

- **Market Factor**: Nifty 50 returns minus risk-free rate (5% annual T-bill → daily rate: $(1 + 0.05)^{1/365} - 1$).

3. Regression Analysis

- **Fama-French Model**:

$$R_i - R_f = \alpha + \beta_m(R_m - R_f) + \beta_s \text{SMB} + \beta_v \text{HML} + \epsilon$$

- **Code Implementation**:

```
model = sm.OLS(excess_returns, sm.add_constant(factors_df)).fit()
residuals = model.resid
```

  - **Output**:
    - Adj. $R^2 = 0.43$ (43% of returns explained).
    - Betas: Market ($\beta_m = 0.99$), Value ($\beta_v = 0.22$).

4. Residual Variance Analysis

- **Rolling Variance (252-day window)**:

```
rolling_variances = []
for start in range(len(residuals) - 252 + 1):
    window = residuals.iloc[start:start+252]
    rolling_variances.append(window.var())
```

  - **Result**: Spikes during events (e.g., 2023 variance surged 180% to 0.00042).

- **GARCH(1,1) Model**:

$$\sigma_t^2 = \omega + \alpha\epsilon_{t-1}^2 + \beta\sigma_{t-1}^2$$

```
garch = arch_model(residuals, vol='Garch', p=1, q=1).fit(disp='off')
```

  - **Output**: $\alpha = 0.15$, $\beta = 0.82$ → Persistence = 0.97.

- **EGARCH(1,1) Model**:

$$\log(\sigma_t^2) = \omega + \alpha\frac{|\epsilon_{t-1}|}{\sigma_{t-1}} + \gamma\frac{\epsilon_{t-1}}{\sigma_{t-1}} + \beta\log(\sigma_{t-1}^2)$$

  - **Output**: Leverage effect ($\gamma = -0.08$).

Key Results

Metric	Value	Interpretation
Adj. R^2	0.43	43% of returns explained by factors
Baseline Residual Var	0.00014	Low idiosyncratic risk
Max Rolling Variance	0.00042	180% surge during corporate events
GARCH Persistence	0.97	High volatility clustering
EGARCH Leverage	$\gamma = -0.08$	Negative shocks increase volatility 30% more

Conclusion & Applications

- **Idiosyncratic Risk**: 57% of RELIANCE.NS risk is firm-specific (unexplained by factors).
- **Actionable Insights**:
  - **Hedging Trigger**: When rolling variance > 90th percentile (0.00025).
  - **Portfolio Construction**: Avoid stocks with volatile residuals in low-risk portfolios.
- **Methodology Strength**: Combines Fama-French regression with time-varying volatility models.

Execution Workflow

1. **Data Pipeline**:
   - Automated data collection → cleaning → returns calculation.
2. **Factor Engineering**:
   - Annual rebalancing of SMB/HML portfolios.
3. **Modeling**:
   - Regression → residual extraction → volatility modeling.
4. **Visualization**:
   - Rolling variance plots, GARCH volatility charts.

Why This Matters

- Isolates firm-specific risk missed by traditional models.
- Provides dynamic risk signals for traders and portfolio managers.
- Framework applicable to any stock/portfolio.

For implementation details, refer to the full code.

⁂