Politechnika Warszawska

Wydział Matematyki i Nauk Informacyjnych

RiskFolio: prognozowanie wyników inwestycyjnych z kwantyfikacją ryzyka

Adam Bagiński, nr indeksu 338978
Aleksandra Dmitruk, nr indeksu 338981
Barbara Gawlik, nr indeksu 338987

Zaawansowane Programowanie Obiektowe i Funkcyjne, sem. zimowy 2025/2026

Warszawa, 26 stycznia 2026

Spis treści

1 Opis tematu

1.1 Charakterystyka projektu

RiskFolio to aplikacja desktopowa stworzona w środowisku Java, której głównym zadaniem jest budowanie wirtualnych portfeli inwestycyjnych oraz przeprowadzanie symulacji ich przyszłej wartości. System wykorzystuje algorytmy statystyczne oraz metodę Monte Carlo, aby dostarczyć użytkownikowi prognozy oparte na realnych danych historycznych z rynków finansowych.

1.2 Cel biznesowy i grupa docelowa

Grupą docelową są inwestorzy indywidualni często podejmują decyzje w oparciu jedynie o historyczne stopy zwrotu, ignorując kluczowy aspekt, jakim jest zmienność i ryzyko rynkowe. Odpowiedzią na ten problem jest aplikacja RiskFolio, która:

1. Dostarcza narzędzia do świadomego zarządzania inwestycjami;
2. Kwantyfikuje ryzyko za pomocą miar takich jak VaR (Value at Risk - maksymalna strata z 95% przedziałem ufności) oraz CVaR (Conditional Value at Risk - średnia strata wśród 5% najgorszych przypadków niekwalifikujących się do VaR);
3. Umożliwia wizualizację różnych scenariuszy rynkowych (od pesymistycznych po optymistyczne), co pomaga w planowaniu strategii inwestycyjnych.

Aplikacja jest skierowana zarówno do początkujących inwestorów, jak i osób poszukujących szybkich narzędzi do weryfikacji założeń portfelowych bez konieczności budowania skomplikowanych modeli lub używania płatnych platform.

2 Założenia

2.1 Założenia funkcjonalne

1. Zarządzanie portfelem – użytkownik:
    - dodaje aktywa poprzez wpisanie tickera danego aktywa lub poprzez wybranie opcji spośród wyświetlających się podpowiedzi na podstawie wpisanego fragmentu nazwy;
    - określa udział aktywów w portfelu (np. 40% AAPL.US, 60% ALE);
    - podaje kapitał początkowy (np. 100 000 PLN);

- podaje horyzont prognozy (np. 365 dni – UWAGA! jeden dzień odpowiada jednej sesji giełdowej – rok kalendarzowy to ok. 252 sesji giełdowych)

2. Walidacja danych oraz stabilności połączenia – aplikacja zwróci błąd w przypadku wprowadzania nieprawidłowych danych przez użytkownika – np. ujemny horyzont czasowy czy nieistniejący / nieobsługiwany ticker. Dodatkowo, aplikacja poinformuje o braku połączenia / błędzie połączenia z serwerem.

3. Pobieranie danych – aplikacja automatycznie pobiera historyczne notowania z wprowadzonego przez użytkownika okresu z zewnętrznego API (Stooq).

4. **Analiza statystyczna – system na podstawie wykonanej symulacji Monte Carlo (10 000 scenariuszy) oblicza:**

   - podstawowe metryki dla portfela (np. średni zwrot, maksymalny zwrot);
   - parametry kwantyfikacji ryzyka (VaR, CVaR);
   - parametry modelu GARCH(1,1) optymalne dla konfiguracji początkowej podanej przez użytkownika;

5. Wizualizacja – wizualizacja wartości portfela dla scenariusza najlepszego, średniego, VaR, CVaR oraz raport tekstowy informujący o wyliczonych parametrach i ewentualnych ostrzeżeniach.

2.2 Założenia niefunkcjonalne

1. Programowanie funkcyjne – wykorzystanie Java Streams API do przetwarzania danych finansowych (np. map, filter);

2. Programowanie obiektowe – architektura oparta na wielu współpracujących ze sobą klasach oraz strukturach danych typu record (np. SimulationResult) dla zapewnienia niemodyfikowalności;

3. Wydajność – wykorzystanie Parallel Streams, co pozwala na równoległe wykorzystanie wielu wątków procesora, do pobierania danych i obliczania symulacji w ciągu maksymalnie dziesięciu sekund;

4. Czysty i czytelny kod – podział wykonywanych zadań pomiędzy różne pliki, generalny opis zadania każdego z plików, jednolite i intuicyjne nazewnictwo oraz plik README;

5. Intuicyjność aplikacji – przyjazny interfejs w języku polskim pozwalający na skonfigurowanie własnego portfela w uruchomienia symulacji w czasie krótszym niż dwie minuty;

6. Bezpieczeństwo i prywatność – aplikacja nie przechowuje danych użytkownika, wszystkie wprowadzone informacje są automatycznie kasowane po zakończeniu sesji;

7. Niezawodność – system oferuje dwa tryby obsługi niekompletnych danych historycznych – uzupełnianie brakujących notowań notowaniami indeksu WIG20 dla spółek polskich oraz notowaniami indeksu S&P 500 dla spółek amerykańskich, oraz skracanie historii analizowanych danych do najdłuższej wspólnej serii danych dla aktywów spoza rynków polskiego i amerykańskiego.

3 Schemat systemu

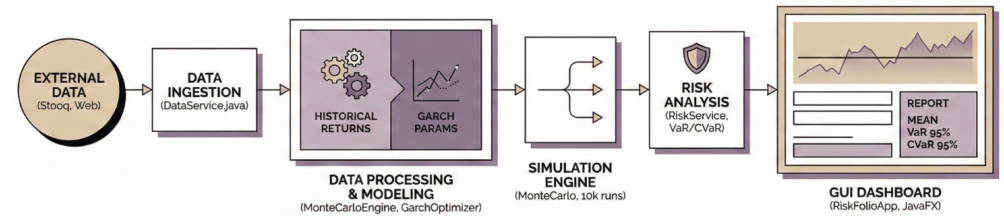
System RiskFolio składa się z 4 logicznych warstw:

1. Warstwa danych – pobieranie i przetwarzanie historycznych danych.

2. Warstwa modelowania – estymacja optymalnych parametrów GARCH, obliczanie miar ryzyka, prognozowanie wyniku portfela przy użyciu metody Monte Carlo.

3. Warstwa koordynacji – sterowanie kolejnością operacji i przepływem danych.

4. Warstwa prezentacji – interakcja z użytkownikiem, walidacja danych wejściowych, uruchamianie obliczeń w wątku roboczym, wizualizacja wyników i raportowanie.



Rysunek 1: Schemat przepływu danych w RiskFolio.

Przepływ sterowania:

1. Użytkownik wprowadza dane portfela inwestycyjnego w GUI:

   - wybrane instrumenty finansowe,
   - wagi portfela,
   - kapitał początkowy,

- horyzont czasowy,
- zakres danych historycznych.

2. GUI waliduje dane wejściowe – poprawność formatu, suma wag musi równać się 1, kapitał i horyzont muszą być dodatnie.
3. Po kliknięciu *Uruchom symulację* tworzony jest nowy wątek roboczy
4. Silnik MonteCarloEngine przejmuje sterowanie:
   - pobiera dane historyczne dostosowane do konfiguracji portfela,
   - znajduje optymalne parametry modelu GARCH dla danego portfela,
   - uruchamia symulację Monte Carlo, której wynikiem jest prognoza dynamiki portfela dzień po dniu, w tym wynik końcowy portfela.
5. Do GUI wracają następujące wyniki:
   - statystyki końcowe portfela – maksymalny, średni i minimalny wynik,
   - miary ryzyka – wartości VaR 95% oraz CVaR 95%,
   - wizualizacja dynamiki portfela na przestrzeni horyzontu czasowego – najlepszy scenariusz, średni scenariusz, wartości VaR oraz CVaR,
   - wyznaczone optymalne parametry modelu GARCH – jeśli któryś z parametrów będzie przyjmował bardzo niską bądź bardzo wysoką wartość, wyświetla się też krótki komentarz związany z interpretacją tej wartości.

4 Wykorzystane technologie

- Java 21 – język implementacji rozwiązania,
- JavaFX – warstwa graficzna,
- Maven – zarządzanie zależnościami,
- IntelliJ IDEA 2024 – środowisko, w którym programowaliśmy,
- Bitbucket – wersjonowanie plików projektu,
- Stooq – źródło historycznych danych finansowych,
- programowanie obiektowe,
- programowanie funkcyjne,
- programowanie równoległe.

5 Opis implementacji

5.1 Model ekonomiczny

1. Dla każdej ceny zamknięcia instrumentu P_t liczymy dzienną stopę zwrotu:

$$r_t = \frac{P_t - P_{t-1}}{P_{t-1}}$$

2. Dzienna stopa zwrotu portfela składającego się z n instrumentów:

$$R_t = \sum_{i=1}^{n} w_i r_{i,t} \quad \text{gdzie} \quad \sum_{i=1}^{n} w_i = 1 \quad \text{oraz} \quad w_i - \text{waga } i\text{-tego instrumentu}$$

3. Model GARCH(1,1). Wariancja stóp zwrotu opisana jest równaniem:

$$\sigma_t^2 = \omega + \alpha r_{t-1}^2 + \beta \sigma_{t-1}^2 \quad \text{gdzie } \alpha + \beta < 1 \text{ oraz } \omega > 0, \alpha \geq 0, \beta \geq 0$$

4. Proces symulacyjny Monte-Carlo.

   Losowy szok rynkowy:

$$\varepsilon_t = \sigma_t \cdot Z_t \quad \text{gdzie } Z_t \sim N(0,1)$$

   Dzienna stopa zwrotu portfela:

$$R_t = \mu + \varepsilon_t$$

   Ewolucja wartości portfela:

$$V_{t+1} = V_t \cdot (1 + R_t)$$

5. Miary ryzyka

   VaR (Value at Risk) na poziomie ufności α:

$$VaR_\alpha = \inf\{x : P(V_T \leq x) \geq 1 - \alpha\}$$

   CVaR (Conditional Value at Risk) na poziomie ufności α:

$$CVaR_\alpha = E\left[V_T \mid V_T \leq VaR_\alpha\right]$$

5.2 Przetwarzanie danych

Klasa DataService realizuje pełny pipeline przepływu danych:

1. Budowa URL do API Stooq.
2. Pobranie danych z ustawionym nagłówkiem User-Agent.
3. Parsowanie CSV z pominięciem nagłówka.
4. Filtrowanie rekordów (tak, aby mieć tylko te prawidłowe).
5. Ekstrakcja ceny zamknięcia.
6. Ograniczenie zakresu danych do liczby dni: $liczba\_lat \cdot 252$
7. Transformacja cen zamknięcia na stopy zwrotu.

5.3 Estymacja parametrów modelu GARCH(1,1)

Kalibracja modelu realizowana jest metodą *Random Search*, mającą na celu maksymalizację funkcji wiarygodności. Proces ten przebiega w następujących krokach:

1) **Inicjalizacja i losowanie parametrów:**
Wykonujemy równolegle 2000 iteracji. W każdej z nich losujemy parametry α i β z przyjętych przedziałów. Parametr ω jest wyznaczany analitycznie na podstawie wariancji historycznej oraz danych α i β. Zestaw parametrów jest akceptowany tylko wtedy, gdy spełniony jest warunek stacjonarności procesu, tj gdy $\alpha + \beta < 1$.

$$\begin{aligned} \alpha &\sim U(0.01, 0.26) \\ \beta &\sim U(0.50, 0.99) \\ \alpha + \beta &< 1 \\ \omega &= \mathrm{Var}(r) \cdot (1 - \alpha - \beta) \end{aligned}$$

2) **Symulacja i ocena błędu:**
Dla wylosowanego zestawu parametrów przechodzimy przez całą historię stóp zwrotu. Dla każdego momentu t obliczamy wkład do funkcji straty (ujemny logarytm wiarygodności) oraz aktualizujemy wariancję na krok $t + 1$:

- Obliczenie wartości funkcji celu (błędu) L:

$$L = \sum_{t=1}^{T} \left(\ln(\sigma_t^2) + \frac{r_t^2}{\sigma_t^2} \right)$$

- Aktualizacja wariancji warunkowej (równanie GARCH):

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

Gdzie σ_1^2 inicjowane jest jako $\mathrm{Var}(r)$, wariancja obliczona na podstawie danych historycznych.

3) **Wybór optymalnego rozwiązania:**
Algorytm porównuje wartości L dla wszystkich iteracji i zwraca zestaw parametrów $\{\omega, \alpha, \beta\}$, dla którego suma błędów jest najmniejsza (najlepiej dopasowuje zmienność modelu do danych historycznych).

5.4 Prognozowanie z metodą Monte Carlo

Silnik symulacyjny realizuje:

1. Agregację historycznych zwrotów portfela.
2. Estymację średniego dziennego zwrotu.
3. Kalibrację GARCH
4. Uruchomienie 10 000 równoległych, niezależnych ścieżek.
5. Dynamiczną aktualizację wariancji GARCH po każdym kroku czasowym.
6. Zapis wyników do SynchronizedList.

5.5 Mechanika GUI

GUI zostało zaprojektowane tak, aby całość obliczeń wykonywana była w osobnym wątku. Interfejs odświeżamy wyłącznie przez Platform.runner().

6 Instrukcja wdrożeniowa

6.1 Wymagania systemowe i środowiskowe

Do uruchomienia aplikacji *RiskFolioApp* są konieczne:

1. Java Development Kit (JDK) w wersji 21 lub nowszej (język implementacji rozwiązania),
2. środowisko programistyczne, najlepiej IntelliJ IDEA 2024,

3. stabilne połączenie z Internetem, ponieważ dane pobierane są w czasie rzeczywistym z serwisu Stooq,
4. narzędzie Maven do zarządzania bibliotekami zewnętrznymi zdefiniowanymi w pliku *pom.xml*,
5. kod źródłowy aplikacji *RiskFolioApp.java*,
6. pliki konieczne do funkcjonowania aplikacji, umieszczone w folderze źródłowym (*DataSerive.java*, *GarchOptimizer.java*, *GarchParams.java*, *Instrument.java*, *InstrumentDatabase.java*, *MainLauncher.java*, *MonteCarloEngine.java*, *RiskService.java*, *SimulationResult.java*)
7. plik *tickers.csv* w folderze głównym projektu
8. klient Git,
9. konto na serwisie bitbucket.org.

6.2 Instrukcja uruchomienia aplikacji

W celu uruchomienia aplikacji należy wykonać następujące kroki:

1. Z repozytorium projektu należy uzyskać link konieczny do skopiowania repozytorium.
2. W środowisku IntelliJ IDEA otworzyć projekt z systemu kontroli wersji za pomocą linku z repozytorium.
3. Po wykryciu przez środowisko pliku pom.xml kliknąć powiadomienie *Load Maven Project*.
4. Uruchomić aplikację poprzez plik pomocniczy *MainLauncher.java*. Jest to konieczne, aby obejść restrykcje systemu modułów Javy (JPMS). Pozwala to na poprawne ładowanie bibliotek JavaFX ze ścieżki Classpath bez konieczności pełnej modularyzacji projektu.

7 Instrukcja użytkownika

Po uruchomieniu skryptu*MainLauncher.java*, na ekranie użytkownika powinno pojawić się okno aplikacji RiskFolio. Aplikacja jest podzielona na dwa obszary, panel centralny z wizualizacją oraz panel boczny. Do zarządzania aplikacją służy panel boczny znajdujący się po lewej stronie ekranu. Panel boczny zawiera zestaw

interaktywnych pól, które pozwalają użytkownikowi zdefiniować parametry inwestycji. Każde pole opatrzone jest ikoną (?). Po najechaniu na nią kursorem myszy wyświetla się dymek z dodatkowym wyjaśnieniem danej funkcji. Poniżej zostaną przedstawione wszystkie pola.

1. **Dodaj aktywa:** W tym polu użytkownik może wybrać aktywa, jakie mają znaleźć się w jego portfelu. Aplikacja obsługuje automatyczne wyszukiwanie aktywów po wpisywanych przez użytkownika literach. Aktywa można wpisywać jako symbole giełdowe i za pomocą pełnych nazw. Aplikacja obsługuje wyszukiwanie 2364 aktywów, których słownik tworzy plik tickers.csv. Kliknięcie w podpowiedź dodaje instrument do portfela.

2. **Twój portfel:** Wybrane aktywa pojawiają się w tej sekcji w formie interaktywnych *kafelków*. Każdy kafelek reprezentuje jedną spółkę lub aktywo. Aby usunąć instrument z portfela, należy kliknąć symbol x, znajdujący się na kafelku.

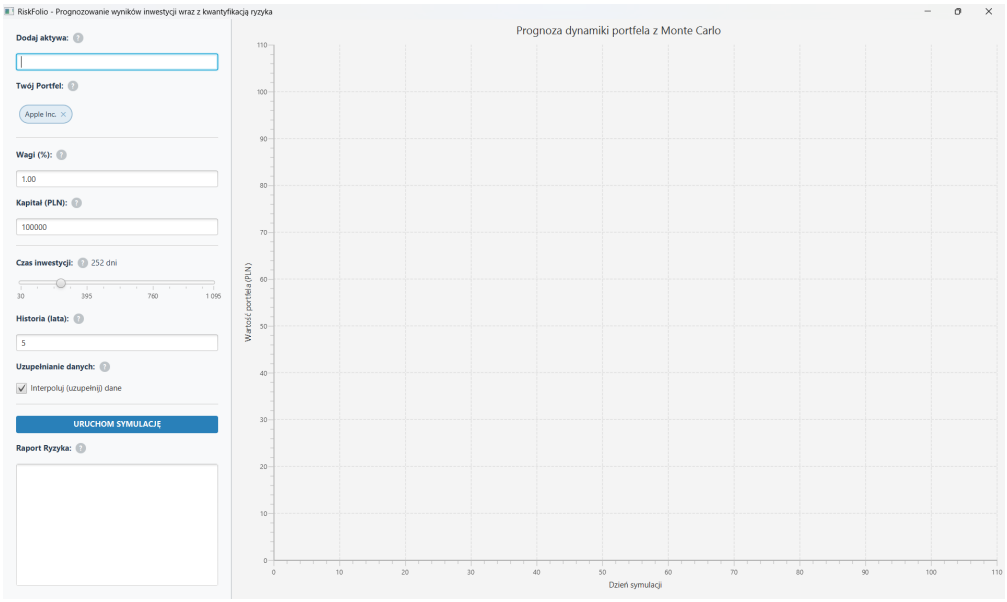
3. **Wagi:** Określa procentowy udział każdego instrumentu w portfelu. Domyślnie aplikacja automatycznie przelicza wagi po równo dla każdego dodanego aktywa (np. dla 2 spółek po 0.50, dla 4 spółek po 0.25). Użytkownik może ręcznie edytować wartości (pamiętając, że ich suma musi wynosić 1.0). W przypadku błędów zaokrągleń, silnik symulacji automatycznie znormalizuje wagi do 1.0 przed uruchomieniem obliczeń.

4. **Kapitał:** Kwota początkowa inwestycji wyrażona w Polskich Złotych. Stanowi punkt odniesienia dla obliczania zysków i strat.

5. **Czas inwestycji:** Interaktywny suwak pozwalający określić horyzont czasowy symulacji (od 30 dni do 3 lat). Przesuwanie suwaka dynamicznie aktualizuje liczbę dni wyświetlaną powyżej. Domyślna wartość to 252 dni, co odpowiada jednemu rokowi giełdowemu.

6. **Historia:** Określa, jak daleko wstecz aplikacja ma sięgnąć po dane historyczne, aby *nauczyć się* zachowania rynku (zmienności i korelacji).

7. **Uzupełnianie danych:** Jeżeli historyczne dane notowań dla któregoś z wprowadzonych przez użytkownika aktywów są krótsze niż wybrana historia (Zob. 6.) i aktywo jest notowane na rynku polskim bądź amerykańskim, to brakujące notowania zostaną zastąpione notowaniami indeksów, odpowiednio WIG20 i S&P 500. Jeżeli aktywo nie pochodzi z żadnego z wymienionych wyżej rynków, to dane historyczne zostaną ucięte do najdłuższej możliwej, wspólnej serii danych.

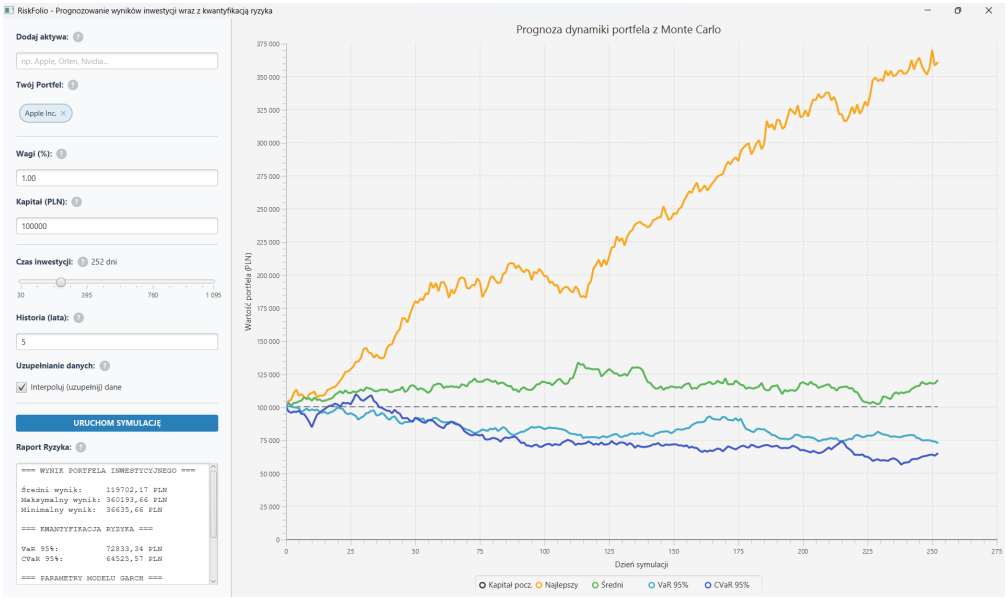


Rysunek 2: Wygląd aplikacji RiskFolio.

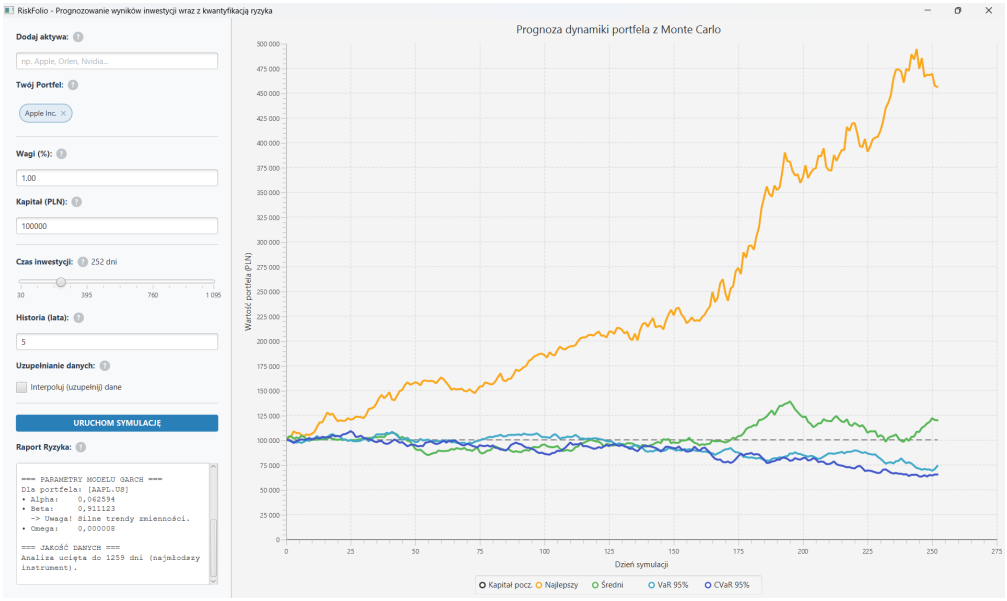
Aby rozpocząć proces, należy kliknąć niebieski przycisk URUCHOM SYMULACJĘ. Aplikacja połączy się z serwisem Stooq, pobierze dane, skalibruje model GARCH i wygeneruje 10,000 scenariuszy przyszłości. Wykres prezentuje możliwe ścieżki wartości portfela w czasie. Linia przerywana oznacza poziom kapitału początkowego. Linie kolorowe reprezentują różne scenariusze (najlepszy, średni, oraz scenariusze pesymistyczne oparte o miary VaR i CVaR). Legenda wyświetla się pod wykresem. W dolnej części panelu bocznego Raport Ryzyka generowany jest raport tekstowy zawierający kluczowe metryki:

1. **Statystyki:** Średni, maksymalny i minimalny przewidywany wynik.
2. **Kwantyfikacja Ryzyka:**
   - VaR 95% (Value at Risk): Maksymalna strata w typowych warunkach rynkowych (z ufnością 95%).
   - CVaR 95% (Conditional VaR): Przewidywana strata w przypadku wystąpienia najgorszego przypadku.
3. **Diagnostyka GARCH:** Parametry modelu (Alpha, Beta, Omega) opisujące zmienność i *nerwowość* wybranych aktywów.

Symulacją można dowolnie manipulować zmieniając parametry i ponownie klikając przycisk URUCHOM SYMULACJĘ.



Rysunek 3: Aplikacja RiskFolio po uruchomieniu symulacji.



Rysunek 4: W lewym dolnym rogu wyświetlają się również wyznaczone parametry modelu GARCH.

8 Podsumowanie

8.1 Wnioski – praca projektowa

Praca nad projektem była podzielona na dwa etapy. Najpierw powstały główne funkcjonalności projektu, takie jak:

a) pobieranie danych historycznych z serwisu Stooq,

b) uruchamianie symulacji Monte Carlo dla stałych parametrów alpha, beta, omega dla GARCH,

c) utworzenie pierwszej wersji aplikacji z wizualizacją symulacji,

d) funkcje zajmujące się obliczaniem VaR i CVaR.

W drugim etapie pracy powstały bardziej zaawansowane elementy projektu:

a) obsługa błędów połączeń sieciowych w DataService.java,

b) model ML do wyznaczania parametrów alpha, beta, omega do GARCH,

c) biblioteka aktywów i system podpowiedzi w aplikacji,

d) interaktywna informacja o aplikacji, w trakcie jej działania.

Projekt był zadaniem nie tylko programistycznym, ale również matematycznym i technicznym, ze względu na obecną symulację Monte Carlo oraz korzystanie z zewnętrznych danych. Połączenie intuicyjnego interfejsu graficznego z zaawansowaną matematyką finansową było wyzwaniem, przed jakim stanęło rozwiązanie.

8.2 Wnioski stworzone rozwiązanie

Finalna wersja aplikacji RiskFolio stanowi funkcjonalne i wydajne narzędzie wspierające procesy decyzyjne inwestora. Do kluczowych osiągnięć w ramach stworzonego rozwiązania należy zaliczyć:

1) Zastosowanie uczenia maszynowego do kalibracji parametrów modelu GARCH pozwoliło na dynamiczne dopasowywanie *osobowości* modelu do specyfiki danego instrumentu finansowego. Symulacja uwzględnia charakterystykę poszczególnych rodzajów papierów wartościowych, np. spółek dywidendowych, kryptowalut czy surowców, co znacząco podnosi jej wiarygodność.

2) Aplikacja w przejrzysty sposób prezentuje zaawansowane miary ryzyka, takie jak VaR czy CVaR, przez co jest przydatnym narzędziem edukacyjnym.

3) Architektura wielowątkowa oraz oddzielenie warstwy obliczeniowej od interfejsu użytkownika sprawiają, że aplikacja działa płynnie i jest odporna na chwilowe problemy z dostępnością serwisu Stooq.

4) System inteligentnych podpowiedzi oraz automatyczne przeliczanie wag portfela eliminują bariery wejścia dla początkujących użytkowników. Prosta obsługa idzie w parze z precyzją narzędzia analitycznego.

RiskFolio pokazuje, że możliwe jest stworzenie narzędzia, które w czasie rzeczywistym pobiera dane zewnętrzne, przeprowadza złożone obliczenia i prezentuje wyniki w formie przystępnej dla użytkownika. Stworzone rozwiązanie realizuje cel główny pracy, dostarczając wiarygodnej kwantyfikacji ryzyka inwestycyjnego.

8.3 Możliwe kierunki rozwoju

Pomimo osiągniętych sukcesów, RiskFolio jest rozwiązaniem, które ma wyjątkowo dużo kierunków dalszego rozwoju. Przedstawmy niektóre z nich:

1) Użytkownik sam dobiera wagi aktywów w portfelu, bądź aplikacja sama przypisuje wszystkim te same wagi. W przyszłości kierunkiem byłoby zaimplementowanie algorytmu Współczesnej Teorii Portfelowej (MPT), który automatycznie sugerowałyby optymalny podział kapitału w celu maksymalizacji zysku przy określonym poziomie ryzyka.

2) Obecnie aplikacja tworzy jeden model GARCH dla zagregowanego portfela. Naturalnym kierunkiem rozwoju jest zastosowanie modeli klasy DCC-GARCH (Dynamic Conditional Correlation). Pozwoliłoby to na modelowanie zmienności każdego aktywa z osobna oraz na prognozowanie, jak korelacja między nimi zmienia się w czasie. Jest to istotne, ponieważ na rynkach finansowych korelacje często gwałtownie rosną w momentach paniki, czego statyczna agregacja historyczna może nie w pełni uchwycić.

3) Dodanie możliwości generowania profesjonalnych raportów w formacie PDF lub Excel, zawierających wykresy i podsumowanie ryzyka, co ułatwiłoby archiwizację wyników lub prezentację ich osobom trzecim.