**README**

במסמך זה נסביר בקצרה מה מטרתו של כל קובץ ומאיפה נלקח במידת הצורך.

התיקייה הראשית של הפרויקט

basic\_data\_processing.ipynb: מחברת (jupyter notebook) לטובת עיבוד קובץ הנתונים המקורי (המאגר שזמין ב-kaggle) ויצירת קובץ csv לטובת הסימולטור שלנו

Simulator.py: משתמש ב-CustomEnv (פירוט בהמשך) כדי לבצע הרצה יחידה של הסימולטור בהינתן משקיע מסוג מסוים

TestsLogger.py: הרצת הסימולטור מספר רב של פעמים ושמירת התוצאות לקובץ logger, שמהווה את הפלט של הניסויים שביצענו (ולאחר מכן אנחנו משתמשים בקובץ הזה כדי לנתח את התוצאות).

results\_analysis.ipynb: מחברת (jupyter notebook) לניתוח קבצי ה-logger שמתקבלים מ-TestsLogger.py כדי להבין את תוצאות הניסויים עבור משקיעים שונים.

investors\_types:

* HumanHeuristicsInvestors.py: מכיל את המשקיעים מבוססי היוריסטיקות האנושיות
* Investor.py: כולל אובייקט גנרי שממנו יורשים המשקיעים השונים
* PseudoAgents.py: מכיל את הגדרות הפסאודו-משקיעים: המשקיע הטוב ביותר, הגרוע ביותר והאקראי
* RLInvestor.py: בקובץ זה נמצאים שני המשקיעים מבוססי הבינה המלאכותית שלנו: המשקיע בשיטת Q-learning והמשקיע בשיטת Q-Learning with Value Function Approximation

our\_simulator:

* SimulatorCore.py: התשתית הבסיסית של הסימולטור שלנו. הפעולה הזו כוללת את הלוגיקה הבסיסית של הסימולטור, כמו למשל הבחירה האקראית של הקרנות בתחילת הריצה, ביצוע פעולה בסביבה (בהינתן קלט של הפעולה הרצויה) ועוד
* CustomEnv.py: מעטפת ל-SimulatorCore על בסיס ממשק זהה לממשק הנפוץ של ספריית gym (שכוללת מספר סביבות להרצת אלגוריתמי RL), אך לא משתמש בספרייה ישירות. האובייקט CustomEnv משתמש ב-SimulatorCore אך כולל פונקציונליות נוספת. לדוגמה, ב-CustomEnv ישנה פעולת reset שמייצרת אובייקט SimulatorCore חדש עבור הריצה הנוכחית. המטרה של CustomEnv היא "ליישר קו" בין הלוגיקה של הסימולטור שלנו לבין ממשק אחיד שאלגוריתמים קיימים מכירים
* State.py: משמש לייצוג של מצב הסימולטור ברגע נתון- רשימת הפיצ'רים של 10 הקרנות ברבעון שבו נמצא הסימולטור
* Printer.py: קובץ המשמש להדפסות שונות במהלך ריצת הסימולטור
* Fund.py: מחלקה המכילה את המידע על קרן של הסימולטור בריצה הנוכחית

RL\_Trainer:

* TrainerRL.py: בקובץ זה נמצאים אלגוריתמי האימון של הסוכנים הנבונים שלנו.

אימון המשקיע מסוג RLQinvestor נכתב בהשראת הקוד הבא:

<https://github.com/monokim/framework_tutorial/blob/master/main.py>

אימון המשקיע מסוג RLApproximateQInvestor נכתב על בסיס פעולת האימון של RLQinvestor, תוך התאמת המימוש לאלגוריתם Value Function Approximation (מימוש עצמאי שלנו)

* QTable.py: מבנה נתונים המשמש לאימון של משקיע מסוג RLQInvestor והרצת הסימולטור איתו. הקובץ כולל את טבלת ה-QTable שהסוכן משתמש בה.
* FunctionApproximation.py: משמש לאימון משקיע מסוג RLApproximateQInvestor והרצת הסימולטור איתו. הקובץ כולל מחלקה בשם Estimator שדואגת ללוגיקה של אימון סוכן Approximate Q Learning, לדוגמה שמירת המשקולות, עדכון שלהם, חילוץ הפיצ'רים המהונדסים מתוך מצב (state) כפי שהוא מוגדר בסימולטור ועוד.

תיקיות נוספות

* approximate\_q\_learning\_weights: בתיקייה זו נשמרים קבצי האימון של סוכני RLApproximateQInvestor. צירפנו להגשה את קבצי האימון של סוכן מאומן.
* q\_learning\_q\_table: בתיקייה זו נשמרים קבצי QTable, קבצי האימון של סוכני RLQInvestor. עם ההגשה ישנו קובץ דוגמה של סוכן שהתאמן על מספר נמוך של אפיזודות.
* experiments\_results: בתיקייה זו נשמרים התוצאות של הרצת TestsLogger.py.