Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка програмного забезпечення для вибору стратегії в системах автоматизації трейдингу

Виконав(ла): студент(ка) 4 курсу, групи СН-43
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)

(підпис) Синявський О.Ю. (прізвище та ініціали)

Керівник (підпис) Марценко С.В. (прізвище та ініціали)

Нормоконтроль (підпис) Липак Г.І. (прізвище та ініціали)

Завідувач кафедри (підпис) Боднарчук І.О. (прізвище та ініціали)

Рецензент (підпис) (прізвище та ініціали)

Тернопіль
2025

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
"23" червня 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

студенту Синявський Олег Юрійович
(прізвище, ім'я, по батькові)

1. Тема роботи **Розробка програмного забезпечення для вибору стратегії в системах автоматизації трейдингу**

Керівник роботи **к.т.н., доц. Марценко С.В.**
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від "*07*" *травня* 2025 року № *4/7-444*

2. Термін подання студентом завершеної роботи 23 червня 2025 р.

3. Вихідні дані до роботи Літературні джерела з тематики роботи

4. Зміст роботи (перелік питань, які потрібно розробити)
ВСТУП 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 1.1 Визначення автоматизованого трейдингу 1.2 Принципи роботи автоматизованого трейдингу 1.3 Переваги використання автоматизованих систем трейдингу 1.4 Ризики та виклики автоматизованого трейдингу 1.5 Популярні автоматизовані торгові стратегії 1.6 Важливі фактори при виборі автоматизованої системи трейдингу 1.7 Як розпочати роботу з автоматизованим трейдингом 2 ОГЛЯД СИСТЕМ АВТОМАТИЗОВАНОГО ТРЕЙДИНГУ 2.1 Загальний огляд програмного забезпечення для алгоритмічного трейдингу 2.2 TradeStation – найкраще безкоштовне програмне забезпечення для алгоритмічного трейдингу 2.3 Stock Market Guides як рішення для користувачів 3 ПРАКТИЧНА ЧАСТИНА РОБОТИ 3.1 Машинне навчання для трейдингу 3.2 Поява квантових фондів 3.3 Розробка та реалізація стратегії на основі машинного навчання 4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ; ВИСНОВКИ; СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ; ДОДАТКИ

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Тема. 2. Вступ. 3. Основні поняття. 4. Переваги автоматизованого трейдингу. 5. Ризики та виклики. 6. Популярні торгові стратегії. 7. Характеристики систем автоматизованого трейдингу. 8, 9, 10. Огляд існуючих систем. 11. Машинне навчання в трейдингу. 12. Розробка власної ситсеми.13. Принципи оптимізації стратегій. 14. Тестове виконання програми. 15. Висновки

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці	Окіпний І.Б., к.т.н., доцент кафедри МТ		

7. Дата видачі завдання 29 січня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	24.01.2025 – 27.01.2025	*Виконано*
2.	Підбір джерел по темі роботи	28.01.2025 – 01.04.2025	*Виконано*
3.	Оформлення першого розділу	15.04.2025	*Виконано*
4.	Оформлення другого розділу	20.04.2025	
5.	Оформлення третього розділу	30.04.2025	*Виконано*
6.	Виконання завдання до підрозділу "Безпека		
7.	життєдіяльності, основи охорони праці"	15.05.2025	*Виконано*
8.	Оформлення кваліфікаційної роботи	07.06.2025	*Виконано*
9.	Перевірка на плагіат	07.06.2025	*Виконано*
10.	Нормоконтроль	09.06.2025	*Виконано*
11.	Попередній захист кваліфікаційної роботи	11.06.2025	*Виконано*
12.	Захист кваліфікаційної роботи	28.06.2025	
13.			
14.			
15.			

Студент _____ Синявський О.Ю.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Марценко С.В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

"Розробка програмного забезпечення для вибору стратегії в системах автоматизації трейдингу" // Кваліфікаційна робота освітнього рівня "Бакалавр" // Синявський Олег Юрійович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп’ютерних наук, група СН-43 // Тернопіль, 2025 // с. – 69, рис. – 9, таблиць – 2, джерел – 26, додатків – 1, сторінок додатків – 27.

Ключові слова: прогнозування, ордер, стратегія трейдингу, автоматизований трейдинг, біржа, шаблони, машинне навчання

У сучасних умовах автоматизація трейдингу набуває дедалі більшого значення, оскільки дозволяє підвищити ефективність трейдингу шляхом використання алгоритмічних методів та спеціалізованого програмного забезпечення. Дана бакалаврська робота за спеціальністю 122 – Комп'ютерні науки присвячена розробці автоматизованої системи трейдингу мовою Python, що базується на сучасних методах аналізу ринкових даних та алгоритмічних стратегій.

У роботі розглядаються основні поняття автоматизованого трейдингу, базова термінологія та аналіз існуючих програмних рішень у цій сфері. Особливу увагу приділено теоретичним та практичним аспектам розробки власної системи, включаючи вибір алгоритмів, обробку даних, управління ризиками та інтеграцію із торговими платформами.

Python є однією з найбільш популярних мов програмування у фінансовій сфері завдяки широкій підтримці бібліотек для машинного навчання, аналізу даних та роботи з API торгових платформ. У межах дослідження розроблена програмна система, що реалізує автоматизоване прийняття рішень на основі математичних моделей, статистичного аналізу та історичних даних ринку.

Запропонований підхід спрямований на оптимізацію торгових стратегій, зменшення впливу людського фактору та підвищення точності прогнозування ринкових трендів. Робота містить аналіз проблем та перспектив розвитку алгоритмічного трейдингу, що може бути корисним для студентів, дослідників та професіоналів у сфері комп'ютерних наук та фінансових технологій.

ANNOTATION

"Software Development for Strategy Selection in Automated Trading Systems" // Qualification work of the educational level "Bachelor" // Syniavskyi Oleh // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science, Group CH-43 // Ternopil, 2025 // p. – 69, fig. – 9, tables – 2, references – 26, annexes – 1, pages for annexes – 27.

Keywords: forecasting, order, trading strategy, automated trading, stock exchange, templates, machine learning

In today's financial market conditions, the automation of trading is becoming increasingly significant, as it enhances trading efficiency through the use of algorithmic methods and specialized software. This bachelor's thesis in the field of Computer Science (specialty 122) focuses on the development of an automated trading system in Python, based on modern market data analysis methods and algorithmic strategies.

The thesis explores the fundamental concepts of automated trading, basic terminology, and the analysis of existing software solutions in this domain. Particular attention is given to the theoretical and practical aspects of developing a proprietary system, including algorithm selection, data processing, risk management, and integration with trading platforms.

Python is one of the most popular programming languages in the financial sector due to its extensive support for libraries related to machine learning, data analysis, and interaction with trading platform APIs. As part of the research, a software system has been developed that implements automated decision-making based on mathematical models, statistical analysis, and historical market data.

The proposed approach aims to optimize trading strategies, reduce the impact of human factors, and improve the accuracy of market trend forecasting. The thesis includes an analysis of the challenges and prospects of algorithmic trading

development, which may be useful for students, researchers, and professionals in the fields of computer science and financial technologies.

ЗМІСТ

ВСТУП

Автоматизований трейдинг, також відома як алгоритмічний трейдинг або роботизований трейдинг, здійснив революцію у функціонуванні фінансових ринків. Використовуючи передові комп'ютерні алгоритми, автоматизований трейдинг дозволяє трейдерам здійснювати угоди без втручання людини, роблячи процес швидшим, ефективнішим та менш схильним до людських помилок. Ця передова технологія набула значної популярності в останні роки, трансформуючи ландшафт трейдингу.

Зі зростанням автоматизації в різних галузях промисловості було лише питанням часу, коли трейдинг піде цим шляхом. Автоматизовані торгові системи аналізують величезні обсяги даних, виявляють торгові можливості та виконують угоди на основі заздалегідь визначених критеріїв. Це усуває людську упередженість та емоції з процесу прийняття рішень, що призводить до більш дисциплінованої та об'єктивної трейдингу.

Основна мета автоматизованого трейдингу полягає у використанні неефективності ринку, волатильності та цінових коливань, які трейдери-люди можуть не мати змоги використати. Використовуючи складні математичні моделі та попередньо визначені торгові стратегії, автоматизовані торгові системи можуть діяти швидко та виконувати угоди за частки секунди, що робить їх особливо вигідними в умовах високочастотної трейдингу.

Ця робота має на меті дослідити принципи автоматизованого трейдингу, надаючи представлення про її визначення, функціонування, переваги, ризики та популярні торгові стратегії. Вона також висвітлює важливі фактори, які слід враховувати під час вибору автоматизованої торгової системи, та надає рекомендації щодо початку роботи з автоматизованим трейдингом. Для цього також пропонується програмна система для вибору стратегії автоматизованого трейдингу на основі алгоритмів машинного навчання.

Алгоритмічний трейдинг спирається на комп'ютерні програми, які виконують алгоритми для автоматизації деяких або всіх елементів торгової

стратегії. Алгоритми – це послідовність кроків або правил, розроблених для досягнення мети. Вони можуть приймати різні форми та сприяти оптимізації протягом усього інвестиційного процесу, від генерування ідей до розподілу активів, виконання торгів та управління ризиками.

Машинне навчання (ML) включає алгоритми, які вивчають правила або закономірності з даних для досягнення мети, такої як мінімізація помилки прогнозування. Приклади в цій роботі проілюструють, як алгоритми ML можуть витягувати інформацію з даних для підтримки або автоматизації ключових інвестиційних видів діяльності. Ці види діяльності включають спостереження за ринком та аналіз даних для формування очікувань щодо майбутнього та прийняття рішень щодо розміщення ордерів на купівлю або продаж, а також управління результуючим портфелем для отримання привабливої прибутковості відносно ризику.

Незалежно від того, чи ви досвідчений трейдер, який прагне оптимізувати свою торговельну діяльність, чи початківець, зацікавлений у дослідженні алгоритмічного трейдингу, ця робота надасть необхідні знання, щоб стати спеціалістом у сфері автоматизованого трейдингу.

1АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Визначення автоматизованого трейдингу

Автоматизований трейдинг (автоматизований трейдинг), яку також називають алгоритмічною трейдингом або роботизованою трейдингом, – це метод виконання угод на фінансових ринках з мінімальним втручанням людини або без нього. Він передбачає використання складних комп'ютерних алгоритмів, які аналізують ринкові дані, виявляють торгові можливості та автоматично виконують угоди на основі заздалегідь визначених правил і стратегій.

Ці торгові алгоритми запрограмовані на дотримання набору правил і параметрів, які можуть включати такі фактори, як ціна, обсяг, час і технічні індикатори. Алгоритми використовують історичні та ринкові дані в режимі реального часу для прийняття обґрунтованих торгових рішень і виконання угод за оптимальними цінами та часом.

Автоматизовані торгові системи можуть бути розроблені для трейдингу різноманітними фінансовими інструментами, включаючи акції, облігації, валюту, товари та деривативи. Вони можуть виконувати угоди на кількох ринках та біржах одночасно, використовуючи ринкові можливості, які можуть виникнути протягом кількох секунд.

Однією з ключових особливостей автоматизованих торгових систем є їхня здатність виконувати угоди набагато швидше, ніж трейдери-люди. Це особливо корисно у високочастотному трейдингу, де швидкість має вирішальне значення для використання швидкоплинних ринкових можливостей та максимізації прибутковості.

Ще однією відмінною рисою автоматизованого трейдингу є його здатність виключати людські емоції з процесу трейдингу. Такі емоції, як страх і жадібність, часто можуть затьмарювати судження та призводити до ірраціональних торгових рішень. Автоматизовані торгові системи працюють на основі заздалегідь

визначених правил і не залежать від емоцій, що забезпечує дисциплінований та об'єктивний підхід до трейдингу.

Важливо зазначити, що хоча автоматизовані трейдингові системи можуть сприяти трейдинговій діяльності, вони не є безвідмовними. Вони розроблені для допомоги трейдерам у виконанні угод на основі заздалегідь визначених стратегій, але не гарантують прибутку. Успіх автоматизованого трейдингу зрештою залежить від якості торговельної стратегії, точності алгоритмів та поточних ринкових умов.

У наступному розділі ми розглянемо те, як працюють автоматизовані торгові системи та механізми виконання ними угод.

1.2 Принципи роботи автоматизованого трейдингу

Автоматизовані торгові системи покладаються на передові алгоритми та комп'ютерні програми для аналізу ринкових даних, визначення торгових можливостей та виконання угод. Давайте детальніше розглянемо ключові компоненти та процеси, що беруть участь у функціонуванні автоматизованого трейдингу:

1. Аналіз даних. Автоматизовані трейдингові системи збирають величезні обсяги ринкових даних, включаючи історичні дані про ціни та обсяги, ринкові стрічки в режимі реального часу, новини та іншу відповідну інформацію. Ці системи використовують складні математичні та статистичні моделі для аналізу цих даних та виявлення потенційних торговельних можливостей.

2. Трейдингові стратегії. Трейдери можуть розробляти або налаштовувати трейдингові стратегії на основі своїх уподобань та цілей. Ці стратегії визначають правила та умови для входу та виходу з торгів, а також параметри управління ризиками, такі як ордери стоп-лосс та рівні тейк-профіту. Стратегії можуть охоплювати широкий спектр факторів, включаючи технічні індикатори, аналіз трендів та фундаментальний аналіз.

3. Автоматизоване розміщення ордерів. Після виявлення торгової можливості автоматизована торгова система генерує та розміщує ордери

відповідно до попередньо визначеної торгової стратегії. Ці ордери надсилаються безпосередньо на торгову платформу або біржу, що усуває необхідність ручного розміщення ордерів.

4. Виконання ордерів. Автоматизована торгова система постійно стежить за ринком і виконує угоди на основі заздалегідь визначених правил. Вона може виконувати кілька угод одночасно на різних ринках і таймфреймах, використовуючи можливості, які можуть виникнути протягом мілісекунд.

5. Управління ризиками. Ефективне управління ризиками є ключовим аспектом автоматизованого трейдингу. Трейдери можуть впроваджувати різні функції управління ризиками, такі як встановлення максимальних розмірів позицій, застосування стоп-лосів для обмеження потенційних збитків та використання стоп-ордерів для захисту прибутку.

6. Моніторинг та оптимізація. Автоматизовані торгові системи постійно моніторять та аналізують ринкові умови та результати виконаних угод. Трейдери можуть використовувати ці дані для оптимізації своїх стратегій, внесення необхідних коректив та вдосконалення свого підходу для досягнення кращих торгових результатів.

Спрощена схема роботи автоматизованого трейдингу показана на рисунку 1.1. На основі наданого рисунку можна виділити такі ключові принципи автоматизованого трейдингу.

1. Розробка торгової стратегії:

- Трейдер або розробник визначає правила трейдингу: коли купувати, коли продавати, які індикатори використовувати.
- Стратегія може базуватися на технічному аналізі, математичних моделях або машинному навчанні.

2. Отримання ринкових даних:

Біржа є джерелом котирувань і фінансових даних.

- Автоматизовані системи отримують інформацію про ціни, обсяги трейдингу та інші параметри.
- Дані надходять у реальному часі через API торгових платформ.

Рисунок 1.1 – Схема роботи автоматизованого трейдингу

3. Автоматичне виконання угод:

- Алгоритм аналізує ринкові умови та виконує угоди без втручання людини.
- Використовуються механізми управління ризиками, такі як стоп-лоси та тейк-профіти.

4. Аналіз результатів трейдингу:

- Система оцінює ефективність угод, коригує стратегію та адаптується до змін ринку.

- Використовується бек-тестинг для перевірки алгоритму на історичних даних.

Цей рисунок наочно демонструє, як працює автоматизований трейдинг: від розробки стратегії до виконання угод і аналізу результатів.

Важливо зазначити, що хоча автоматизовані торгові системи працюють незалежно, трейдери все ще відіграють життєво важливу роль у їхній ефективності. Трейдери відповідають за розробку, тестування та вдосконалення торгових стратегій, що використовуються автоматизованою системою. Вони також повинні контролювати продуктивність системи, забезпечувати належне управління ризиками та бути в курсі ринкових умов, які можуть вимагати коригування торгової стратегії.

1.3 Переваги використання автоматизованих систем трейдингу

Тепер, коли ми дослідили, як працює автоматизований трейдинг, проаналізуємо переваги, які вона пропонує трейдерам.

Автоматизований трейдинг пропонує трейдерам численні переваги, надаючи їм можливості, які можуть значно покращити торговельну діяльність. Розглянемо деякі ключові переваги використання автоматизованих торгових систем:

1. Швидкість та ефективність. Автоматизовані торгові системи можуть виконувати угоди з блискавичною швидкістю, що значно перевершує можливості трейдерів-людей. Вони можуть миттєво аналізувати ринкові умови, виявляти можливості та виконувати угоди за частки секунди. Ця швидкість та ефективність допомагають трейдерам отримувати вигоду навіть від найменших коливань ринку та скористатися можливостями високочастотної трейдингу.

2. Усунення емоцій. Такі емоції, як страх і жадібність, часто можуть затьмарювати судження та призводити до ірраціональних торгових рішень. Автоматизовані торгові системи працюють на основі заздалегідь визначених правил та алгоритмів, усуваючи вплив емоцій з торгового процесу. Це забезпечує

дисциплінований та об'єктивний підхід до трейдингу, що призводить до більш послідовного та раціонального прийняття рішень.

3. Цілодобовий моніторинг ринку. На відміну від трейдерів-людей, автоматизовані торгові системи можуть працювати цілодобово, моніторячи ринки на предмет потенційних можливостей та виконуючи угоди навіть тоді, коли трейдери сплять або не можуть активно торгувати. Такий постійний моніторинг ринку усуває ризик втрати вигідних торгових можливостей через обмеження в часі.

4. Тестування на історичних даних та оптимізація. Автоматизовані торгові системи дозволяють трейдерам проводити тестування своїх торгових стратегій на історичних даних, використовуючи історичні ринкові дані. Це дає трейдерам змогу оцінювати ефективність своїх стратегій за різних ринкових умов та часових рамок. Крім того, трейдери можуть оптимізувати свої стратегії на основі результатів тестування на історичних даних , вносячи необхідні корективи для покращення продуктивності системи.

5. Покращене виконання ордерів. Автоматизовані торгові системи можуть виконувати угоди за найкращими можливими цінами завдяки своїй здатності миттєво обробляти величезні обсяги ринкових даних. Це може допомогти мінімізувати прослизання та забезпечити виконання угод з мінімальною затримкою, покращуючи загальну ефективність трейдингу.

6. Диверсифікація. Автоматизовані торгові системи можуть торгувати на кількох ринках та інструментах одночасно. Це дозволяє трейдерам диверсифікувати свою торговельну діяльність та зменшити ризик залежності від одного ринку чи інструменту. Диверсифікація може допомогти розподілити ризики та потенційно покращити загальну ефективність портфеля.

Використовуючи можливості автоматизації, трейдери можуть використовувати ці переваги для отримання конкурентної переваги на ринку. Однак важливо визнати, що автоматизований трейдинг має свій власний набір ризиків і викликів, які ми обговоримо в наступному розділі.

1.4 Ризики та виклики автоматизованого трейдингу

Хоча автоматизований трейдинг пропонує значні переваги, трейдерам важливо усвідомлювати ризики та проблеми, пов'язані з цим підходом. Розуміння цих ризиків може допомогти трейдерам приймати обґрунтовані рішення та впроваджувати стратегії для пом'якшення потенційних пасток. Давайте розглянемо деякі ключові ризики та проблеми автоматизованого трейдингу:

1. Технічні збої. Автоматизовані торгові системи залежать від технологічної інфраструктури, включаючи комп'ютерне обладнання, програмне забезпечення та підключення до Інтернету. Будь-які технічні збої або перебої можуть призвести до збоїв у роботі системи та призвести до пропущених угод або помилкових виконань. Трейдери повинні мати резервні системи та регулярно контролювати продуктивність і надійність своєї автоматизованої торгової системи.

2. Складність та системні помилки. Розробка та впровадження успішної автоматизованої торгової системи вимагає досвіду в програмуванні та розробці алгоритмів. Помилки в коді або неправильне впровадження торгових стратегій можуть призвести до непередбачуваних наслідків та фінансових втрат. Трейдери повинні ретельно протестувати свої системи та забезпечити належне управління ризиками.

3. Волатильність ринку та події типу "чорного лебедя". Автоматизовані торгові системи можуть бути чутливими до волатильності ринку та екстремальних подій, які можуть відрізнятися від історичних даних або звичайних торгових умов. Різкі коливання цін, раптові новини або маніпуляції ринком можуть вплинути на продуктивність автоматизованих торгових систем і призвести до неочікуваних результатів. Трейдерам слід впроваджувати заходи управління ризиками для подолання невизначеності ринку.

4. Надмірна оптимізація та підгонка кривої. Трейдери можуть спокуситися надмірно оптимізувати свої трейдингові стратегії на основі минулих

показників, що призводить до підгонки кривої. Підгонка кривої відбувається, коли стратегія занадто специфічна для історичних даних, що робить її менш ефективною в умовах ринку в режимі реального часу. Трейдерам слід знаходити баланс між оптимізацією своїх стратегій та забезпеченням їхньої адаптивності до змін у ринковому середовищі.

5. Залежність від ринкових даних. Автоматизовані трейдингові системи значною мірою залежать від точних та надійних ринкових даних. Неточні або затримані дані можуть призвести до помилкового аналізу та неправильних торговельних рішень. Трейдери повинні мати доступ до високоякісних потоків даних та встановити процедури для перевірки цілісності даних, що використовуються їхніми системами.

6. Нормативні та правові аспекти. Трейдери, які займаються автоматизованим трейдингом, повинні дотримуватися відповідних регуляторних вимог та юридичних зобов'язань. Різні юрисдикції мають конкретні правила та положення щодо алгоритмічного трейдингу, включаючи контроль ризиків, зобов'язання щодо звітності та запобігання маніпуляціям на ринку. Трейдери повинні бути в курсі регуляторного ландшафту та забезпечувати відповідність своєї автоматизованої торговельної діяльності чинному законодавству.

Розуміючи та керуючи цими ризиками та викликами, трейдери можуть ефективно орієнтуватися у світі автоматизованого трейдингу та максимізувати її потенційні переваги. У наступному розділі ми розглянемо деякі популярні стратегії автоматизованого трейдингу, які трейдери можуть розглянути.

1.5 Популярні автоматизовані торгові стратегії

Автоматизовані торгові системи можуть використовувати широкий спектр стратегій, залежно від цілей трейдера, толерантності до ризику та ринкових умов. Ці стратегії зазвичай реалізуються за допомогою комбінації технічних індикаторів, математичних моделей та статистичного аналізу. Давайте розглянемо деякі популярні автоматизовані торгові стратегії:

1. Трейдинг на основі імпульсу (Momentum Trading). Стратегії трейдингу на основі імпульсу прагнуть скористатися тенденціями та ціновими рухами на ринку. Ці стратегії спрямовані на виявлення активів, які демонструють значний висхідний або низхідний імпульс, та укладання угод у напрямку переважаючого тренду.

2. Повернення до середнього значення (Mean Reversion). Стратегії повернення до середнього значення функціонують виходячи з того, що ціни зрештою повернуться до свого середнього значення після відхилення від нього. Ці стратегії використовують умови перекупленості або перепроданості, укладаючи угоди проти переважаючого тренду.

3. Трейдинг на прориві (Breakout Trading). Стратегії прориву спрямовані на визначення ключових рівнів підтримки та опору та використання цінових проривів за межі цих рівнів. Коли ціна пробиває рівень опору або опускається нижче рівня підтримки, автоматизована торгова система може увійти в угоду в напрямку прориву.

4. Арбітраж. Арбітражні стратегії передбачають використання цінових розбіжностей між двома або більше ринками чи фінансовими інструментами. Ці стратегії спрямовані на отримання прибутку від тимчасових цінових дисбалансів шляхом одночасної купівлі та продажу активів, щоб скористатися ціновою різницею.

5. Парний трейдинг. Парний трейдинг передбачає одночасну торгівлю двома корельованими активами. Стратегія спрямована на визначення пар активів, які мають історично стабільне співвідношення цін. Коли співвідношення цін відхиляється від своєї історичної норми, автоматизована торгова система може укладати угоди, щоб скористатися очікуваною конвергенцією цін.

Це лише кілька прикладів багатьох автоматизованих торгових стратегій, які можуть впроваджувати трейдери. Кожна стратегія має свої сильні та слабкі сторони, і для трейдерів важливо ретельно протестувати та оцінити ефективність стратегії, перш ніж використовувати її в реальному трейдингу. Крім того,

трейдери також можуть комбінувати кілька стратегій або налаштовувати існуючі стратегії, щоб створити свій унікальний торговий підхід.

Варто зазначити, що ефективність торгової стратегії може змінюватися залежно від ринкових умов та інших факторів. Трейдери повинні регулярно контролювати та коригувати свої стратегії за потреби, щоб адаптуватися до змінної динаміки ринку та оптимізувати продуктивність.

Тепер, коли ми розглянули популярні стратегії автоматизованого трейдингу, перейдемо до обговорення важливих факторів, які слід враховувати під час вибору автоматизованої торгової системи.

1.6 Важливі фактори при виборі автоматизованої системи трейдингу

Вибираючи автоматизовану систему трейдингу, важливо враховувати кілька факторів, які можуть вплинути на її продуктивність, надійність та придатність для трейдингових потреб. Розглянемо деякі важливі міркування щодо вибору автоматизованої системи трейдингу.

1. Стратегія та налаштування. Варто розглянути трейдингові стратегії, що пропонуються системою, та переконатися, що вони відповідають торговим цілям та толерантності до ризику. Слід шукати системи, які пропонують налаштування, що дозволить змінювати або створювати власні трейдингові стратегії за потреби.

2. Тестування на історичних даних (бек-тест) та оптимізація. Здатність тестувати на історичних даних та оптимізувати трейдингові стратегії є надзвичайно важливою. Треба переконатися, що система забезпечує надійні можливості тестування на історичних даних та дозволяє точно налаштовувати свої стратегії на основі результатів.

3. Дані у режимі реального часу та зв'язок. Доступ до своєчасних та точних ринкових даних є життєво важливим для автоматизованого трейдингу. Слід перевірити, чи має система надійне з'єднання з провайдером даних та біржами, щоб забезпечити отримання актуальної інформації для точного прийняття та виконання рішень.

4. Функції управління ризиками. Слід оцінити інструменти та функції управління ризиками, що пропонуються системою. Варто звертати увагу на такі функції, як стоп-лосс-ордери, трейлінг-стопи та опції визначення розміру позиції, які можуть допомогти керувати ризиками та захистити капітал.

5. Технічна підтримка та документація. Заслуговує на увагу рівень технічної підтримки, що надається розробниками або постачальниками системи. Слід переконатися, що наявна доступна документація, навчальні посібники та оперативна служба підтримки клієнтів, яка допоможе вирішити будь-які питання чи проблеми, що можуть виникнути.

6. Безпека та надійність. Автоматизовані торгові системи обробляють конфіденційні торгові дані та виконують угоди від імені користувача. Треба перевірити заходи безпеки, що надаються системою для захисту даних, та переконатися, що система має послужний список надійності та безвідмовної роботи.

7. Витрати та комісії. Потрібно оцінити структуру ціноутворення та комісії, пов'язані з системою. Враховуються такі фактори, як початкові витрати, поточні абонентські плати, комісійні збори за угоду та будь-які додаткові платежі за канали даних або додаткові функції.

8. Інтеграція з брокерською компанією. Якщо у наявна бажана брокерська або торгова платформа, слід перевірити, чи сумісна автоматизована торгова система та чи може вона безперешкодно інтегруватися. Це дозволяє ефективно виконувати угоди та оптимізувати управління рахунками.

Перш ніж приймати рішення, вкрай важливо ретельно дослідити та оцінити різні автоматизовані торгові системи. Доцільним буде розглянути демо-рахунки або пробні періоди, що пропонуються постачальниками систем, щоб оцінити продуктивність системи та її придатність для стилю трейдингу конкретного клієнта.

Завжди слід пам'ятати, що автоматизована торгова система – це інструмент, який має допомагати та доповнювати торговельну діяльність. Перш

ніж вибирати автоматизовану торгову систему, важливо чітко розуміти свої торгові цілі, схильність до ризику та торговельну стратегію.

Тепер, коли ми розглянули фактори, які слід враховувати при виборі автоматизованої торгової системи, перейдемо до обговорення того, як розпочати роботу з автоматизованим трейдингом.

1.7 Як розпочати роботу з автоматизованим трейдингом

Початок шляху автоматизованого трейдингу вимагає ретельного планування та виконання. Ось кілька ключових кроків, які допоможуть розпочати таку діяльність.

1. Користувачеві слід весь час навчатися – отримувати ґрунтовне розуміння того, як працює автоматизований трейдинг, включаючи різні стратегії, технічні індикатори, методи управління ризиками та динаміку ринку. Корисним буде приділити час вивченню програмування, розробки алгоритмів та інструментів, що використовуються в автоматизованому трейдингові.

2. Визначення своїх торгових цілей. Потрібно чітко визначити свої торгові цілі, толерантність до ризику та бажаний стиль трейдингу. Визначити ринки, на яких користувач хочете торгувати, та часові рамки, на яких він хоче зосередитися. Ця ясність допоможе вибрати відповідні автоматизовані трейдингові системи та стратегії, що відповідають визначеним цілям.

3. Вибір автоматизованої торгової платформи. Слід обрати надійну та авторитетну автоматизовану торгову платформу, яка пропонує необхідні функції та можливості. Враховуються тут такі фактори, як простота використання, варіанти налаштування, можливості тестування на попередніх даних та сумісність з бажаною брокерською або терйдинговою платформою.

4. Розробка чи налаштування своєї торговельної стратегії. Великим позитивом буде створення або налаштування торговельної стратегії, яка відповідає цілям та толерантності до ризику. Можна використовувати інструменти та функції, що надаються обраною автоматизованою трейдинговою платформою, для впровадження та тестування власної стратегії. Доцільно

розглянути можливість ретельного тестування на попередніх даних, щоб оцінити ефективність стратегії за різних ринкових умов.

5. Оптимізація та управління ризиками. Варто здійснювати вдосконалення та оптимізацію своєї стратегії на основі результатів тестування на історичних даних. Слід впроваджувати методи управління ризиками, такі як встановлення стоп-лосс-ордерів та розмірів позицій, для контролю ризиків та захисту капіталу. Потрібно постійно відстежувати та коригувати свою стратегію відповідно до зміни ринкових умов.

6. Трейдинг в реальному часі з невеликими інвестиціями. Після розробки та тестування власної стратегії слід почати торгувати в реальному часі з невеликими інвестиціями або використовуючи демо-рахунок. Одночасно потрібно слідкувати за продуктивністю автоматизованої торгової системи та аналізувати результати. На цій основі вносити необхідні корективи та вдосконалення, набуваючи досвіду та навчаючись на реальних результатах трейдингу.

7. Безперервне навчання та адаптація. Потрібно бути в курсі ринкових тенденцій, новин та подій, які впливають на обрані ринки. Необхідним є постійне вивчайте нових стратегій та методів, які можуть покращити продуктивність автоматизованої трейдингової системи. Свій підхід має бути вдосконалений та адаптований на основі змін ринкових умов.

8. Моніторинг та оцінка ефективності. Регулярний контроль ефективності автоматизованої трейдингової системи є необхідним процесом кожного користувача. Слід оцінювати її ефективність на основі попередньо визначених показників, таких як прибутковість, просідання та скоригована на ризик прибутковість. Ці дані використовуються для прийняття обґрунтованих рішень та постійної оптимізації стратегії.

9. Звернення за професійною порадою в разі потреби. Якщо користувач – новачок в автоматизованому трейдингові або йому потрібна допомога, потрібно звернутися за порадою до фахівців, таких як фінансові

консультанти або досвідчені трейдери. Вони можуть надати цінну інформацію та допомогти розібратися в складнощах автоматизованого трейдингу.

Успіх в автоматизованому трейдингові вимагає як технічних знань, так і постійного навчання.

2 ОГЛЯД СИСТЕМ АВТОМАТИЗОВАНОГО ТРЕЙДИНГУ

2.1 Загальний огляд програмного забезпечення для алгоритмічного трейдингу

Програмісти відіграють дедалі більшу роль в процесах трейдингу на різних біржах, і не без підстав. Розробники можуть створювати та впроваджувати алгоритмічні торгові стратегії, які враховують величезні обсяги даних, аналізують їх у режимі реального часу та виконують угоди на основі попередньо розроблених правил і формул.

Після того, як знайдено життєздатну торговельну стратегію, вона працює в автопілоті, зазвичай з дуже мінімальним ручним втручанням. Найкраще в цьому те, що людині не потрібна робота на умовній Уолл-стріт, щоб створити свій власний трейдинговий бізнес.

Якщо користувач автоматизованої системи трейдингу має навички програмування, який хоче створити власну стратегію алгоритмічного трейдингу або просто збирати дані та проводити тестування на попередніх даних, то для такого клієнта є найкращі платформи, які він може використовувати для свого робочого процесу. А якщо користувач не програміст, то для нього також є варіанти підбору систем трейдингу.

В таблиці 2.1 наведено перелік деяких систем автоматизованого трейдингу, про котрих вдалось знайти найбільше інформації. Більшість зазначених цін стосуються місячних планів, але багато з цих трекерів також пропонують (зі знижкою) річні ціни.

Кожен із цих інструментів пропонує деякі свої функції безкоштовно, але інші платні.

Проаналізуємо тепер детальніше наведені в таблиці 1.1 системи.

2.2 TradeStation – найкраще безкоштовне програмне забезпечення для алгоритмічного трейдингу

TradeStation – це брокерська система трейдингу, створена власне для трейдингу на біржах. Використовуючи цей програмний продукт, можна торгувати акціями, ETF, опціони, ф'ючерси та криптовалюти. Це швидко, недорого ($0 за угоду акціями, $0,60 за угоду опціонами) та пропонує потужний набір технологій.

Таблиця 2.1 – Характеристики систем автоматизації трейдингу

Назва системи автоматизованого трейдингу	**Найкраще застосування**	**Вартість**
TradeStation	Найкращий загалом	Безкоштовно
Stock Market Guides	Найкраще для тих, хто не є програмістом	від $19 /місяць
QuantConnect	Найкраща платформа з відкритим кодом	від $60 /місяць
Interactive Brokers	Друге місце в номінації "Найкращий алгоритмічний брокер"	Безкоштовно
NinjaTrader	Найкраще для ф'ючерсів	Безкоштовно
Mindful Trader	Найкращий сервіс торгових сповіщень на основі алгоритмів	$47 /місяць
Ваша власна інфраструктура	Найкраще для повного налаштування та контролю	Варіюється

Декілька слів про ETF (Exchange-Traded Fund) – це біржовий інвестиційний фонд, який торгується на біржі, як звичайні дії. У контексті трейдингу ETF можуть бути використані для автоматизованих стратегій, оскільки вони забезпечують ліквідність, диверсифікацію та доступ до різних класів активів.

Основні характеристики ETF у трейдингу:

- Диверсифікація – ETF створює набір активів, таких як дії, зобов'язання або сировинні товари, що знижує ризик.
- Ліквідність – великий популярний ETF має високу ліквідність, що дозволяє швидко відмовитися від угоди.
- Низькі витрати – замість традиційних інвестиційних фондів, ETF мають нижчі комісії.
- Алгоритмічний трейдинг – багато торгових ботів та скриптів інтегрують ETF у ваші стратегії для оптимізації прибутковості.
- Трейдери часто використовують ETF для хеджування ризиків або отримання експозиції до певного сектора ринку.

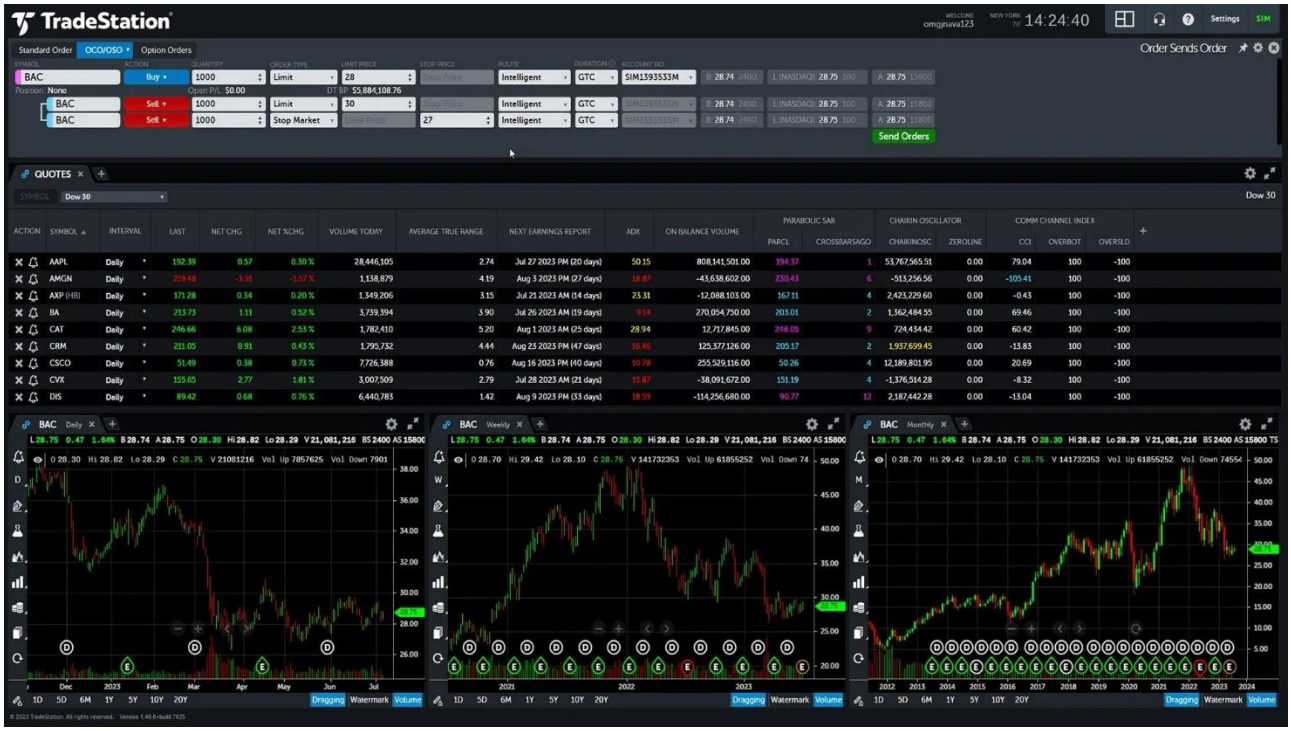


Рисунок 2.1 – Інтерфейс TradeStation

Крім того, програмісти можуть використовувати ту саму технологію, яку TradeStation використовує у власних додатках.

Ось перелік властивостей, що TradeStation пропонує програмістам.

- Комплексний API. Його API підтримує дані в режимі реального часу, виконання замовлень та управління обліковими записами. Він інтегрується з

кількома мовами програмування, включаючи C#, C++, Python, Ruby, PHP та EasyLanguage.

- EasyLanguage. Його власна мова програмування EasyLanguage розроблена спеціально для торгових стратегій та індикаторів. Вона зручна для початківців (простіша за Python та має вбудовані індикатори та терміни, пов'язані з акціями), але достатньо потужна для більш досвідчених трейдерів та програмістів.
- Побудова стратегії та тестування на історичних даних (бек-тест). Конструктор стратегій та бек-тест є основними інструментами для створення, тестування та оптимізації стратегій. Конструктор стратегій дозволяє створювати стратегії як візуально, так і на основі коду, тоді як бек-тест допомагає вдосконалити алгоритми, використовуючи історичні дані, перед початком трейдингу на реальних даних.
- Трейдинг на папері. TradeStation пропонує симульоване торгове середовище для тестування стратегій з даними в режимі реального часу, без ризику вкладеним капіталом.
- Розширені ринкові дані. Платформа надає доступ до високоякісних ринкових даних у режимі реального часу та за минулі періоди, що є важливим для тестування та оптимізації алгоритмів.
- Налаштування та гнучкість. Підтримує високо настроювані торговельні стратегії та має можливість автоматизувати майже кожен аспект торговельних операцій.
- Хмарні та хостингові рішення. Варіанти хмарного хостингу для цілодобової роботи алгоритму, що зменшує потребу в обслуговуванні локальної інфраструктури.
- Освітні ресурси. Доступ до вебінарів, навчальних посібників та інших навчальних матеріалів, які допомагають користувачам зрозуміти платформу та її інструменти.

Головним недоліком використання TradeStation для алгоритмічного трейдингу є те, що ви можете застосовувати свій алгоритм лише до відкритих

графіків – якщо ви хочете застосувати алгоритм, наприклад, до акцій Apple (AAPL), вам потрібно відкрити графік Apple.

Це може бути проблемою для трейдерів, які хочуть застосувати свій алгоритм до понад 100 акцій одночасно.

Коротше кажучи, TradeStation має все необхідне для створення, тестування та виконання власних алгоритмічних торгових стратегій як у демо-, так і в реальному середовищі, що робить її найкращою платформою для алгоритмічних трейдерів.

2.3 Stock Market Guides як рішення для користувачів – непрограмістів

Stock Market Guides використовують сканер на основі алгоритмів, який шукає налаштування процесу трейдингу в режимі реального часу.

Алгоритм був створений шляхом проведення тисяч тестів на ретроспективному рівні з використанням широкого спектру графіків, свічкових моделей та фондових індикаторів.

Потім алгоритм було налаштовано на фільтрацію всіх комбінацій акцій та патернів, окрім найприбутковіших (наприклад, TSLA).

Вигляд цієї технології показано на рисунку 2.2.

За допомогою Stock Market Guides можна використовувати алгоритм, розроблений професійними трейдерами, без необхідності самостійно програмувати.

Сам сканер безкоштовний у використанні, але результати тестування на історичних даних обмежені платним доступом.

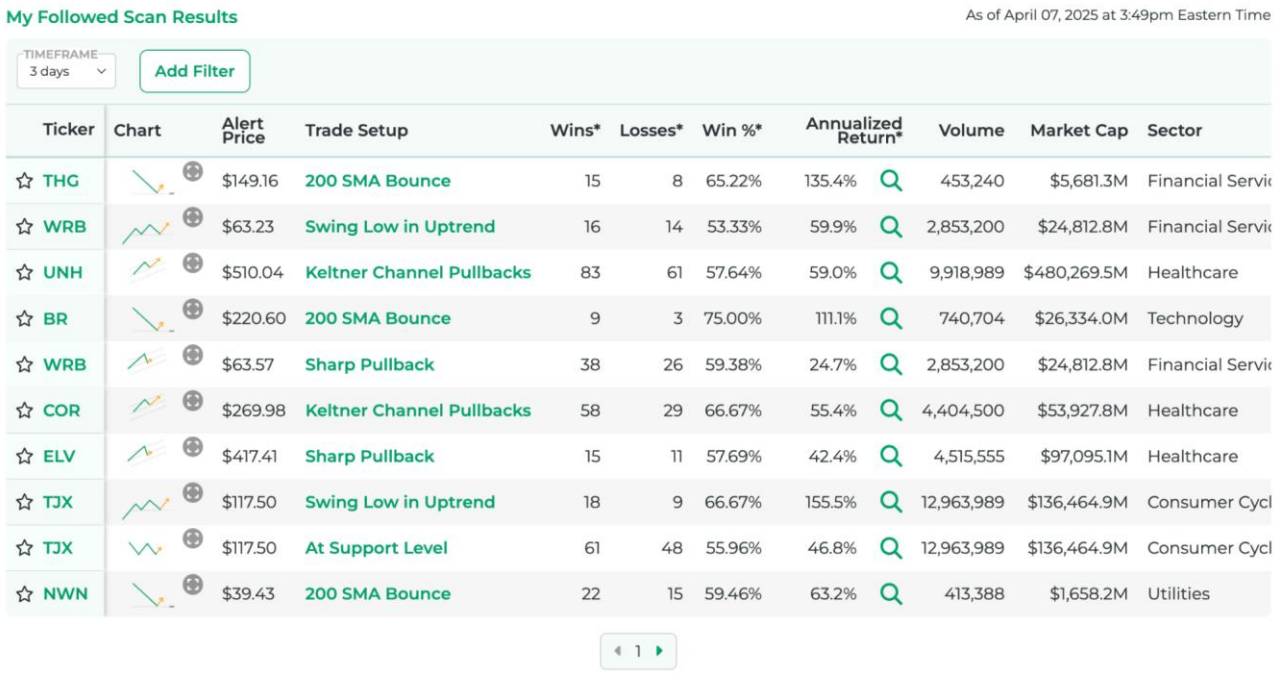


Рисунок 2.2 – Інтерфейс користувача системи Stock Market Guides

Існує два рівні сервісу, а також вибір між довгостроковим інвестуванням, свінг-трейдингом та трейдингом опціонами – все за різними ціновими категоріями:

а) Сканер (самостійне дослідження):
- інвестування в акції: $19/місяць;
- свінг-трейдинг: $39/місяць;
- трейдинг опціонами: $59/місяць.

б) Вибір (сповіщення про готові угоди):
- інвестування в акції: $29/місяць;
- свінги: $49/місяць;
- опціони: $69/місяць.

Сервіси свінгів та опціонів є найпопулярнішими в кожній з категорій, тому варто вибрати один з них залежно від того, яким активом користувач хоче торгувати. Можете переглянути всі сервіси та дізнатися більше, ознайомившись з документацією цієї системи.

2.4 QuantConnect – платформа для алгоритмічного трейдингу з відкритим кодом

QuantConnect – це хмарна платформа з відкритим кодом, розроблена спеціально для алгоритмічних трейдерів та квантових трейдерів.

Він надає повний набір інструментів для побудови стратегій, проведення тестів на минулих даних та виконання угод (через зв'язки з брокерськими компаніями, такими як TradeStation, Interactive Brokers, TD Ameritrade, Coinbase та Binance) – все на одній платформі (див. рис. 2.3).

Рисунок 2.3 – Вигляд платформи QuantConnect

Платформа надає доступ до історичних та реальних ринкових даних, а також до мільйонів точок альтернативних даних, що може забезпечити реальну перевагу в розробці унікальної стратегії.

Його LEAN-фреймворк – це рушій, на якому працює QuantConnect, і який дозволяє розробникам створювати свої стратегії за допомогою C# та Python.

QuantConnect забезпечує масштабованість та надійність, що робить її чудовим вибором для трейдерів, яким потрібно безперервно запускати алгоритми, не турбуючись про обслуговування локального сервера.

Хоча QuantConnect є комплексним рішенням для свінг-трейдерів, його інтерфейс користувача та система ордерів дещо незручні для внутрішньоденних таймфреймів.

Якщо ви плануєте денну торгівлю, і особливо якщо плануєте здійснювати понад 10 угод на день, вам може знадобитися спеціальний фреймворк.

Крім того, активна спільнота QuantConnect (понад 367 000 алгоритмічних трейдерів), обширна документація та освітні ресурси допоможуть розробити та вдосконалити власні стратегії.

2.5 Interactive Brokers

Interactive Brokers (IBKR) – це ще один зручний для трейдерів брокерський сервіс, відомий своїми низькими витратами, потужними інструментами та широким спектром інвестиційних активів.

Торговані цінні папери включають понад 47000 акцій, понад 6000 ETF, понад 7000 опціонів, понад 13000 ф'ючерсних контрактів, понад 2 мільйони облігацій тощо, що охоплюють як ринки США, так і міжнародні ринки. Вигляд інтерфейсу Interactive Brokers показано на рисунку 2.4.

Він також пропонує гнучкі API та обширні ринкові дані, що є ідеальним поєднанням для програмістів, які прагнуть створити власні стратегії на основі алгоритмів.

IBKR – це складніша платформа, ніж TradeStation, і вона краще підходить для більш досвідчених трейдерів та більш досвідчених програмістів.

Interactive Brokers надає бібліотеку навчальних матеріалів про свої API, включаючи курси в Академії трейдерів та блозі Quant.

Рисунок 2.4 – Інтерфейс користувача Interactive Brokers

Тим не менш, багато алгоритмічних трейдерів хочуть отримати доступ до якомога більшої кількості ринків, а глобальний доступ IBKR до ринку – 160 ринків у 36 країнах – не має собі рівних.

2.6 NinjaTrader – система для трейдингу ф'ючерсами

NinjaTrader є популярним брокером серед ф'ючерсних та форекс-трейдерів завдяки низьким комісіям та можливостям побудови графіків.

Програмісти можуть створювати, тестувати та розгортати алгоритмічні торговельні стратегії за допомогою торгового фреймворку на основі C#. Вигляд інтерфейсу системи показано на рисунку 2.5.

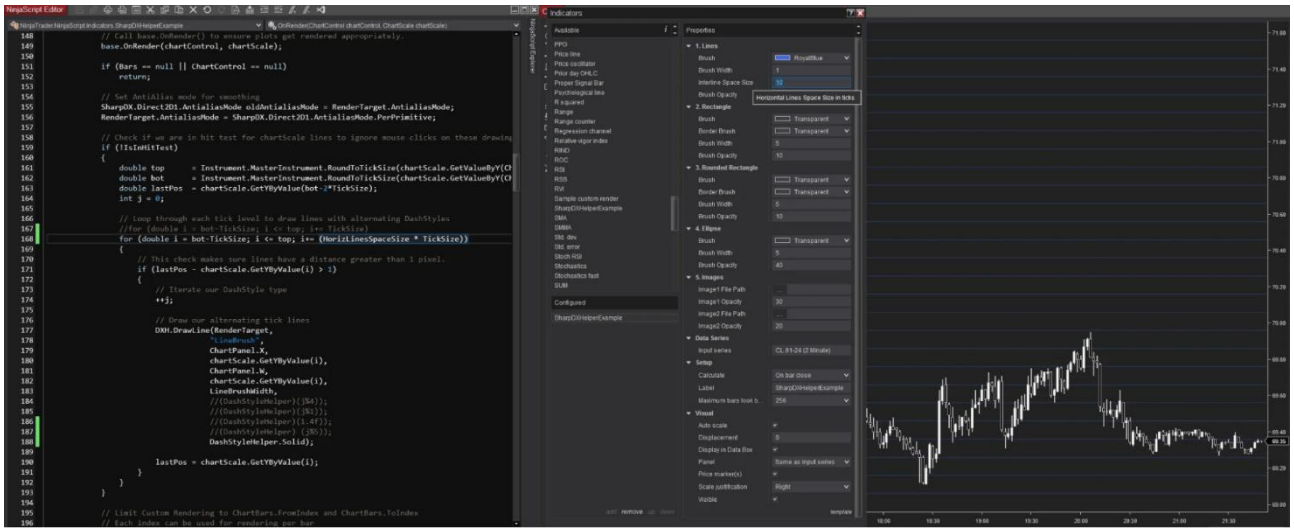


Рисунок 2.5 – інтерфейсу системи NinjaTrader

Фреймворк надає розробникам низькорівневий доступ до:

- залишки та позиції;
- ордери та виконання;
- дані в режимі реального часу та історичні дані;
- користувацькі інтерфейси;
- елементи керування;
- індикатори.

Непрограмісти також можуть створювати власні автоматизовані торгові стратегії, використовуючи конструкцію платформи "вкажи та клацни", хоча їхні можливості налаштування будуть обмежені.

Функції аналізу стратегій та повтору ринку NinjaTrader можуть допомогти протестувати та оптимізувати власні стратегії, вдосконалюючи їх перед реальним розгортанням.

Загалом, NinjaTrader пропонує збалансоване поєднання зручності використання та гнучкості, що робить його надійним варіантом для ф'ючерсних трейдерів усіх рівнів.

2.7 Mindful – сервіс торгових сповіщень на основі алгоритмів

Mindful Trader – це сервіс свінг-трейдингу (акціями та опціонами) на основі алгоритмів, які створив Ерік Фергюсон.

Фергюсон, студент факультету економіки зі Стенфорда, розробив свої торгові стратегії, проводячи сотні тисяч тестів на минулих даних протягом чотирьох років.

Усі його угоди базуються на суворих правилах трейдингу. Його входи та виходи з ринку визначені заздалегідь, і він не сумнівається в результатах угоди, коли його алгоритм надсилає йому сповіщення. Вигляд цієї системи показано на рисунку 2.6.

Triggered Date	Triggered Time	Strategy	Ticker	Share Count Calculation	Target Entry	Profit Target	Stoploss	Stock Position Value	Close-By	Earnings Date	Dividends Date
04/07/2025	07:43:39	Double Down	ED	44	105.64	110.13	101.15	$4648	04/15/2025	N/A	N/A
04/07/2025	06:51:58	Double Down	MNST	92	56.14	58.30	53.97	$5164	04/15/2025	N/A	N/A
04/07/2025	06:51:52	Double Down	MCK	6	652.19	683.34	621.04	$3913	04/15/2025	N/A	N/A
04/07/2025	06:51:27	Double Down	WRB	47	63.41	67.65	59.18	$2980	04/15/2025	N/A	N/A
04/07/2025	06:30:01	Double Down	HDB	76	62.75	65.38	60.12	$4769	04/15/2025	N/A	N/A
04/07/2025	06:30:01	Main Pullback	HDB	76	64.06	65.38	61.44	$4868	04/15/2025	N/A	N/A
04/07/2025	06:30:01	Main Pullback	MCK	6	667.77	683.34	636.62	$4006	04/15/2025	N/A	N/A
04/07/2025	06:30:00	Main Pullback	ED	44	107.89	110.13	103.39	$4747	04/15/2025	N/A	N/A
04/04/2025	12:59:35	Double Down	WCN	32	187.98	194.08	181.89	$6015	04/14/2025	N/A	N/A
04/04/2025	12:54:21	Double Down	VRSN	22	241.17	249.94	232.40	$5305	04/14/2025	N/A	N/A

View More

Рисунок 2.6 – Система Mindful Trader

Кожне сповіщення автоматично публікується на веб-сайті та надсилається електронною поштою, тож користувач отримує сповіщення одночасно з ним.

Дотримуючись рекомендацій Е. Фергюсона (приблизно 1–3 на день), користувач можете скористатися перевагами підходу до трейдингу на основі даних, не писавши жодного коду, не тестуючи жодних стратегій чи не виконуючи жодних тестів на минулих даних.

2.8 Власна інфраструктура – найкращий варіант для повного налаштування та контролю

Багато програмістів вирішують створювати власну інфраструктуру для максимальної налаштування та контролю. Найпоширенішими мовами для цього є Python, C++/C#, Java та Go.

Якщо клієнт хоче піти цим шляхом, то знадобиться комбінація інструментів, які дозволять отримувати доступ до даних, розробляти стратегії, проводити тестування на історичних даних та виконувати реальні ордери.

Ці інструменти показані в таблиці 2.2.

Таблиця 2.2 – Складові елементи системи трейдингу власного створення

Інструмент	**Мета**	**Популярний вибір**
API брокерства	Для доступу до ринкових даних (в режимі реального часу та історичних) та виконання ордерів	TradeStation, IBKR, Alpaca, Tradier
Фреймворк для зворотного тестування	Для моделювання стратегій на історичних даних	QuantConnect, Backtrader, Zipline
Середовище розробки стратегії (IDE)	Для написання, тестування та оптимізації коду	Jupyter Notebooks, PyCharm, Visual Studio Code

Можна також додати інструменти для оптимізації (Hyperopt, Optuna, Talib), розгортання хмарних технологій (AWS, Google Cloud, Azure) та моніторингу (Quantconnect, Backtrader).

Налаштування власної інфраструктури – це складний та трудомісткий процес, тому слід переконатися, що ви готові інвестувати, перш ніж розпочинати цей шлях.

Алгоритмічний трейдинг – це використання комп'ютерів для ідентифікації та виконання угод на основі заздалегідь визначеної комбінації математичних формул, графіків та індикаторів, цінової динаміки, обсягу та інших ринкових даних.

Алгоритмічний трейдинг найчастіше використовується у денній та свінг-трейдингу, але також може застосовуватися довгостроковими інвесторами. Вона застосовується на низці фінансових ринків, включаючи акції, опціони, ф'ючерси, форекс та криптовалюту.

Щоб створити власні стратегії алгоритмічного трейдингу, окрім навичок програмування, вам знадобиться програмне забезпечення або платформа, яка надає можливості тестування на попередніх даних, дані в реальному часі та історичні, опції налаштування та зв'язок з брокером.

Для вибору платформи для алгоритмічного трейдингу слід користуватись такими критеріями.

- Ціна: загальна ціна, співвідношення ціни та якості, середня вартість на місяць та будь-які приховані платежі.
- Функціональність: що дозволяє робити програмне забезпечення, наскільки воно просте у використанні, наскільки воно гнучке та який рівень контролю воно надає.
- Достовірність: якість інформації та даних, а також репутація компанії та бренду.
- Аудиторія: для кого призначений продукт, діапазон використання та застосування, чи дійсно він працює для своєї цільової аудиторії, чи є він найкращим доступним варіантом, та будь-які його обмеження.
- Пропозиції: чи є спеціальна пропозиція для реєстрації (включаючи безкоштовні пробні періоди) або будь-які знижки.

Незалежно від того, чи цінує користувач системи автоматизованого трейдингу налаштування та контроль, чи швидкість та простоту використання, але серед проаналізованих систем є гідні представники для кожного цілі.

Для програмістів найшвидший і найдешевший спосіб розпочати роботу – це скористатися послугами алгоритмічно зручного брокера, такого як TradeStation, Interactive Brokers або NinjaTrader.

QuantConnect та створення власної інфраструктури на замовлення також є популярними варіантами.

Для тих, хто не є програмістом і хоче торгувати алгоритмічно, найкращими варіантами будуть Stock Market Guides та Mindful Trader. Також можна спробувати інструмент NinjaTrader з функцією "вкажи та клацни".

.

3 ПРАКТИЧНА ЧАСТИНА РОБОТИ

3.1 Машинне навчання для трейдингу

Алгоритмічний трейдинг опирається на комп'ютерні програми, які виконують алгоритми для автоматизації деяких або всіх елементів торгової стратегії. Алгоритми – це послідовність кроків або правил, розроблених для досягнення мети. Вони можуть приймати різні форми та сприяють оптимізації протягом усього інвестиційного процесу, від генерування ідей до розподілу активів, виконання угод та управління ризиками.

Машинне навчання (ML) включає алгоритми, які вивчають правила або закономірності з даних для досягнення мети, такої як мінімізація помилки прогнозування. Приклади в цій роботі проілюструють, як алгоритми ML можуть витягувати інформацію з даних для підтримки або автоматизації ключових інвестиційних видів діяльності. Ці види діяльності включають спостереження за ринком та аналіз даних для формування очікувань щодо майбутнього та прийняття рішень щодо розміщення ордерів на купівлю чи продаж, а також управління результуючим портфелем для отримання привабливої прибутковості відносно ризику.

Зрештою, метою активного управління інвестиціями є генерування альфи, яка визначається як дохідність портфеля, що перевищує бенчмарк, що використовується для оцінки. Фундаментальний закон активного управління постулює, що ключем до генерування альфи є наявність точних прогнозів дохідності в поєднанні зі здатністю діяти відповідно до цих прогнозів [9].

Він визначає **інформаційний коефіцієнт** (Information Ratio – IR) для вираження вартості активного управління як відношення різниці в дохідності між портфелем та бенчмарком до волатильності цієї дохідності. Він також апроксимує IR як добуток:

- Інформаційного коефіцієнту (IK), який вимірює якість прогнозу як його рангову кореляцію з результатами.

- Квадратний корінь із широти стратегії**,** виражений як кількість незалежних ставок на ці прогнози.

Конкуренція досвідчених інвесторів на фінансових ринках означає, що для створення точних прогнозів для отримання альфи потрібна краща інформація, або через доступ до кращих даних, або через кращу здатність їх обробляти, або через те й інше. Саме тут на допомогу приходить машинне навчання (ML): застосування ML для трейдингу (ML4T) зазвичай спрямоване на більш ефективне використання швидко диверсифікованого діапазону даних для створення як кращих, так і більш дієвих прогнозів, тим самим покращуючи якість інвестиційних рішень та результатів.

Історично алгоритмічний трейдинг вужче визначалася як автоматизація виконання угод для мінімізації витрат, що пропонуються стороною продажу. Ця робота розглядає це питання більш комплексно, оскільки використання алгоритмів загалом та машинного навчання зокрема почало впливати на ширший спектр діяльності: від генерування ідей та вилучення сигналів з даних до розподілу активів, визначення розміру позицій, тестування та оцінки стратегій.

У цьому розділі розглядаються галузеві тенденції, які призвели до появи машинного навчання (ML) як джерела конкурентної переваги в інвестиційній галузі. Ми також розглянемо, яке місце ML займає в інвестиційному процесі для реалізації стратегій алгоритмічного трейдингу.

3.1.1 Зростання машинного навчання (ML) в інвестиційній галузі

Інвестиційна галузь зазнала кардинальних змін протягом останніх кількох десятиліть і продовжує розвиватися на тлі посилення конкуренції, технологічного прогресу та складного економічного середовища. У цьому розділі розглядаються ключові тенденції, які сформували загальне інвестиційне середовище, а також контекст алгоритмічного трейдингу та використання машинного навчання зокрема.

Тенденції, які призвели до нинішнього зростання алгоритмічного трейдингу та машинного навчання, включають:

- Зміни в мікроструктурі ринку, такі як поширення електронного трейдингу та інтеграція ринків між класами активів та географічними регіонами.
- Розробка інвестиційних стратегій, сформульованих з точки зору впливу факторів ризику, на відміну від класів активів.
- Революції в обчислювальній потужності, генерації та управлінні даними, а також статистичних методах, включаючи прориви в глибокому навчанні.
- Перевага піонерів алгоритмічного трейдингу порівняно з людьми, інвесторами на власний розсуд.

Крім того, фінансові кризи 2001 та 2008 років вплинули на підхід інвесторів до диверсифікації та управління ризиками. Одним із результатів є зростання популярності недорогих пасивних інвестиційних інструментів у формі біржових інвестиційних фондів (ETF). На тлі низької дохідності та низької волатильності після кризи 2008 року, яка спровокувала масштабні придбання активів провідними центральними банками, інвестори, які дбають про витрати, перевели понад 3,5 трильйона доларів з активно керованих пайових фондів у пасивно керовані ETF.

Конкурентний тиск також відображається у зниженні комісій хедж-фондів, які знизилися з традиційних 2% річної комісії за управління та 20% фіксації прибутку до середнього рівня 1,48% та 17,4% відповідно у 2017 році.

3.1.2 Від електронного до високочастотного трейдингу

Електронна торгівля значно розвинулася з точки зору можливостей, обсягів, охоплення класів активів та географічних регіонів з моменту початку мережевого перенаправлення цін до комп'ютерних терміналів у 1960-х роках.

Дохідність, яку забезпечує актив, залежить від невизначеності або ризику, пов'язаного з фінансовими інвестиціями. Інвестиції в акції передбачають,

наприклад, прийняття бізнес-ризику компанії, а інвестиції в облігації передбачають прийняття ризику дефолту.

Оскільки конкретні характеристики ризику прогнозують прибутковість, визначення та прогнозування поведінки цих факторів ризику стає основним пріоритетом під час розробки інвестиційної стратегії. Це дає цінні торгові сигнали та є ключем до вищих результатів активного управління. Розуміння факторів ризику в галузі з часом суттєво змінилося та вплинуло на те, як машинне навчання використовується для алгоритмічного трейдингу.

Фактори, що пояснювали прибутковість понад CAPM, були включені в інвестиційні стилі, які схиляють портфелі на користь одного або кількох факторів, і активи почали мігрувати в портфелі, що базуються на факторах. Фінансова криза 2008 року підкреслила, як маркування класів активів може бути дуже оманливим і створювати хибне відчуття диверсифікації, коли інвестори не враховують ризики, пов'язані з основними факторами, оскільки класи активів одночасно обвалилися.

Протягом останніх кількох десятиліть кількісне факторне інвестування еволюціонувало від простого підходу, що базується на двох або трьох стилях, до багатофакторних розумних або екзотичних бета-продуктів. У 2017 році активи фондів Smart Beta перевищили 1 трильйон доларів, що свідчить про популярність гібридної інвестиційної стратегії, яка поєднує активне та пасивне управління. Фонди Smart Beta використовують пасивну стратегію, але модифікують її відповідно до одного або кількох факторів, таких як здешевлення акцій або їх відбір за виплатою дивідендів, щоб отримувати кращу прибутковість. Це зростання збіглося зі зростанням критики високих комісій, що стягуються традиційними активними менеджерами, а також посиленням контролю за їхньою діяльністю.

Постійне виявлення та успішне прогнозування факторів ризику, які окремо або в поєднанні з іншими факторами ризику суттєво впливають на майбутню прибутковість активів у різних класах активів, є ключовим фактором зростання

машинного навчання (ML) в інвестиційній галузі та буде ключовою темою всієї цієї книги.

Алгоритмічні засоби трейдингу перевершують людей

Досягнення та зростання активів під управлінням (Assets Under Management – AUM) фірм, які очолили алгоритмічну торгівлю, відіграли ключову роль у залученні інтересу інвесторів та подальших зусиллях галузі щодо повторення їхнього успіху.

Систематичні стратегії, що переважно або виключно спираються на алгоритмічне прийняття рішень, були найвідомішими завдяки математику Джеймсу Саймонсу, який заснував Renaissance Technologies у 1982 році та перетворив її на провідну квантову фірму. Її секретний фонд Medallion Fund, закритий для сторонніх осіб, з 1982 року приносив приблизно 35% річної прибутковості.

DE Shaw, Citadel та Two Sigma, три найвідоміші кількісні хедж-фонди, що використовують систематичні стратегії на основі алгоритмів, вперше у 2017 році потрапили до першої десятки найкращих фондів за загальним обсягом зароблених інвесторами коштів після вирахування комісій з моменту заснування.

У 2017 році Morgan Stanley оцінив, що алгоритмічні стратегії зростали на 15% на рік протягом останніх шести років і контролюють близько 1,5 трильйона доларів США між хедж-фондами, взаємними фондами та ETF з інтелектуальними бета-фондами. Інші звіти свідчать про те, що індустрія кількісних хедж-фондів мала перевищити 1 трильйон доларів США під управлінням, майже вдвічі збільшившись з 2010 року на тлі відтоку капіталу з традиційних хедж-фондів. Для порівняння, загальний капітал індустрії хедж-фондів досяг 3,21 трильйона доларів США, згідно з останнім глобальним звітом Hedge Fund Researchи [15].

3.2 Поява квантових фондів

В управлінні активними інвестиціями розвинулися два різних підходи: систематичне (або кількісне) та дискреційне інвестування. Систематичні підходи

спираються на алгоритми для повторюваного та керованого даними підходу для виявлення інвестиційних можливостей для багатьох цінних паперів; навпаки, дискреційний підхід передбачає поглиблений аналіз меншої кількості цінних паперів. Ці два підходи стають все більш схожими, оскільки менеджери застосовують підходи, що більше ґрунтуються на науці про дані.

Навіть фундаментальні трейдери зараз озброєні кількісними методами, що становить 55 мільярдів доларів систематичних активів. Незалежно від конкретних компаній, кількісні фонди торгують моделями та динамікою в широкому спектрі цінних паперів. Дані, зібрані Barclays, показують, що кількісні фонди зараз становлять близько 17% від загальної кількості активів хедж-фондів [16].

Хедж-фонди давно шукають альфу завдяки інформаційній перевазі та здатності виявляти нові некорельовані сигнали. Історично це включало такі речі, як власні опитування покупців або виборців перед виборами чи референдумами. Час від часу використання інсайдерів компаній та експертних мереж для розширення знань про галузеві тенденції чи компанії перетинає межі закону: серія судових переслідувань трейдерів, портфельних менеджерів та аналітиків за використання інсайдерської інформації після 2010 року сколихнула галузь.

На противагу цьому, інформаційна перевага від використання традиційних та альтернативних джерел даних за допомогою машинного навчання пов'язана не з експертними та галузевими мережами чи доступом до корпоративного управління, а радше з можливістю збирати великі обсяги даних та аналізувати їх у режимі реального часу.

Три тенденції революціонізували використання даних в алгоритмічних торгових стратегіях і можуть ще більше змістити інвестиційну галузь від дискреційного до кількісного стилів:

- Експоненціальне збільшення обсягу цифрових даних.
- Збільшення обчислювальної потужності та ємності зберігання даних за менших витрат.

- Досягнення методів машинного навчання для аналізу складних наборів даних.
- Використання машинного навчання для управління ризиками [17].

3.3 Розробка та реалізація стратегії на основі машинного навчання

Машинне навчання (ML) може додавати цінність на кількох етапах життєвого циклу торгової стратегії та спирається на ключову інфраструктуру та ресурси даних. Отже, ця книга має на меті розглянути, як методи ML вписуються в ширший процес проектування, виконання та оцінки стратегій.

Стратегія алгоритмічного трейдингу керується комбінацією альфа-факторів, які перетворюють одне або кілька джерел даних на сигнали, що, у свою чергу, прогнозують майбутню прибутковість активів та запускають ордери на купівлю або продаж. Розділ 2 "Ринкові та фундаментальні дані" та розділ 3 "Альтернативні дані для фінансів" охоплюють джерела та управління даними, сировину та найважливіший рушій успішної торгової стратегії.

Різке зростання доступності даних з точки зору обсягу, різноманітності та швидкості є ключовим доповненням до застосування машинного навчання (ML) у трейдингу, що, своєю чергою, збільшило витрати галузі на придбання нових джерел даних. Однак, зростаючий обсяг даних вимагає ретельного відбору та управління для виявлення потенційної цінності, включаючи такі кроки:

1. Визначити та оцінити ринкові, фундаментальні та альтернативні джерела даних, що містять альфа-сигнали, які не згасають надто швидко.
2. Розгорніть або отримайте доступ до хмарної масштабованої інфраструктури даних та аналітичних інструментів, таких як Hadoop або Spark, для забезпечення швидкого та гнучкого доступу до даних.
3. Ретельно керуйте та куруйте дані, щоб уникнути упередженого прогнозування, коригуючи їх до бажаної частоти на певний момент часу. Це означає, що дані повинні відображати лише інформацію, доступну та відому на даний момент часу. Алгоритми машинного навчання, навчені на спотворених історичних даних, майже напевно дадуть збій під час реального трейдингу.

Альфа-фактори призначені для вилучення сигналів з даних для прогнозування дохідності активів для заданого інвестиційного простору протягом торгового горизонту. Фактор набуває одного значення для кожного активу під час оцінки, але може поєднувати одну або кілька вхідних змінних. Процес включає кроки, описані на наступному рисунку:

Фаза дослідження робочого процесу торгової стратегії включає розробку, оцінку та комбінування альфа-факторів. ML відіграє велику роль у цьому процесі, оскільки складність факторів зростає, оскільки інвестори реагують як на затухання сигналу простіших факторів, так і на набагато багатші дані, доступні сьогодні.

Альфа-фактори продукують сигнали входу та виходу, що призводять до ордерів на купівлю або продаж, а виконання ордерів призводить до утримання активів у портфелі. Профілі ризику окремих позицій взаємодіють, створюючи специфічний профіль ризику портфеля. Управління портфелем включає оптимізацію ваг позицій для досягнення бажаного ризику портфеля та отримання профілю, який відповідає загальним інвестиційним цілям. Цей процес є дуже динамічним, щоб враховувати постійно мінливі ринкові дані.

Включення інвестиційної ідеї в алгоритмічну стратегію вимагає ретельного тестування з використанням наукового підходу, який намагається відхилити ідею на основі її ефективності в альтернативних позавибіркових ринкових сценаріях. Тестування може включати симульовані дані для фіксації сценаріїв, які вважаються можливими, але не відображені в історичних даних.

3.4 ML для трейдингу на практиці: стратегії та варіанти використання

На практиці ми застосовуємо машинне навчання (ML) до трейдингу в контексті конкретної стратегії для досягнення певної бізнес-мети. У цьому розділі ми коротко опишемо, як торговельні стратегії розвивалися та диверсифікувалися, а також наведемо реальні приклади застосування ML, підкреслюючи, як вони пов'язані з матеріалом цієї книги.

Кількісні стратегії розвивалися та ставали більш складними у три хвилі:

1. У 1980-х та 1990-х роках сигнали часто виникали в результаті академічних досліджень та використовували один або дуже мало вхідних даних, отриманих з ринкових та фундаментальних даних. AQR, один з найбільших кількісних хедж-фондів на сьогодні, був заснований у 1998 році для впровадження таких стратегій у великих масштабах. Ці сигнали зараз значною мірою комерціалізовані та доступні як ETF, такі як базові стратегії повернення до середнього значення.

2. У 2000-х роках факторне інвестування поширилося завдяки новаторській роботі Юджина Фами, Кеннета Френча та інших [8]. Фонди використовували алгоритми для ідентифікації активів, що піддаються впливу факторів ризику, таких як вартість або імпульс, щоб шукати можливості для арбітражу. Викуп активів на початку фінансової кризи спровокував кількісний землетрус серпня 2007 року, який поширився на індустрію факторних фондів. Ці стратегії зараз також доступні як довгострокові розумні бета-фонди, які нахиляють портфелі відповідно до заданого набору факторів ризику.

3. Третя ера зумовлена інвестиціями в можливості машинного навчання та альтернативні дані для генерування прибуткових сигналів для повторюваних торгових стратегій. Розпад факторів є серйозною проблемою: було показано, що надлишкова дохідність від нових аномалій падає на чверть від відкриття до публікації та понад 50 відсотків після публікації через конкуренцію та скупчення ресурсів.

Сьогодні трейдери переслідують низку різних цілей, використовуючи алгоритми для виконання правил:

- Алгоритми виконання торгів, спрямовані на досягнення вигідного ціноутворення.
- Короткострокові угоди, спрямовані на отримання прибутку від невеликих коливань цін, наприклад, через арбітраж.
- Поведінкові стратегії, спрямовані на передбачення поведінки інших учасників ринку.

- Торгові стратегії, засновані на прогнозах абсолютних та відносних цін і прибутковості.

ML витягує сигнали з широкого спектру ринкових, фундаментальних та альтернативних даних і може застосовуватися на всіх етапах процесу алгоритмічної торгової стратегії. Ключові застосування включають:

- Аналіз даних для виявлення закономірностей, вилучення ознак та отримання аналітичних висновків.
- Кероване навчання для генерації факторів ризику або альф та створення торгових ідей.
- Агрегація окремих сигналів у стратегію.
- Розподіл активів відповідно до профілів ризику, вивчених алгоритмом.
- Тестування та оцінка стратегій, зокрема за допомогою синтетичних даних.
- Інтерактивне, автоматизоване вдосконалення стратегії за допомогою навчання з підкріпленням.

Ми коротко виділимо деякі з цих застосувань і визначимо, де ми продемонструємо їх використання в наступних розділах.

3.4.1 Інтелектуальний аналіз даних для вилучення ознак та аналітики

Економічно ефективна оцінка великих, складних наборів даних вимагає виявлення сигналів у великих масштабах.

Теорія інформації допомагає оцінити вміст сигналу в ознаках-кандидатах, що є корисним для вилучення найцінніших вхідних даних для моделі машинного навчання.

Самонавчання пропонує широкий спектр методів для визначення структури даних з метою отримання розуміння або вирішення подальших завдань.

Прозорість моделі – ми наголошуємо на специфічних для моделі способах отримання розуміння прогностичної сили окремих змінних та представляємо новий теоретико-ігровий підхід, який називається адитивними поясненнями

Шеплі (SHAP). Ми застосовуємо його до градієнтних буст-машин з великою кількістю вхідних змінних.

3.4.2 Кероване навчання для створення та агрегації альфа-фактора

Найбільш відомим обґрунтуванням застосування машинного навчання (ML) у трейдингу є отримання прогнозів фундаментальних показників активів, руху цін або ринкових умов. Стратегія може використовувати кілька алгоритмів ML, які базуються один на одному:

- Нижченаведені моделі можуть генерувати сигнали на рівні портфеля, інтегруючи прогнози щодо перспектив окремих активів, очікувань ринку капіталу та кореляції між цінними паперами.
- Як альтернатива, прогнози машинного навчання можуть впливати на дискреційні угоди , як у квантовому підході, описаному раніше.

Прогнози машинного навчання також можуть бути спрямовані на конкретні фактори ризику, такі як вартість або волатильність, або впроваджувати технічні підходи, такі як слідування за трендом або повернення до середнього значення:

Машинне навчання (ML) використовувалося для розподілу портфелів на основі моделей дерев рішень, які обчислюють ієрархічну форму паритету ризиків. В результаті, характеристики ризику визначаються закономірностями цін на активи, а не класами активів, і досягають вищих характеристик співвідношення ризику та прибутковості.

3.4.3 Тестування торгових ідей

Тестування на попередніх даних є критично важливим кроком для вибору успішних стратегій алгоритмічного трейдингу. Перехресна перевірка з використанням синтетичних даних є ключовим методом машинного навчання (ML) для отримання надійних результатів поза вибіркою в поєднанні з відповідними методами для корекції багаторазового тестування. Характер

фінансових даних, що базується на часових рядах, вимагає модифікацій стандартного підходу, щоб уникнути упередженості прогнозування або іншого забруднення даних, що використовуються для навчання, перевірки та тестування. Крім того, обмежена доступність історичних даних призвела до появи альтернативних підходів, що використовують синтетичні дані: ми продемонструємо різні методи тестування моделей ML з використанням ринкових, фундаментальних та альтернативних моделей, які отримують обґрунтовані оцінки помилок поза вибіркою.

3.4.4 Навчання з підкріпленням

Трейдинг відбувається на конкурентному, інтерактивному ринку. Навчання з підкріпленням має на меті навчити агентів вивчати функцію політики, засновану на винагородах; це часто вважається однією з найперспективніших галузей у фінансовому машинному навчанні. Див., наприклад, [18] щодо застосування до виконання торгів.

3.5 Оптимізація портфеля та оцінка ефективності

У контексті портфеля позитивна дохідність активів може компенсувати негативні цінові коливання. Позитивні зміни ціни на один актив з більшою ймовірністю компенсують збитки на інший, чим нижча кореляція між двома позиціями. Виходячи з того, як ризик портфеля залежить від коваріації позицій, Гаррі Марковіц розробив теорію сучасного управління портфелем на основі диверсифікації в 1952 році [19]. Результатом є оптимізація середньої дисперсії, яка вибирає ваги для заданого набору активів, щоб мінімізувати ризик, виміряний як стандартне відхилення дохідності для заданої очікуваної дохідності.

Модель ціноутворення капітальних активів (Capital Asset Pricing Model – CAPM) вводить премію за ризик, виміряну як очікувана дохідність, що перевищує безризикову інвестицію, як рівноважну винагороду за володіння

активом. Ця винагорода компенсує вплив одного фактора ризику – ринку – який є систематичним, на відміну від ідіосинкратичного для активу, і тому не може бути диверсифікований.

Управління ризиками стало більш складним, оскільки з'явилися додаткові фактори ризику та більш детальні варіанти вибору експозиції. Критерій Келлі – це популярний підхід до динамічної оптимізації портфеля, який полягає у виборі послідовності позицій з часом; він був адаптований у [20] з моменту свого початкового застосування в азартних іграх до фондового ринку.

В результаті існує кілька підходів до оптимізації портфелів, які включають застосування машинного навчання (ML) для вивчення ієрархічних зв'язків між активами та трактування їх активів як доповнень або замінників відносно профілю ризику портфеля.

Щоб оцінити та порівняти різні стратегії або вдосконалити існуючу стратегію, нам потрібні показники, що відображають їхню ефективність відносно наших цілей. В інвестуванні та торгівлі найпоширенішими цілями є дохідність та ризик інвестиційного портфеля.

Цілі щодо дохідності та ризику передбачають компроміс: більший ризик може дати вищу дохідність за певних обставин, але також передбачає більший збиток. Щоб порівняти, як різні стратегії дотримуються цього компромісу, дуже популярні коефіцієнти, які обчислюють міру дохідності на одиницю ризику. Ми по черзі обговоримо коефіцієнт Шарпа та інформаційний коефіцієнт (Information Ratio – IR).

Коефіцієнт Шарпа (Sharpe Ratio – SR) порівнює очікуваний надлишковий портфель з волатильністю цього надлишкового доходу, виміряного його стандартним відхиленням. Він вимірює компенсацію як середній надлишковий дохід на одиницю прийнятого ризику. Його можна оцінити на основі даних.

Фінансова дохідність часто порушує припущення щодо неідентифікованого надлишкового доходу (IID). Автор у [21] вивів необхідні корективи до розподілу та часової агрегації для доходності, яка є стаціонарною, але автокорельованою. Це важливо, оскільки властивості часових рядів

інвестиційних стратегій (наприклад, середнє повернення, імпульс та інші форми серійної кореляції) можуть мати нетривіальний вплив на сам оцінювач SR, особливо при річному обчисленні SR з даних вищої частоти.

Цікаво, що Renaissance Technologies (RenTec), найефективніший квантовий фонд, заснований Джимом Саймонсом, про який ми згадували вище, приніс подібну дохідність, як і Воррен Баффет, незважаючи на надзвичайно різні підходи. Інвестиційна фірма Воррена Баффета Berkshire Hathaway тримає близько 100-150 акцій протягом досить тривалих періодів, тоді як RenTec може виконувати 100 000 угод на день. Як ми можемо порівняти ці різні стратегії?

ML – це оптимізація цільових функцій. В алгоритмічній торгівлі цілями є дохідність і ризик загального інвестиційного портфеля, зазвичай відносно бенчмарку (яким можуть бути готівка, безризикова процентна ставка або індекс цін на активи, такий як S&P 500).

Високий інформаційний коефіцієнт (IR) передбачає привабливе перевищення показників відносно додаткового прийнятого ризику. Фундаментальний закон активного управління розбиває IR на інформаційний коефіцієнт (IC) як міру навички прогнозування та здатність застосовувати цю навичку через незалежні ставки. Він підсумовує важливість як частих ставок (висока широта охоплення), так і надійних ставок (високий IC).

Індекс ризику (IC) вимірює кореляцію між альфа-фактором та прогнозованою прибутковістю, що виникає в результаті його сигналів, і відображає точність прогнозних навичок менеджера. Широта стратегії вимірюється незалежною кількістю ставок, які інвестор робить за певний період часу, а добуток обох значень пропорційний IR, також відомому як оціночний ризик [22].

Фундаментальний закон важливий, оскільки він висвітлює ключові чинники перевищення результатів: важливі як точні прогнози, так і здатність робити незалежні прогнози та діяти на їх основі. На практиці оцінити широту стратегії складно, враховуючи перехресну та часову рядову кореляцію між прогнозами.

3.6 Як керувати ризиком та дохідністю портфеля

Управління портфелем спрямоване на вибір та визначення розміру позицій у фінансових інструментах, які досягають бажаного компромісу між ризиком та дохідністю відносно бенчмарку. Як портфельний менеджер, у кожному періоді ви обираєте позиції, які оптимізують диверсифікацію для зменшення ризиків, одночасно досягаючи цільової дохідності. Протягом періодів ці позиції можуть потребувати ребалансування, щоб врахувати зміни ваг, що виникають внаслідок руху цін, для досягнення або підтримки цільового профілю ризику.

Диверсифікація дозволяє нам зменшувати ризики для заданої очікуваної дохідності, використовуючи те, як недосконала кореляція дозволяє прибуткам одного активу компенсувати втрати іншого активу. Автор [19] винайшов сучасну теорію портфеля (MPT) у 1952 році та надав математичні інструменти для оптимізації диверсифікації шляхом вибору відповідних ваг портфеля.

Сучасна теорія портфеля розв'язує задачу визначення оптимальних ваг портфеля для мінімізації волатильності для заданої очікуваної дохідності або максимізації дохідності для заданого рівня волатильності. Ключовими необхідними вхідними даними є очікувана дохідність активів, стандартні відхилення та коваріаційна матриця.

Розглянемо короткий приклад оптимізації середньої дисперсії. Програмний код розв'язує задачу для визначення оптимальних ваг портфеля, щоб мінімізувати волатильність для заданої очікуваної прибутковості або максимізувати прибутковість для заданого рівня волатильності. Ключовими необхідними вхідними даними є очікувана прибутковість активів, стандартні відхилення та коваріаційна матриця.

Диверсифікація працює, оскільки дисперсія прибутковості портфеля залежить від коваріації активів і може бути зменшена нижче середньозваженого значення дисперсій активів шляхом включення активів з кореляцією, що не є ідеальною.

Зокрема, для заданого вектора ω ваг портфеля та коваріаційної матриці, Σ, дисперсії портфеля, σ_{PF} визначається як:

$$\sigma_{\text{PF}} = \omega^T \Sigma \omega. \tag{3.1}$$

У [19] автор показав, що задача максимізації очікуваної дохідності портфеля з урахуванням цільового ризику має еквівалентне подвійне представлення мінімізації ризику портфеля з урахуванням цільового рівня очікуваної дохідності μ_{PF}. Отже, задача оптимізації стає такою:

$$\begin{aligned} \min_{\omega} \quad & \sigma_{\text{PF}}^2 = \omega^T \Sigma \omega, \\ \text{при умові} \quad & \mu_{\text{PF}} = \omega^T \mu \\ & \|\omega\| = 1. \end{aligned} \tag{3.2}$$

Ми можемо розрахувати ефективну межу, використовуючи scipy.optimize.minimize та історичні оцінки дохідності активів, стандартних відхилень та коваріаційної матриці. Докладний програмний код наведено в додаках. Зображення змодельованого портфеля активів показано на рис. 3.1.

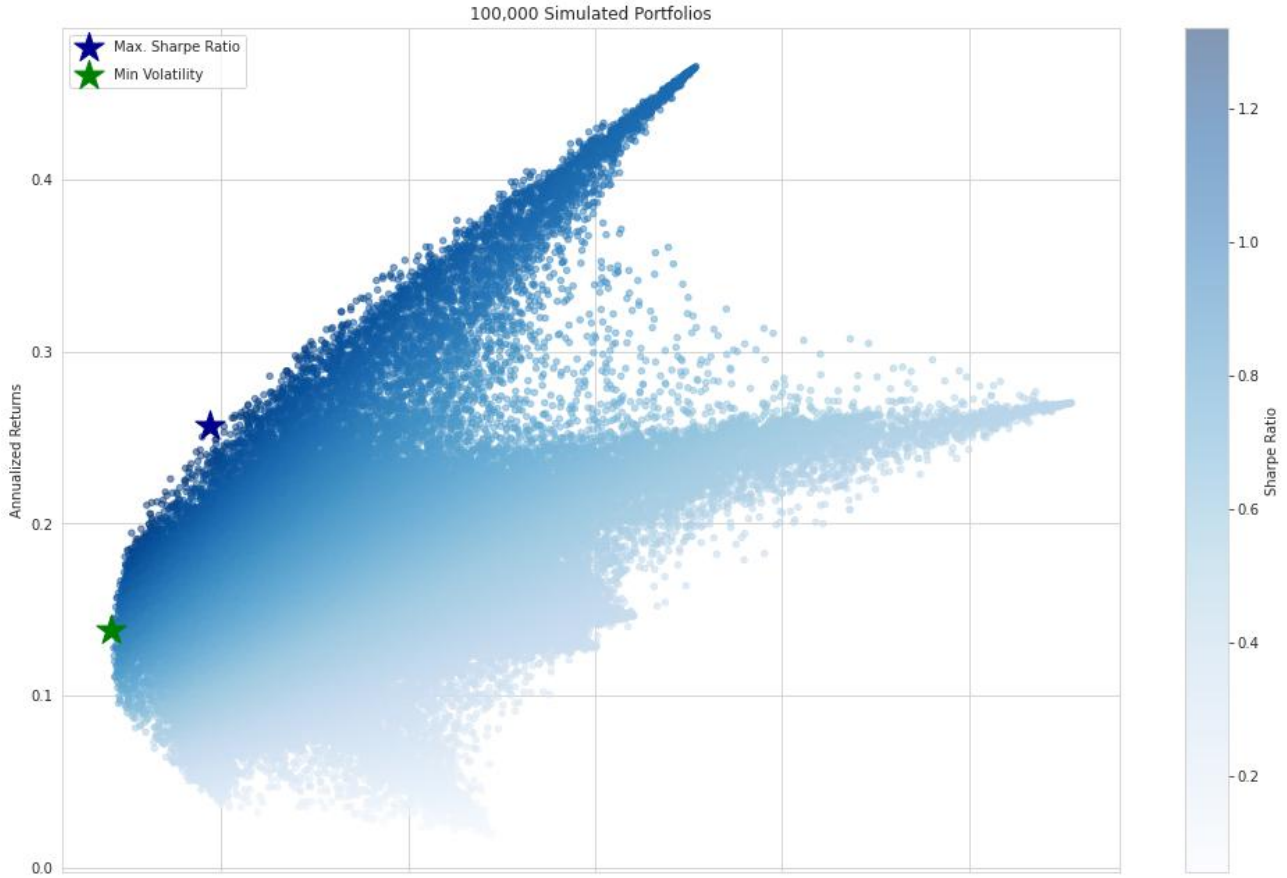


Рисунок 3.1 – Модель портфеля активів

Моделювання дає підмножину можливих портфелів, а ефективна межа визначає оптимальні комбінації дохідності та ризику у вибірці, які були досяжні на основі історичних даних.

На рисунку 3.2 показано результат, включаючи портфель з мінімальною дисперсією та портфель, який максимізує SR, а також кілька портфелів, отриманих за допомогою альтернативних стратегій оптимізації (див. рис. 3.2).

3.7 Альтернативи оптимізації середньої дисперсії

Труднощі з точними вхідними даними для задачі оптимізації середньої дисперсії призвели до прийняття кількох практичних альтернатив, які обмежують середнє значення, дисперсію або обидва, або пропускають оцінки дохідності, які є більш складними, такі як підхід паритету ризиків, який ми обговоримо пізніше в цьому розділі.

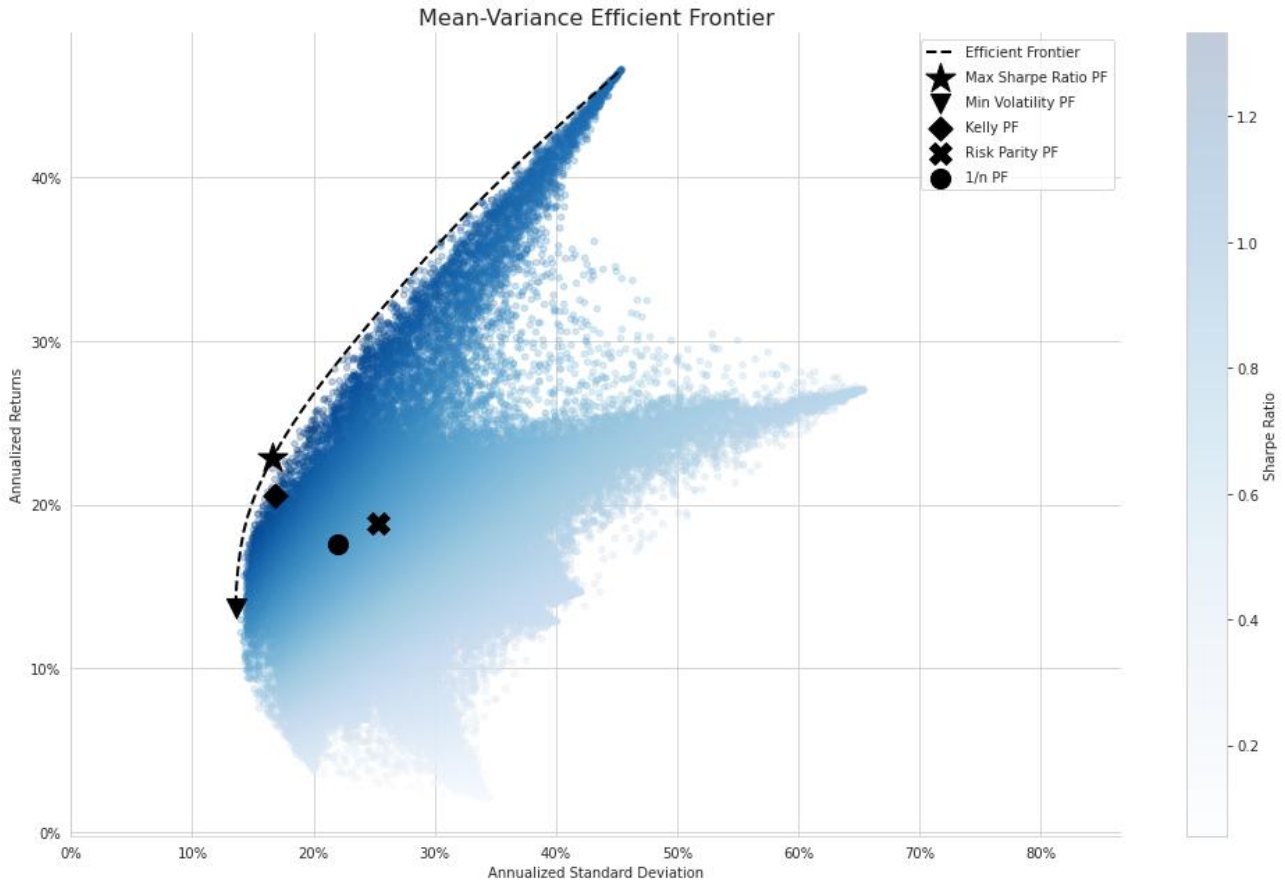


Рисунок 3.2 – Співвідношення між дохідністю портфеля активів та волатильністю активів

3.7.2 Портфель 1/N

Прості портфелі забезпечують корисні контрольні показники для оцінки доданої вартості складних моделей, які генерують ризик перенавчання. Найпростіша стратегія – портфель з рівною вагою – показала себе однією з найкращих.

3.7.3 Портфель з мінімальною дисперсією

Іншою альтернативою є портфель з глобальною мінімальною дисперсією (Global Minimum-Variance – GMV), який надає пріоритет мінімізації ризику. Він показаний на рисунку ефективної межі та може бути розрахований н, мінімізуючи стандартне відхилення портфеля за допомогою структури середньої дисперсії.

3.7.4 Підхід Блека-Літтермана

Підхід глобальної оптимізації портфеля Блека та Літтермана поєднує економічні моделі зі статистичним навчанням і є популярним, оскільки він генерує оцінки очікуваної дохідності, які є правдоподібними в багатьох ситуаціях. Цей метод припускає, що ринок є портфелем із середньою дисперсією, як це випливає з моделі рівноваги CAPM. Він ґрунтується на тому факті, що спостережувану ринкову капіталізацію можна розглядати як оптимальні ваги, що присвоюються кожному цінному паперу ринком. Ринкові ваги відображають ринкові ціни, які, у свою чергу, втілюють очікування ринку щодо майбутньої прибутковості.

3.7.5 Правило Келлі

Правило Келлі має довгу історію в азартних іграх, оскільки воно дає вказівки щодо того, скільки ставити на кожну з (нескінченної) послідовності ставок з різними (але сприятливими) коефіцієнтами, щоб максимізувати кінцеве багатство.

Келлі пов'язав теорію інформації Шеннона, щоб знайти ставку, яка є оптимальною для довгострокового зростання капіталу, коли коефіцієнти сприятливі, але невизначеність залишається. Його правило максимізує логарифмічне багатство як функцію шансів на успіх кожної гри та включає неявний захист від банкрутства, оскільки log(0) дорівнює мінус нескінченності, тому гравець Келлі природним чином уникне втрати всього.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Охорона праці та її актуальність в ІТ-сфері

Для підвищення ефективності системи управління охорони праці (СУОП) дуже важлива роль належить формуванню і розвитку інформаційної культури фахівців ІТ-технологій, яка впливає на удосконалення інформаційного контуру сучасних підприємств, дозволяє створювати надійні прогнози щодо стану умов праці, показників здоров'я та працездатності, виробничого травматизму і професійної захворюваності, визначати політику розвитку підприємств, установ та організацій на основі різноманітних стратегій охорони праці (інноваційні, маркетингові, інвестиційні, фінансові, технологічні, диверсифікаційні). Поряд з інформаційною культурою важливо використовувати в рамках СУОП "трикутник" її складових: правову, організаційну, управлінську.

В управлінні охороною праці потрібно реалізувати основні положення, окремі теоретико-методологічні підходи інформаційного менеджменту. Головну роль та відповідальність за стан СУОП мають нести фахівці служби охорони праці сучасного підприємства.

Сучасне суспільство називають постіндустріальним, постекономічним, інформаційним, оскільки йдеться про багатосторонні і кардинальні зміни у розвитку цивілізації.

На постіндустріальному етапі розвитку суспільства вирішальним фактором стає інформація. Її домінування ініціювала науково-технічна революція, яку ще іменують інформаційною, оскільки нею охоплена будь-яка інтелектуальна діяльність, починаючи з інформаційних образів штучного інтелекту у нових технологіях, економіки, і продовжуючи інформатизацією суспільства в умова світової глобалізації науки й освіти тощо.

Інформаційні технології розглядаються як потужний важіль економічного зростання України. Для цього необхідні значні стратегічні інвестиції у

комп’ютерну та комунікаційну інфраструктуру, програми досліджень і розробок, освітню галузь.

Під інформаційною культурою розуміють сукупність, складову НІТ (новітні інформаційні технології), технологічну, правову, психологічну, соціологічну та ергономічну підсистеми, що сприяють спрямованому впливу на протікання соціальних процесів у суспільстві, колективі і вихованню свідомого відношення людини до праці, виконання прав та обов’язків [23].

Поняття інформаційної культури виникло в процесі активізації дослідницької уваги до механізмів інформаційного обміну у зв’язку зі значним підвищення ролі інформації в соціокультурних процесах суспільства, яке розглядають як інформаційне суспільство знань, де в центрі знаходяться інформаційні технології.

Робота з інформацією та інформаційна культура в цілому є одним з найважливіших компонентів спроб компанії управляти змінами. Є три принципові причини, в силу яких сьогодні необхідно дбати про інформаційну культуру компанії.

По-перше, вона все більше і більше стає найважливішою частиною загальної організаційної (корпоративної) культури компанії. Все більше компаній розуміють необхідність перетворень, орієнтованих на задоволення очікувань споживача. Щоб сьогодні впливати на майбутнє, потрібно уявляти собі на що вона буде схожа. А для цього потрібно працювати з різноманітною діловою, професійною, технологічною, соціальною, ринковою та політичною інформацією.

По-друге, інформаційні технології роблять можливим створення в компаніях комп'ютерних мереж, за допомогою яких йде спілкування між менеджерами, але важливо знати, як люди використовують цю інформацію. Саме по собі створення такої мережі з усіма її робочими станціями і мультимедійними можливостями не гарантує того, що інформація буде використовуватися більш розумно і більш ефективно.

По-третє, для різних функціональних служб, підрозділів та робочих груп сучасних підприємств в сфері охорони праці інформаційна культура різна, а це означає відмінність методологічних підходів до процесів усвідомлення, збору, організації, обробки, поширення і використання інформації. Тому багато менеджерів погодяться з тим, що корпоративна інформаційна культура важлива для вироблення різних стратегій охорони праці та запровадження відповідних заходів з її вдосконалення.

Для деяких галузей, таких як розробка програмного забезпечення, інформаційна культура є необхідною умовою виживання, тому що зміна технологій в розробці програмного забезпечення відбувається кожні 6-8 місяців, а інвестиції на підготовку персоналу і освоєння нової технології величезні і у великих компаніях варіюються від 1,5 до 2 млрд. доларів на рік.

Аналіз свідчить, що інформатизація та інтеграція комунікаційного простору України сприяє різкому підвищенню інформаційної та професійної компетентності, ділової активності, стимулюванню конкуренції, створенню інноваційних підприємств та організацій, нових робочих місць, зниженню витрат на утримання управлінського апарату.

Поряд із задачами і здобутками окреслилися негативи використання інформаційних технологій:

1) надмірне інформаційне навантаження, суть якого полягає у тому, що кількість корисної інформації, яка надходить до мережі, перевищує психофізіологічні можливості її сприйняття людиною;

2) велика кількість інформації, яка сприймається, але не є корисною для фахівців в даний момент;

3) інформаційний голод, причиною якого є саме надлишок інформації, викликаний інформаційним перенавантаженням;

4) "інформоманія" як хвороба людини, яка робить останню знеособленою, залежною від перебування в інформаційному просторі і роботи з комп’ютером і чому вона віддає перевагу, уникаючи "живого" спілкування з людьми;

5) поява "кіберспільнот", що за своїми соціокультурними характеристиками набагато ближчі до представників інших культур у глобальному інформаційному просторі, ніж до своєї етнонаціональної спільноти чи решти населення, не охопленого Інтернетом;

6) індивідуалізм і дегуманізація способу життя "мешканців" Інтернету – відсутність готовності ділитися своїми знаннями.

Слід розуміти, що комп'ютерні технології, а особливо їх мережі істотно впливають на життєдіяльність людини, припускаючи глобалізацію і технократизацію суспільства. Але в ще більшій мірі цей вплив поширюється безпосередньо на центральну нервову систему, яка звикає працювати в дуже інтенсивному режимі багатозадачності, де вже переважають не тривалі логічні роздуми, а інтуїтивно-реактивні ланцюжки розумових формулювань у зв'язку з величезним обсягом оброблюваної щодня інформації, кількість якої зростає за експоненціальною швидкістю. Виникає припущення, що саме збільшення обсягу інформації та прискорення її обробки людиною може згубно вплинути на розвиток розумових здібностей людини.

Аналіз продуктивності розумової праці в найбільших за чисельністю фахівців IT-фірм показав, що велике значення з точки зору впливу на її результати має організаційна (корпоративна) культура. В цьому напрямі влаштовуються різні тімбілдинги, заходи, тренінги для розвитку персоналу. Також кожен керівник повинен добре розуміти свого співробітника, що саме для нього важливо, що його мотивує. Важливо відвести потрібну роль відповідному співробітнику, щоб він виконував ті завдання, які йому цікаві.

На подібних тренінгах в тому числі повинна розглядатися інформаційна культура працівника, в освоєнні, володінні, мотивуванні, застосуванні, перетворенні інформації із застосуванням сучасних інформаційних технологій і використанням цих умінь в навчанні з охорони праці і в подальшій професійній діяльності. Особливо вони будуть корисні, як доповнення до існуючих інструктажів з охорони праці на підприємстві, або як контроль психологічного стану та взаємовідносин у колективі.

Інформаційна культура як інтегративне утворення абсолютно не зводиться до розрізнених знань, вмінь та навичок роботи за комп'ютером. Вона передбачає інформативну спрямованість цілісної особистості, яка володіє мотивацією до застосування і засвоєння нових даних. Інформаційну культуру можна розглядати, як одну з граней особистісного розвитку промислових робітників. Це шлях універсалізації якостей людини.

Оволодіння інформаційною культурою сприяє реальному розумінню особистістю свого місця, себе і своєї ролі у виробничому колективі. Вона має сприяти формуванню нового покоління фахівців інформаційного суспільства, який повинен володіти наступними навичками: виділення релевантної, значущої інформації, диференціації вихідних даних, розробки інформативних критеріїв її оцінки інформації, вміло використовувати її в рамках СУОП.

Сьогодні продовжує діяти стратегічне правило "Можливості комп’ютерної техніки обмежені тільки нашими уявленнями".

4.2 Шкідлива дія шуту та вібрації і захист від неї

Для запобігання шкідливої дії шуму і вібрації на організм працюючих проводяться технічні, організаційні і медикопрофілактичні заходи [25].

Одним з основних технічних заходів є зменшення при експлуатації та на стадії проектування, конструювання обладнання причин шуму і вібрації в самому джерелі утворення. Досягають цього завдяки використанню раціональної конструкції обладнання, заміни ударної дії деталей і машин коливальною, з’єднання елементів гнучкими зв’язками, врівноважування обертових частин механізмів, заміни металевих деталей пластмасовими, забезпечення різних власних частот коливань механізму з частотою збуджуючої сили. Аеродинамічний шум може бути зменшений застосуванням глушників та повітропроводів зі змінним перерізом. Шум трансформаторів (електромагнітний шум) знижується, якщо застосувати листи заліза як складових осердя трансформатора з малою магнітострикцією, серцевини.

Якщо неможливо ізолювати чи знизити шум і вібрацію самого джерела, потрібно:

- ізолювати джерело шуму або вібрації від навколишнього середовища засобами вібро- та звукоізоляції;
- раціонально планувати виробничі приміщення, що мають інтенсивні джерела шуму;
- збільшувати звукопоглинання внутрішніх поверхонь приміщення шляхом звукопоглинальних покриттів.

Принцип роботи звукоізоляційних екранів оснований на відбиванні звукової хвилі від різних екранів, стін, кожухів обладнання. Шумливі агрегати слід закривати звукоізоляційними кожухами з виводом назовні органів керування та контрольних приладів. Звукоізоляційні екрани виготовляють з металу, деревини, пластмаси та інших щільних матеріалів. Екрани зсередини покривають звукопоглинаючими матеріалами (скловатою пінополіуретаном), а по периметру кожуха – віброізоляційними підкладками (гума).

Вихідними даними для розрахунку параметрів необхідного екрану є спектр шуму, який необхідно ослабити, кількість екранів, через які проходить шум, їх площа, акустичні характеристики приміщення. За розрахованими значеннями необхідної звукової ізоляційної здатності екрану підбирається матеріал конструкції й екрану.

Принцип звукопоглинання оснований на явищі трансформації коливальної енергії звуку в теплову через втрати при терті. Найбільші втрати при терті мають пористі, волокнисті і перфоровані матеріали: поролон, пемзолітові і деревоволокнисті плити тощо. Енергія звукової хвилі переходить у теплову енергію, причому, ефект звукоізоляції збільшується з ростом частоти звукової хвилі. Звукопоглинаючими матеріалами оббивають стелі, стіни. Щоб одержати ефективну звукоізоляцію, найбільш доцільно застосовувати багатошарові огородження з м’якими прошарками (мінеральна вата).

Важливим технічним рішенням у забезпеченні виробничих умов є вдосконалення ручних віброінструментів. Для цього використовують

віброгасіння, змінюють ударний вузол, проводять балансування частин, що обертаються.

Послаблення локальної вібрації і передачі вібрації на підлогу і сидіння досягається засобами віброізоляції і вібропоглинання, застосуванням пружинних і гумових амортизаторів, прокладок тощо. Для обмеження поширення вібрацій через ґрунт, між фундаментом і ґрунтом залишають повітряні проміжки, які називаються акустичними розривами.

В останні роки знаходять застосування динамічні віброгасники, в яких створюються вібрації, що співпадають по частоті і протилежні по фазі вібрації машини, коливання якої необхідно зменшити.

До організаційних заходів по боротьбі з шумом та вібрацією на виробництві відносяться: впровадження раціонального режиму праці і відпочинку, обмеження часу роботи при використанні ручного інструменту, який створює вібрацію.

Глушники звуку застосовуються для зменшення шуму аеродинамічних установок (вентиляторів, пневмоінструментів, газотурбінних, дизельних, компресорних установок). Вони поділяються на активні, які поглинають звукову енергію, що на них поступила, і реактивні, які відбивають цю енергію. Потужні джерела шуму як правило розміщують в окремих приміщеннях, які віддалені від постійних робочих місць. Ізоляційні кабіни або екрани застосовують як екрани робочих місць для зменшення зовнішніх шумів.

Якщо не вдається зменшити рівень шуму і вібрації на робочому місці до нормативних значень та необхідно використовувати засоби індивідуального захисту: рукавиці, взуття, навушники, м’які шоломи, які зменшують рівень звукового тиску на 40-50 дБ.

У процесі виробництв, експлуатації і зберігання радіоелектронних засобів можуть виникати механічні і динамічні дії, що характеризуються широким діапазоном частот коливань, а також амплітудою, прискоренням і часом дії. Рівень механічних дій визначається умовами транспортування й експлуатації.

Необхідно розрізняти два види механічних дій: удари і вібрації. Удар виникає, коли апаратура отримує швидку зміну прискорення (піддаються удару входи кабелів, джгути, резистори, конденсатори, напівпровідникові діоди і тріоди, силові трансформатори, дроселі тощо). Вібрації – довготривалі знакозмінні процеси, які впливають на роботу апаратури при безпосередньому контакті з джерелом коливань або через повітряне середовище.

У результаті дії вібрацій і удару можуть бути наступні ушкодження апаратури: порушення герметичності через псування паяльних, зварних і клеєних швів і появи тріщин у метало-скляних спаях; повне руйнування корпусів або окремих їх частин через механічний резонанс або циклічну втому; обривання монтажних зв’язків, відшарування багатошарових друкованих плат, руйнування підставок; вихід з ладу електричних контактів; модуляція розмірів хвилеводних трактів; коаксіальних кабелів, конденсаторів змінної ємності, коливальних контурів, електровакуумних приладів, зміщення положення органів настроювання і управління.

Під впливом вібрацій може статись зміна параметрів напівпровідникових приладів, вольт амперних характеристик діодів, транзисторів. Все це призводить до руйнування конструкцій за рахунок явищ втоми. Радіоелектронна апаратура (РЕА) повинна мати віброміцність, вібростійкість, ударостійкість.

Захист РЕА здійснюється наступними групами методів:

- зменшується інтенсивність джерел вібрації шляхом балансування, зменшення зазорів, віброізоляції джерела вібрацій;
- зменшується величина дій, що передається апаратом шляхом віброізоляції, демпфірування, виключення резонансів, активного віброзахисту за допомогою ексцентриків, маятників, гіроскопів;
- використання найбільш добротні і жорсткі компоненти і вузли;
- застосовуються амортизатори.

Захист часом, захист віддалю, усунення джерела тепловиділення, теплоізоляція, охолодження гарячої поверхні, забезпечення тепловіддачі тіла людини та індивідуальні засоби захисту. Захист часом передбачає обмеження

часу перебування робітника в зоні дії інфрачервоного випромінювання. Потужність випромінювання можна знизити за рахунок конструкторських і технологічних рішень (зміною нагрівання виробів у нагрівальних пічках індукційним нагріванням та ін.) і за рахунок покриття поверхні, яка нагрівається, тепло ізолювальним матеріалом.

Якщо теплоізоляція неможлива, тоді захист від прямої дії інфрачервоного випромінювання здійснюється екрануванням. Екрани можуть бути прозорими, напівпрозорими і непрозорими. У свою чергу вони поділяються на тепловідбивальні, тепловідвідні та теплопоглинальні; стаціонарні і нестаціонарні.

Застосовують також прозору водяну завісу у вигляді суцільної тонкої водяної плівки. Вода є активним поглиначем інфрачервоного випромінювання.

Перегрівання людини попереджують раціональним режимом пиття, режимом праці та гідро процедурами. Спецодяг виготовляється з незаймистого, стійкого до інфрачервоного випромінювання, м’якого і повітронепроникного матеріалу (тканина з металевим покриттям відбиває 90% інфрачервоного випромінювання). Для захисту очей застосовують світлофільтри зі спеціального жовто-зеленого або синього скла.

Першочергові заходи – це конструкторські і технологічні рішення, які виключають генерацію або понижують інтенсивність випромінювання. Спеціальні засоби захисту (екранування джерел випромінювання, фарбування стін у світлі кольори) попереджують розповсюдження і знижують інтенсивність цих випромінювань у виробничих приміщеннях. Очі захищають окулярами або щитками зі склом – світлофільтром. Для захисту шкіри використовують мазі з речовинами – світлофільтрами для цих променів (салол, саліцилово-метиловий ефір та ін.), а також спецодяг з бавовняних тканин і грубововняного сукна. Руки захищають рукавицями.

ВИСНОВКИ

Автоматизований трейдинг революціонізувала спосіб участі трейдерів на фінансових ринках, пропонуючи швидкість, ефективність та об'єктивність. Використовуючи передові алгоритми, автоматизовані торгові системи усувають людські емоції та упередження з процесу прийняття рішень, що призводить до більш дисциплінованої та потенційно прибуткової торгової діяльності.

У цьому розділі ми розглянули визначення автоматизованого трейдингу, принципи її роботи, переваги, ризики, популярні торгові стратегії та важливі фактори, які слід враховувати під час вибору автоматизованої торгової системи. Ми також обговорили, як розпочати роботу з автоматизованим трейдингом, наголошуючи на важливості навчання, налаштування, управління ризиками та постійного навчання.

Хоча автоматизований трейдинг надає численні переваги, важливо підходити до неї з обережністю та розумінням пов'язаних з нею ризиків. Технічні збої, волатильність ринку, складність та надмірна оптимізація – це все фактори, про які трейдери повинні пам'ятати, використовуючи автоматизовані торгові системи.

Автоматизований трейдинг не є гарантованим шляхом до прибутку, вона вимагає ретельного планування, розробки стратегії та постійного моніторингу. Важливо ретельно досліджувати та тестувати автоматизовані трейдингові системи, вибираючи ті, що відповідають торговим цілям та толерантності до ризику.

Постійне навчання, слідкування за ринковими тенденціями та звернення за професійною порадою за потреби допоможуть зорієнтуватися в складнощах автоматизованого трейдингу та збільшити ваші шанси на успіх.

Зрештою, автоматизований трейдинг може бути потужним інструментом у торговому арсеналі, що дозволить приймати рішення на основі даних та ефективно та точно використовувати ринкові можливості.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Готович, В. А., С. В. Марценко, and Т. Л. Щербак. "Створення мобільного апаратно-програмного пристрою моніторингу характеристик якості електроенергії." Збірник наукових праць Інституту проблем моделювання в енергетиці ім. ГЄ Пухова 70 (2014): 98-105.

2. Марценко, Сергій Володимирович. Математичне моделювання та статистичні методи обробки даних вимірювань в задачах моніторингу електронавантаження. Diss. Тернопільський національний технічний університет імені Івана Пулюя, 2011.

3. Lytvynenko, I., et al. "Simulation of gas consumption process based on the mathematical model in the form of cyclic random process considering the scale factors." 1st International Workshop on Information Technologies: Theoretical and Applied Problems, ITTAP 2021. 2021.

4. Машлій, Г. Б., & Ганущак, В. (2016). Електронний трейдинг – сучасна технологія біржової торгівлі. In II-га Міжнародна науково-практична конференція "Теоретичні та практичні аспекти розвитку науки"(м. Київ, 29–30 листопада 2016 року). Київ: МЦНД.

5. Лупенко, А. М., & Степчук, В. Ю. (2023). Трейдинг криптовалютами: ризики, можливості та важливі фактори успіху в цифровій торгівлі. Матеріали XI науково-технічної конференції „Інформаційні моделі, системи та технології “, 179 – 179.

6. Лупенко, А. М., & Степчук, В. Ю. (2023). Ризик-менеджмент у трейдингу: стратегії зниження ризиків та керування капіталом. Матеріали XI науково-технічної конференції „Інформаційні моделі, системи та технології “, 178 – 178.

7. Markowitz, H. (1952). Portfolio Selection/H. Markovitz. The Journal of Finance, 7(1), 77-91.

8. Fama, E. F., & French, K. R. (2004). The capital asset pricing model: Theory and evidence. Journal of economic perspectives, 18(3), 25-46.

9. Grinold, R. C., & Kahn, R. N. (2000). Active portfolio management.

10. Treynor, J. L., & Black, F. (1973). How to use security analysis to improve portfolio selection. The journal of business, 46(1), 66-86.

11. Clarke, R., De Silva, H., & Thorley, S. (2002). Portfolio constraints and the fundamental law of active management. Financial Analysts Journal, 58(5), 48-66.

12. Lo, A. W. (2002). The statistics of Sharpe ratios. Financial analysts journal, 58(4), 36-52.

13. Black, F., & Litterman, R. (1992). Global portfolio optimization. Financial analysts journal, 48(5), 28-43.

14. CFA Institute. Dark pools [Електронний ресурс] / CFA Institute. – 2020. – Режим доступу: https://rpc.cfainstitute.org/policy/positions/dark-pools, (10.04.2025).

15. Economist. (2019). The stockmarket is now run by computers, algorithms and passive managers. The Economist.

16. Barclays PLC. Barclays PLC [Електронний ресурс] / Barclays PLC. – 2025. – Режим доступу: https://www.londonstockexchange.com/stock/BARC/barclays-plc/company-page, (10.04.2025).

17. Perlin, M. S., Caldeira, J. F., Santos, A. A., & Pontuschka, M. (2017). Can we predict the financial markets based on Google's search queries?. Journal of Forecasting, 36(4), 454-467.

18. Hendricks, D., & Wilcox, D. (2014, March). A reinforcement learning extension to the Almgren-Chriss framework for optimal trade execution. In 2014 IEEE Conference on computational intelligence for financial engineering & economics (CIFEr) (pp. 457-464). IEEE.

19. Portfolio Selection, Harry Markowitz, The Journal of Finance, 1952.

20. Beat the Market: A Scientific Stock Market System , Edward O. Thorp,1967.

21. Lo, A. W. (2002). The statistics of Sharpe ratios. Financial analysts journal, 58(4), 36-52.

22. Treynor, J. L., & Black, F. (1973). How to use security analysis to improve portfolio selection. The journal of business, 46(1), 66-86.

23. Стручок, В. С., Стручок, О. С., & Мудра, Д. В. (2017). Навчальний посібник до написання розділу дипломного проекту та дипломної роботи ″Безпека в надзвичайних ситуаціях ″для студентів всіх спец. денної, заочної (дистанційної) та екстернатної форм навчання.

24. Стручок, В. С. (2022). Техноекологія та цивільна безпека. Частина "Цивільна безпека". Навчальний посібник.

25. Жидецький, В. Ц., Джигирей, В. С., & Мельников, О. В. (2000). Основи охорони праці. Львів: Афіша, 350, 132-136.

26. Навакатікян, О. О., Кальниш, В. В., & Стрюков, С. М. (1997). Охорона праці користувачів комп'ютерних відеодисплейних терміналів. О. Навакатікян.

ДОДАТКИ

Додаток А

Програмний код

Imports

In [1]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
import sys
import numpy as np
import pandas as pd
from pytz import UTC

from logbook import (NestedSetup, NullHandler, Logger, StreamHandler,
StderrHandler,
                     INFO, WARNING, DEBUG, ERROR)

from zipline import run_algorithm
from zipline.api import (attach_pipeline,
                         date_rules,
                         time_rules,
                         get_datetime,
                         order_target_percent,
                         pipeline_output,
                         record,
                         schedule_function,
                         get_open_orders,
                         calendars,
                         set_commission,
                         set_slippage)
from zipline.finance import commission, slippage
from zipline.pipeline import Pipeline, CustomFactor
from zipline.pipeline.factors import Returns, AverageDollarVolume

from pyfolio.utils import extract_rets_pos_txn_from_zipline

import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:

```
sns.set_style('whitegrid')
```

Logging Setup

In [4]:

```
# setup stdout logging
format_string = '[{record.time: %H:%M:%S.%f}]: {record.level_name}:
{record.message}'
zipline_logging = NestedSetup([NullHandler(level=DEBUG),
                               StreamHandler(sys.stdout,
format_string=format_string, level=INFO),
                               StreamHandler(sys.stderr, level=ERROR)])
zipline_logging.push_application()
log = Logger('Algorithm')
```

Algo Settings

In [5]:

```
# Settings
MONTH = 21
YEAR = 12 * MONTH
N_LONGS = 50
N_SHORTS = 50
VOL_SCREEN = 500
```

In [6]:

```
start = pd.Timestamp('2013-01-01', tz=UTC)
end = pd.Timestamp('2017-01-01', tz=UTC)
capital_base = 1e7
```

Mean Reversion Factor

In [7]:

```
class MeanReversion(CustomFactor):
    """Compute ratio of latest monthly return to 12m average,
       normalized by std dev of monthly returns"""
    inputs = [Returns(window_length=MONTH)]
    window_length = YEAR

    def compute(self, today, assets, out, monthly_returns):
        df = pd.DataFrame(monthly_returns)
        out[:] = df.iloc[-1].sub(df.mean()).div(df.std())
```

Create Pipeline

The Pipeline created by the `compute_factors()` method returns a table with a long and a short column for the 25 stocks with the largest negative and positive deviations of their last monthly return from its annual average, normalized by the standard deviation. It also limited the universe to the 500 stocks with the highest average trading volume over the last 30 trading days.

In [8]:

```
def compute_factors():
    """Create factor pipeline incl. mean reversion,
        filtered by 30d Dollar Volume; capture factor ranks"""
```

```
    mean_reversion = MeanReversion()
    dollar_volume = AverageDollarVolume(window_length=30)
    return Pipeline(columns={'longs'  : mean_reversion.bottom(N_LONGS),
                             'shorts' : mean_reversion.top(N_SHORTS),
                             'ranking': mean_reversion.rank(ascending=False)},
                    screen=dollar_volume.top(VOL_SCREEN))
```

Before_trading_start() ensures the daily execution of the pipeline and the recording of the results, including the current prices.

In [9]:

```
def before_trading_start(context, data):
    """Run factor pipeline"""
    context.factor_data = pipeline_output('factor_pipeline')
    record(factor_data=context.factor_data.ranking)
    assets = context.factor_data.index
    record(prices=data.current(assets, 'price'))
```

Set up Rebalancing

The new rebalance() method submits trade orders to the exec_trades() method for the assets flagged for long and short positions by the pipeline with equal positive and negative weights. It also divests any current holdings that are no longer included in the factor signals:

In [10]:

```
def rebalance(context, data):
    """Compute long, short and obsolete holdings; place trade orders"""
    factor_data = context.factor_data
    assets = factor_data.index

    longs = assets[factor_data.longs]
    shorts = assets[factor_data.shorts]
    divest = context.portfolio.positions.keys() - longs.union(shorts)
    log.info('{} | Longs: {:2.0f} | Shorts: {:2.0f} |
{:,.2f}'.format(get_datetime().date(),
                                                                    len(longs),
                                                                    len(shorts),

context.portfolio.portfolio_value))

    exec_trades(data, assets=divest, target_percent=0)
    exec_trades(data, assets=longs, target_percent=1 / N_LONGS if N_LONGS else 0)
    exec_trades(data, assets=shorts, target_percent=-1 / N_SHORTS if N_SHORTS else
0)
```

In [11]:

```
def exec_trades(data, assets, target_percent):
    """Place orders for assets using target portfolio percentage"""
    for asset in assets:
        if data.can_trade(asset) and not get_open_orders(asset):
```

```
            order_target_percent(asset, target_percent)
```

Initialize Backtest

The `rebalance()` method runs according to `date_rules` and `time_rules` set by the `schedule_function()` utility at the beginning of the week, right after market_open as stipulated by the built-in US_EQUITIES calendar (see docs for details on rules).

You can also specify a trade commission both in relative terms and as a minimum amount. There is also an option to define slippage, which is the cost of an adverse change in price between trade decision and execution

In [12]:

```
def initialize(context):
    """Setup: register pipeline, schedule rebalancing,
        and set trading params"""
    attach_pipeline(compute_factors(), 'factor_pipeline')
    schedule_function(rebalance,
                      date_rules.week_start(),
                      time_rules.market_open(),
                      calendar=calendars.US_EQUITIES)

    set_commission(us_equities=commission.PerShare(cost=0.00075,
                                                   min_trade_cost=.01))
    set_slippage(us_equities=slippage.VolumeShareSlippage(volume_limit=0.0025,
                                                          price_impact=0.01))
```

Run Algorithm

The algorithm executes upon calling the run_algorithm() function and returns the backtest performance DataFrame.

In [13]:

```
backtest = run_algorithm(start=start,
                         end=end,
                         initialize=initialize,
                         before_trading_start=before_trading_start,
                         bundle='quandl',
                         capital_base=capital_base)
```

Extract pyfolio Inputs

The `extract_rets_pos_txn_from_zipline` utility provided by `pyfolio` extracts the data used to compute performance metrics.

In [14]:

```
returns, positions, transactions = extract_rets_pos_txn_from_zipline(backtest)
```

Persist Results for use with `pyfolio`

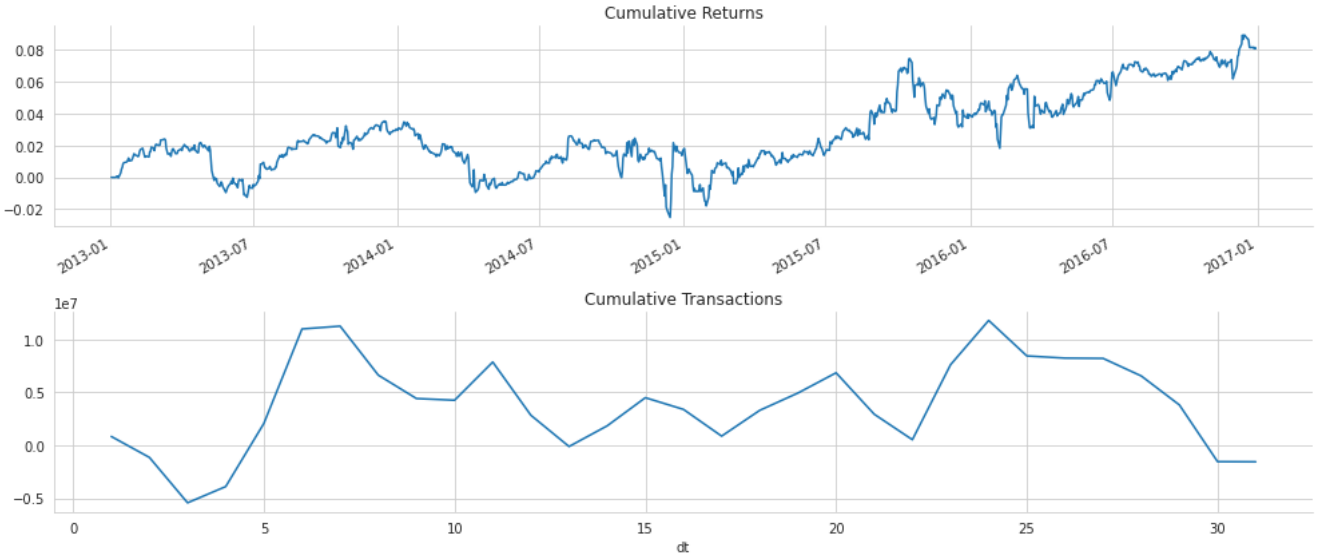
In [15]:

```
with pd.HDFStore('backtests.h5') as store:
    store.put('backtest/equal_weight', backtest)
    store.put('returns/equal_weight', returns)
    store.put('positions/equal_weight', positions)
    store.put('transactions/equal_weight', transactions)
```

Plot Results

In [16]:

```
fig, axes= plt.subplots(nrows=2, figsize=(14,6))
returns.add(1).cumprod().sub(1).plot(ax=axes[0], title='Cumulative Returns')
transactions.groupby(transactions.dt.dt.day).txn_dollars.sum().cumsum().plot(ax=ax
es[1], title='Cumulative Transactions')
fig.tight_layout()
sns.despine();
```

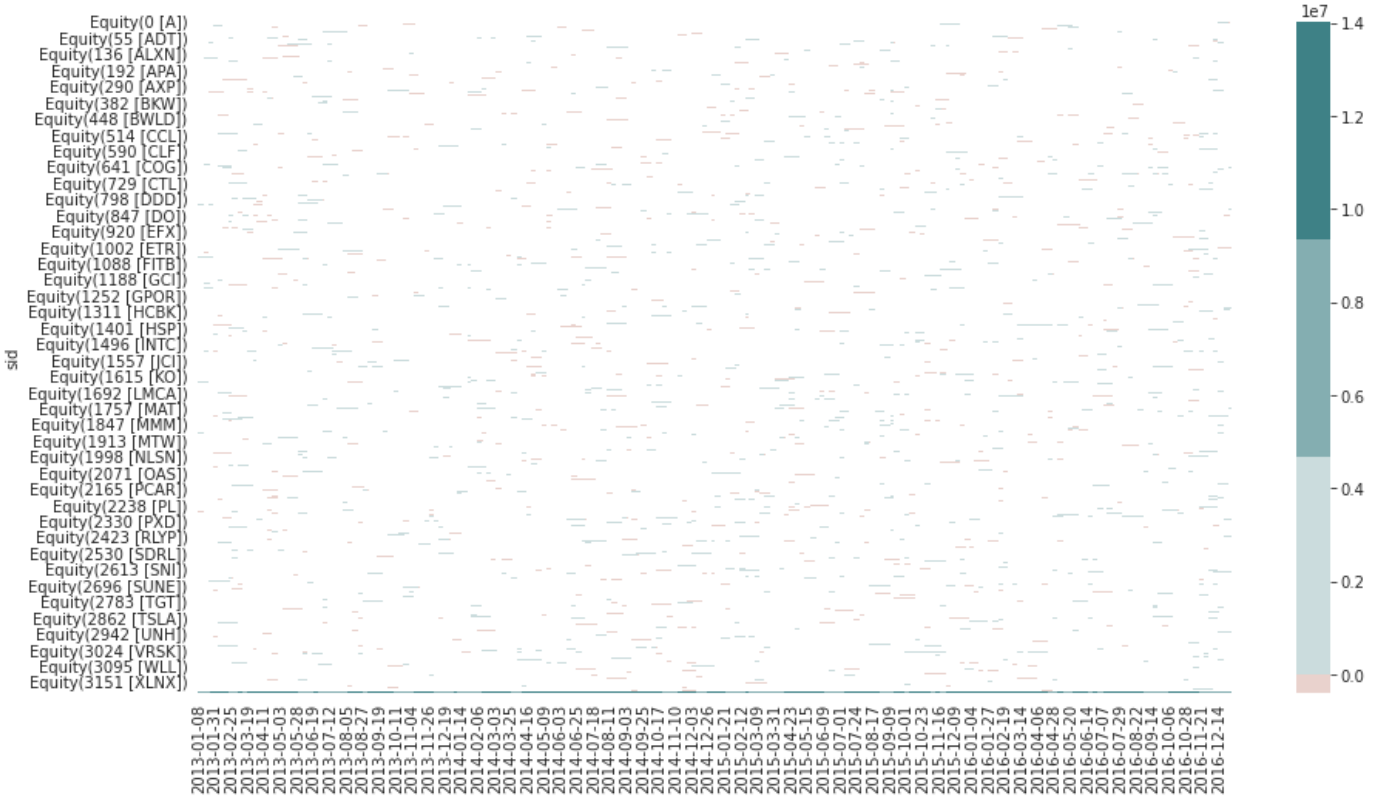


In [17]:

```
positions.index = positions.index.date
```

In [18]:

```
fig, ax = plt.subplots(figsize=(15, 8))
sns.heatmap(positions.replace(0, np.nan).dropna(how='all', axis=1).T,
            cmap=sns.diverging_palette(h_neg=20, h_pos=200), ax=ax, center=0);
```



In [19]:

```
positions.head()
```

Out[19]:

sid	Equity(0 [A])	Equity(1 [AA])	Equity(2 [AAL])	Equity(7 [AAP])	Equity(8 [AAPL])	Equity(12 [ABBV])	...	Equity(3197 [ZTS])	cash
2013-01-08	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	8.441446e+06
2013-01-09	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	8.379079e+06
2013-01-10	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	8.379079e+06
2013-01-11	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	8.379079e+06
2013-01-14	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	8.379079e+06

5 rows × 750 columns

In [20]:

```
transactions.info()
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 6505 entries, 2013-01-08 21:00:00+00:00 to 2016-12-30
21:00:00+00:00
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
```

```
 0   sid          6505 non-null   object
 1   symbol       6505 non-null   object
 2   price        6505 non-null   float64
 3   order_id     6502 non-null   object
 4   amount       6505 non-null   int64
 5   commission   0 non-null      object
 6   dt           6505 non-null   datetime64[ns, UTC]
 7   txn_dollars  6505 non-null   float64
dtypes: datetime64[ns, UTC](1), float64(2), int64(1), object(4)
memory usage: 457.4+ KB
```

In [ ]:

Imports & Settings

In [1]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
%matplotlib inline
from pathlib import Path

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from pyfolio.utils import extract_rets_pos_txn_from_zipline
from pyfolio.plotting import (plot_perf_stats,
                              show_perf_stats,
                              plot_rolling_beta,
                              plot_rolling_returns,
                              plot_rolling_sharpe,
                              plot_drawdown_periods,
                              plot_drawdown_underwater)

from pyfolio.timeseries import perf_stats, extract_interesting_date_ranges
```

In [3]:

```
sns.set_style('whitegrid')
```

Converting data from zipline to pyfolio

In [4]:

```
with pd.HDFStore('backtests.h5') as store:
    backtest = store['backtest/equal_weight']
backtest.info()
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1008 entries, 2013-01-02 00:00:00+00:00 to 2016-12-30
00:00:00+00:00
Data columns (total 39 columns):
```

```
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   period_open              1008 non-null   datetime64[ns, UTC]
 1   period_close             1008 non-null   datetime64[ns, UTC]
 2   starting_cash            1008 non-null   float64
 3   ending_cash              1008 non-null   float64
 4   portfolio_value          1008 non-null   float64
 5   returns                  1008 non-null   float64
 6   longs_count              1008 non-null   int64
 7   shorts_count             1008 non-null   int64
 8   long_value               1008 non-null   float64
 9   short_value              1008 non-null   float64
 10  long_exposure            1008 non-null   float64
 11  pnl                      1008 non-null   float64
 12  short_exposure           1008 non-null   float64
 13  capital_used             1008 non-null   float64
 14  orders                   1008 non-null   object
 15  transactions             1008 non-null   object
 16  gross_leverage           1008 non-null   float64
 17  positions                1008 non-null   object
 18  net_leverage             1008 non-null   float64
 19  starting_exposure        1008 non-null   float64
 20  ending_exposure          1008 non-null   float64
 21  starting_value           1008 non-null   float64
 22  ending_value             1008 non-null   float64
 23  factor_data              1008 non-null   object
 24  prices                   1008 non-null   object
 25  treasury_period_return   1008 non-null   float64
 26  trading_days             1008 non-null   int64
 27  period_label             1008 non-null   object
 28  algorithm_period_return  1008 non-null   float64
 29  algo_volatility          1007 non-null   float64
 30  benchmark_period_return  1008 non-null   float64
 31  benchmark_volatility     1007 non-null   float64
 32  alpha                    0 non-null      object
 33  beta                     0 non-null      object
 34  sharpe                   1004 non-null   float64
 35  sortino                  1004 non-null   float64
 36  max_drawdown             1008 non-null   float64
 37  max_leverage             1008 non-null   float64
 38  excess_return            1008 non-null   float64
dtypes: datetime64[ns, UTC](2), float64(26), int64(3), object(8)
memory usage: 315.0+ KB
```

`pyfolio` relies on portfolio returns and position data, and can also take into account the transaction costs and slippage losses of trading activity. The metrics are computed using the empyrical library that can also be used on a standalone basis. The performance DataFrame produced by the zipline backtesting engine can be translated into the requisite pyfolio input.

In [5]:

```
returns, positions, transactions = extract_rets_pos_txn_from_zipline(backtest)
```

In [6]:

```
returns.head().append(returns.tail())
```

Out[6]:

```
2013-01-02 00:00:00+00:00    0.000000
2013-01-03 00:00:00+00:00    0.000000
2013-01-04 00:00:00+00:00    0.000000
2013-01-07 00:00:00+00:00    0.000000
2013-01-08 00:00:00+00:00   -0.000005
2016-12-23 00:00:00+00:00   -0.000233
2016-12-27 00:00:00+00:00    0.000160
2016-12-28 00:00:00+00:00   -0.000847
2016-12-29 00:00:00+00:00    0.000735
2016-12-30 00:00:00+00:00   -0.000606
Name: returns, dtype: float64
```

In [7]:

```
positions.info()
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1004 entries, 2013-01-08 00:00:00+00:00 to 2016-12-30
00:00:00+00:00
Columns: 750 entries, Equity(0 [A]) to cash
dtypes: float64(750)
memory usage: 5.8 MB
```

In [8]:

```
positions.columns = [c for c in positions.columns[:-1]] + ['cash']
positions.index = positions.index.normalize()
positions.info()
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1004 entries, 2013-01-08 00:00:00+00:00 to 2016-12-30
00:00:00+00:00
Columns: 750 entries, Equity(0 [A]) to cash
dtypes: float64(750)
memory usage: 5.8 MB
```

In [9]:

```
transactions.symbol = transactions.symbol.apply(lambda x: x.symbol)
```

In [10]:

```
transactions.head().append(transactions.tail())
```

Out[10]:

	sid	symbol	price	order_id	amount	commission	dt	txn_dollars
2013-01-08 21:00:00+00:00	Equity(85 [AGN])	AGN	86.680005	eb7dbd283656403ca8c9d6c1188dc727	2334	None	2013-01-08 21:00:00+00:00	-202311.131306
2013-01-08 21:00:00+00:00	Equity(213 [ARIA])	ARIA	19.651001	682c967689d540748db436fd51d5163a	7590	None	2013-01-08 21:00:00+00:00	-149151.099321
2013-01-08 21:00:00+00:00	Equity(367 [BIIB])	BIIB	144.390001	a428261b64d4412db94a4573faa1cc04	1365	None	2013-01-08 21:00:00+00:00	-197092.351185
2013-01-08 21:00:00+00:00	Equity(811 [DFS])	DFS	40.090001	956824d7a6064717b7aef1594daa65dc	5073	None	2013-01-08 21:00:00+00:00	-203376.572741
2013-01-08 21:00:00+00:00	Equity(1059 [FDO])	FDO	57.320002	863bd8a8917b476390c04d8924fa923d	3496	None	2013-01-08 21:00:00+00:00	-200390.728571

	sid	symbol	price	order_id	amount	commission	dt	txn_dollars
2016-12-29 21:00:00+00:00	Equity(66 [AEO])	AEO	15.270000	ba96247cd5ee435da2c104312b81c7f7	3461	None	2016-12-29 21:00:00+00:00	-52849.470534
2016-12-29 21:00:00+00:00	Equity(833 [DKS])	DKS	52.380000	027a5f84d39549e5ad05470eb14ed268	1063	None	2016-12-29 21:00:00+00:00	-55679.940343
2016-12-29 21:00:00+00:00	Equity(1757 [MAT])	MAT	27.630000	638e5a7fee514e4091b637db8da6903c	1183	None	2016-12-29 21:00:00+00:00	-32686.290035
2016-12-30 21:00:00+00:00	Equity(2181 [PDCO])	PDCO	41.030000	5fdc0d539a0f4ccd8554dc6db26f36b2	-119	None	2016-12-30 21:00:00+00:00	4882.569999
2016-12-30 21:00:00+00:00	Equity(2953 [URBN])	URBN	28.480000	d49b7ce165dd4daaa515da54b6f2c497	-1006	None	2016-12-30 21:00:00+00:00	28650.879685

In [11]:

```
HDF_PATH = Path('..', 'data', 'assets.h5')
```

Sector Map

In [12]:

```
assets = positions.columns[:-1]
with pd.HDFStore(HDF_PATH) as store:
    df = store.get('us_equities/stocks')['sector'].dropna()
    df = df[~df.index.duplicated()]
sector_map = df.reindex(assets).fillna('Unknown').to_dict()
```

Benchmark

In [13]:

```
with pd.HDFStore(HDF_PATH) as store:
    benchmark_rets = store['sp500/fred'].close.pct_change()
benchmark_rets.name = 'S&P500'
benchmark_rets = benchmark_rets.tz_localize('UTC').filter(returns.index)
benchmark_rets.tail()
```

Out[13]:

```
DATE
2016-12-23 00:00:00+00:00    0.001252
2016-12-27 00:00:00+00:00    0.002248
2016-12-28 00:00:00+00:00   -0.008357
2016-12-29 00:00:00+00:00   -0.000293
2016-12-30 00:00:00+00:00   -0.004637
Name: S&P500, dtype: float64
```

In [14]:

```
perf_stats(returns=returns,
           factor_returns=benchmark_rets)
#            positions=positions,
#            transactions=transactions)
```

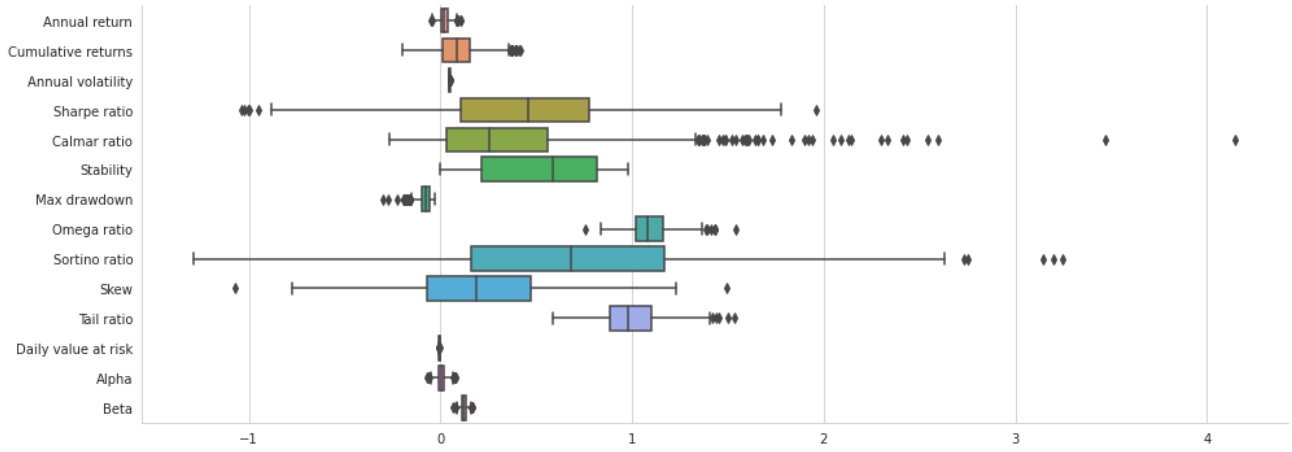
Out[14]:

```
Annual return          0.019619
Cumulative returns     0.080817
Annual volatility      0.047487
Sharpe ratio           0.432879
Calmar ratio           0.336024
```

```
Stability               0.555919
Max drawdown           -0.058387
Omega ratio             1.085094
Sortino ratio           0.630497
Skew                    0.223701
Kurtosis                6.125539
Tail ratio              0.988875
Daily value at risk    -0.005901
Alpha                   0.005922
Beta                    0.121033
dtype: float64
```

In [15]:

```
fig, ax = plt.subplots(figsize=(14, 5))
plot_perf_stats(returns=returns,
                factor_returns=benchmark_rets,
                ax=ax)
sns.despine()
fig.tight_layout();
```



Returns Analysis

Testing a trading strategy involves backtesting against historical data to fine-tune alpha factor parameters, as well as forward-testing against new market data to validate that the strategy performs well out of sample or if the parameters are too closely tailored to specific historical circumstances.

Pyfolio allows for the designation of an out-of-sample period to simulate walk-forward testing. There are numerous aspects to take into account when testing a strategy to obtain statistically reliable results, which we will address here.

In [16]:

```
oos_date = '2016-01-01'
```

In [17]:

```
show_perf_stats(returns=returns,
                factor_returns=benchmark_rets,
                positions=positions,
                transactions=transactions,
```

```
                    live_start_date=oos_date)
```

Start date			2013-01-02
End date			2016-12-30
In-sample months			36
Out-of-sample months			12
	In-sample	**Out-of-sample**	**All**
Annual return	1.3%	3.974%	1.962%
Cumulative returns	3.951%	3.974%	8.082%
Annual volatility	4.7%	4.899%	4.749%
Sharpe ratio	0.30	0.82	0.43
Calmar ratio	0.22	1.28	0.34
Stability	0.15	0.76	0.56
Max drawdown	-5.839%	-3.108%	-5.839%
Omega ratio	1.06	1.17	1.09
Sortino ratio	0.43	1.21	0.63
Skew	0.16	0.40	0.22
Kurtosis	5.47	7.74	6.13
Tail ratio	1.00	0.92	0.99
Daily value at risk	-0.587%	-0.601%	-0.59%
Gross leverage	0.35	0.36	0.35
Daily turnover	24.367%	23.364%	24.116%
Alpha	-0.00	0.03	0.01
Beta	0.13	0.09	0.12

Rolling Returns OOS

The `plot_rolling_returns` function displays cumulative in and out-of-sample returns against a user-defined benchmark (we are using the S&P 500):

In [18]:

```
plot_rolling_returns(returns=returns,
                     factor_returns=benchmark_rets,
                     live_start_date=oos_date,
                     cone_std=(1.0, 1.5, 2.0))
plt.gcf().set_size_inches(14, 8)
sns.despine()
plt.tight_layout();
```

The plot includes a cone that shows expanding confidence intervals to indicate when out-of-sample returns appear unlikely given random-walk assumptions. Here, our strategy did not perform well against the benchmark during the simulated 2017 out-of-sample period

Summary Performance Statistics

pyfolio offers several analytic functions and plots. The perf_stats summary displays the annual and cumulative returns, volatility, skew, and kurtosis of returns and the SR. The following additional metrics (which can also be calculated individually) are most important:

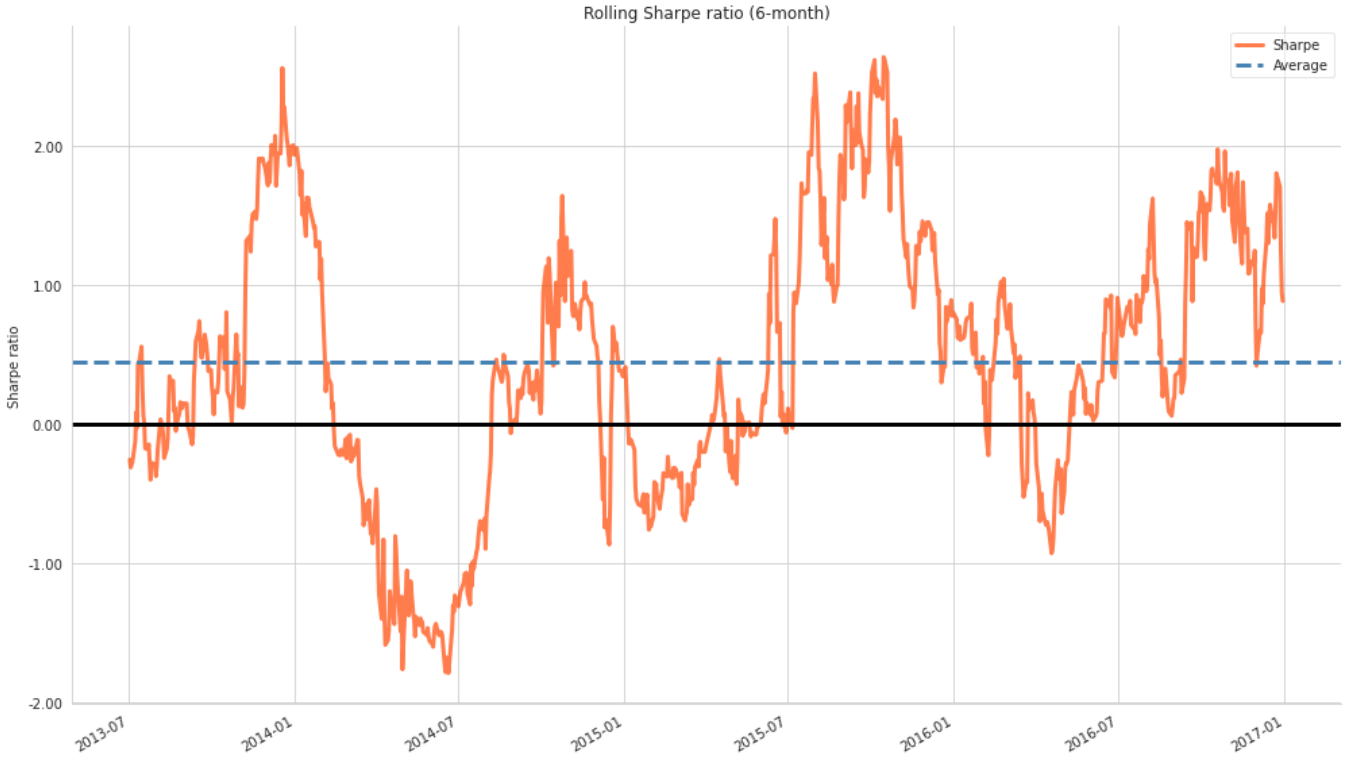
- Max drawdown: Highest percentage loss from the previous peak
- Calmar ratio: Annual portfolio return relative to maximal drawdown
- Omega ratio: The probability-weighted ratio of gains versus losses for a return target, zero per default
- Sortino ratio: Excess return relative to downside standard deviation
- Tail ratio: Size of the right tail (gains, the absolute value of the 95th percentile) relative to the size of the left tail (losses, abs. value of the 5th percentile)
- Daily value at risk (VaR): Loss corresponding to a return two standard deviations below the daily mean
- Alpha: Portfolio return unexplained by the benchmark return
- Beta: Exposure to the benchmark

Rolling Sharpe

In [19]:

```
plot_rolling_sharpe(returns=returns)
plt.gcf().set_size_inches(14, 8)
sns.despine()
```

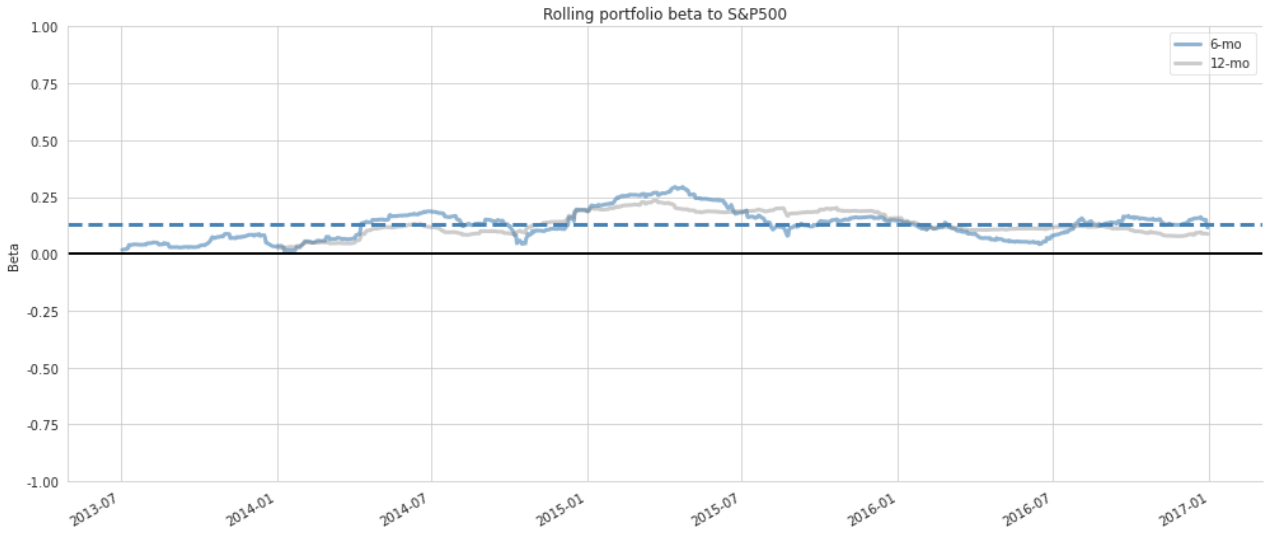
```
plt.tight_layout();
```



Rolling Beta

In [20]:

```
plot_rolling_beta(returns=returns, factor_returns=benchmark_rets)
plt.gcf().set_size_inches(14, 6)
sns.despine()
plt.tight_layout();
```



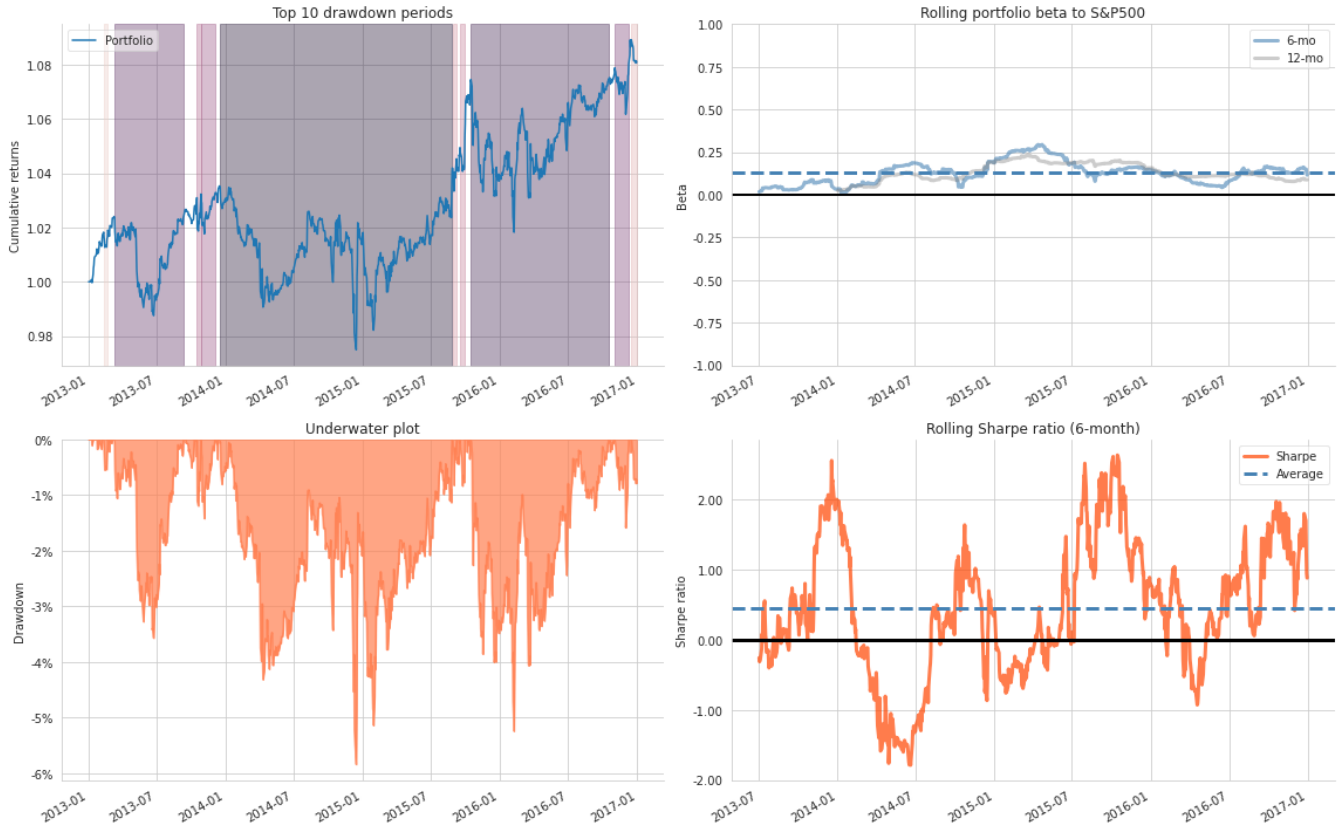
Drawdown Periods

The plot_drawdown_periods(returns) function plots the principal drawdown periods for the portfolio, and several other plotting functions show the rolling SR and rolling factor exposures to the market beta or the Fama French size, growth, and momentum factors:

In [21]:

```
fig, ax = plt.subplots(nrows=2, ncols=2, figsize=(16, 10))
axes = ax.flatten()

plot_drawdown_periods(returns=returns, ax=axes[0])
plot_rolling_beta(returns=returns, factor_returns=benchmark_rets, ax=axes[1])
plot_drawdown_underwater(returns=returns, ax=axes[2])
plot_rolling_sharpe(returns=returns)
sns.despine()
plt.tight_layout();
```



This plot, which highlights a subset of the visualization contained in the various tear sheets, illustrates how pyfolio allows us to drill down into the performance characteristics and exposure to fundamental drivers of risk and returns.

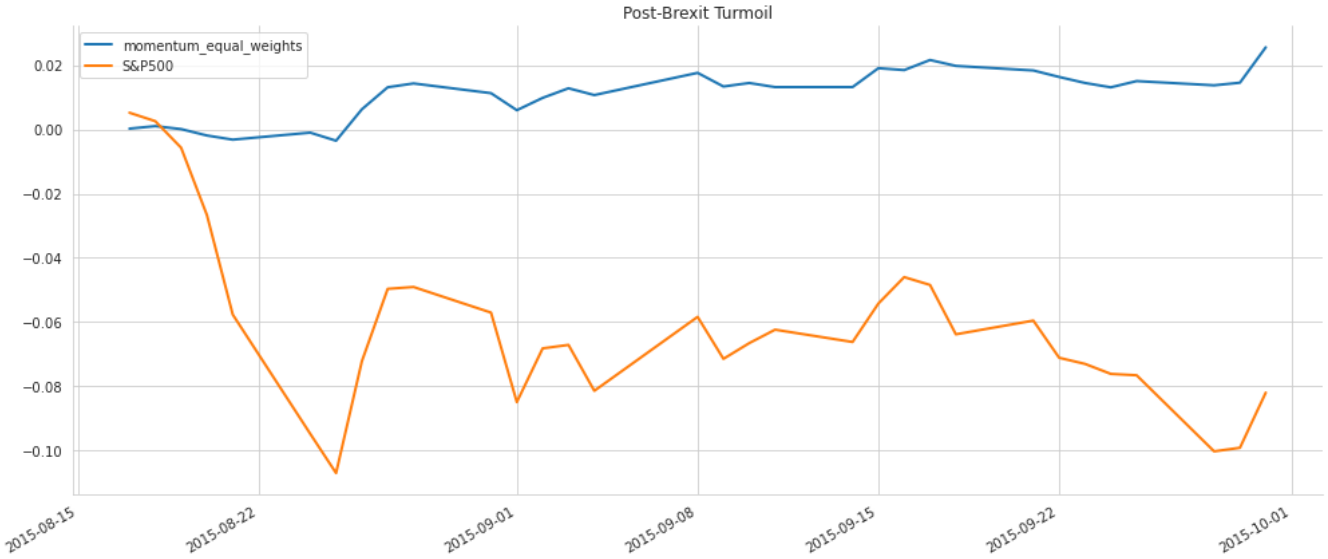
Modeling Event Risk

Pyfolio also includes timelines for various events that you can use to compare the performance of a portfolio to a benchmark during this period, for example, during the fall 2015 selloff following the Brexit vote.

In [22]:

```
interesting_times = extract_interesting_date_ranges(returns=returns)
```

```
(interesting_times['Fall2015']
 .to_frame('momentum_equal_weights').join(benchmark_rets)
 .add(1).cumprod().sub(1)
 .plot(lw=2, figsize=(14, 6), title='Post-Brexit Turmoil'))
sns.despine()
plt.tight_layout();
```



Mean-Variance Optimization

Imports & Settings

In [1]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
%matplotlib inline

import pandas as pd
import numpy as np
from numpy.random import random, uniform, dirichlet, choice
from numpy.linalg import inv

from scipy.optimize import minimize

import pandas_datareader.data as web
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter
import seaborn as sns
```

In [3]:

```
sns.set_style('whitegrid')
np.random.seed(42)
```

In [4]:

```
cmap = sns.diverging_palette(10, 240, n=9, as_cmap=True)
```

Prepare Data

We select historical data for tickers included in the S&P500 (according to Wikipedia) from 1998-2017.

In [5]:

```
with pd.HDFStore('../data/assets.h5') as store:
    sp500_stocks = store['sp500/stocks']
```

In [6]:

```
sp500_stocks.head()
```

Out[6]:

	name	gics_sector	gics_sub_industry	location	first_added	cik	founded
ticker							
MMM	3M Company	Industrials	Industrial Conglomerates	St. Paul, Minnesota	1976-08-09	66740	1902
ABT	Abbott Laboratories	Health Care	Health Care Equipment	North Chicago, Illinois	1964-03-31	1800	1888
ABBV	AbbVie Inc.	Health Care	Pharmaceuticals	North Chicago, Illinois	2012-12-31	1551152	2013 (1888)
ABMD	ABIOMED Inc	Health Care	Health Care Equipment	Danvers, Massachusetts	2018-05-31	815094	1981
ACN	Accenture plc	Information Technology	IT Consulting & Other Services	Dublin, Ireland	2011-07-06	1467373	1989

In [7]:

```
with pd.HDFStore('../data/assets.h5') as store:
    prices = (store['quandl/wiki/prices']
              .adj_close
              .unstack('ticker')
              .filter(sp500_stocks.index)
              .sample(n=30, axis=1))
```

Compute Inputs

Compute Returns

In [8]:

```
start = 2008
end = 2017
```

Create month-end monthly returns and drop dates that have no observations:

In [9]:

```
weekly_returns =
prices.loc[f'{start}':f'{end}'].resample('W').last().pct_change().dropna(how='all'
)
weekly_returns = weekly_returns.dropna(axis=1)
weekly_returns.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 521 entries, 2008-01-13 to 2017-12-31
Freq: W-SUN
Data columns (total 25 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   WYNN    521 non-null    float64
 1   FIS     521 non-null    float64
 2   VLO     521 non-null    float64
 3   MRK     521 non-null    float64
 4   DTE     521 non-null    float64
 5   REGN    521 non-null    float64
 6   TEL     521 non-null    float64
 7   YUM     521 non-null    float64
 8   HST     521 non-null    float64
 9   AES     521 non-null    float64
 10  XOM     521 non-null    float64
 11  NEE     521 non-null    float64
 12  HAL     521 non-null    float64
 13  TJX     521 non-null    float64
 14  AXP     521 non-null    float64
 15  IR      521 non-null    float64
 16  CERN    521 non-null    float64
 17  IVZ     521 non-null    float64
 18  XLNX    521 non-null    float64
 19  LOW     521 non-null    float64
 20  WEC     521 non-null    float64
 21  ETFC    521 non-null    float64
 22  XEL     521 non-null    float64
 23  CDNS    521 non-null    float64
 24  BXP     521 non-null    float64
dtypes: float64(25)
memory usage: 105.8 KB
```

Set Parameters

In [10]:

```
stocks = weekly_returns.columns
```

In [11]:

```
n_obs, n_assets = weekly_returns.shape
n_assets, n_obs
```

Out[11]:

```
(25, 521)
```

In [12]:

```
NUM_PF = 100000 # no of portfolios to simulate
```

In [13]:

```
x0 = uniform(0, 1, n_assets)
x0 /= np.sum(np.abs(x0))
```

Annualization Factor

In [14]:

```
periods_per_year = round(weekly_returns.resample('A').size().mean())
periods_per_year
```

Out[14]:

```
52
```

Compute Mean Returns, Covariance and Precision Matrix

In [15]:

```
mean_returns = weekly_returns.mean()
cov_matrix = weekly_returns.cov()
```

The precision matrix is the inverse of the covariance matrix:

In [16]:

```
precision_matrix = pd.DataFrame(inv(cov_matrix), index=stocks, columns=stocks)
```

Risk-Free Rate

Load historical 10-year Treasury rate:

In [17]:

```
treasury_10yr_monthly = (web.DataReader('DGS10', 'fred', start, end)
                         .resample('M')
                         .last()
                         .div(periods_per_year)
                         .div(100)
                         .squeeze())
```

In [18]:

```
rf_rate = treasury_10yr_monthly.mean()
```

Simulate Random Portfolios

The simulation generates random weights using the Dirichlet distribution, and computes the mean, standard deviation, and SR for each sample portfolio using the historical return data:

In [19]:

```
def simulate_portfolios(mean_ret, cov, rf_rate=rf_rate, short=True):
    alpha = np.full(shape=n_assets, fill_value=.05)
```

```
    weights = dirichlet(alpha=alpha, size=NUM_PF)
    if short:
        weights *= choice([-1, 1], size=weights.shape)

    returns = weights @ mean_ret.values + 1
    returns = returns ** periods_per_year - 1
    std = (weights @ weekly_returns.T).std(1)
    std *= np.sqrt(periods_per_year)
    sharpe = (returns - rf_rate) / std
    return pd.DataFrame({'Annualized Standard Deviation': std,
                         'Annualized Returns': returns,
                         'Sharpe Ratio': sharpe}), weights
```

In [20]:

```
simul_perf, simul_wt = simulate_portfolios(mean_returns, cov_matrix, short=False)
```

In [21]:

```
df = pd.DataFrame(simul_wt)
df.describe()
```

Out[21]:

	0	1	2	3	...	23	24
count	1.000000e+05	1.000000e+05	1.000000e+05	1.000000e+05	...	1.000000e+05	1.000000e+05
mean	4.014051e-02	4.021212e-02	3.990815e-02	3.998121e-02	...	4.026256e-02	3.993907e-02
std	1.312552e-01	1.311073e-01	1.303379e-01	1.306601e-01	...	1.315626e-01	1.302771e-01
min	5.487810e-97	6.391219e-103	2.537639e-96	3.103047e-92	...	1.990480e-122	3.521496e-97
25%	6.548585e-13	7.526395e-13	7.532093e-13	7.098138e-13	...	6.467348e-13	6.834165e-13
50%	6.616167e-07	7.502422e-07	7.797426e-07	7.198818e-07	...	6.695123e-07	7.623604e-07
75%	2.277156e-03	2.395975e-03	2.408796e-03	2.264291e-03	...	2.369465e-03	2.459960e-03
max	9.977419e-01	9.979261e-01	9.993984e-01	9.998279e-01	...	9.990703e-01	9.993066e-01

8 rows × 25 columns

Plot Simulated Portfolios

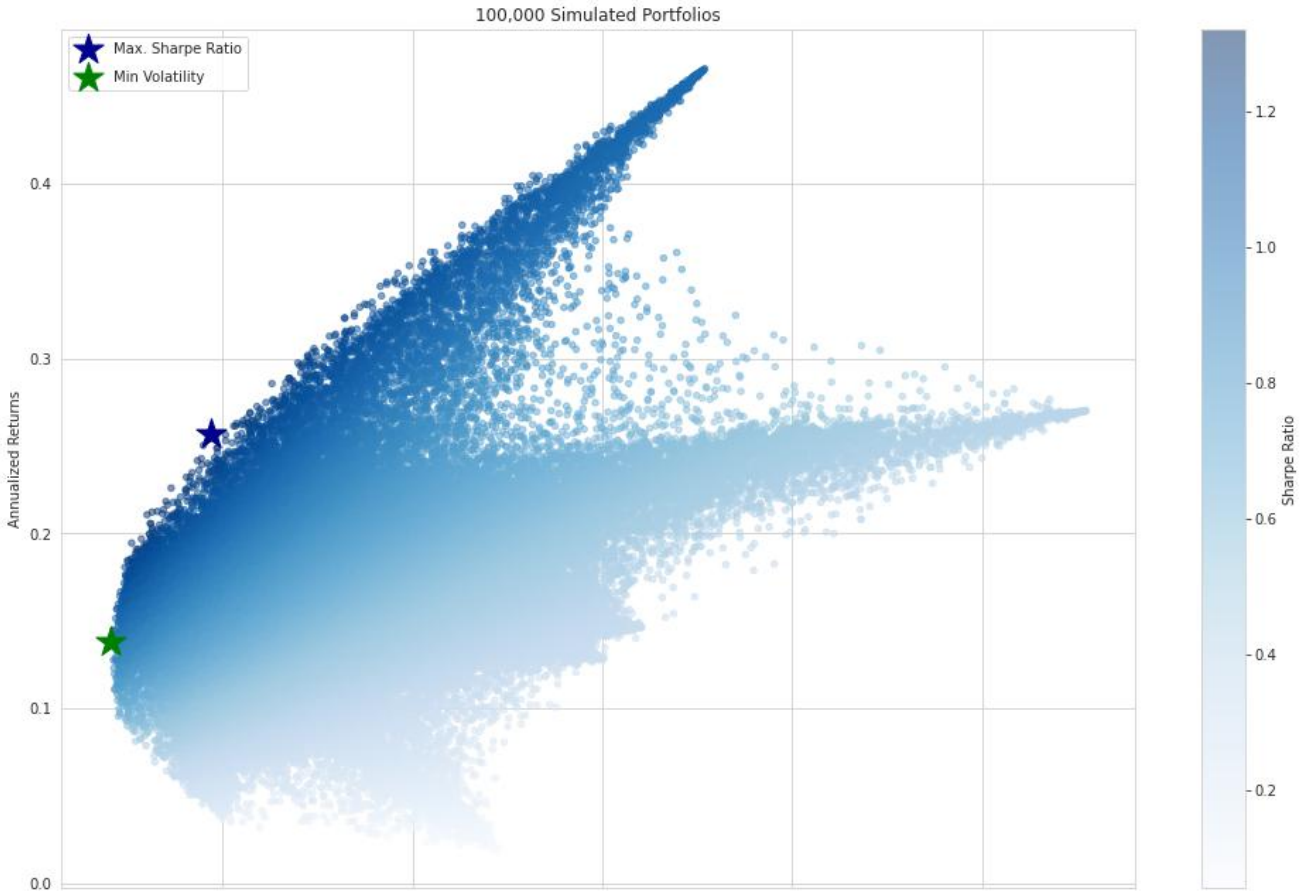
In [22]:

```
ax = simul_perf.plot.scatter(x=0, y=1, c=2, cmap='Blues',
                             alpha=0.5, figsize=(14, 9), colorbar=True,
                             title=f'{NUM_PF:,d} Simulated Portfolios')

max_sharpe_idx = simul_perf.iloc[:, 2].idxmax()
sd, r = simul_perf.iloc[max_sharpe_idx, :2].values
print(f'Max Sharpe: {sd:.2%}, {r:.2%}')
ax.scatter(sd, r, marker='*', color='darkblue', s=500, label='Max. Sharpe Ratio')

min_vol_idx = simul_perf.iloc[:, 0].idxmin()
sd, r = simul_perf.iloc[min_vol_idx, :2].values
ax.scatter(sd, r, marker='*', color='green', s=500, label='Min Volatility')
plt.legend(labelspacing=1, loc='upper left')
```

```
plt.tight_layout()
Max Sharpe: 19.42%, 25.70%
```



Compute Annualize PF Performance

Now we'll set up the quadratic optimization problem to solve for the minimum standard deviation for a given return or the maximum SR.

To this end, define the functions that measure the key metrics:

In [23]:

```
def portfolio_std(wt, rt=None, cov=None):
    """Annualized PF standard deviation"""
    return np.sqrt(wt @ cov @ wt * periods_per_year)
```

In [24]:

```
def portfolio_returns(wt, rt=None, cov=None):
    """Annualized PF returns"""
    return (wt @ rt + 1) ** periods_per_year - 1
```

In [25]:

```
def portfolio_performance(wt, rt, cov):
    """Annualized PF returns & standard deviation"""
    r = portfolio_returns(wt, rt=rt)
    sd = portfolio_std(wt, cov=cov)
    return r, sd
```

Max Sharpe PF

Define a target function that represents the negative SR for scipy's minimize function to optimize, given the constraints that the weights are bounded by [-1, 1], if short trading is permitted, and [0, 1] otherwise, and sum to one in absolute terms.

In [26]:
```
def neg_sharpe_ratio(weights, mean_ret, cov):
    r, sd = portfolio_performance(weights, mean_ret, cov)
    return -(r - rf_rate) / sd
```

In [27]:
```
weight_constraint = {'type': 'eq',
                     'fun': lambda x: np.sum(np.abs(x))-1}
```

In [28]:
```
def max_sharpe_ratio(mean_ret, cov, short=False):
    return minimize(fun=neg_sharpe_ratio,
                    x0=x0,
                    args=(mean_ret, cov),
                    method='SLSQP',
                    bounds=((-1 if short else 0, 1),) * n_assets,
                    constraints=weight_constraint,
                    options={'tol':1e-10, 'maxiter':1e4})
```

Compute Efficient Frontier

The solution requires iterating over ranges of acceptable values to identify optimal risk-return combinations

In [29]:
```
def min_vol_target(mean_ret, cov, target, short=False):

    def ret_(wt):
        return portfolio_returns(wt, mean_ret)

    constraints = [{'type': 'eq',
                    'fun': lambda x: ret_(x) - target},
                   weight_constraint]

    bounds = ((-1 if short else 0, 1),) * n_assets
    return minimize(portfolio_std,
                    x0=x0,
                    args=(mean_ret, cov),
                    method='SLSQP',
                    bounds=bounds,
                    constraints=constraints,
                    options={'tol': 1e-10, 'maxiter': 1e4})
```

Min Volatility Portfolio

In [30]:

```
def min_vol(mean_ret, cov, short=False):
    bounds = ((-1 if short else 0, 1),) * n_assets

    return minimize(fun=portfolio_std,
                    x0=x0,
                    args=(mean_ret, cov),
                    method='SLSQP',
                    bounds=bounds,
                    constraints=weight_constraint,
                    options={'tol': 1e-10, 'maxiter': 1e4})
```

In [31]:

```
def efficient_frontier(mean_ret, cov, ret_range, short=False):
    return [min_vol_target(mean_ret, cov, ret) for ret in ret_range]
```

Run Calculation

Get random PF

In [32]:

```
simul_perf, simul_wt = simulate_portfolios(mean_returns, cov_matrix, short=False)
```

In [33]:

```
print(simul_perf.describe())
```

```
       Annualized Standard Deviation  Annualized Returns   Sharpe Ratio
count                  100000.000000       100000.000000  100000.000000
mean                        0.272889            0.177186       0.675622
std                         0.076975            0.055340       0.213748
min                         0.139966            0.019720       0.055767
25%                         0.217795            0.148334       0.511378
50%                         0.260579            0.173843       0.665910
75%                         0.314480            0.202062       0.834405
max                         0.654543            0.466036       1.332866
```

In [34]:

```
simul_max_sharpe = simul_perf.iloc[:, 2].idxmax()
simul_perf.iloc[simul_max_sharpe]
```

Out[34]:

```
Annualized Standard Deviation    0.166130
Annualized Returns               0.221928
Sharpe Ratio                     1.332866
Name: 67110, dtype: float64
```

Get Max Sharpe PF

In [35]:

```
max_sharpe_pf = max_sharpe_ratio(mean_returns, cov_matrix, short=False)
max_sharpe_perf = portfolio_performance(max_sharpe_pf.x, mean_returns, cov_matrix)
```

In [36]:

```
r, sd = max_sharpe_perf
pd.Series({'ret': r, 'sd': sd, 'sr': (r-rf_rate)/sd})
```

Out[36]:

```
ret    0.228716
sd     0.166360
sr     1.371822
dtype: float64
```

From simulated pf data

Get Min Vol PF

In [37]:

```
min_vol_pf = min_vol(mean_returns, cov_matrix, short=False)
min_vol_perf = portfolio_performance(min_vol_pf.x, mean_returns, cov_matrix)
```

Get Efficent PFs

In [38]:

```
ret_range = np.linspace(simul_perf.iloc[:, 1].min(), simul_perf.iloc[:, 1].max(),
50)
eff_pf = efficient_frontier(mean_returns, cov_matrix, ret_range, short=True)
eff_pf = pd.Series(dict(zip([p['fun'] for p in eff_pf], ret_range)))
```

Plot Result

The simulation yields a subset of the feasible portfolios, and the efficient frontier identifies the optimal in-sample return-risk combinations that were achievable given historic data.

The below figure shows the result including the minimum variance portfolio and the portfolio that maximizes the SR and several portfolios produce by alternative optimization strategies. The efficient frontier

In [39]:

```
fig, ax = plt.subplots()
simul_perf.plot.scatter(x=0, y=1, c=2, ax=ax, cmap='Blues',alpha=0.25,
                        figsize=(14, 9), colorbar=True)

eff_pf[eff_pf.index.min():].plot(linestyle='--', lw=2, ax=ax, c='k',
                                 label='Efficient Frontier')

r, sd = max_sharpe_perf
ax.scatter(sd, r, marker='*', color='k', s=500, label='Max Sharpe Ratio PF')

r, sd = min_vol_perf
ax.scatter(sd, r, marker='v', color='k', s=200, label='Min Volatility PF')

kelly_wt = precision_matrix.dot(mean_returns).clip(lower=0).values
kelly_wt /= np.sum(np.abs(kelly_wt))
r, sd = portfolio_performance(kelly_wt, mean_returns, cov_matrix)
ax.scatter(sd, r, marker='D', color='k', s=150, label='Kelly PF')
```

```
std = weekly_returns.std()
std /= std.sum()
r, sd = portfolio_performance(std, mean_returns, cov_matrix)
ax.scatter(sd, r, marker='X', color='k', s=250, label='Risk Parity PF')

r, sd = portfolio_performance(np.full(n_assets, 1/n_assets), mean_returns,
cov_matrix)
ax.scatter(sd, r, marker='o', color='k', s=200, label='1/n PF')


ax.legend(labelspacing=0.8)
ax.set_xlim(0, eff_pf.max()+.4)
ax.set_title('Mean-Variance Efficient Frontier', fontsize=16)
ax.yaxis.set_major_formatter(FuncFormatter(lambda y, _: '{:.0%}'.format(y)))
ax.xaxis.set_major_formatter(FuncFormatter(lambda y, _: '{:.0%}'.format(y)))
sns.despine()
fig.tight_layout();
```