
# Önálló laboratóriumi beszámoló

Távközlési és Médiainformatikai Tanszék - TMIT2021-009


| | |
|---|---|
| Készítette: | **Imets Tamás** |
| Neptun-kód: | **JIT72J** |
| Ágazat: | **Mélytanulás** |
| E-mail cím: | **imetstamas@gmail.com** |
| Konzulens: | **Dr. Gyires-Tóth Bálint** |
| E-mail címe: | **toth.b@tmit.bme.hu** |


## Deep learning alapú algoritmikus végrehajtási stratégiák kereskedési platformokon

## Feladat

Az önálló laboratóriumi projekt célja egy olyan mélytanulási algoritmuson alapuló modell fejlesztése, mely segítségével gyakorlatban lehet potenciális profitáló helyzeteket detektálni felhasználva az ármozgásokat, rendeléskönyvi adatokat és múltbéli tranzakciókat.

**2020/2021. 2. félév**

## Bevezető

Ma már a világ legtöbb tőzsdéjén és kriptovaluta-váltó központjában elektronikus „limit order book” (rendelésállomány, továbbiakban LOB) alapján fut a kereskedés, melyek tartalmazzák az adott tőzsdén a kereskedők által publikusan közzétett árajánlatokat és rendelési mennyiséget. A LOB monitorizálásából és elemzéséből a spekulánsok rezisztencia szinteket, piaci hangulatot és likviditási lehetőséget próbálnak megfigyelni, melyet jelen korszakban egyre többen igyekeznek automatizálni [1]. Az ilyen automatizálási eljárások között használnak többek között gépi tanulást és újabban „black box” jellegű modelleket is, mint a deep learning. Utóbbi nagyvállalati környezetben viszont kevésbé használt, ennek az oka meglehetősen egyszerű: a legtöbb ilyen modell nem működik túl jól a zajos pénzügyi adatokon. Deep learning nagyszerűen működik olyan adatokon, ahol könnyen ki lehet bontani egy adott minta vonásait matematikai eljárásokkal, viszont az ár és értékvolumen idősor nem ilyen. Nem véletlen, hogy a publikusan elérhető kutatások nagy része is sok helytelen információt és következtetést tartalmaz. A legtöbb mélytanulás alapú modell könnyen túltanul sok paraméterrel a tanulási adaton és így gyakran teszteléskor csak az előző idősor elemet adják eredményként - ez megtörténhet árakkal, árváltozással, volumennel stb.

Sajnos a legtöbb publikáció, melyet olvastam mind azzal zárult, hogy a kifejlesztett modell nem igazán képes elkapni a gyakorlatban az árváltozásokat, ami indokolható azzal is, hogy túl zajos a modellek számára a pénzügyi adat, vagy azzal is, hogy akinek van működő megoldása az nem publikálja, hogy ne veszítse el a piaci előnyét [2]. Egyetemi témalaboratóriumi projektem alatt sikerült egy jó érzelmi analízis modellt felépíteni hírklasszifikációra, viszont gyakran nem volt használható korreláció az érzelmi index és az ármozgások között. Sokan ezt azzal magyarázzák, hogy az árban már benne van minden információ, amit az adott eszközről ismerünk [3]. Jelen projektemben emiatt azzal próbálkozom meg, hogy LOB és árváltozás alapján keressek anomáliákat kripto valuták mozgásában.

A projektnek abban a tudatban fogtam neki, hogy bár nagyon nagy az esély arra, hogy a többi publikációhoz hasonlóan nem lesznek túl jó eredményeim, minél többet meg akarok tudni a témakörről, hogy ezeket a későbbiekben akár más modellekkel is tudjam alkalmazni. Továbbá, az eddig rekurrens architektúrájú kísérleteimet ki szerettem volna próbálni konvolúciós modellekkel, hátha jobban sikerül a jellemzők detektálása.

## Kivonat

Saját megközelítésemben igyekszem arra fókuszálni, hogy ne egy jövőbeli ármozgást vagy pár időegységgel későbbi adatpontot próbáljak megjósolni, hanem először az adatból próbáljak matematikai és statisztikai alapon kivonni fontos összefüggéseket és használható tudást. Ugyan a deep learnig modellek fő tulajdonsága, hogy képesek a jellemzők automatikus felismerésére, célom, hogy jó minőségű adatot tudjak átadni a kísérleti modelleknek, mely akár önmagában is sok hasznos információt hordoz. A zajos adatsor miatt hasonló praktikákhoz fordulnak a nagy alapkezelők is, projektem során ilyen jól használható adatból igyekszem kialakítani deep learning segítségével még jobb kereskedési modelleket.

Az első dolog, amit nagyon sok helyen olvastam, hogy időbeli mozgás szerint próbálnak építeni gépi tanulási modelleket, mellyel a nagy hiba talán az, hogy a piacot nem az idő, hanem a volumen hajtja. Mintákat felismerni viszonylag egyszerű a tőzsdéken, viszont olyat találni, mely hosszútávon is megbízható jeleket ad nem egyszerű feladat. Pontosan emiatt a kísérletem célja tehát az, hogy anomáliákat, kevésbé helytálló mozgásokat detektáljak mélytanulás segítségével, olyan piaci változásokat, melyek egy nagy volumen nélküli eladási pánikra utalnak vagy ennek az ellentétére, egy mikrobuborékra. Az, hogy a megjelenő anomáliák valóban jelentenek-e piaci előnyt vagy nem egy backtest-tel fogom ellenőrzöm kiszámítva különböző mutatókat egy saját tesztelési környezetben. Ilyen modellekhez előnyös a nagy volatilitás, ezért a legnagyobb kriptovaluta platformról szereztem adatokat különböző eszközök áráról az amerikai dollárt követő Tether-höz viszonyítva.

Több megközelítést is alkalmaztam a projektem során, címkéztem az adatot, de megpróbáltam nem felügyelt tanulást is használni anomáliakereséshez, melyet teszteltem, hogy van-e köze az adott anomáliának olyan piaci mozgáshoz, melyen profitálni lehet. Utóbbi hozta a legjobb eredményt, a címkézett tanítás hasonlóan az árpredikciós modellekhez nem működött jól és egy lineáris regresszióhoz hasonló élesben nem alkalmazható eredményt adott. A felügyeletlen anomáliakeresés jó eredményt hozott a backtesten, viszont szükség van nagyobb adathalmazon való tesztelésre, hogy kiderüljön valóban jól működik-e minden környezetben, ez a modell ugyanis nem feltételez semmilyen címkét az anomália mögött, csak megpróbál nem az idősorba illő elemeket detektálni.

Továbbiakban az önálló laboratóriumi projektem megvalósítása során meghozott döntéseket és eredményeket mutatom be részletesen az adatgyűjtéstől kezdve a backtest eredményekig.

## Adatszerzés

Pénzügyi adatokkal az az egyik fő probléma, hogy szinte minden alapkezelő és bank saját maga szeretné értékesíteni a gyűjtött adatot, mely nem elérhető a nyilvánosság számára. Hasonlóan a LOB adat is egy ilyen, egy díjat kifizetése nélkül nem igazán lehet hozzáférni ilyen forrásokhoz. Szerencsére viszont a modern kereskedési platformokon nagyon könnyen lehet jó minőségű adatot lekérni élőben. Ugyan ezek nem nyúlnak vissza 2017-nél messzebbre, számos nagy likviditású eszközről sikerült adatot szerezni, mely mindegyike tartalmazza OHLCV (open, high, low, close, volume) formátumban időbélyeggel ellátva az árakat és volument.

Ennek megvalósítására egy Python szkriptet írtam, mely a Binanceről kér le adatokat aszinkron módon, vagyis tickerként kapja meg az LOB és tranzakciós történet változásait, melyből egy adatfeldolgozó pipeline automatikusan 1 másodperces bontású adatot állít elő. Az így kapott viszonylag nagy méretű struktúrát egy külső merevlemezre mentettem le későbbi felhasználásra. Összesen kicsivel több, mint 8 gigabájt adatsort használok fel tanítási és tesztelési célokra. Ezen adat teljessége egy pipeline programon megy keresztül, mely mindegyiket ugyan olyan struktúrába rendezi és menti le, hogy alkalmas legyen stratégiák definiálására és modellek tanítására a backtest környezetben.

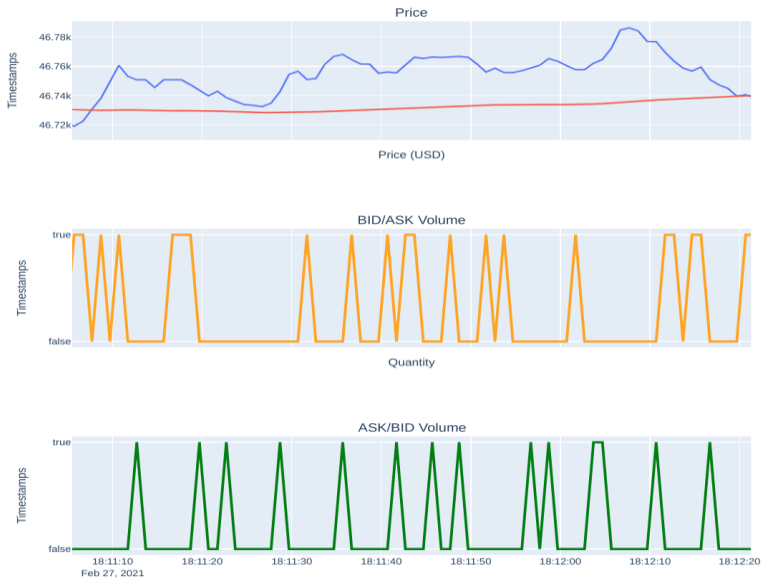
## Adatelemzés

A projektem fontos része az adatelemzés és manuális „feature engineering”. Első lépés ismerkedés az adattal és előzetes pénzügyi és kereskedési ismereteim felhasználása új attribútumok kiszámítására. Ezek közé tartozik mozgóátlagok felvétele, kereszteződések jelzése, relatív erősség index kiszámítása és még néhány művelet a volument tekintve. Bár az effajta indikátorok valós működése nem megalapozott modellalkotáskor úgy tekinthetünk rá, mint egy aluláteresztő szűrőre, mely kevésbé zajos jelet eredményez.

Az LOB-t visszaépítettem árszintekre hozva, megadott árkülönbségek szerint summáztam az adott szintre jutó volument. Ezt mindkét irányba összesen 20 szinten állítottam vissza, hogy az váltóközponton is hasonló rendeléskönyvet kapjak eredményül. Ezzel az eljárással az volt a fő reményem, hogy rezisztencia vagy szupport szinteket fogok tudni a modell segítségével találni. (Pl.: anomália, ha egy ilyen pár milliós rezisztencia szint pár perc alatt felvásárolódik.)

Konzulensem tanácsára, hogy ne csak olyan adatokban gondolkodjunk, melyek emberként használunk, ennek inverzét is kiszámoltam, vagyis adott volumenre az átlagos árszintet, ami szintén sok információt elárulhat a piaci hangulatról. Az átlagos eladási- és vételári LOB közötti különbség elárulhatja milyen irányba alakulhat ki új trend.

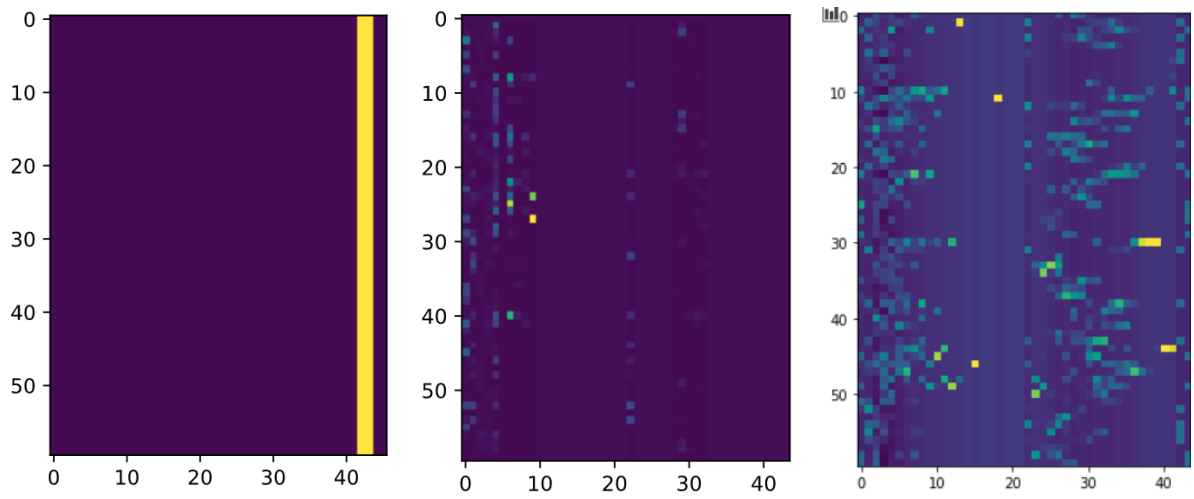
Érdekes jelenséget fedeztem fel a nagy esések volumenváltozása és az ellentétes trend kialakulása között. Erre különösen figyeltem a jellemzők hozzáadásakor, ezért bevezettem az átlagos eladási ár és átlagos vételár volumeneket hányadosát és ennek fordítottan arányos megfelelőjét. Ez hasonló az előző bekezdésben említett jellemzőhöz, melyet a neuronháló akár meg tud magától is közelíteni, viszont manuális hozzáadásával kisebb háló is elégséges lehet, mely kevésbé könnyen tanul túl a zajos adaton [4].




***1. ábra**: Árfolyam árbrázolása viszonyítva az eladási és vételi volumenek hányadosához. Hogy pontosabb*

Ehhez hozzáadtam a volumen függvényében történt árváltozást, mely erős indikációja lehet egy nagy, ugyanakkor fals és gyorsan korrigáló piaci mozgásnak, százalékos árváltozást, volumen változását, mozgóátlagok alapján trend bélyeget, zajcsökkentést és trend erősséget, illetve túlértékelt és alulértékelt indikátor jellemzőket. Erre szintén azért lehet szükség, mert egy megfelelő méretű neurális hálózat nem biztos, hogy felismeri ezeket a jellemzőket egy konvolúciós architektúra segítségével. Lehet, hogy csak a zajra tanul rá, amit rengeteg publikációban és interneten közzétett munkában láttam. Az új jellemzők hozzáadása után egy 174.000 rekordot tartalmazó adathalmazt kaptam eredményül, melyet későbbi felhasználásra lementettem. (Elérhető a projekt GitHub oldalán [5].)

A rekordok ilyen állapotban csak a pillanatnyi LOB és áradatokat tartalmazza, ezekhez a tanítás előtt érdemes az előző rekordokat hozzávenni, hogy így egy nagyobb mátrix alakuljon ki a struktúrából. Ebben a struktúrában az oszlopok jelentik a jellemzőket és a sorok az idő szerinti változást, melynek dimenziója 60x44, azaz az elmúlt 1 perc adatait használom fel anomália keresésre és klasszifikációra. Az így kapott monokróm képek viszont teljesen eltérő nagyságrendű oszlopokat tartalmaznak, mely problémát okozhat a neuronháló tanításakor. Több megoldást próbáltam ezen akadály megoldására, többek között az adat teljes normalizálást, ablakonkénti standardizálást, ablakonkénti normalizálást és min-max skálázást.




***2. ábra***: *Bal oldalon: skálázás nélküli adat; középső: ablakonként normalizált adat; jobb oldali: standardizált adat.*

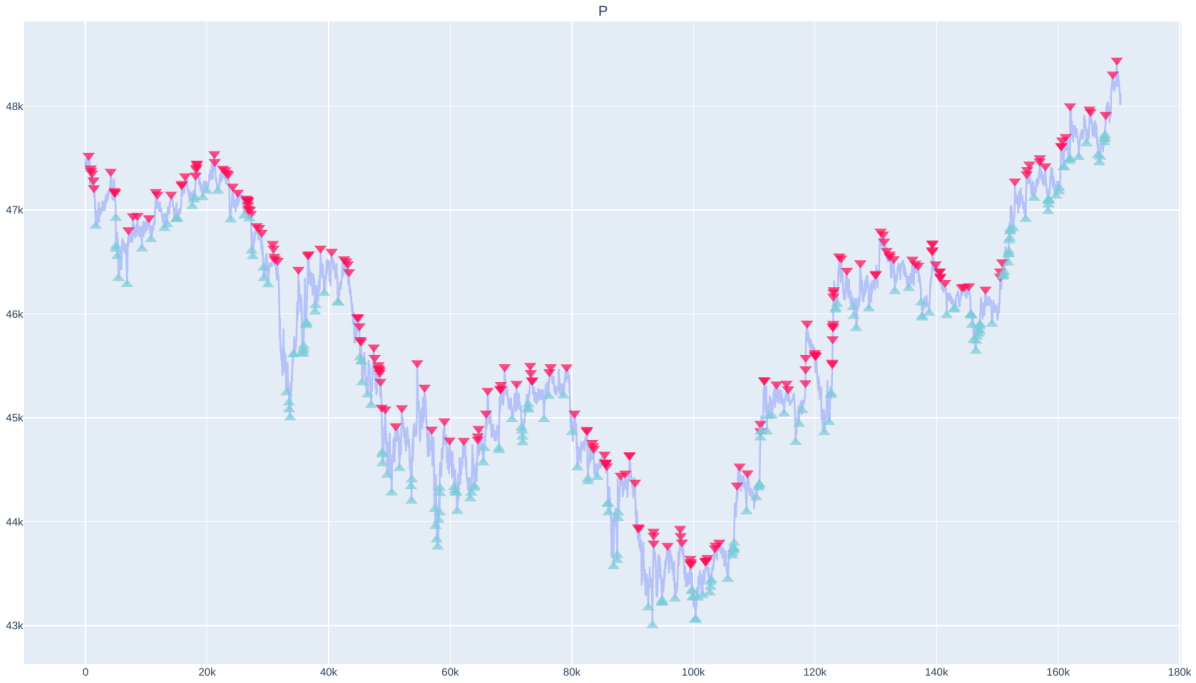
Ahogy 2. ábrán is látszik a nem feldolgozott ablakokon minden elsötétül a nagyságrendekkel nagyobb eladási- és vételári oszlopok mellett, ami azt jelenti, hogyha ezeket ábrázolnánk egy koordinátarendszerben teljesen el lennének tolva a rendeléskönyv és többi jellemzőkhöz képest. Az első módszer melyet próbáltam a tanító és tesztelési adat skálázása. (Az illesztés a tanító adaton történt.) Mivel idősor adatról van szó, mely változik az idő előrehaladtával és gyakran véletlenszerű, nem repetitív ez a megoldás nem bizonyult hatékonynak. Következő megoldás az ablakonkénti normalizálás volt. Ennek eredménye a 2. ábrán középen tekinthető meg, melyet úgy számol ki a program, hogy leosztja az első sorral az időben utána lévő sorokat, ezzel kifejezve a nagyságrendbeli változást. A módszer előnye, hogy nagyon jól reprezentálja az ár változását, viszont az LOB százalékos változása nem biztos, hogy így a legjobb

reprezentációt kapja. Gyakran *nan* és *inf* értékek keletkeztek, melyet külön kezelni kellett, sok szint 0-s értéket kapott, ha az első sorban szereplő érték 0 vagy túl nagy szám volt.

Egy olyan megoldás kellett tehát, melyen egy ablakon belül helyesen skálázódnak a saját jellemzők, az LOB és az áradatok is. A *Deep Learning a gyakorlatban Python és Lua* nevű tárgy egyik előadására alapozva képenként skáláztam az adatokat, mindegyik képet önmagára illesztve és standardizálva [6]. Ez helyesen normalizálja az árakat és az LOB adatok sem vésznek el *null* vagy *inf* értékként. Az eredmény az X. ábrán figyelhető meg. Akár szemmel is kivehető néhány összefüggés a képek között, de ez valószínűleg csak a véletlen műve.

Mivel a projekt célja profitszerzési lehetőségek detektálása, ezért eleinte úgy gondoltam, hogy szükség van címkézésre. Régebbi projektjeimben, többek között a témalaboratóriumi projektemben is, úgy oldottam meg a címkézést, hogy bizonyos időközzel egy adott árváltozási értéket vagy osztályt (pl.: fel/le) próbál megtanulni a modell. Eddigi tapasztalataim és az olvasott publikációk alapján úgy döntöttem, hogy inkább belépési pontokat keressen a program és ne jövőbeli mozgásokat próbáljon megjósolni, hátha így jobban lehet alkalmazni egy kereskedési stratégiában.

Hogy ne manuálisan kelljen definiáljak idő és átlagos ármozgás vagy kvartilisek szerint címkéket ezért egy új módszert próbáltam ki: ablakokra bontva az áradaton kerestem a *scipy* könyvtár segítségével lokális minimum és maximum értékeket, melyek konkrét trendváltozást mutatnak. (Az így kapott címkék biztosan profitáló belépési és kilépési pontokat adnak meg, melyet a 3. ábra szemléltet.)




***3. ábra**: Belépési (felfele nyíl) és kilépési (lefele nyíl) címkék lokális maximum és minimum kereséssel.*

## Benchmark

A projekt célja egy gyakorlati hasznú gépi tanulási modell fejlesztése, ezért nagy hangsúlyt fektettem a benchmarkinkra. A tesztelésre egy saját backtest környzetet használok, melyet a témalaboratóriumi projektem során fejlesztettem, de a végleges verzió az önálló laboratórium alatt lett kész. A keretrendszer lényege, hogy Pythonban könnyen lehessen definiálni kereskedési stratégiákat indikátorok vagy gépi tanulás alapján és ezeket egy teljes körű elemzésnek alávetve kapjunk képet a program teljesítményéről. A *plotly* nevű adatvizualizációs könyvtár segítségével összerakott grafikus felület kijelzi az egyenleg változását, drawdown változást, profitokat és veszteségeket az idő folyamán és rövid eladási, valamint vételi pozíciókat és ezek kilépését vizualizálva az árgrafikonon. Külön grafikinon jeleníti meg a stratégia teljesítményét viszonyítva az S&P 500, DAX és DJIA tőzsdeindexekhez valamint a B&H-hoz (buy and hold, pozíció tartása) képest. Továbbá, a program kiszámolja a leggyakrabban használt kereskedési metrikákat: Sortino ratio [7], Sharpe ratio [8], drawdown kockázata (mennyi az esély arra, hogy egy adott százalékot csökken az egyenleg), szórás, átlagos nyereség és veszteség, ezek szórása, profit tényező és nyertes trédek hányadosa.

A végleges deep learning modellt pontosságát és hozamát egy lineáris regresszió alapú trendfordulási kereskedési stratégiával vetem össze egy backtesten. Jó eredménynek mondhatunk egy olyan nem látott adaton végzett éles tesztet, mely növeli a hozamot vagy csökkenti a maximális egyenlegsüllyedést (drawdownt). Mivel a modell százalékos pontossága nem teljesen mérvadó pénzügyi környezetben ezért olyan metrikának vetem alá a modelleket, mely figyelembe veszi egy predikció profitabilitását. Például egy 50%-os pontosságú modell sokkal jobb, ha képes előre jelezni átlagosan kétszer nagyobb mértékű helyes mozgásokat, mint helytelent egy olyannal szemben, amely 80%-ban jó pozíciókat nyit és zár, viszont a maradék 20% kitörli a teljes nyereséget. Erre jó indikációt ad a backtest és a predikciót követő átlagos ármozgás nagyságához viszonyítva a pontosság. A pénzügyekben használt egyenlegcsökkenés kockázatának képletéből kihozható ellentétesen a következő profitabilitási metrika pszeudokódban megadva [9]:

```
// valószínűségi tényező
p = 0.5 * (1 + ((win_rate * average_win) – (losing_rate * average_loss)) /
sqrt((win_rate * average_win) ^ 2 – (losing_rate * average_loss) ^ 2)))

// profitabilitási valószínűség
pop = abs(p / (1-p))

// annak a valószínűségét, hogy elérünk 10% profitot így kapjuk meg
pop10 = min(ror ** (0.1 / (sqrt((win_rate * average_win) ^ 2 – (losing_rate
* average_loss) ^ 2)), 1)
```

## Modellépítés és Eredmények

Baseline modellnek a kísérlet során egyszerű logisztikus regressziót használtam, mely ezen az adaton teljesen véletlen eredményt adott, vagyis azt tanulta meg, hogy mindegyik mintát ugyan abba az osztályba sorolja be. Ez 50%-os AUC (area under curve) pontosságot eredményezett így ennek a backtest teljesítményét nem is részletezem.

Első megközelítésben a címkézett adathoz építettem egy előre csatolt (feedforward - FFW) architektúrájú neurális hálózatot mely koncepcióját hamar eldobtam és 2D-s konvolúciós hálózatokkal folytattam a kísérletet. Mivel az egyik osztály sokszorosan alulreprezentált ezért a tanítást úgy végeztem konzulensem tanácsára, hogy a *keras* könyvtár segítségével súlyoztam az osztályokat. A féléves munkám kicsit lehangoló része az, hogy bár nagyon sokat dolgoztam hiperparaméter hangolással, a modellek méretével, kipróbáltam az előző félévben tanult *hyperopt* alapú automatikus hangolást is, még ez a megközelítés sem működött túl jól. Az egész rendszer pontossága attól függően konvergált egy adott pontba, hogy milyen súlyokat adtam meg az osztályoknak, vagyis a becslés körülbelül véletlen volt. Kicsit sikerült ezt azzal javítani, hogy csökkentettem a modell méretét, ez által kevesebb paraméterrel dolgozik és kevésbé tud túltanulni, de ez sem vezetett gyakorlatban használható eredményre. Ezen még sikerült enyhén javítani, amikor ránéztem a konkrét prediktált értékekre és azt figyeltem meg, hogy mind ugyan azok az értékek lesznek a kimenetek. Ez csak akkor történhet meg, ha a súlyok zérus közeliek és csak a bias tanult rá erre az értékre. Miután minden rétegben letiltottam a biasokat a háló sokkal jobban tanult, de a bináris pontossága még így is a véletlenhez konvergált.

Következő módszer, melyet kipróbáltam az anomáliakeresés volt. Ebben az esetben a címkék nélkül közelítettem meg a problémát, mert úgy tapasztaltam, hogy ezen az adaton nem fog jó eredményre vezetni a címke. Egy autoencodert építettem szintén 2D-s konvolúciós rétegekkel, mely bemenetként a standardizált 60x44-es monokróm ablakot várja és kimenetként egy ugyan ekkora dimenziójú képet ad vissza. Ebben az esetben is az működött a legjobban, ha kevés paraméterrel rendelkező modellt tanítottam Adam optimizációs algoritmussal és lineáris kimenettel. Így összesen 3217 paramétere van, bias-ok nélkül.

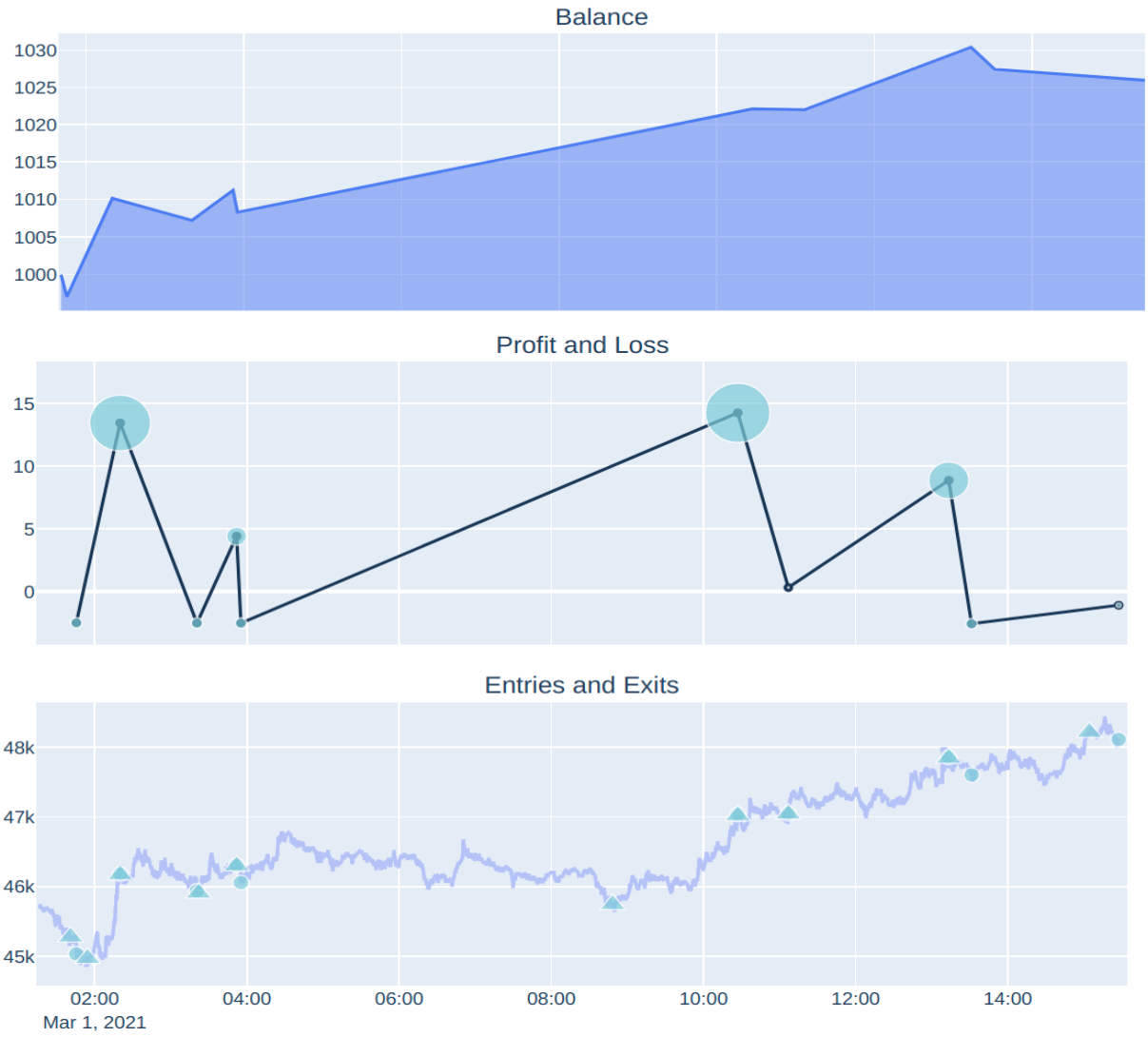
Ennél a megoldásnál az elkészített backtest keretrendszer lehetőségével éltem. A modellel sikerült nagyon jól kiszűrni a ritka mintákat, mert voltak olyan esetek, amikor teljesen eltérő képet kaptam a bementhez képest. Így azt viszont nem tudjuk, hogy milyen prediktív erővel és relevanciával rendelkeznek ezek az anomáliák. A modellhez egy kereskedési ágenst (agent) írtam, mely a tesztadaton (30% tesztadat) az idő előrehaladását szimulálva ad predikciókat mindegy egyes ablakra. Ha anomáliát detektálunk az ágens azt feltételezi, hogy potenciális


***4. ábra:*** *A stratégia (középső görbe) viszonyítva a DJIA, S&P 500 (alsók) és B&H-hoz (felső) képest.*

belépési pontnál vagyunk és egy pozíciót nyit az adott vételáron. A pozíció nyitásának feltétele ezen kívül egy mozgóátlag értéke, mely kisebb kell legyen a jelenlegi középárfolyamnál.

Hogy kiküszöböljem a slippaggel (csúszás az árfolyamban késés vagy kihatás miatt) járó ármozgást az eredeti komisszió 125%-át vettem, mely így 0.05%-os költséget jelent nyitáskor és záráskor is. Ha az ágens tehát képes pár másodperc alatt 0.1%-nál nagyobb pozitív ármozgást eltalálni, akkor profitban van. Ehhez természetesen nagyon közeli *take profit* és *stop loss* szinteket helyez el a program, hiszen csak egy pár másodperces mozgást akar elkapni.

Az elgondolás bevált és sokkal jobban működött, mint a címkézett adattal felépített neuronháló. A jó eredmény lehet csak a piac természete miatt van, ezért a félév végén elindítottam még egy adatgyűjtést, hogy ezt teljes egészében csak tesztelésre használjam. Sajnos ezeket az eredményeket még nem tudom feltüntetni a beszámolómban, viszont a 4-5. ábrák képet adnak az eddigi sikerekről. A szimulációt fiktív 1000 dollárral indítottam, melyen a két napos tesztelési időszak alatt 2.5% profitot generált a program. A következő adatgyűjtés végén várhatóan 20-30-szor több adattal lesz lehetőségem tesztelni a gyűjtés végén.




***5. ábra:*** *Fent: egyenlegváltozás a stratégia trédjei alapján; középen: az egyes pozíciók profitjai és veszteségei; alul: be- és kilépő pontok az áron vizualizálva.*

| Trédek száma | Profit és veszteség | Profit Faktor | Nyereségi százalék | Maximális Drawdown | PoP 10% | Sharpe |
|---|---|---|---|---|---|---|
| 10 | +25.95 | 2.96 | 50% | -4.5 (-0.44%) | 38.12% | 0.55 |

***1. táblázat:*** *A backtest környzet által számolt teljesítménymetrikák.*

## Konklúzió

Az önálló laboratóriumi projektem által tovább bővítettem a tudásom a pénzügyi adatfeldolgozás terén és kipróbáltam eddig általam pénzügyi környezetben még nem használt technikákat. Az adatok zajos természete miatt nem igazán lehet felépíteni jövőt jósló modelleket és bár az autoencoder jó eredményhez vezetett, nem tudok még megbizonyosodni a gyakorlatban való használhatóságáról. Amit viszont megtanultam és talán a szakdolgozatomban is tovább fogok vinni az az, hogy deep learning segítségével és bayes-i közelítéssel (Tree-structured Parzen Estimator - TPE, fa struktúrájú parzen becslési algoritmus) jó eredménnyel lehet optimalizálni a paraméterezett stratégiákat.

## Hivatkozások és felhasznált irodalom


[1] Dani Burger, „The U.S. Stock Market Belongs to Bots”, https://www.bloomberg.com/news/articles/2017-06-15/it-s-a-quant-s-stock-market-as-computer-programs-keep-on-buying, letöltés ideje: 2020. 05. 03.

[2] Zihao Zhang, Stefan Zohren, and Stephen Roberts, „DeepLOB: Deep Convolutional Neural Networks for Limit Order Books”, 2020. 01. 23., elérhető: https://arxiv.org/pdf/1808.03668.pdf

[3] Burton G. Malkiel, „The Efficient Market Hypothesis and Its Critics”, Princeton University, 2003 április, elérhető: https://www.princeton.edu/~ceps/workingpapers/91malkiel.pdf

[4] „Overfit and underfit”, https://www.tensorflow.org/tutorials/keras/overfit_and_underfit, letöltés ideje: 2020. 05. 03.

[5] Forráskód tárolója, https://github.com/arathus/onlab2021, letöltés ideje: 2020. 05. 03.

[6] Dr. Gyires-Tóth Bálint, Dr. Csapó Tamás Gábor, Dr. Zainkó Csaba, “Deep Learning a gyakorlatban Python és LUA alapon”, VITMAV45, 2020

[7] Will Kenton, Margaret James, „Sortino Ratio Definition”, https://www.investopedia.com/terms/s/sortinoratio.asp, letöltés ideje: 2020. 05. 03.

[8] Jason Fernando, Margaret James, „Sharpe Ratio”, https://www.investopedia.com/terms/s/sharperatio.asp, letöltés ideje: 2020. 05. 03.

[9] François Dufresne, Hans U. Gerber, „Three Methods to Calculate the Probability of Ruin”, Cambridge University Press, 2014. 08. 29., elérhető: https://www.cambridge.org/core/journals/astin-bulletin-journal-of-the-iaa/article/three-methods-to-calculate-the-probability-of-ruin/BD33EDF2AEE7D364C99B9E2EE32406F2#article

[10] Nishant Kumar, „Why Machine Learning Hasn't Made Investors Smarter”, https://www.bloomberg.com/news/articles/2019-07-11/why-machine-learning-hasn-t-made-investors-smarter-quicktake, letöltés ideje: 2020. 05. 03.

[11] Gregory Zuckerman, „The Man Who Solved the Market: How Jim Simons Launched the Quant Revolution”, Portfolio kiadó, 2019

[12] Maria Giuseppina Bruni, „An alternate method for calculating the probability of ruin is risk theory”, La Sapienza University, elérhető: http://www.actuaries.org/ASTIN/Colloquia/Porto_Cervo/Bruno.pdf

[13] Simon Hawkins, Hongxing He, Graham Williams and Rohan Baxter, „Outlier Detection Using Replicator Neural Networks”, CSIRO Mathematical and Information Sciences, elérhető: https://togaware.com/papers/dawak02.pdf