# ארכיטקטורה למערכת מסחר אלגוריתמי מבוססת סוכנים מרובים והטמעת נתונים

## 1. תכנון מערכת המסחר האלגוריתמי מבוססת סוכנים מרובים: סקירה כללית

### 1.1. חזון המערכת: סינרגיה בין רכיבים

החזון המוצע מתאר מערכת מסחר אלגוריתמי מורכבת, המבוססת על מספר מטריצות נתונים, כאשר כל מטריצה מייצגת סוג מידע ייחודי (כגון נתוני מחיר/נפח, ספר פקודות, סנטימנט חדשותי, נתונים פונדמנטליים). ליבת המערכת כוללת רשת נוירונים שתפקידה להטמיע (embed) מטריצות אלו למרחב וקטורי מאוחד, ובכך להפחית את ההטרוגניות הטבועה במקורות המידע המגוונים. סביב הטמעה זו פועלים סוכנים חכמים, כאשר לכל מטריצת נתונים מוקצה סוכן ייעודי. סוכנים אלו ממנפים את ההטמעות הווקטוריות לצורך קבלת החלטות מסחר. המטרה המרכזית היא ליצור מערכת בה סוכנים מומחים משתפים פעולה, כשהם מוזנים מתמונה כוללת ומאוחדת של הנתונים, לצורך ביצוע מסחר בזמן אמת.

גישה זו, בה לכל מטריצת נתונים מוקצה סוכן ייעודי, מרמזת על ארכיטקטורה דמוית מיקרו-שירותים (microservices) בשכבת הבינה של המערכת. כל צמד סוכן-מטריצה יכול להיחשב כ"שירות" מומחה, האחראי על עיבוד נתונים וייצור אותות ראשוניים מסוג מידע ספציפי. מודולריות זו מועילה לפיתוח, תחזוקה והרחבה של המערכת, אך היא גם מחייבת מנגנוני תקשורת חזקים בין הסוכנים ושכבת אגרגציה או קבלת החלטות מרכזית. שכבה זו עשויה להתממש כסוכן "מנהל" או כמנגנון קונצנזוס. הדבר מעלה שאלות לגבי המבנה הכולל של מערכת הסוכנים המרובים (MAS - Multi-Agent System), כגון האם יהיה היררכי, מבוסס צוותים, או מבנה אחר, נושא שיידון בהמשך.

### 1.2. יתרונות מרכזיים של הגישה המוצעת

* **מודולריות והתמחות:** הקצאת מטריצות נתונים ספציפיות לסוכנים ייעודיים מאפשרת התמחות ממוקדת ופיתוח ותחזוקה קלים יותר של רכיבים בודדים. כל סוכן יכול להפוך ל"מומחה" בניתוח סוג הנתונים הייחודי לו.
* **טיפול בהטרוגניות נתונים:** האתגר המשמעותי של מגוון פורמטים ומאפיינים של נתונים פיננסיים מטופל באמצעות שכבת ההטמעה. שכבה זו מהווה פתרון מתוחכם ליצירת קלט קוהרנטי ומאוחד לתהליכי קבלת ההחלטות. הצלחתה של שכבת ההטמעה היא קריטית; היא מהווה את הגשר בין נתונים גולמיים ומגוונים לבין בינה מאוחדת וניתנת לפעולה. אם ההטמעה תיכשל ביצירת וקטורים משמעותיים ומייצגים, הלוכדים יחסים בין-מודאליים (למשל, כיצד סנטימנט חדשותי מתואם עם תנועות מחירים), הסוכנים המומחים, גם אם יתוכננו היטב, יפעלו על בסיס קלט פגום. איכות ההטמעה משפיעה ישירות על המטרה של "הפחתת הטרוגניות" , ומדגישה את הצורך בבחירה קפדנית ובאימון של מתודולוגיית ההטמעה.
* **פוטנציאל לייצור אותות משופר:** על ידי שילוב תובנות ממקורות נתונים מגוונים (למשל, טכניים, פונדמנטליים, סנטימנט), המערכת עשויה לחשוף אותות מסחר חזקים ומורכבים יותר מאשר מערכות המסתמכות על סוג נתונים יחיד.

### 1.3. אתגרים ארכיטקטוניים מרכזיים

* **אינטגרציה וסנכרון נתונים:** הבטחת זרימה מדויקת ובזמן של נתונים למטריצות המתאימות.
* **אפקטיביות ההטמעה:** תכנון רשת הטמעה שתלכוד באמת את יחסי הגומלין בין נתונים הטרוגניים.
* **תיאום סוכנים:** הגדרת האופן שבו אותות או תובנות מסוכנים בודדים מאוחדים לכדי החלטת מסחר סופית.
* **עקביות בין בדיקה לאחור (Backtesting) למסחר חי:** האתגר הקריטי של הבטחת ביצועי המערכת כצפוי בעת פריסתה בשווקים חיים. נושא זה יהיה מרכזי לאורך כל הדיון.

## 2. הנחת היסודות: הכנת נתונים ובניית מטריצות

### 2.1. שיטות עבודה מומלצות להזנה, ניקוי ואימות של נתונים פיננסיים

* **מקורות נתונים והזנה:** דיון במקורות נפוצים לנתונים פיננסיים (APIs, ספקי נתונים למחיר/נפח, APIs חדשותיים לסנטימנט, ספקי נתוני ספר פקודות). יש להדגיש את הצורך בהזנות נתונים אמינות ובעלות השהיה נמוכה, במיוחד למסחר בזמן אמת.
* **ניקוי נתונים:**
  + טיפול בערכים חסרים (למשל, טכניקות השלמה כגון ממוצע, חציון, או שיטות מתוחכמות יותר כמו Multiple Imputation (MI), אם כי יש לנקוט זהירות עם נתוני סדרות זמן).
  + זיהוי וטיפול בחריגים (למשל, ניתוח Z-score, טווח בין-רבעוני (IQR), וינזוריזציה), תוך הכרה בכך שחלק מה"חריגים" בתחום הפיננסי הם אירועי קיצון אמיתיים.
  + תיקון שגיאות, חוסר עקביות (למשל, סימולי טיקר שונים לאותו נכס ממקורות שונים), ואי-דיוקים.
* **אימות נתונים:**
  + הבטחת דיוק, שלמות ועקביות הנתונים (למשל, חותמות זמן אחידות, מטבע).
  + בדיקות אימות אוטומטיות בצנרת הנתונים לאיתור מוקדם של בעיות. הצלבת נתונים מול מספר מקורות במידת האפשר.
  + הבנת ניואנסים בנתונים, כגון הטיית שרידות (survivorship bias) והטיית הצצה לעתיד (look-ahead bias), וכיצד למתן אותם במהלך איסוף ועיבוד הנתונים.

### 2.2. הנדסת תכונות אסטרטגית לסוגי נתונים פיננסיים מגוונים

* **תפיסה:** הפיכת נתונים גולמיים לקלטים משמעותיים (תכונות) המשפרים את יכולת החיזוי של המודל. שלב זה חיוני לפני אכלוס המטריצות.
* **נתוני מחיר/נפח (תכונות טכניות):**
  + נתוני OHLC (Open, High, Low, Close), תשואות, ממוצעים נעים (SMA, EMA), מדדי תנודתיות (למשל, ATR, סטיית תקן), מדדי מומנטום (RSI, MACD), VWAP (Volume Weighted Average Price).
  + שיקולים לגבי סטציונריות (למשל, שימוש בתשואות במקום במחירים גולמיים, מבחן דיקי-פולר, דיפרנציאציה חלקית).
* **נתוני ספר פקודות:**
  + תכונות כגון מרווח קנייה-מכירה (bid-ask spread), עומק, חוסר איזון פקודות (order imbalance), חוסר איזון בזרימת עסקאות (trade flow imbalance). נתונים אלו הם בתדירות גבוהה ודורשים עיבוד מיוחד.
  + LOBFrame ככלי קוד פתוח לעיבוד נתוני ספר פקודות.
* **נתוני סנטימנט (למשל, מחדשות, רשתות חברתיות):**
  + שימוש במודלים מאומנים מראש כמו FinBERT לניקוד סנטימנט פיננסי.
  + אגרגציה של ניקודי סנטימנט על פני חלונות זמן עבור נכסים ספציפיים.
  + אתגרים: רעש, סובייקטיביות, וקישור סנטימנט להשפעה על השוק. יש לשקול תגובות שוק למילים כמדד אמין יותר.
* **נתונים פונדמנטליים (אם רלוונטי):**
  + מדדים כגון יחס מחיר לרווח (P/E ratio), רווח למניה (EPS), צמיחת הכנסות. בדרך כלל בתדירות נמוכה יותר; דורשים התאמה קפדנית לנתונים בתדירות גבוהה יותר.
  + אתגרי מיזוג עם תדירויות נתונים שונות.
* **נתונים אלטרנטיביים:** אזכור קצר של מקורות נתונים פוטנציאליים אחרים (למשל, תמונות לוויין, נתוני שרשרת אספקה) ואתגרי הנדסת התכונות שהם מציבים.
* **שיטות עבודה מומלצות:**
  + ידע תחום (domain knowledge) הוא מפתח לבחירת תכונות.
  + מזעור רעש תוך שמירה על אותות חשובים.
  + ניתוח מתאמים (correlation analysis) להסרת תכונות מיותרות.

השאלה של המשתמש לגבי "חלוקה אקראית של נתונים" מתייחסת ככל הנראה לחלוקה לקבוצות אימון/בדיקה/אימות, אך יישומה חייב להיות מותאם לסדרות זמן פיננסיות ולסוגי נתונים שונים בתוך המטריצות. בעוד שערבוב אקראי הוא סטנדרטי עבור נתונים בלתי תלויים ושווי התפלגות (i.i.d) , סדרות זמן פיננסיות מפגינות לעיתים קרובות מתאם-עצמי (auto-correlation) ואי-סטציונריות. חלוקה אקראית של נתונים סדורים בזמן עלולה להוביל להטיית הצצה לעתיד, כאשר המודל לומד בטעות ממידע עתידי, וכתוצאה מכך מתקבלות תוצאות בדיקה לאחור אופטימיות מדי. עבור מטריצות המכילות תכונות של סדרות זמן (למשל, טכניקות מחיר, מצבי ספר פקודות), חלוקה מודעת-זמן (למשל, חלוקה כרונולוגית, TimeSeriesSplit מ-) היא חיונית לאימון הסוכנים ואימות רשת ההטמעה. עבור סוגי נתונים אחרים שעשויים להיות פחות תלויים בזמן (למשל, בסיס ידע סטטי של נתונים פונדמנטליים של חברות, אם כי גם זה משתנה עם הזמן), חלוקה אקראית עשויה להיות מתאימה יותר לבדיקות מסוימות של אימון מקדים או רכיבים. הדו"ח חייב להנחות את המשתמש מתי וכיצד ליישם אסטרטגיות חלוקה שונות עבור חלקים שונים של המערכת שלהם.

הנדסת תכונות אינה משימה חד-פעמית אלא תהליך איטרטיבי שחייב להיות עקבי בין בדיקה לאחור למסחר חי. בחירת התכונות בתוך כל מטריצה מגדירה את ה"שפה" שהסוכן שלה "מדבר". מדגיש את תפקידה של הנדסת תכונות במסחר בתדירות גבוהה (HFT), ו- מדגישים עקביות. אם סוכן מאומן על מטריצה עם קבוצת תכונות A במהלך בדיקה לאחור, הוא *חייב* לקבל מטריצה עם קבוצת תכונות A (שנגזרה באופן זהה) במסחר חי. כל אי-התאמה שוברת את העקביות הלוגית. יתר על כן, התכונות שנבחרו למטריצה (למשל, מומנטום לטווח קצר למטריצה 1 לעומת ערך לטווח ארוך למטריצה 4) מעצבות ישירות את היכולות והמיקוד של הסוכן המוקצה לאותה מטריצה. זהו קשר קריטי בין הכנת נתונים לתכנון סוכנים.

### 2.3. הגדרה ואכלוס של מטריצות נתונים: התאמת סוגי נתונים למטריצות

* **הרעיון המרכזי של המשתמש:** כל מטריצה מכילה סוג נתונים ספציפי. סעיף זה מפרט זאת.
* **מבנה מטריצה לדוגמה:**
  + **מטריצה 1 (טכניקות מחיר/נפח):** שורות = צעדי זמן, עמודות = תכונות כגון (מחיר מנורמל, תשואה, ממוצע נע 20 יום, RSI, שינוי בנפח).
  + **מטריצה 2 (מיקרו-מבנה ספר פקודות):** שורות = צעדי זמן (פוטנציאלית בגרנולריות עדינה הרבה יותר), עמודות = תכונות כגון (מרווח קנייה-מכירה, עומק ב-N רמות, חוסר איזון פקודות).
  + **מטריצה 3 (סנטימנט חדשותי):** שורות = צעדי זמן (מותאמות לנתוני מחיר), עמודות = תכונות כגון (ניקוד סנטימנט לנכס A, ניקוד סנטימנט לשוק, סוג אירוע חדשותי).
  + **מטריצה 4 (נתונים פונדמנטליים):** שורות = צעדי זמן (למשל, יומי או רבעוני, ולאחר מכן מילוי קדימה או אינטרפולציה בזהירות), עמודות = (P/E, צמיחת EPS).
* **סנכרון:** אתגר קריטי של התאמת נתונים ממקורות ותדירויות שונות למטריצות אלו. חותמות זמן הן מפתח; יש להחליט על תדירות מאסטר למערכת או לטפל בזרימות נתונים אסינכרוניות.
* **נורמליזציה/סקיילינג:** כל מטריצה עשויה לדרוש סקיילינג פנימי משלה (למשל, סטנדרטיזציה לתכונות מחיר, מינימום-מקסימום ל-RSI) לפני הזנתה לרשת ההטמעה. זה נפרד מהפחתת ההטרוגניות על ידי שכבת ההטמעה עצמה. מדגיש צורך מוגבר בנורמליזציה עם מערכי נתונים הטרוגניים.
* **בנייה עבור MAS הטרוגני:** האופן שבו נתונים בנויים למטריצות משפיע ישירות על האופן שבו ניתן להתמחות בסוכנים. כל מטריצה הופכת ל"תמונת העולם" עבור הסוכן הייעודי שלה.

הפרדת נתונים למטריצות נפרדות, על אף שהיא טובה להתמחות, מציגה אתגר של "סילואי מידע" (information siloing) ששכבת ההטמעה חייבת להתגבר עליו. על ידי יצירת מטריצות נפרדות למחיר, סנטימנט וכו', אנו מפשטים את משימות הסוכנים הבודדים. עם זאת, ה"אלפא" האמיתית טמונה לעיתים קרובות ב*אינטראקציה* בין סוגי נתונים שונים אלה (למשל, כיצד סנטימנט חדשותי *מקדים* או *מאשר* תנועת מחיר). אם שכבת ההטמעה (סעיף 3) לא תלמד ביעילות את היחסים הבין-מטריציוניים הללו, המערכת תהיה פחות חזקה ממערכת שיכולה לראות באופן הוליסטי את כל הנתונים הגולמיים. זה מדגיש את חשיבות טכניקת ההטמעה (למשל, מיקוד ה-CODE על התרחשות משותפת ב-).

## 3. איחוד נתונים מגוונים: הטמעת מטריצות להפחתת הטרוגניות

### 3.1. אתגר ההטרוגניות בנתונים פיננסיים רב-מקורות

נתונים פיננסיים ממקורות מרובים (מחיר, נפח, חדשות, נתונים פונדמנטליים, ספרי פקודות) הם הטרוגניים מטבעם: סולמות, יחידות, התפלגויות שונות (למשל, תשואות מחירים לעומת ציוני סנטימנט), תדירויות זמניות ומבני נתונים שונים. שרשור פשוט או שילוב בסיסי של נתונים כה מגוונים עלול להוביל לכך שמודלים יושפעו באופן דומיננטי מתכונות מסוימות או ייכשלו בלכידת תלויות מורכבות.

### 3.2. עקרונות ליבה של הטמעת נתונים ליישומים פיננסיים

* **הגדרה:** הטמעה כתהליך של הפיכת נתונים בעלי ממדיות גבוהה, בדידים או הטרוגניים לייצוגים וקטוריים רציפים בעלי ממדיות נמוכה יותר.
* **מטרה:** יצירת "שפה משותפת" או מרחב תכונות מאוחד שבו ניתן להשוות ולשלב באופן משמעותי סוגי נתונים מגוונים. מטרת המשתמש של "הפחתת הטרוגניות" מושגת על ידי מיפוי נתונים שונים למרחב וקטורי משותף זה.
* **יתרונות:** מאפשר קלט לאלגוריתמים של למידת מכונה המצפים לווקטורים רציפים , יכול לחשוף דפוסים ויחסים חבויים , ויכול לשפר את ביצועי המודל ואת יכולת ההכללה שלו.

### 3.3. צלילה עמוקה: יישום הטמעת נתוני התרחשות משותפת (CODE) עבור מטריצות פיננסיות הטרוגניות

* **רלוונטיות:** CODE תוכנן במפורש להטמעת אובייקטים מ*סוגים שונים* למרחב אוקלידי משותף בהתבסס על סטטיסטיקות ההתרחשות המשותפת שלהם. הדבר תואם ישירות את צורך המשתמש לשלב מידע ממטריצות נתונים פיננסיות שונות.
* **כיצד CODE פועל (סיכום מ-):**
  + ממדל את ההסתברות המותנית p(y|x) בהתבסס על המרחק האוקלידי d^2\_{x,y} במרחב ההטמעה: p(y|x) = \bar{p}(y) / Z(x) \* e^{-d^2\_{x,y}}.
  + לומד מיפויים φ~(X) \rightarrow R^d ו-ψ~(Y) \rightarrow R^d עבור שתי קבוצות משתנים הטרוגניות X ו-Y.
  + ממקסם פונקציית נראות (likelihood) המאזנת בין מזעור מרחק ריבועי ממוצע לבין איבר רגולריזציה.
* **יישום למטריצות הפיננסיות של המשתמש:**
  + **התרחשות משותפת זוגית:** CODE ייושם על זוגות של מטריצות (או ליתר דיוק, על סוגי אירועים/תכונות הנגזרים ממטריצות אלו). לדוגמה:
    - מטריצה 1 (אירועי מחיר/נפח) ומטריצה 3 (אירועי סנטימנט חדשותי).
    - מטריצה 1 (אירועי מחיר/נפח) ומטריצה 2 (אירועי ספר פקודות).
  + **הגדרת "אירועים" ו"התרחשות משותפת":**
    - נתונים רציפים במטריצות (למשל, שינוי מחיר, ציון סנטימנט) צריכים לעבור דיסקרטיזציה ל"אירועים" משמעותיים (למשל, "עליית מחיר גדולה", "סנטימנט ניטרלי", "חוסר איזון גבוה בצד המכירה").
    - התרחשות משותפת מוגדרת אז כתדירות שבה אירועים בדידים אלו ממטריצות שונות מופיעים יחד באותו חלון זמן (למשל, "עליית מחיר גדולה" המתרחשת במשותף עם "סנטימנט חדשותי חיובי").
  + **לימוד הטמעות משותפות:** עבור כל זוג, CODE לומד הטמעות לאירועים משתי המטריצות ל*אותו* מרחב בעל ממדיות נמוכה. משמעות הדבר היא שניתן להשוות ישירות וקטור "עליית מחיר גדולה" ווקטור "סנטימנט חדשותי חיובי" באמצעות המרחק ביניהם.
* **שלבי יישום מעשיים (מבוסס על ):**
  1. **עיבוד מקדים ודיסקרטיזציה של נתונים:** כפי שנדון בסעיף 2. יצירת "אירועים" קטגוריאליים מנתוני מטריצה רציפים.
  2. **בניית מטריצות/טבלאות התרחשות משותפת:** עבור כל זוג סוגי אירועים ממטריצות נתונים שונות, יש לספור את ההתרחשויות המשותפות שלהם מנתונים היסטוריים. זה יוצר את \bar{p}(x, y) הדרוש ל-CODE. ( מזכיר חישוב מטריצת התרחשות משותפת עבור הזמנות רכש, שניתן להתאים לכאן).
  3. **אימון מודל CODE:** יישום או שימוש בספרייה עבור CODE. האופטימיזציה יכולה להתבצע באמצעות עלייה בגרדיאנט מצומד (conjugate gradient ascent) או על ידי ניסוח מחדש כאופטימיזציה קמורה על פני מטריצות חיוביות למחצה (PSD).
  4. **חילוץ הטמעות:** ה-φ~(event\_A) וה-ψ~(event\_B) הנלמדים הם הייצוגים הווקטוריים.
* **שילוב הטמעות מזוגות מרובים:** אם CODE מיושם באופן זוגי, נדרשת אסטרטגיה ליצירת וקטור מאוחד יחיד עבור מצב המערכת הכולל. זה יכול לכלול שרשור של ההטמעות הקשורות לחותמת זמן ספציפית או שימוש בשכבת רשת נוירונים נוספת לשילובן. לחלופין, ניתן לחקור הרחבות של CODE ליותר משני סוגי משתנים.

בחירת טכניקת ההטמעה היא החלטה ארכיטקטונית קריטית המכתיבה כיצד יילמדו וייוצגו יחסים בין-מטריציוניים. CODE לומד במפורש הטמעות המבוססות על *התרחשות משותפת* בין סוגים הטרוגניים, מה שאידיאלי למציאת קשרים ישירים (למשל, "סנטימנט חיובי מתרחש לעיתים קרובות במשותף עם עליית מחיר"). אוטואנקודרים לומדים ייצוגים יעילים לשחזור, שעשויים ללכוד היטב דפוסים תוך-מטריציוניים אך יחסים בין-מטריציוניים פחות ישירות, אלא אם כן תוכננו כאוטואנקודר משותף. הטמעות גרפים מצטיינות אם ניתן להגדיר מבנה גרף משמעותי המייצג יחסים אלו. המשתמש צריך לבחור בהתבסס על האם הוא נותן עדיפות להתרחשות משותפת סטטיסטית מפורשת (CODE), לדחיסה לא ליניארית כללית (אוטואנקודרים), או ליחסים דמויי רשת (הטמעות גרפים).

דיסקרטיזציה של נתונים פיננסיים רציפים עבור שיטות כמו CODE היא שלב של הנדסת תכונות בפני עצמו ויכולה להשפיע באופן משמעותי על איכות ההטמעה. CODE, כפי שתואר ב-, עובד עם משתנים קטגוריאליים לחישוב התרחשות משותפת. נתונים פיננסיים במטריצות (מחיר, נפח, ציוני סנטימנט) הם לעיתים קרובות רציפים. תהליך החלוקה לקטגוריות (binning) או הגדרת "אירועים" (למשל, "עליית מחיר > 2%", "ציון סנטימנט 0.5-0.7") אינו טריוויאלי. דיסקרטיזציה גרועה עלולה להסתיר יחסים אמיתיים או להכניס רעש, ולהוביל להטמעות לא אופטימליות. שלב זה דורש מומחיות תחום ושיפור איטרטיבי.

"רשת ההטמעה" אינה רק טרנספורמטור פסיבי; מטרת האימון שלה מגדירה באופן מרומז מה משמעות "הפחתת הטרוגניות" עבור המערכת. אם רשת ההטמעה מאומנת באופן בלתי מונחה (כמו אוטואנקודרים בסיסיים או CODE על התרחשויות משותפות היסטוריות), היא לומדת יחסים כלליים. אם היא מאומנת (או מכווננת) כחלק ממשימה מונחית מקצה לקצה (למשל, חיזוי תנועת המחיר הבאה), ההטמעות יעברו אופטימיזציה עבור *אותה משימה ספציפית*. משמעות הדבר היא ש"הפחתת ההטרוגניות" מכוונת לייצור וקטורים שהם דיסקרימינטיביים עבור יעד החיזוי. זהו שיקול חיוני לאופן שבו הסוכנים ישתמשו בסופו של דבר בהטמעות אלו.

### 3.4. טכניקות הטמעה חלופיות ומשלימות

* **אוטואנקודרים (Autoencoders) :**
  + **עיקרון:** רשתות נוירונים המאומנות לשחזר את הקלט שלהן, עם שכבת צוואר בקבוק הלומדת ייצוג דחוס (מוטמע).
  + **יישום:** כל מטריצה יכולה להיות מוזנת דרך אוטואנקודר משלה כדי לקבל הטמעה, או שאוטואנקודר גדול יותר יכול לנסות ללמוד הטמעה משותפת מקלטים של מטריצות משורשרות (אך מנורמלות מראש). ייצוג מרחב סמוי (latent space) הוא ההטמעה.
  + **יתרונות:** טוב להפחתת ממדיות, יכול ללמוד יחסים לא ליניאריים.
  + **חסרונות:** ייתכן שלא יעבור אופטימיזציה מפורשת ללכידת יחסים *בין* סוגים הטרוגניים באופן ישיר כמו CODE.
* **הטמעות גרפים (למשל, Node2Vec, GraphSAGE, GCNs) :**
  + **עיקרון:** אם ניתן למדל יחסים בין נקודות נתונים שונות (אפילו על פני מטריצות) כגרף (למשל, צמתים הם נכסים, תכונות או אירועים, וקשתות מייצגות מתאמים, התרחשויות משותפות או קשרים סיבתיים), אז טכניקות הטמעת גרפים יכולות ללמוד וקטורי צמתים.
  + **יישום:** בניית גרף הטרוגני שבו צמתים מייצגים, לדוגמה, מניות ספציפיות, אירועי מחיר בדידים, קטגוריות סנטימנט ומצבי ספר פקודות. קשתות יכולות לייצג את ההתרחשויות המשותפות הנצפות שלהם או יחסים מוגדרים אחרים.
  + **יתרונות:** חזק ללכידת מבנים יחסיים מורכבים.
  + **חסרונות:** דורש בניית גרף קפדנית, שיכולה להיות מורכבת עבור נתונים פיננסיים.
* **מודלים מבוססי טרנספורמר (קשב עצמי - Self-Attention):** אמנם לא פורטו במפורש בהקשר זה במקורות, אך ארכיטקטורות מודרניות כמו טרנספורמרים ניתנות להתאמה. מנגנון הקשב העצמי יכול לשקול את חשיבותן של תכונות/אלמנטים שונים ממטריצות מגוונות בעת יצירת הטמעה הקשרית.

### 3.5. בניית רשת ההטמעה

זוהי ה"רשת שמטמיעה אותם כווקטור ומורידה הטרוגניות" משאלת המשתמש. היא יכולה להיות יישום ייעודי של CODE, קבוצת אוטואנקודרים שהייצוגים הסמויים שלהם משולבים לאחר מכן, או ארכיטקטורת למידה עמוקה מורכבת יותר.

* **קלט:** מטריצות הנתונים הנפרדות.
* **פלט:** וקטור מאוחד יחיד (או קבוצת וקטורים קשורים) המייצג את המצב המשולב מכל המטריצות עבור צעד זמן נתון.
* **אימון:** רשת זו צריכה להיות מאומנת. אם משתמשים ב-CODE, זה מבוסס על נראות התרחשות משותפת. אם אוטואנקודרים, זהו אובדן שחזור. אם עבור משימה במורד הזרם (למשל, חיזוי מחירים), ניתן לאמן אותה מקצה לקצה.

**טבלה 1: השוואת טכניקות הטמעת נתונים עבור נתונים פיננסיים הטרוגניים**

| טכניקה | עיקרון | טיפול בהטרוגניות | יתרונות | חסרונות | התאמה למערכת רב-מטריציונית | מקורות |
| --- | --- | --- | --- | --- | --- | --- |
| **CODE** | סטטיסטיקות התרחשות משותפת | מיועדת במפורש להטמעת סוגי אובייקטים שונים למרחב משותף על בסיס התרחשותם. | לוכדת יחסים סטטיסטיים ישירים בין סוגי נתונים שונים. | דורשת דיסקרטיזציה של נתונים רציפים; מיושמת לרוב זוגית. | גבוהה מאוד |  |
| **אוטואנקודרים** | מזעור אובדן שחזור, למידת ייצוג דחוס | כל מטריצה יכולה להיות מוטמעת בנפרד, או אוטואנקודר משותף יכול ללמוד על נתונים משולבים. | טובים להפחתת ממדיות ולמידת יחסים לא ליניאריים בתוך כל סוג נתונים. | יחסים בין-מטריציוניים עשויים לא להילכד באופן מפורש אלא אם כן תוכננו במיוחד. | בינונית עד גבוהה |  |
| **הטמעות גרפים** | אגרגציית שכנים, לכידת מבנה רשתי | דורשת בניית גרף הטרוגני המייצג יחסים בין אלמנטים ממטריצות שונות. | חזקות בלכידת מבנים יחסיים מורכבים אם ניתן להגדיר גרף משמעותי. | בניית גרף יכולה להיות מורכבת ולא טריוויאלית עבור נתונים פיננסיים מגוונים. | בינונית |  |
| **טרנספורמרים** | קשב עצמי (Self-Attention) | יכולים לשקול באופן דינמי את החשיבות של תכונות שונות ממטריצות שונות ליצירת ייצוג הקשרי. | גמישים מאוד, יכולים ללמוד יחסים מורכבים וארוכי טווח. | דורשים כמויות גדולות של נתונים לאימון; עשויים להיות יקרים חישובית. | גבוהה | (ידע כללי) |

טבלה זו מספקת סקירה השוואתית ברורה של אסטרטגיות ההטמעה העיקריות, ומאפשרת למשתמש לקבל החלטה מושכלת בהתבסס על מאפייני הנתונים הספציפיים שלו, יכולותיו הטכניות וסוגי היחסים שהוא מעוניין ללכוד בין מטריצות הנתונים שלו. היא עונה ישירות על חלק ה"כיצד להטמיע" בשאלתו על ידי הצגת אפשרויות ופשרות.

## 4. תכנון ליבת הבינה: מערכת הסוכנים המרובים (MAS)

### 4.1. התמחות סוכנים: פרדיגמת "סוכן אחד, מטריצה אחת"

* **תפיסת המשתמש:** הרעיון שלכל מטריצת נתונים יהיה סוכן ייעודי האחראי על עיבודה ופרשנותה (שאילתת המשתמש).
* **יתרונות:**
  + **מומחיות ממוקדת:** כל סוכן יכול להיות מכוונן עם אלגוריתמים המתאימים ביותר לסוג הנתונים הספציפי שלו (למשל, סוכן לנתוני ספר פקודות עשוי להשתמש במודלי למידת מכונה שונים מסוכן לסנטימנט חדשותי).
  + **מודולריות והרחבה:** קל יותר לפתח, לבדוק ולעדכן סוכנים בודדים. ניתן להוסיף פוטנציאלית מקורות נתונים (מטריצות) חדשים עם סוכנים חדשים.
* **אחריות של סוכן ספציפי למטריצה:**
  + הזנת נתוני המטריצה הייעודיים (המוטמעים) שלו.
  + ביצוע ניתוח נוסף או זיהוי דפוסים על נתונים אלו.
  + יצירת אותות, ציונים או תחזיות בהתבסס על התצוגה המומחית שלו.
  + תקשורת הפלט שלו לסוכנים אחרים או ליחידת קבלת החלטות מרכזית.
* **תפקידי דוגמה הנובעים מ"סוכן אחד, מטריצה אחת":**
  + **סוכן טכני מחיר/נפח:** מנתח דפוסים במטריצה 1.
  + **סוכן מיקרו-מבנה ספר פקודות:** מנתח דפוסים במטריצה 2.
  + **סוכן סנטימנט חדשותי:** מנתח דפוסים במטריצה 3.
  + (יש לשקול אם רשת ההטמעה עצמה היא "סוכן" או שכבת עיבוד מקדים לכל הסוכנים).

תפיסת "סוכן אחד, מטריצה אחת" מהווה נקודת מוצא ל*התמחות בנתונים*, אך מערכת מסחר חזקה תדרוש ככל הנראה סוכנים *פונקציונליים* נוספים (למשל, לניהול סיכונים, ביצוע, אגרגציה) היוצרים היררכיה או צוות MAS מורכב יותר. בעוד שכל סוכן מטריצה (למשל, "סוכן מחיר", "סוכן סנטימנט") מטפל בנתונים שלו , הפלטים שלהם הם רק אותות ביניים. החלטת מסחר קוהרנטית דורשת סינתזה של אותות אלו, הערכת סיכונים ולאחר מכן ביצוע. הדבר מרמז על תפקידים כמו "סוכן אסטרטג ראשי" לאיחוד אותות , "סוכן ניהול סיכונים" להחלת אילוצי תיק, ו"סוכן ביצוע" לאינטראקציה עם השוק. המודל הראשוני של המשתמש צריך להתפתח כדי לכלול תפקידים פונקציונליים אלו.

### 4.2. קביעת מספר הסוכנים: מעבר לספירת מטריצות פשוטה

* **אמונתו הראשונית של המשתמש:** מספר הסוכנים = מספר המטריצות (שאילתת המשתמש). זוהי נקודת התחלה טובה לפירוק פונקציונלי.
* **ניואנסים:**
  + **מורכבות בתוך מטריצה:** מטריצה יחידה ומורכבת מאוד (למשל, נתוני ספר פקודות רב-ממדיים עבור נכסים רבים) עשויה להפיק תועלת מ*מספר* תת-סוכנים המשתפים פעולה בניתוחה.
  + **יחסים בין-מטריציוניים:** ייתכן שיידרשו סוכנים לא רק למטריצות בודדות אלא גם לניתוח ההטמעות ה*משולבות* או היחסים *בין* אותות הנגזרים ממטריצות.
  + **תפקידים פונקציונליים:** מעבר לסוכני נתונים-מטריצה, יש לשקול תפקידים אחרים:
    - **סוכן אגרגציה/איחוד נתונים:** משלב אותות מסוכנים ספציפיים למטריצה.
    - **סוכן קבלת החלטות/ביצוע:** מקבל אותות מאוחדים ומקבל החלטות מסחר סופיות, מנהל פקודות.
    - **סוכן ניהול סיכונים:** מנטר את סיכון התיק הכולל, יכול לעקוף או להתאים עסקאות.
    - **סוכן למידה/הסתגלות:** מפקח על אימון מחדש והסתגלות של מודלים עבור כל הסוכנים.
* **מספר אופטימלי :** אין נוסחת קסם. מדובר באיזון בין התמחות, תקורת תקשורת ומורכבות הבעיה. דן בסוכני מטא-רמה לאופטימיזציה ברשתות, ומציע אפשרויות היררכיות. המספר ה"אופטימלי" תלוי בבעיה ולעיתים קרובות נמצא באמצעות תכנון ובדיקה איטרטיביים.

### 4.3. דפוסים ארכיטקטוניים ל-MAS פיננסי והתאמתם

* **ארכיטקטורות MAS :**
  + **מבנה היררכי:**
    - **תיאור:** ארגון דמוי עץ, סוכנים ברמה גבוהה יותר שולטים/מתאמים סוכנים ברמה נמוכה יותר. דוגמה: "סוכן מסחר ראשי" מקבל קלטים מ"סוכני ניתוח מטריצות" מומחים (למשל, סוכן מחיר, סוכן סנטימנט) ומקבל את ההחלטה הסופית. מזכיר קרן גידור עם מבנה היררכי (ידע זורם מסוכנים בטווח זמן גבוה יותר לסוכנים בטווח זמן נמוך יותר).
    - **רלוונטיות:** מתאים היטב לרעיון של סוכנים מומחים המזינים נקודת החלטה מרכזית. מבנה פיקוד ברור.
  + **מבנה מבוסס צוות:**
    - **תיאור:** סוכנים עובדים בשיתוף פעולה הדוק, מסתמכים זה על זה, לעיתים קרובות עם תפקידים מוגדרים, לקראת מטרה משותפת.
    - **רלוונטיות:** סוכני המטריצות יכולים ליצור "צוות ניתוח נתונים", שהפלט הקולקטיבי שלו משמש לאחר מכן "צוות ביצוע מסחר". מסגרת TradingAgents מתארת צוותי אנליסטים, חוקרים, סוחרים וניהול סיכונים.
  + **מבנה הולוני:**
    - **תיאור:** סוכנים מקובצים ל"הולונים" (יחידות אוטונומיות ושיתופיות) שיכולים להיות חלק מהולונים גדולים יותר.
    - **רלוונטיות:** יכול לשמש אם קבוצות שונות של מטריצות/סוכנים מתמקדות בסוגי נכסים או אסטרטגיות שונות, ויוצרות הולונים שמתאמים לאחר מכן ברמה גבוהה יותר.
  + **מבנה קואליציוני:**
    - **תיאור:** סוכנים יוצרים בריתות זמניות למשימות ספציפיות.
    - **רלוונטיות:** סוכני מטריצות עשויים ליצור קואליציות בהתבסס על תנאי שוק ספציפיים (למשל, תנודתיות גבוהה עשויה להפעיל קואליציה של סוכני ספר פקודות וסנטימנט). פחות יציב לארכיטקטורת ליבה אך שימושי לתגובות דינמיות.
  + **מבוזר/עמית לעמית (Peer-to-Peer):**
    - **תיאור:** סוכנים מקיימים אינטראקציה ישירה ללא שליטה מרכזית. תיאום באמצעות פרוטוקולים.
    - **רלוונטיות:** סוכני מטריצות יכולים לשתף תובנות ישירות ולנהל משא ומתן על קונצנזוס. מורכב יותר לניהול עבור החלטות מסחר קוהרנטיות.
* **מסגרת Windsurf :** בעוד ש-Windsurf הוא IDE לקידוד AI, המושגים שלו של "Flows = Agents + Copilots" וסוכן "Cascade" המבין כוונה והקשר יכולים להוות השראה לתכנון MAS. הרעיון של AI שגם משתף פעולה וגם מתמודד עם משימות מורכבות באופן עצמאי רלוונטי לסוכני מסחר מתוחכמים. האסטרטגיה ה"אגנוסטית למודל" של Windsurf מרמזת גם על גמישות בבחירת יישומי סוכנים.

בחירת ארכיטקטורת MAS (למשל, היררכית לעומת מבוזרת) נושאת השלכות עמוקות על יכולת ההסתגלות של המערכת, עמידותה בפני תקלות ומורכבות מנגנוני התיאום שלה. מערכת היררכית עשויה להציע מסלולי החלטה ברורים אך עלולה ליצור צווארי בקבוק או נקודות כשל בודדות ברמות גבוהות יותר. מערכת מבוזרת מציעה חוסן רב יותר במקרה של כשל בסוכנים בודדים, אך הופכת את השגת אסטרטגיית מסחר אופטימלית גלובלית למאתגרת יותר בשל תיאום מורכב. המשתמש צריך לאזן בין שליטה לגמישות. עבור מסחר אלגוריתמי, גישה היברידית, אולי מבנה מבוסס צוות עם מנהיג או מתאם ברור, משיגה לעיתים קרובות איזון טוב.

**טבלה 2: דפוסים ארכיטקטוניים של מערכות סוכנים מרובים (MAS) למסחר אלגוריתמי**

| דפוס | תיאור | מודל אינטראקציה בין סוכנים | התאמת סוכנים ספציפיים למטריצה | יתרונות למסחר | חסרונות למסחר | מקורות |
| --- | --- | --- | --- | --- | --- | --- |
| **היררכי** | ארגון דמוי עץ, רמות שליטה ותיאום. | מלמעלה למטה (Top-down), סוכנים ברמה נמוכה מדווחים/נשלטים על ידי רמות גבוהות. | סוכני מטריצה כסוכנים ברמה נמוכה המזינים סוכן אסטרטגיה/החלטה ברמה גבוהה. | מסלולי החלטה ברורים, שליטה מרכזית. | צווארי בקבוק פוטנציאליים, נקודות כשל בודדות, פחות גמישות. |  |
| **מבוסס צוות** | סוכנים עובדים בשיתוף פעולה הדוק, תפקידים מוגדרים, מטרה משותפת. | שיתוף פעולה אינטנסיבי, תלות הדדית, תקשורת תכופה. | סוכני מטריצה כחברים בצוות ניתוח, המשתפים פעולה עם צוותי ביצוע או ניהול סיכונים. | סינרגיה גבוהה, ניצול מומחיות משולבת. | תלות חזקה בין חברי צוות, כשל של סוכן אחד עלול להשפיע על כל הצוות. |  |
| **הולוני** | יחידות אוטונומיות ושיתופיות ("הולונים") המרכיבות היררכיות גדולות יותר. | שיתוף פעולה פנימי בתוך הולונים, אינטראקציה בין הולונים. | כל הולון יכול להתמקד בקבוצת מטריצות/אסטרטגיות, עם סוכני מטריצה פנימיים. | גמישות, הרחבה, ארגון עצמי. | מורכבות בתיאום בין הולונים, הגדרת גבולות הולונים. |  |
| **מבוזר/P2P** | סוכנים מקיימים אינטראקציה ישירה ללא שליטה מרכזית. | תקשורת עמית-לעמית, פרוטוקולי תיאום מבוזרים. | סוכני מטריצה משתפים תובנות ישירות ומנהלים משא ומתן על החלטות. | חוסן גבוה לכשלים, אין נקודת כשל מרכזית. | השגת קונצנזוס ואופטימיזציה גלובלית יכולות להיות מאתגרות, תקורת תקשורת גבוהה. |  |
| **קואליציוני** | סוכנים יוצרים בריתות זמניות למשימות ספציפיות. | אינטראקציות דינמיות, יצירה ופירוק של קואליציות. | סוכני מטריצה רלוונטיים יוצרים קואליציות בתגובה לאירועי שוק ספציפיים. | יכולת תגובה מהירה להזדמנויות חולפות, גמישות. | חוסר יציבות מבני, עלויות הקמה ופירוק של קואליציות. |  |

טבלה זו תסייע למשתמש להבין את הדרכים הבסיסיות לארגן את הסוכנים המרובים שלו. על ידי ראיית האופן שבו "הסוכנים הספציפיים למטריצה" שלו יפעלו בתוך כל דפוס והפשרות הנלוות, הוא יכול לבחור ארכיטקטורה המתאימה ביותר למורכבות המערכת שלו, לרמת השליטה הרצויה ולצורכי התיאום. זה עובר מ"כמה סוכנים" ל"כיצד יש לארגן סוכנים אלו".

### 4.4. מנגנונים לתיאום סוכנים ואגרגציית החלטות

* **פרוטוקולי תקשורת:**
  + **שפות תקשורת סוכנים (ACLs):** שפות פורמליות כמו FIPA-ACL, KQML להודעות מובנות.
  + **תקשורת מבוססת אונטולוגיה:** אוצר מילים משותף לפרשנות עקבית.
  + **שפה טבעית:** הופכת לישימה יותר עם מודלי שפה גדולים (LLMs), אך עם פוטנציאל לעמימות.
  + **אינטראקציה מבוססת פרוטוקול:** רצפי הודעות מוגדרים (למשל, contract net, הצבעה). עבור MAS פיננסי, אלו יכולים להיות פרוטוקולים מותאמים אישית לשיתוף אותות והצבעה.
  + **מערכות לוח מודעות (Blackboard Systems):** מרחב מידע משותף שבו סוכנים יכולים לפרסם/לאחזר מידע. סוכני מטריצה מפרסמים אותות; סוכן החלטה קורא.
  + **פרוטוקול הקשר מודל (MCP - Model Context Protocol):** לחיבור סוכנים לכלים/משאבים וניהול הקשר. יכול לשמש סוכנים לגישה לנתונים או להפעלת כלי ניתוח.
* **אסטרטגיות אגרגציית החלטות :**
  + **הצבעה/ממוצע משוקלל:** שיטות פשוטות שבהן אותות מסוכני מטריצה משוקללים ומשולבים.
  + **מערכות מבוססות חוקים:** סוכן ברמה גבוהה יותר משתמש בחוקים מוגדרים מראש לשילוב אותות (למשל, "IF Price\_Agent\_Signal=BUY AND Sentiment\_Agent\_Signal=POSITIVE THEN Trade=BUY"). לוגיקה עמומה (fuzzy logic) יכולה לטפל באי-ודאות.
  + **סוכן מטא-לומד (Meta-Learner Agent):** מודל למידת מכונה נפרד הלומד לשלב את הפלטים של סוכני המטריצה.
  + **משא ומתן/דיון:** סוכנים עשויים "לטעון" את עמדתם, כפי שנראה במערכות סוכנים מבוססות LLM מסוימות. מסגרת TradingAgents משתמשת בדיונים בין חוקרים שוריים לדוביים.
  + **הצטלבות (Confluence):** שילוב מספר מדדים/אותות לקבלת אימות חזק יותר. זה ישים ישירות לשילוב פלטים מסוכני מטריצה שונים.
* **תיאום בפועל:**
  + האצלת משימות (למשל, סוכן מנהל מקצה משימות ניתוח).
  + בסיס ידע משותף או מודל עולם.
  + מנגנוני פתרון קונפליקטים.

תקשורת סוכנים אפקטיבית אינה עוסקת רק בהעברת הודעות; היא עוסקת בהבנה משותפת ובהקשר. פרוטוקולים כמו MCP מסייעים בניהול הקשר. הייצוגים הווקטוריים המאוחדים משכבת ההטמעה הם, במהותם, צורה דחוסה של הקשר הנגזר מכל מטריצה. כאשר סוכן מתקשר אות המבוסס על המטריצה שלו, אות זה כבר מעוגן במרחב המוטמע המשותף, מה שעשוי להקל על סוכנים אחרים או על אגרגטור להבין ולשלב אותו. זה מפחית את "הפער הסמנטי" בין סוכנים המתמחים בנתונים הטרוגניים.

## 5. אימון ואימות המערכת: סוכנים ורשת ההטמעה

### 5.1. אסטרטגיות חלוקת נתונים לאימון מודלים

* **התייחסות ל"חלוקה אקראית של נתונים" של המשתמש:**
  + **חלוקות אקראיות ראשוניות:** עבור רכיבים שאינם לומדים תלות זמנית ישירות (למשל, אימון מקדים של מחלצי תכונות מסוימים אם הם סטטיים, או צורות מסוימות של הטמעה בלתי מונחית אם סדר הזמן אינו בעל חשיבות עליונה להתרחשות משותפת), ניתן להשתמש בחלוקות אימון/בדיקה/אימות אקראיות סטנדרטיות (X\_train, X\_test, y\_train, y\_test מ-). random\_state מבטיח שחזוריות.
  + **מדוע אקראיות לעיתים קרובות אינה מספיקה בתחום הפיננסי:** נתונים פיננסיים הם בדרך כלל נתוני סדרות זמן עם מתאם-עצמי ואי-סטציונריות פוטנציאלית. ערבוב אקראי עלול להוביל לדליפת נתונים (המודל רואה נתונים "עתידיים" במהלך האימון) ולמדדי ביצועים אופטימיים מדי.
* **חלוקה מודעת-סדרות זמן (חיונית לנתונים פיננסיים):**
  + **חלוקה כרונולוגית:** אימון על נתונים ישנים יותר, אימות על נתונים עדכניים יותר, בדיקה על הנתונים החדשים ביותר ( "Time-Based Splitting"). זה מחקה פריסה בעולם האמיתי.
  + **TimeSeriesSplit (scikit-learn):** מספק אימות צולב עם מקור מתגלגל (rolling-origin cross-validation). בכל חלוקה, קבוצת הבדיקה מורכבת מ-'n' הנקודות הבאות אחרי קבוצת האימון. זה מצוין לאימות מודלים תלויי-זמן.
  + **K-Fold Cross Validation לסדרות זמן:** גרסאות כמו StratifiedKFold ניתנות להתאמה אם תוויות המחלקה (למשל, אותות קנייה/מכירה/החזקה) אינן מאוזנות, ומבטיחות שהקיפולים שומרים על התפלגות המחלקות , אך יש ליישמן בזהירות עם סדרות זמן כדי לכבד את הסדר הזמני (למשל, על ידי קיפול על פני בלוקי זמן).
  + **אימות צועד קדימה (Walk-Forward Validation):** שיטה חזקה יותר שבה המודל מאומן על חלון של נתונים קודמים, נבדק על התקופה הבאה, ואז החלון מחליק קדימה, והמודל מאומן מחדש. מומלץ מאוד למסחר אלגוריתמי.
  + **תקופת חיץ/אמברגו:** בעת חלוקת נתוני סדרות זמן, במיוחד עם תכונות בעלות מבט לאחור (כמו ממוצעים נעים), חשוב שיהיה חיץ בין קבוצות האימון והאימות/בדיקה כדי למנוע דליפת מידע.

בקשת המשתמש ל"חלוקת נתונים אקראית" זקוקה להבהרה זהירה. בעוד שחלוקות אקראיות פשוטות הן נקודת התחלה, האופי הזמני של נתונים פיננסיים והתלות ההדדית בין מטריצות מחייבים טכניקות אימות מתקדמות של סדרות זמן כדי למנוע הערכות ביצועים מטעות. לשווקים פיננסיים יש זיכרון. מודל שאומן על נתונים מעורבבים אקראית עלול ללמוד מתאמים מזויפים שאינם מתקיימים במסחר חי. טכניקות כמו TimeSeriesSplit , אימות צועד קדימה , ושימוש בתקופות חיץ/אמברגו הן חיוניות להערכה חזקה הן של רשת ההטמעה והן של הסוכנים, במיוחד אם הם לומדים דפוסים זמניים. ההיבט ה"אקראי" ישים יותר ל*בחירה* של נקודות נתונים *בתוך* קיפול אימון המוגדר כרונולוגית אם משתמשים בטכניקות כמו bootstrapping, או לאתחול משקולות, אך לא למבנה החלוקה הראשי לאימון/בדיקה.

**טבלה 3: אסטרטגיות חלוקת נתונים למודלי סדרות זמן פיננסיות**

| שיטה | אופן פעולה | מתי להשתמש בנתונים פיננסיים | יתרונות | חסרונות/אזהרות | מקורות |
| --- | --- | --- | --- | --- | --- |
| **חלוקה אקראית** | ערבוב אקראי של הנתונים וחלוקה לאימון/בדיקה. | בעיקר לרכיבים שאינם לומדים תלות זמנית, או לבדיקות ראשוניות מאוד (בזהירות רבה). | פשוטה ליישום. | סיכון גבוה להטיית הצצה לעתיד, ביצועים אופטימיים מדי, אינה משקפת תנאי מסחר אמיתיים. |  |
| **חלוקה כרונולוגית** | אימון על נתונים ישנים, בדיקה על נתונים חדשים יותר. | ברירת מחדל טובה למודלים של סדרות זמן, מחקה פריסה בעולם האמיתי. | פשוטה להבנה ויישום, מכבדת את סדר הזמן. | קבוצת בדיקה יחידה עשויה לא להיות מייצגת, רגישה לבחירת נקודת החלוקה. |  |
| **TimeSeriesSplit / אימות צולב מתגלגל** | יצירת מספר קיפולים, כאשר בכל קיפול קבוצת הבדיקה היא המשך ישיר של קבוצת האימון. | הערכה חזקה יותר של ביצועי מודל על פני תקופות זמן שונות. | ניצול טוב יותר של הנתונים, הערכה פחות מוטה של יכולת הכללה. | יקרה יותר חישובית, עדיין יש צורך בזהירות עם תכונות בעלות מבט לאחור. |  |
| **אימות צועד קדימה (Walk-Forward)** | אימון על חלון נתונים, בדיקה על התקופה הבאה, הזזת החלון קדימה ואימון מחדש. | נחשבת לשיטה החזקה ביותר לבדיקה לאחור של אסטרטגיות מסחר, מדמה הסתגלות לשינויים בשוק. | מציאותית מאוד, מאפשרת הערכת יציבות האסטרטגיה והצורך באימון מחדש. | דורשת משאבים חישוביים רבים, הגדרת גודל החלון וצעד ההזזה דורשת שיקול דעת. |  |
| **Stratified K-Fold (לסדרות זמן - עם אזהרות)** | חלוקה ל-K קיפולים תוך שמירה על יחס המחלקות (למשל, קנייה/מכירה) בכל קיפול. יש ליישם על בלוקי זמן כדי לכבד סדר. | כאשר יש חוסר איזון משמעותי במחלקות היעד, ורוצים להבטיח ייצוג הולם בכל קיפול, תוך שמירה על סדר זמנים ככל האפשר. | מבטיחה שכל מחלקה מיוצגת כראוי בקיפולי האימון/בדיקה. | קשה ליישום נכון בסדרות זמן מבלי להפר את הסדר הכרונולוגי; עלולה להוביל לדליפת מידע אם לא נזהרים. |  |

טבלה זו עונה ישירות על שאלת המשתמש לגבי "חלוקה אקראית של נתונים" על ידי הצבתה בהקשר של שיטות חזקות יותר המתאימות לסדרות זמן פיננסיות. היא מספקת מדריך מעשי לבחירת אסטרטגיית האימות הנכונה כדי למנוע מלכודות נפוצות כמו הטיית הצצה לעתיד והתאמת יתר, מה שמוביל להערכת מודל אמינה יותר.

### 5.2. אסטרטגיות אימון לרשת ההטמעה

* **אם משתמשים ב-CODE :**
  + האימון כולל מקסום פונקציית נראות ההתרחשות המשותפת. זהו בדרך כלל תהליך בלתי מונחה המבוסס על דפוסי התרחשות משותפת היסטוריים בין אירועים בדידים ממטריצות שונות.
* **אם משתמשים באוטואנקודרים :**
  + האימון הוא בלתי מונחה, וממזער את אובדן השחזור. לכל מטריצה עשוי להיות אוטואנקודר משלה, או שניתן לאמן אוטואנקודר משותף על קלטי מטריצות משולבים (מנורמלים).
* **אם משתמשים בהטמעות גרפים :**
  + האימון תלוי באלגוריתם הספציפי (למשל, DeepWalk משתמש בהילוכים אקראיים, GraphSAGE מאחד מידע שכנים). בדרך כלל בלתי מונחה או מונחה-עצמית.
* **אימון מונחה מקצה לקצה:** רשת ההטמעה יכולה להיות השכבות הראשוניות של מודל למידה עמוקה גדול יותר המאומן על משימה מונחית (למשל, חיזוי תשואות עתידיות, סיווג משטרי שוק). ההטמעות נלמדות כחלק מהאופטימיזציה למשימה זו. זה הופך את ההטמעות לספציפיות למשימה.

אימון רשת ההטמעה והסוכנים יכול להיות מנותק או מצומד (מקצה לקצה). בחירה זו משפיעה על האם ההטמעות הן לשימוש כללי או ספציפיות למשימה, ויש לה השלכות על מודולריות המערכת ומורכבות האימון. אימון בלתי מונחה של הטמעות (למשל, CODE, אוטואנקודרים לפי ) יוצר ייצוגים לשימוש כללי של מטריצות הנתונים. אלו יכולים לשמש לאחר מכן סוכנים שונים המאומנים בנפרד. זה מודולרי. לחלופין, רשת ההטמעה יכולה להיות שכבות הקלט של סוכן למידת חיזוק עמוקה או מודל חיזוי מונחה, המאומנים מקצה לקצה. זה הופך את ההטמעות לספציפיות מאוד למשימת הסוכן אך עלול להפוך את המערכת למונוליטית יותר וקשה יותר לניפוי שגיאות. המשתמש צריך לשקול פשרה זו.

### 5.3. אסטרטגיות אימון לסוכנים בודדים

* **סוג הסוכן מכתיב את האימון:**
  + **סוכנים מבוססי חוקים:** ייתכן שלא ידרשו "אימון" במובן של למידת מכונה אלא כיול ואופטימיזציה של חוקים (למשל, באמצעות אלגוריתמים גנטיים או סריקות פרמטרים על נתונים היסטוריים).
  + **סוכני למידה מונחית:** אם משימת הסוכן היא חיזוי (למשל, סוכן סנטימנט החוזה השפעה קצרת טווח על המחיר), הוא יאומן על נתונים היסטוריים מתויגים (קלט: תכונות סנטימנט, פלט: שינוי מחיר בפועל).
  + **סוכני למידת חיזוק (RL):** אם סוכנים לומדים על ידי אינטראקציה עם סביבה (מדומה או אמיתית) כדי למקסם תגמול (למשל, סוכן ביצוע הלומד למזער החלקה (slippage), או סוכן מסחר הלומד מדיניות). זה מורכב אך חזק.

### 5.4. התמודדות עם אתגרים בלמידת חיזוק לא מקוונת (Offline RL) עם נתונים פיננסיים היסטוריים

* **הקשר של Offline RL:** אימון סוכני RL באמצעות מערך נתונים קבוע של אינטראקציות היסטוריות, ללא אינטראקציה נוספת עם סביבה חיה. זה נפוץ כאשר סימולטור טוב אינו זמין או יקר מדי.
* **אתגרים מרכזיים :**
  + **הסטת התפלגות (Distribution Shift):** התפלגות הנתונים ההיסטוריים עשויה להיות שונה מההתפלגות שתיתקל בה אם המדיניות הנלמדת הייתה נפרסת.
  + **היעדר נתונים נגדיים (Counterfactuals):** קשה לדעת מה *היה קורה* אם הסוכן היה נוקט פעולה אחרת בעבר.
  + **יחס אות לרעש (SNR) נמוך בנתונים פיננסיים:** מקשה על הבחנה בין אות אמיתי לרעש.
  + **אי-סטציונריות:** דינמיקת השוק משתנה עם הזמן, מה שהופך נתונים היסטוריים לפחות מייצגים של תנאים עתידיים.
  + **נתונים אנונימיים בהקשרים רב-סוכנים:** אם נתונים היסטוריים אינם מפרטים איזה סוכן נקט איזו פעולה, קשה להקצות קרדיט/אשמה או להסיק מצבים/תגמולים של סוכנים בודדים.
* **"מסע בזמן עקבי בנתונים" (Consistent Data Time Travel) :**
  + **הבעיה שהיא פותרת:** מתמודדת עם הבעיה של שימוש נאיבי בנתונים היסטוריים באופן רציף, כאשר פעולות סוכן ה-RL עשויות להיות לא עקביות עם אירועים היסטוריים עוקבים.
  + **מנגנון:** במקום לעבור לנקודת הנתונים ה*כרונולוגית הבאה* לאחר פעולת סוכן, היא "נוסעת" בנתונים היסטוריים כדי למצוא נקודה שבה מצב המערכת *לאחר* פעולה היסטורית עקבי עם המצב *לאחר* הפעולה ההיפותטית של סוכן ה-RL במצב הנוכחי. זה הופך את ההתפתחות העתידית הנצפית למציאותית יותר בהינתן התערבות הסוכן.
  + **יתרון:** מוביל לחישוב תגמול ולמידת מדיניות מציאותיים יותר בסביבות לא מקוונות עבור מערכות רגישות כמו שווקים פיננסיים. טוען שזה יכול לשפר משמעותית רווחים.

אם משתמשים בסוכני RL, טכניקת "מסע בזמן עקבי בנתונים" היא שיטה מתוחכמת לשיפור ריאליזם האימון הלא מקוון, אך היא מוסיפה מורכבות. אימוצה תלוי בחומרת פער הסימולציה-למציאות הנתפס על ידי המשתמש. למידת חיזוק לא מקוונת סטנדרטית על נתונים פיננסיים היסטוריים נכשלת לעיתים קרובות מכיוון שהנתונים אינם מגיבים לפעולות הסוכן. "מסע בזמן עקבי בנתונים" מנסה למתן זאת על ידי מציאת מקטעים היסטוריים ש*כן* עקביים עם פעולות הסוכן. זהו צעד משמעותי קדימה מעיבוד רציף נאיבי. עם זאת, יישומו דורש התאמת מצבים קפדנית וניהול ה"קפיצות" בזמן. המשתמש צריך לשקול את הרווח הפוטנציאלי בביצועים מול מאמץ היישום. עבור סוכנים מונחים פשוטים יותר, זה עשוי להיות מוגזם.

### 5.5. מדדי אימות וביצועים

* **עבור רשת ההטמעה:**
  + איכותי: ויזואליזציה של הטמעות (למשל, באמצעות t-SNE, UMAP מ-) כדי לראות אם מושגים דומים מתקבצים יחד.
  + כמותי: אם אומנה למשימה במורד הזרם, ביצועים במשימה זו. עבור הטמעות בלתי מונחות, אובדן שחזור (אוטואנקודרים) או מדדים הקשורים לשימור מבני שכנות.
* **עבור סוכנים:**
  + מדדי למידת מכונה סטנדרטיים (דיוק, precision, recall, F1 לסיווג; MSE, MAE לרגרסיה).
  + מדדים פיננסיים: יחס שארפ, יחס סורטינו, משיכה מקסימלית (max drawdown), רווח והפסד (P&L), שיעור זכייה.
  + השוואה מול שיטות מסורתיות (למשל, MVO לאופטימיזציית תיקים, כמו ב-) או קווי בסיס פשוטים יותר.
* **מערכת כוללת:** ביצועי בדיקה לאחור על נתונים מחוץ לדגימה (out-of-sample) באמצעות אימות סדרות זמן מתאים.

## 6. מסימולציה למציאות: הבטחת עקביות בין בדיקה לאחור למסחר חי

### 6.1. הצורך הקריטי בעקביות לוגית בין סביבות

* **הבעיה:** אסטרטגיות שנראות מצוין בבדיקה לאחור נכשלות לעיתים קרובות במסחר חי. פער זה נובע מהבדלים באופן עיבוד הנתונים, אופן ביצוע האסטרטגיות, או גורמים מהעולם האמיתי שלא נלקחו בחשבון.
* **מטרה:** להבטיח שסביבת הבדיקה לאחור תשקף את סביבת המסחר החי באופן הדוק ככל האפשר, כך שתוצאות הבדיקה לאחור יהיו מנבא אמין של ביצועים חיים ( "100% עקביות בלוגיקה").
* **השפעת חוסר עקביות:** בזבוז מאמץ פיתוח, הפסדים כספיים, אובדן אמון במערכת.

השגת עקביות בין בדיקה לאחור למסחר חי אינה רק משימה טכנית אלא פילוסופיית תכנון בסיסית שחייבת לחלחל לכל ארכיטקטורת המערכת, מהזנת נתונים ועד לוגיקת ביצוע פקודות. צוותים רבים מתייחסים לבדיקה לאחור ולמסחר חי כמערכות נפרדות, מה שמוביל לסטיות עדינות (או גדולות). ארכיטקטורה מונעת-אירועים מספקת בסיס חזק, אך יש לאכוף עקביות גם בכל טרנספורמציית נתונים , בכל חישוב תכונה, ובכל פיסת לוגיקת החלטה של סוכן. רצונו של המשתמש "להמשיך לעבוד עם אותו מספר סוכנים" הוא ביטוי לצורך זה בזהות מבנית ולוגית.

### 6.2. יישום ארכיטקטורה מונעת-אירועים למעבר חלק

* **תפיסה:** רכיבי המערכת מגיבים לאירועים (למשל, נתוני שוק חדשים, אות שנוצר, פקודה שמולאה) במקום לפעול במרווחי זמן קבועים או על נתוני אצווה טעונים מראש.
* **רכיבי מפתח :**
  + **תור אירועים (Event Queue):** רכזת מרכזית לכל האירועים.
  + **מטפל נתונים (DataHandler):** מזין נתוני שוק (היסטוריים לבדיקה לאחור, חיים למסחר) כ-MarketEventים. *כאן מתרחש המעבר.*
  + **אסטרטגיה (או סוכנים בהקשר MAS זה):** צורכת MarketEventים (או נתוני מטריצה מעובדים), מייצרת SignalEventים.
  + **תיק (Portfolio):** מנהל פוזיציות בהתבסס על SignalEventים, מייצר OrderEventים.
  + **מטפל ביצוע (ExecutionHandler):** מדמה ביצוע פקודות (בדיקה לאחור) או מתחבר לברוקר חי (מסחר חי), מייצר FillEventים.
* **יתרונות לעקביות :**
  + **שימוש חוזר בקוד:** הלוגיקה המרכזית לאסטרטגיה, ניהול תיקים, ואפילו היבטים של טיפול בביצוע יכולים להיות זהים לבדיקה לאחור ולמסחר חי. יש להחליף מינימום רכיבים.
  + **סימולציה מציאותית:** מטפלת בנתונים בסגנון "טפטוף", ומחקה הגעת נתונים בזמן אמת ומונעת הטיית הצצה לעתיד.
  + **טיפול בגורמים מהעולם האמיתי:** קל יותר למדל עלויות עסקה, החלקה, דינמיקת תור פקודות במסגרת מונעת-אירועים.

### 6.3. צנרת נתונים, הנדסת תכונות ולוגיקת סוכנים עקביות

* **צנרת נתונים :**
  + יש ליישם בדיוק את אותם שלבי ניקוי, נורמליזציה וטרנספורמציה של נתונים על נתונים היסטוריים (לבדיקה לאחור של המטריצות) ונתונים בזמן אמת (לאכלוס מטריצות חיות).
  + זרימות עבודה אוטומטיות הן מפתח להבטחת עקביות זו.
* **הנדסת תכונות :**
  + תכונות המוזנות לכל מטריצה חייבות להיות מחושבות באופן זהה בשתי הסביבות. כל שינוי בהגדרת תכונה או בחישובה פוגע בעקביות.
  + התאמת חותמות זמן וטיפול בנתונים המגיעים באיחור דורשים לוגיקה עקבית.
* **לוגיקת סוכנים:**
  + הקוד לכל סוכן (כיצד הוא מעבד את המטריצה שלו, מייצר אותות) חייב להיות *זהה*.
  + ארכיטקטורת רשת ההטמעה והמשקולות המאומנות שלה חייבות להיות זהות.
  + הלוגיקה לאיחוד אותות וקבלת החלטות סופיות חייבת להיות זהה.

### 6.4. ניהול מצב מערכת ופעולות סוכנים בזמן אמת (שאילתת המשתמש)

* **שימור מצב:** המשתמש מודאג לגבי האופן שבו סוכנים מאומנים ומצביהם נשמרים.
  + **מצב סוכן:** אם סוכנים הם בעלי מצב (stateful) (למשל, סוכן RL עם מצב אמונה פנימי, או סוכן העוקב אחר היסטוריית האותות האחרונה שלו), לוגיקת ניהול מצב זו חייבת להיות עקבית. במסחר חי, יש לשמר מצב באופן אמין.
  + **מצב תיק:** פוזיציות נוכחיות, יתרת מזומנים, פקודות פתוחות – מנוהלים על ידי רכיב התיק במערכת מונעת-אירועים.
* **הזנת נתונים בזמן אמת למטריצות:**
  + הזנות נתונים חיות (WebSockets, APIs) מחליפות קוראי נתונים היסטוריים.
  + נתונים מעובדים, תכונות מהונדסות, ומטריצות מתעדכנות בזמן אמת (או קרוב לזמן אמת, תלוי בתדירות המסחר).
* **שמירה על אותו מספר סוכנים/מטריצות:**
  + הארכיטקטורה (מספר המטריצות, מספר ותפקידי הסוכנים) צריכה להישאר קבועה בין בדיקה לאחור למסחר חי, מכיוון שזה מה שנבדק.
  + המערכת החיה פשוט מזינה נתונים חיים לאותם מבני מטריצות, המעובדים על ידי אותם סוכנים מאומנים.
* **דוגמה: Agentic IDP ו-Financial Crews :** למרות שאינם עוסקים ישירות במסחר, מקורות אלו מראים כיצד מערכות סוכנים מנהלות זרימות עבודה, חילוץ נתונים ותיאום תת-סוכנים בזמן אמת, מה שמקביל ל-MAS מסחר חי. תפיסת סוכן המפקח רלוונטית לתיאום סוכני מסחר.

המעבר מנתונים היסטוריים להזנות נתונים חיות הוא נקודת ממשק קריטית. ה-"DataHandler" במערכת מונעת-אירועים מכמס שינוי זה, אך הנתונים שהוא מספק חייבים להיות זהים מבנית וסמנטית למה שהסוכנים אומנו עליו. בעוד שמקור הנתונים החי עשוי להתחבר ל-WebSocket API, ומקור הבדיקה לאחור קורא קובצי CSV, ה-*פלט* של MarketEvents (או הנתונים המשמשים לאכלוס המטריצות) חייב להיות בעל אותו פורמט, אותן תכונות ואותה משמעות. כל אי-התאמה כאן (למשל, דיוק מחיר שונה, טיפול שונה בחותמות זמן, תכונות חסרות בהזנה החיה) תגרום לסוכנים להתנהג באופן בלתי צפוי. הדבר דורש מיפוי נתונים קפדני ואימות בממשק הנתונים החי.

### 6.5. אסטרטגיות ללמידה מתמשכת, הסתגלות ואימון מחדש של מודלים בשווקים חיים

* **דינמיקת השוק משתנה:** שווקים פיננסיים אינם סטציונריים. מודלים שאומנו על נתונים היסטוריים יתדרדרו בסופו של דבר בביצועיהם.
* **ארכיטקטורת למידה מתמשכת :** יש לתכנן את המערכת כך שתאפשר אימון מחדש תקופתי או מופעל-אירוע.
* **טריגרים לאימון מחדש:**
  + מבוסס זמן (למשל, אימון מחדש כל חודש).
  + התדרדרות בביצועים (למשל, יחס שארפ יורד מתחת לסף מסוים).
  + זיהוי שינוי במשטר השוק.
* **תהליך אימון מחדש:**
  + שימוש בנתונים היסטוריים חדשים ומאומתים.
  + יישום אותן טכניקות חלוקת נתונים ואימות (למשל, צועד קדימה).
  + עדכון משקולות רשת ההטמעה ו/או מודלי סוכנים בודדים.
  + בדיקה קפדנית של מודלים שאומנו מחדש לפני פריסתם למסחר חי (למשל, מסחר בצל - shadow trading).
* **סוכנים מסתגלים :** סוכני RL יכולים, מעצם תכנונם, להמשיך ללמוד ולהסתגל בשווקים חיים, אך הדבר דורש הגדרה וניטור קפדניים למניעת התנהגות לא רצויה.
* **בדיקה צועדת קדימה**  היא גישה טובה המשלבת באופן טבעי אימון מחדש.

למידה מתמשכת ואימון מחדש במערכת רב-סוכנים חיה הם מורכבים. לא מדובר רק באימון מחדש של מודלים בודדים; מדובר בהבטחה ש*כל אנסמבל* הסוכנים ורשת ההטמעה יישארו קוהרנטיים ולא יפתחו התנהגויות מתהוות (emergent behaviors) שליליות לאחר עדכונים. אם סוכן אחד מאומן מחדש והתפלגות האותות שלו משתנה באופן משמעותי, הדבר עלול להשפיע על האופן שבו שכבת האגרגציה מפרשת את אותותיו או כיצד סוכנים אחרים מגיבים (אם הם למדו דינמיקה בין-סוכנית). אימון מחדש צריך להתבצע באופן אידיאלי עבור המערכת (או תת-רכיבים משמעותיים) כמכלול, ולאחר מכן אימות מחדש יסודי ברמת המערכת (למשל, מסחר בצל מורחב) לפני שהמודלים החדשים משתלטים לחלוטין. זה יותר מסתם עדכון משקולות; זה ניהול האבולוציה של מערכת אדפטיבית מורכבת.

**טבלה 4: רשימת תיוג להבטחת עקביות בין בדיקה לאחור למסחר חי**

| תחום | בדיקה מרכזית לעקביות | מדוע זה חשוב | מקורות רלוונטיים |
| --- | --- | --- | --- |
| **צנרת נתונים** | אותם שלבי ניקוי, נורמליזציה, טרנספורמציה וטיפול בערכים חסרים/חריגים. | הבדלים בעיבוד נתונים מובילים לקלטים שונים למודלים, ומכאן לביצועים שונים. |  |
| **הנדסת תכונות** | חישוב זהה של כל התכונות, טיפול זהה בחותמות זמן ונתונים מאחרים. | תכונות הן הקלט הישיר למודלים; אי-עקביות כאן משנה את מה שהמודל "רואה". |  |
| **רשת הטמעה** | אותה ארכיטקטורה, אותן משקולות מאומנות (או תהליך אימון זהה אם מאומנת מחדש). | ההטמעה היא ליבת הפחתת ההטרוגניות; שינויים כאן משפיעים על כל הסוכנים. | (נגזר מהגיון המערכת) |
| **לוגיקת סוכנים** | קוד זהה לכל סוכן (עיבוד מטריצה, יצירת אותות), אותם פרמטרים. | התנהגות הסוכן חייבת להיות צפויה ועקבית בין הסביבות. | (נגזר מהגיון המערכת) |
| **אגרגציית אותות** | לוגיקה זהה לשילוב אותות מסוכנים שונים (למשל, שקלול, חוקים, מטא-לומד). | האופן שבו החלטות מתקבלות מהתובנות המשולבות חייב להיות זהה. |  |
| **סימולציית ביצוע פקודות** | מודל ריאליסטי של עלויות עסקה, החלקה, ודינמיקת תור פקודות בבדיקה לאחור. | התעלמות מגורמים אלו בבדיקה לאחור מובילה לתוצאות אופטימיות מדי. |  |
| **גורמי סביבה** | טיפול זהה בהשהיות (latency), זמינות נתונים, מגבלות API של הברוקר. | גורמים אלו יכולים להשפיע באופן משמעותי על ביצועים חיים וקשה למדל אותם במדויק. |  |
| **ניהול מצב** | לוגיקה עקבית לניהול מצב סוכנים ומצב תיק. | מצב לא עקבי יכול להוביל להחלטות שגויות במסחר חי. | (נגזר מהגיון המערכת) |
| **תדירות עדכון/פעולה** | אותה תדירות של קבלת נתונים, חישוב אותות וביצוע פעולות. | הבדלים בתזמון יכולים לשנות את תוצאות האסטרטגיה. |  |

טבלה זו מספקת כלי מעשי למשתמש לאימות ואכיפה שיטתית של עקביות בכל הרכיבים הקריטיים של מערכת המסחר שלו. היא מפרקת את הרעיון המופשט של "עקביות" לפריטים קונקרטיים, ומסייעת במניעת מלכודות נפוצות המובילות לפערים בין ביצועים שנבדקו לאחור לביצועים חיים. זה עונה ישירות על אחת הדאגות העיקריות של המשתמש.

## 7. סיכום והמלצות אסטרטגיות

### 7.1. סינתזה של החלטות ארכיטקטוניות מרכזיות

המסע שתואר כלל הגדרת מטריצות נתונים, הטמעתן להפחתת הטרוגניות, תכנון מערכת רב-סוכנים, אימונה והבטחת עקביות למסחר חי. הודגשה חשיבותה של ארכיטקטורה מונעת-אירועים, צנרת נתונים עקבית ואימות חזק. פרדיגמת "סוכן לכל מטריצה" מהווה בסיס טוב להתמחות, אך יש להרחיבה עם סוכנים פונקציונליים ואסטרטגיית תיאום/אגרגציה ברורה.

### 7.2. השלכות ושיקולים ארוכי טווח

* **הרחבה (Scalability):** כיצד תתרחב המערכת עם הוספת מקורות נתונים/נכסים חדשים?
* **תחזוקתיות (Maintainability):** מורכבות ניהול סוכנים מרובים והאינטראקציות ביניהם.
* **משאבים חישוביים:** אימון והרצה של הטמעות מתוחכמות וסוכנים מרובים יכולים להיות עתירי משאבים.
* **שווקים מתפתחים:** הצורך בניטור, הסתגלות ומחקר מתמשכים כדי לשמור על יעילות המערכת.

### 7.3. הנחיות לפיתוח איטרטיבי, בדיקות קפדניות ופריסה מדורגת

המערכת המוצעת, על אף עוצמתה, מציגה מורכבות משמעותית. יש לאמץ גישת פיתוח מדורגת, תוך התמקדות באימות כל רכיב (מטריצות נתונים, הטמעה, לוגיקת סוכן בודד, אינטראקציות בין סוכנים) לפני שילובם למערכת מלאה.

* **התחלה פשוטה:** יש להתחיל עם קבוצת ליבה של מטריצות וסוכנים, ולאחר מכן להוסיף מורכבות בהדרגה.
* **בדיקות יחידה ואינטגרציה קפדניות:** יש לבדוק ביסודיות כל סוכן, את רשת ההטמעה ואת האינטראקציות ביניהם.
* **בדיקה לאחור מקיפה:** יש להשתמש בטכניקות אימות חזקות של סדרות זמן.
* **מסחר על נייר/מצב צל (Paper Trading/Shadow Mode):** יש לפרוס את המערכת עם נתונים חיים אך ללא הון אמיתי כדי לצפות בהתנהגותה ובהחלטותיה.
* **פריסה חיה מדורגת:** יש להתחיל עם הקצאת הון קטנה ולהגדילה בהדרגה ככל שהביטחון נבנה. בניית מערכת כזו באופן מונוליטי היא מסוכנת מאוד. כל חלק (הזנת נתונים, יצירת מטריצות, הטמעה, לוגיקת סוכנים, תיאום סוכנים, ממשק מסחר חי) נושא אתגרים משלו. גישה איטרטיבית מאפשרת איתור מוקדם של בעיות ובנייה הדרגתית של מערכת חזקה. לדוגמה, ראשית יש לוודא שמטריצה אחת והסוכן שלה עובדים, לאחר מכן להוסיף עוד אחת, ואז לבדוק את האינטראקציה ביניהן, וכן הלאה. הדבר תואם שיטות עבודה מומלצות כלליות בהנדסת תוכנה אך קריטי במיוחד למערכות AI מורכבות בתחום הפיננסי.

### 7.4. המלצות סופיות למשתמש

* יש לתעדף את תכנון צנרת הנתונים ושכבת ההטמעה, שכן אלו יסודיות.
* יש לבחור בקפידה את ארכיטקטורת ה-MAS ואת מנגנוני התקשורת/תיאום בין הסוכנים.
* יש להשקיע רבות ביצירת סביבת בדיקה לאחור הקרובה ככל האפשר לתנאים חיים, כאשר גישה מונעת-אירועים מומלצת מאוד.
* יש לאמץ חשיבה של שיפור והסתגלות מתמשכים.

הצלחתה ארוכת הטווח של מערכת זו תלויה לא רק בתכנון הראשוני אלא גם בתשתית ובתהליכים שיוקמו לניטור, הערכה, אימון מחדש והסתגלות מתמשכים לתנאי שוק משתנים. שווקים פיננסיים הם דינמיים ואדברסריים. מערכת של "שגר ושכח" תיכשל בהכרח. המשתמש צריך לתכנן MLOps (Machine Learning Operations) מההתחלה: אימות נתונים אוטומטי, מעקב אחר ביצועי מודלים, זיהוי סחיפה (drift detection), וצנרת אימון מחדש יעילה. היבט תפעולי זה חשוב לא פחות מתכנון האלגוריתם הראשוני לרווחיות מתמשכת.

#### Works cited

1. What Are Multiagent Systems? The Future of AI in 2025, https://inclusioncloud.com/insights/blog/multiagent-systems-guide/ 2. Hands-On Multi-Agent Systems Tutorials: Building Your First Distributed AI System, https://smythos.com/ai-agents/multi-agent-systems/multi-agent-systems-tutorials/ 3. Multi-agent Systems in Finance: Enhancing Decision ... - SmythOS, https://smythos.com/ai-agents/multi-agent-systems/multi-agent-systems-in-finance/ 4. 10 Key Stats on Heterogeneity in Data Analysis - Number Analytics, https://www.numberanalytics.com/blog/10-key-stats-heterogeneity-data-analysis 5. cdn.aaai.org, https://cdn.aaai.org/AAAI/2006/AAAI06-261.pdf 6. Top 12 Dimensionality Reduction Techniques for Machine Learning - Encord, https://encord.com/blog/dimentionality-reduction-techniques-machine-learning/ 7. Combining Multiple Trade Signals For Better Analysis - FasterCapital, https://fastercapital.com/topics/combining-multiple-trade-signals-for-better-analysis.html 8. What Is Confluence in Trading, and How Can You Use It? | Market Pulse, https://fxopen.com/blog/en/what-is-confluence-in-trading-and-how-can-you-use-it/ 9. The problem of logical consistency between backtesting and live ..., https://quantra.quantinsti.com/community/t/the-problem-of-logical-consistency-between-backtesting-and-live-trading/25335 10. Time Travel is Cheating: Going Live with DeepFund for Real-Time Fund Investment Benchmarking - arXiv, https://arxiv.org/html/2505.11065v1 11. Backtesting vs Live Trading: Key Factors for a Successful Algo Strategy, https://www.hashstudioz.com/blog/backtesting-vs-live-trading-key-factors-for-a-successful-algo-strategy/ 12. Data Preprocessing for Algo Trading - LuxAlgo, https://www.luxalgo.com/blog/data-preprocessing-for-algo-trading/ 13. Financial sentiment analysis using FinBERT with application in predicting stock movement, https://arxiv.org/html/2306.02136v2 14. How To Automate Data Cleaning For Financial Analysis - Mike's F9 Finance, https://www.f9finance.com/data-cleaning/ 15. Data and Feature Engineering for Trading Course by Dr Ernest Chan, https://quantra.quantinsti.com/course/data-and-feature-engineering-for-trading 16. (PDF) Feature Engineering for High-Frequency Trading Algorithms - ResearchGate, https://www.researchgate.net/publication/387558831\_Feature\_Engineering\_for\_High-Frequency\_Trading\_Algorithms 17. LOB-Bench: Benchmarking Generative AI for Finance – an Application to Limit Order Book Data - arXiv, https://arxiv.org/html/2502.09172v1 18. [2403.09267] Deep Limit Order Book Forecasting - arXiv, https://arxiv.org/abs/2403.09267 19. Market-Derived Financial Sentiment Analysis: Context-Aware Language Models for Crypto Forecasting - arXiv, https://arxiv.org/pdf/2502.14897 20. Different types of data splitting methods - Kaggle, https://www.kaggle.com/discussions/general/448072 21. Train Test Split: What it Means and How to Use It | Built In, https://builtin.com/data-science/train-test-split 22. A Review of Reinforcement Learning in Financial Applications - arXiv, https://arxiv.org/html/2411.12746v1 23. FinRL-Meta: Market Environments and Benchmarks for Data-Driven Financial Reinforcement Learning, https://openfin.engineering.columbia.edu/sites/default/files/content/publications/finrl\_meta\_market\_environments.pdf 24. Designing Heterogeneous LLM Agents for Financial Sentiment Analysis - arXiv, https://arxiv.org/html/2401.05799v1 25. Revisited: Machine Intelligence in Heterogeneous Multi Agent Systems - University of Toronto Institute for Aerospace Studies, https://www.utias.utoronto.ca/wp-content/uploads/2019/11/33-Revisited-Machine-Intelligence-in-Heterogeneous-Multi-Agent-Systems-REVISED.pdf 26. Report: Windsurf Business Breakdown & Founding Story | Contrary Research, https://research.contrary.com/company/windsurf 27. What are graph embeddings ? - NebulaGraph, https://www.nebula-graph.io/posts/graph-embeddings 28. Vector Embedding Tutorial & Example - Nexla, https://nexla.com/ai-infrastructure/vector-embedding/ 29. Train custom embeddings based on co-occurrence data with KFP pipeline - Google Cloud, https://cloud.google.com/blog/products/ai-machine-learning/train-custom-embeddings-based-on-co-occurrence-data-with-kfp-pipeline/ 30. The Architecture of AI Agentic Systems from MatrixLabX - Putting AI to Work, https://matrixmarketinggroup.com/the-architecture-of-ai-agentic-systems-matrixlabx/ 31. arxiv.org, https://arxiv.org/pdf/2408.02322 32. How to Build a Multi-Agent AI System : In-Depth Guide, https://www.aalpha.net/blog/how-to-build-multi-agent-ai-system/ 33. Calculating optimal decision using Meta-level agents for Multi-Agents in Networks - CiteSeerX, https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=feb96de77f90a402f99f9ca961f320ce5d322331 34. Multi-Agent Systems and Optimization: Enhancing Efficiency Through Collaborative AI, https://smythos.com/ai-agents/multi-agent-systems/multi-agent-systems-and-optimization/ 35. Architectural Properties of Multi-Agent Systems - Carnegie Mellon University's Robotics Institute, https://www.ri.cmu.edu/pub\_files/pub1/shehory\_onn\_1998\_1/shehory\_onn\_1998\_1.pdf 36. Windsurf Editor | Windsurf (formerly Codeium), https://windsurf.com/editor 37. Windsurf (formerly Codeium) - The most powerful AI Code Editor, https://windsurf.com/ 38. Windsurf Unveils SWE-1 AI Models Amid OpenAI's Billion-Dollar Interest - OpenTools, https://opentools.ai/news/windsurf-unveils-swe-1-ai-models-amid-openais-billion-dollar-interest 39. Advancing Multi-Agent Systems Through Model Context Protocol: Architecture, Implementation, and Applications - arXiv, https://arxiv.org/html/2504.21030v1 40. A Survey of AI Agent Protocols - arXiv, https://arxiv.org/html/2504.16736v2 41. Multi-Agent System: Enhancing Collaboration in AI - Markovate, https://markovate.com/multi-agent-system/ 42. A Comprehensive Survey on Multi-Agent Cooperative Decision-Making: Scenarios, Approaches, Challenges and Perspectives - arXiv, https://arxiv.org/html/2503.13415v1 43. train\_test\_split — scikit-learn 1.6.1 documentation, https://scikit-learn.org/stable/modules/generated/sklearn.model\_selection.train\_test\_split.html 44. Data Splitting: A Crucial Step in Deep Learning - Alooba, https://www.alooba.com/skills/concepts/deep-learning/data-splitting/ 45. How to develop an AI agent for crypto trading - Cointelegraph, https://cointelegraph.com/news/how-to-develop-an-ai-agent-for-crypto-trading 46. Data Splittting Strategy in Supervised ML - DeepLearning.AI, https://community.deeplearning.ai/t/data-splittting-strategy-in-supervised-ml/586385 47. Understanding Training and Testing Data in Machine Learning, https://sqream.com/blog/training-and-testing-data-in-machine-learning/ 48. Deep Reinforcement Learning for Optimal Portfolio Allocation: A Comparative Study with Mean-Variance Optimization - ICAPS 2023, https://icaps23.icaps-conference.org/papers/finplan/FinPlan23\_paper\_4.pdf 49. Event-Driven Backtesting with Python - Part I - QuantStart, https://www.quantstart.com/articles/Event-Driven-Backtesting-with-Python-Part-I/ 50. Event-Driven Backtesting for Trading Strategies - PyQuant News, https://www.pyquantnews.com/free-python-resources/event-driven-backtesting-for-trading-strategies 51. Autonomous mortgage processing using Amazon Bedrock Data Automation and Amazon Bedrock Agents | AWS Machine Learning Blog, https://aws.amazon.com/blogs/machine-learning/autonomous-mortgage-processing-using-amazon-bedrock-data-automation-and-amazon-bedrock-agents/ 52. Agentic AI Systems Applied to tasks in Financial Services: Modeling and model risk management crews - arXiv, https://arxiv.org/html/2502.05439v1 53. Large Language Models and the Elliott Wave Principle: A Multi-Agent Deep Learning Approach to Big Data Analysis in Financial Markets - MDPI, https://www.mdpi.com/2076-3417/14/24/11897