📄 Securities-Based Lending (SBL) Risk Model – Project Documentation

Project Background and Description:

This project simulates the client portfolio analysis process used in securities-based lending (SBL). The goal is to develop a rule-based model to determine client eligibility for a loan based on their investment portfolio. Using simplified criteria—Loan-to-Value (LTV) and volatility score—the model mimics a core responsibility of data analysts at Supernova Technology.

Project Scope:

- This mini project includes:
- Simulating portfolios of 100 clients with 3–5 holdings each
- Assigning market values and volatility scores
- Calculating risk factors (LTV and average volatility)
- Applying eligibility logic using thresholds
- Visualizing and interpreting the results

High-Level Requirements:

- Generate realistic client and stock portfolio data
- Compute total market value and loan request
- Calculate LTV and volatility score per client
- Define business logic:
  → LTV ≤ 0.5
  → Volatility ≤ 0.35
- Classify clients as Eligible / Not Eligible
- Create charts and summary outputs
- Justify the threshold choices using financial reasoning

Affected Business Processes or Systems:

- The implementation of this project demonstrates impacts on:
- Portfolio risk validation processes
- Suitability modelling

- QA/testing for client onboarding and Decision support for lending eligibility

Business Logic Justification: Why These Thresholds?

In the absence of access to Supernova's proprietary suitability or risk models, I created a **rule-based model** that simulates how loan decisions might be made based on portfolio risk. I selected the following conservative thresholds for a client to qualify for a securities-based loan:

- **Loan-to-Value (LTV) ≤ 0.5**
- **Average Volatility Score ≤ 0.35**

Why LTV ≤ 0.5?

- The **Loan-to-Value (LTV)** ratio helps assess the **risk of lending too much money against a portfolio**. If the stock market drops, a high-LTV loan puts the lender at greater risk of loss.

Example:
If a client has a portfolio worth **$100,000** and they request:

- **$80,000** → LTV = 0.8 → High risk
- **$50,000** → LTV = 0.5 → Safer margin

By setting the threshold at **50%**, I'm simulating a conservative risk strategy where the lender always has **50% collateral buffer** in case of market decline.

Why Volatility ≤ 0.35?

Volatility Score is a proxy for how much the underlying portfolio might fluctuate in value.

Example:

- TSLA is highly volatile, often above **0.5**, while MSFT or AAPL might stay near **0.2–0.3**.
- A client heavily invested in **high-volatility stocks** (e.g., TSLA, AMZN) poses **higher default risk** if the market drops quickly.

Setting a threshold of **0.35** creates a **safe range** for clients with relatively stable portfolios to qualify, like the logic used in actual SBL platforms.

What I'm Simulating

These two rules work together to mimic what Supernova might do when testing suitability models:

- If the client's **loan request is too large** (high LTV), or
- Their **portfolio is too volatile**,
- Then they are flagged as **"Not Eligible"**.

This helps illustrate how automated rules can be used for **risk flagging and client-level validation** exactly what a Data Analyst would support at Supernova.

Implementation Plan

- **Research**: Studied risk assessment logic used in SBL
- **Data Simulation**: Built a portfolio dataset from scratch using Python
- **Eligibility Modelling**: Developed the logic and applied thresholds
- **Visualization**: Plotted and interpreted outcomes
- **Documentation**: Prepared detailed artifacts for submission

SUMMARY

This project was my attempt to replicate what a Data Analyst might do at Supernova from client data onboarding and validation to simple rule-based suitability modelling and risk profiling. I designed everything to reflect what your job description outlined: working with portfolio data, calculating eligibility using LTV and volatility, and preparing insights that help product and risk teams refine models. I'd be excited to take this further #with real data, back-testing, and pipeline improvements.