

OTDK DOLGOZAT

OPCIÓÉRTÉKELŐ, MESTERSÉGES INTELLIGENCIÁRA ÉPÜLŐ VOLATILITÁS MODELLEL

Szerző:

Rigó Árpád

Óbudai Egyetem mérnökinformatikus
BSc. szak, IV. évfolyam

Konzulens:

Dr. Tusor Balázs

adjunktus

Szekció:

Műszaki Tudományi

Budapest, 2023.

Tartalomjegyzék

1. Bevezetés	5
1.1 Az opciós kereskedés modellezésének minősége	5
1.2 Az opciós modellezők iránti elvárások	7
1.3 Az opciós modellezés lényege és működése	8
2. Irodalomkutatás és a lehetséges módszertanok feltárása.....	10
2.1 Tőzsdei alapfogalmak	10
2.2 A volatilitás fogalma	11
2.3 A volatilitás meghatározó tényezői.....	11
2.4 A létező modellezők, funkcióik és hibáik	12
2.4.1 Thinkorswim	12
2.5 Volatilitás-jóslás mély neurális hálózat segítségével	14
2.5.1 Deep Learning	14
2.5.2 A hálózat felépítése.....	15
2.5.3 Gradient Descent eljárás.....	16
2.5.4 Neuron felépítése.....	17
2.5.5 Aktivációs függvény.....	17
2.5.6 A megfelelő tanítás és tanító minta	19
3. Követelmény specifikáció	21
3.1 A szoftver használata, tulajdonságai.....	21
4. Fejlesztői specifikáció.....	24
4.1 Python alapú volatilitás-jósló egység.....	24
4.1.1 Histórikus adatbázis árak	27
4.2 Opció feldolgozó	28
4.3 PostgreSQL Database	28
4.4 Backend Node.js (Typescript).....	29
4.5 Frontend React (Typescript).....	29
5. A szoftver megvalósítása.....	30
5.1 Histórikus adatbázist feldolgozó egység	30
5.2 Az adattisztítás folyamatáról.....	30
5.3 Mély neurális hálózat megvalósítása volatilitás jóslására	31
5.4 Statisztika alapú volatilitás-beclés megvalósítása	33
6. Tesztelés	37

6.1	Hasonló opciók és volatilitásuk lekérdezése funkció	37
6.2	Hasonló opciók volatilitásainak átlaga funkció tesztelése	38
6.3	Portfólió elemzésének tesztelése, Thinkorswimmel való összehasonlítása	39
6.4	Portfólió elemzése minimum, maximum volatilitás-beccsléssel, ársáv használatával	41
6.5	Portfólió betöltése Api segítségével	41
7.	Kiértékelés	42
8.	Összegzés	43
	Irodalomjegyzék	45
	Ábrajegyzék	47

Kivonat

Az opciók komoly részt foglalnak el egy valódi kockázatkezelést folytató befektető portfóliójában, viszont általános elterjedésüknek gátat szab látszólagos bonyolultságuk, mivel több tényező is befolyásolja az árazásukat. Egy részvényre egy adott időben akár több száz opció is jegyzésre kerül, ezek határidőkre bontva láthatóak, egy határidőn belül több tucat lehívási adat, majd ezen opciók két csoportra bontva jelennek meg a piacon lévő szoftverekben: eladási és vételiként.

A szakdolgozat célja, hogy rávilágítson a piacon lévő opciós portfóliókkal foglalkozó szoftverek pontatlanságára. Ezen szoftverek közül a Thinkorswim kockázatanalízis funkcióján keresztül szemlélteti a problémát, mely egy Amerikai Egyesült Államokbeli opciós kereskedésekre specializálódott rendszer. A pontatlanság a nem megfelelően meghatározott implikált volatilitásból származik, mely az opció árazásába foglalt tényező. Pontatlan opciós árral számolt portfólió esetén nem teljesül az árfolyamvárakozás beteljesülésekor a szoftver által meghatározott nyereség-veszteség alakulása.

Erre a hibára nyújt megoldást a szakdolgozatban bemutatott rendszer. Jövőbeli implikált volatilitást határoz meg mesterséges intelligencia és statisztikai úton, majd az ebből származó jövőbeli implikált volatilitást használja fel a Black Scholes modell segítségével az opciós portfólió nyereség-veszteség görbe kirajzolásához.

Mély neurális hálózatként a Rekurrens neurális hálózatok egy speciális fajtáját, az LSTM (Long short-term memory) típusú modellt használja az opciók idősorrendiségének a figyelembevételére érdekében.

Statisztikai megközelítésként egy előre meghatározott részvény opcióinak histórikus adatbázisából keres a rendszer a portfólió opcióihoz hasonlókat, majd ennek az átlagát használja fel jövőbeli implikált volatilitásként.

A rendszer React Js webes felhasználói interfészt használ, ez által kommunikál a felhasználóval, Node Js backenddel. A histórikus adatokat PostgreSQL adatbázisban tárolja és Python alapú mesterséges intelligencia microservice-el rendelkezik. Ezek az egységek dockerizáltak vannak megvalósítva.

1. Bevezetés

Az opciók komoly részt foglalnak el egy valódi kockázatkezelést folytató befektető portfóliójában. Az előző mondat fontos szava a „valódi” – a 2020 február-márciusi események következményeképpen a befektetési alapkezelők által kezelt alapoknál megfigyelhető árfolyamesések rávilágítottak, hogy sok piaci szereplő gondolja úgy, hogy jó a kockázatkezelése – miközben nem az. A tőkepiac nagy játékosai viszont nagy mennyiségben és szívesen élnek az opciós univerzum adta lehetőségekkel, és a nyomukban sok kisbefektető is megismerkedett az opciózás néhány alapelemével. Az őszi-téli, egyesült államokbeli közösségi portálokról induló részvényárfolyam-hajszák egyik alappillére volt az opciók használata.

Az opciók általános elterjedésének gátat szab látszólagos bonyolultságuk. Mivel több tényező is befolyásolja az árazásukat, ezért gyakrabban fordul elő negatív meglepetés a használatuk során, mint pozitív. Ezzel szemben a részvényből vagy devizából felvett „egyenest” pozíció, azaz a vétel vagy eladás nyereség-veszteség egyenlete feltűnően egyszerű: darabszám $\times$ árfolyamváltozás. Ezt bárki képes követni, Excel-képletbe foglalása sem okoz gondot senkinek, ezért fordul elő, hogy a sok energiát az árfolyam-változás irányának és mértékének előrejelzésébe fektetik, és jóslatukat a legegyszerűbb módszerekkel igyekeznek nyereséggé alakítani. Még az összetett várakozások meglelte sem ok sokaknak arra, hogy beletanuljanak az opcióba, holott például egy olyan meglátást, hogy „az árfolyam ebben az ársávban marad”, opciókkal könnyűszerrel lehet nyereségre váltani, míg egyenes ügyletekkel semmiképpen.

1.1 Az opciós kereskedés modellezésének minősége

Az opciók eme viszonylag szűk területre szorulásának oka összetett voltuk, árazásuknak különböző tényezőktől való többszörös függése. Eleve megjelenésük már riasztó, ahogy ez az 1. ábrán is látható: egy részvényre egy adott időben akár több száz opció is jegyzésre kerül, ezek határidőkre bontva láthatóak, egy határidőn belül több tucat lehívási adat, egyik oldalt a vételiek, másikon az eladásiak – a legtöbb ember ezen a ponton már feladja a küzdelmet. A tovább lépők egy része akkor száll ki, amikor az alaptermékre vonatkoztatott ár várakozása igazgá válik, de az opción nem lesz meg neki az elvárt eredmény. Valami nem várt történik, és ez csalódást okoz. Akiket ennél tovább visz a szakmai igényük, azok megtalálják – vagy a

brókerük kereskedési felületén, vagy független szolgáltatónál - az opciós pozíciók modellezésére, időbeli áralakulására írt kiértékelő-programokat. Az utolsó csaldódás itt éri a szereplőket: a gondosan kiválogatott pozíciók, az odafigyeléssel kezelt modellezőprogram előrejelzése ellenére is előállhat a veszteség.

The screenshot displays the Thinkorswim desktop interface for AAPL stock. The top section shows the stock price at 122.54. Below this, the 'Option Chain' is expanded, showing a list of call and put options for various expiration dates. The interface includes tabs for 'Underlying', 'Trade Grid', and 'Option Chain'. The 'Option Chain' tab is active, showing a table of options with columns for Implied Volatility, Probability of OTM, Delta, Bid/Ask prices, and the underlying stock price. The table is organized by expiration date, with rows for 26 MAR 21, 1 APR 21, 9 APR 21, 16 APR 21, and 23 APR 21. Each row lists multiple call and put options with their respective prices and volatilities. The interface also includes 'Order Entry Tools' and 'Order and Strategy Book' sections at the bottom.

1. ábra: Opciók bonyolultsága (Forrás: Thinkorswim desktop).

Jó néhány év kereskedési tapasztalattal rendelkezők szerint kijelenthető, hogy készíthető olyan modellező, amely megfelelően tájékoztat a várható kimenetekről. Általánosságban igaz, hogy a forgalomban lévő modellezők egyszerűsítésekkel élnek, túl széles körben számítják a statisztikai adataikat, alapvető változásokat okozó tényezőket nem vesznek időnként figyelembe. Ezen hibák egyike is elég, hogy sok-sok dollárral arrébb tegye a végeredményt, szinte soha nem a felhasználó örömére. Még az adatok pontosságán túl is van azonban néhány kérdés: hogyan is kellene jól használni egy modellezőt, hogyan lehetne széles körben

használhatóvá tenni az opciókat. Az opciók használata ma olyan viszonyban van a széleskörű használhatósággal, mintha az okostelefonunk helyett kis kerekeken húznánk egy számítógépet, tízkilós akkumulátorral, képernyővel és billentyűzettel, ráadásul magunknak kellene javítani is azt.

Az általam javasolt program kísérletet tesz a hibák javításán túl az opciózás „okostelefonjának” megvalósítására is.

1.2 Az opciós modellezők iránti elvárások

Megfogalmazhatjuk, hogy mit várunk el egy kereskedés-támogató rendszertől. Az elvárásaink között rangsort állíthatunk, a rangsorolás alapelvének azt állítva, mennyire pontosan jelzi előre a várható kimeneteket a szoftver. Egyéb szempontjaink is lehetnének sorba rendezéshez, de könnyen beláthatjuk, hogy a mindenféle kereső, adatszolgáltató funkció mellett, ha az általunk kiválasztott opciókról vagy stratégiákról nem kellő biztonsággal tudjuk meg, mit hozhatnak, akkor vakrepülést végzünk a kereskedési tőkénkkel. Az egész kereskedési folyamat oka a profitra törekvés – ha nem lehetünk eléggé bizonyosak a jövőbeli nyereségben, vagy már akkor is, ha nem vagyunk tisztában a bekövetkező kockázatokkal, akkor nem is irányíthatjuk a folyamatokat, nem várhatunk nyereséget, tehát ok nélkül cselekszünk. Az elvárásaink tehát, sorban:

- a kiválasztott opció(k) jövőbeli nyereség/veszteség kockázatainak minél pontosabb feltárása;
- a kockázatokat jelentő események, piaci helyzetek minél pontosabb jelzése;
- pontos volatilitás-statisztikák;
- nyereséggel kecsegtető helyzetek keresése;
- elvárt nyereség/veszteség helyzeteket megvalósító opciók keresése.

A rangsor első utáni helyezettjei egyrészt kiegészítik az első elvárást, másrészt segítik a felhasználót eligazodni az opciók univerzumában. Az utolsó elvárás teljesen saját ötlet, jelenleg nincs ilyen funkcionalitás a nyilvános opciós szoftverek világában. Ez a lehetőség - a nyereség/veszteség meghatározott mértékét potenciálisan előállító opciók sikeres kiválasztása - lehet az opciózás okostelefonja: az általa elviselhető nyereséggörbét bárki meg tudja rajzolni, és ha jól választ szoftver eszközöket, akkor erősen leegyszerűsödik az opciózás „művészete” [1].

1.3 Az opciós modellezés lényege és működése

Az opciós pozíciókból álló portfólió összefogott, tudományos igényű követése egyáltalán nem általános jelenség az értékpapír-piacon. A fejlesztés közben erről nekünk csupán az jut eszünkbe, hogy aki cipőügynök letétre 100 évvel ezelőtt azzal fordult vissza Afrikából, hogy nincs értelme oda cipőt eladni, mert senki nem hord olyat, az nem vette észre, hogy Afrika azért nagy piac, mert még senki nem hord cipőt. Az opciókat szakszerűen értékelni anyagi létkérdés, és aki ezt közérthetően tudja megoldani, az széles tömegek opciókhoz való hozzáféréseinek alapoz meg. Az opciókat azért nem használja ugyanaz a felhasználói kör, aki tőzsdézik, mert a velük való foglalatosság szakmailag kihívások elé állítja őket: nem tudják előre, hogy mi lesz az eredmény, még akkor sem, amikor bejön az árfolyamváromlásuk.

Az opciós modellezés lényege tehát az kell legyen, hogy közérthető formában a lehető legpontosabb nyereség-előrejelzést adja az opciós portfóliókról.

Az opciós modellezés akként működik, hogy a meglévő portfóliónk minden egyes elemének kiszámoljuk egy megadott jövőbeli időpontra vonatkoztatott értékét, és ezen értékeket összeadjuk. Ezen a folyamat legegyszerűbb része a vége, az összeadás. A szakmai munka az elemek jövőjének meghatározásában van – az itt alkalmazott módszerek (az azokban rejlő esetleges egyszerűsítések) különbözősége miatt a kiszámolt értékek között jelentős, akár anyagi bukást jelentő különbségek vannak. Vegyük sorra, mi befolyásolja egy adott opció jövőbeli értékét:

- A lejáratig hátralévő idő. Egy opció mindig megadott határidőig él. Értékét fokozatosan, de nem egyenletesen veszíti a határidőhöz közeledve.
- Az opció lehívási árának az alaptermék pillanatnyi árfolyamától mért távolsága.
- Vételi vagy eladási opcióról van-e szó?
- Fizet-e és mikor az alaptermék valamilyen összeget az opció lejáratáig?
- A kockázatmentes (pl. állampapír) kamatláb a piacon
- A volatilitás

Az első öt tényező lineáris, vagy (az első esetben) attól csak egy kicsit bonyolultabb függvénnyel hat az opció értékére. Ha csak ezekből a tényezőkből állna az opciós matematika, akkor 100 %-hoz közeli biztonsággal lehetne előre jelezni az opciók áralakulását, és senki nem csalódna, azt kapná, amire számított. A volatilitás, mint tényező azonban átírja a számításokat.

A dolgozat felépítése a következő. A 2. fejezetben irodalomkutatás keretein belül bemutatom a témát és a lehetséges módszertanokat, majd a 3. fejezetben a követelmény specifikációt foglalom össze. A 4. fejezetben a fejlesztői specifikációt részletezem, majd az 5. fejezetben a szoftver megvalósítását írom le. Végül a 6. fejezetben összefoglalom a dolgozatot és a továbbfejlesztési irányokat tűzöm ki.

2. Irodalomkutatás és a lehetséges módszertanok feltárása

A tervezett szoftver lefejlesztése előtt fontos lépés az irodalomkutatás, ezáltal hasonló rendszerek, megoldási módszertanok ismeretére tettem szert. Ez a mérnöki szemlélet egy fontos lépése, így a már létező rendszerek hasonló felépítéséből ihletet merítve hatékonyabbá tehetem a sajátom.

2.1 Tőzsdei alapfogalmak

A következőkben tisztázom a témával kapcsolatos alapvető fogalmakat, a probléma mélyebb megértése végett [2]:

- **Részvény:** Egyfajta értékpapír; akinek a tulajdonában áll, annak tulajdonosi jogot biztosít az adott vállalatban a névértékének erejéig. Akinek részvénye van egy adott vállalatban, az jogosult a vállalati nyereség egy arányos részére és az osztalékra is. A részvény névértéke az az érték, amelyet a részvényen feltüntetnek. Ezen érték arányában van szavazati joga a tulajdonosának és ez alapján kap osztalékot is.
- **Kibocsátási ár:** Az az ár, amennyiért a részvényt a kibocsátásakor az első tulajdonos megveheti.
- **Árfolyam:** Az az összeg, amennyiért a részvényt a piacon meg lehet vásárolni.
- **Opció:** Az opciós ügylet egy olyan szerződés, amely az egyik félnek vételi vagy egyéb jogot biztosít valamelyik értékpapír megvásárlására.
- **Az opciónak két típusa van: Call (vételi) opció, illetve a Put (eladási) opció.** A vételi opció vételi jogot biztosít a vevőjének, míg az opció kiírója kötelezettséget vállal az eladásra. Ezzel szemben az eladási opció eladási jogot biztosít jogosultjának, míg az opció kiírója kötelezettséget vállal a megvásárlására.
- **Típusai: Európai és amerikai opció.** Az európai opciók esetében a joggal csak az opciók lejáratakor lehet élni, míg az amerikai típusú opciók esetében a joggal az opciók lejáratáig bármikor lehet élni. Ezért a szoftverem amerikai típusú opciókkal fog kereskedni.
- **ITM: Az In the money (ITM) opciónak van belső értéke,** vagyis Call opció esetén a mögöttes termék árfolyama magasabb, mint a kötési árfolyam, Put opció esetén alacsonyabb a kötési árfolyamnál.

- **ATM:** Az **At the money (ATM)** opciónak a kötési árfolyam megegyezik a mögöttes termék árfolyamával.
- **OTM:** Az **Out of the money (OTM)** opciónak nincs belső értéke, vagyis Call opció esetén a kötési árfolyam magasabb a mögöttes termék árfolyamánál, Put opció esetén a kötési árfolyam alacsonyabb a termék áránál.
- **Strike price:** **Call** opciónál lehívási ár, tehát, ha ilyen fajta opcióm van, akkor a lejáratáig annyi darab opciónak megfelelő részvényt vehetek, amennyi opció a birtokomban van, **Put** opciónál eladási ár, ennyiért adhatom el a határidőig.

2.2 A volatilitás fogalma

A volatilitás [3] meghatározása nem egységes az értékpapírpiacon. Inkább a hatását kell körüljárnunk, a működés közbeni viselkedését kell megértenünk, hogy vele számolni tudjunk. A legegyszerűbb megfogalmazása az, hogy a volatilitás minden olyan árbefolyásoló hatás, ami nem az első öt tényező. Kicsit elkerülő megfogalmazásnak tűnik, de rá fogunk jönni, hogy mégis a legpontosabb. A volatilitást másképp úgy is hívhatjuk, hogy piaci kockázati felár. A szó maga úgy fordítható magyarra: ingadékonyság. A piac tehát ingadozó mértékben árazza az alaptermék ármozgásában rejlő kockázatot. Néhány példával: egy 100 egységnyi árfolyamú áramszolgáltató cég két nappal a lejárat határidő előtt álló, 100 egységnyi árra vonatkoztatott vételi opciójának ára 0,50 egység, ám egy 100 egységnyi árfolyamú rákgyógyszer-cégnek ugyanúgy a lejárat előtt két nappal a 100 egységre vonatkozó vételi opciója 8 egység is lehet – ha annak a cégnek a gyógyszerét éppen a lejárat előttig még jó eséllyel piacképesnek fogadja el a hatóság. Az első példabeli 0,50-es opciós díjra azt mondjuk, alacsony a volatilitása; a másodikra, a 8 egységnyire pedig azt, kiemelkedően magas. Egy elfogadott rákgyógyszer nagyon megdobhatja az alaptermék, a gyógyszercég részvényének értékét, ettől olyan drágák az opciók: nagy a nyerési esély. Egy áramszolgáltató cég nem ugrik két nap alatt az égig, kevesebbet ingadozik, tehát alacsonyabb a volatilitása.

2.3 A volatilitás meghatározó tényezői

Az opciós modellezés legbensőbb működési elvéig jutottunk: jól kell meghatároznunk a volatilitás alakulását ahhoz, hogy pontosan tudjuk árazni az opciókat. A volatilitás értékét befolyásolja:

- A lejáratig hátralévő idő.
- Az opció lehívási árának az alaptermék pillanatnyi árfolyamától mért távolsága.
- Vételi vagy eladási opcióról van-e szó?
- Fizet-e és mikor az alaptermék valamilyen összeget az opció lejáratáig?
- Van-e valamilyen előre láthatóan nagy árfolyamváltóztató képességgel bíró esemény a lejáratig? Ilyen pl. vállalati részvényeknél a negyedéves gazdálkodási jelentések közzététele.

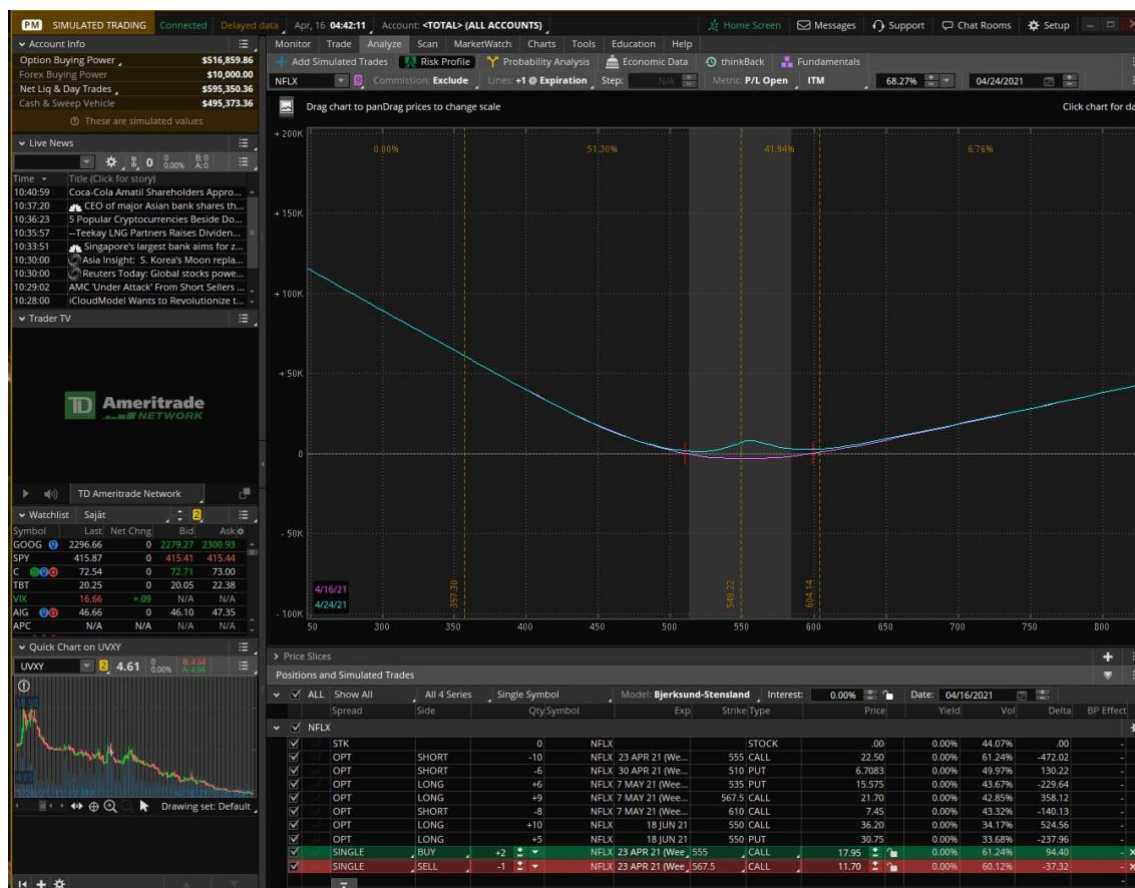
2.4 A létező modellezők, funkcióik és hibáik

Ebben a fejezetben az irodalomkutatás során talált hasonló szoftvert szeretném bemutatni. Ebben a kategóriában a szoftverek nem nyílt forráskódúak, nem részletezik a kapott eredményt a felhasználónak, ez a zártság a biztonsági rések kiküszöbölésének és a tudatlan felhasználók megtévesztésének is betudható, így a bemutatott rendszert a viselkedése alapján tudom jellemezni.

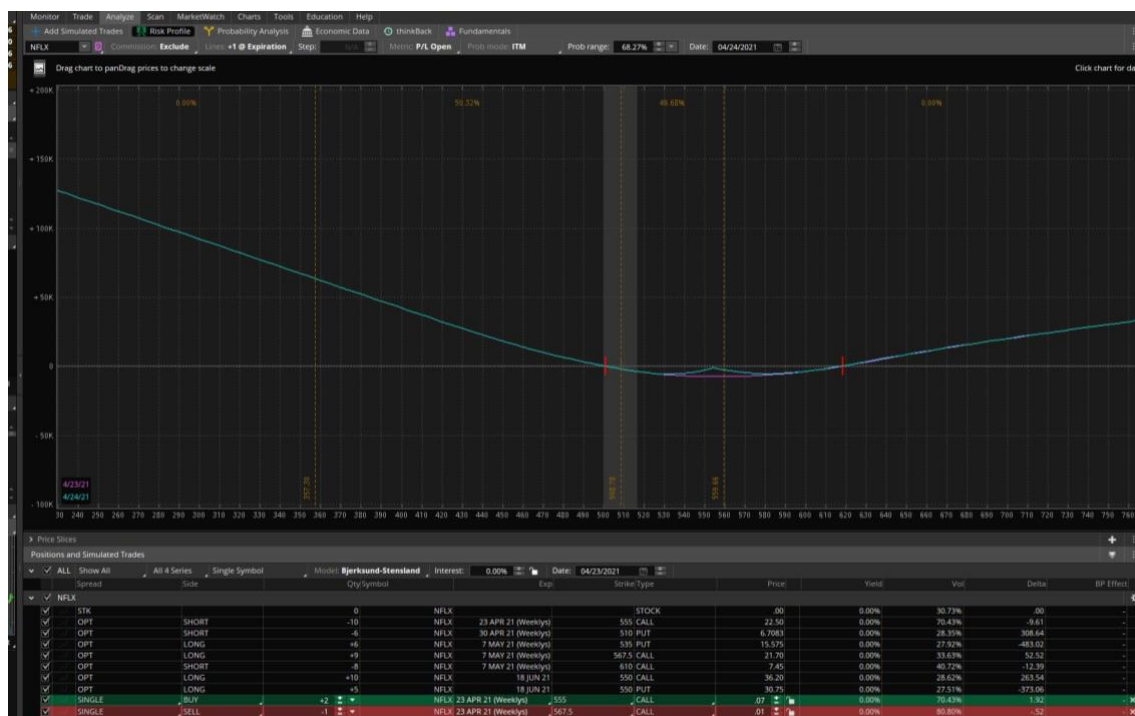
2.4.1 Thinkorswim

Ez egy USA-beli online brókercég, amelyet szakmabeli alapítói kifejezetten az opciós kereskedés támogatására hoztak létre 1999-ben. A szoftverét sok ügyfél a piacon elérhető legjobb követő-értékelő rendszernek tartja, elérhető vastag és vékony kliensen egyaránt. A szoftver hibája a nyereség-veszteség görbe számításában található, mely a következő példában van szemléltetve [4].

A 2. ábrán az „Analyze” funkció „Risk profile” alpontjának ábráját látjuk egy összeválogatott opciós kötés csomag időbeli alakulásáról. A mályvaszínű görbe a nyitáskori aznapi nyereség/veszteség görbe (továbbiakban: Nyereség/Veszteség görbe), a világoskék egy hat nappal későbbi időpontra előjelzett Nyereség/Veszteség görbe. A rendszer maga vette észre a legrövidebb lejáratot, és erre számolja a görbét: ez helyes, mert egy kötés csomag viselkedése teljesen átalakul, ha elemei közül néhány lejár, azaz megszűnik létezni. A kirajzolt görbe alapján azt látjuk, hogy ezt a kötés csomagot birtokolva a kötés utáni 8. napon az alaptermék bármely árfolyamértékénél nyereségesek leszünk, azaz a pozíciónak semmilyen kockázata nincsen. A súlyos hibát azonban a 3. ábrán látjuk.



2. ábra: Thinkorswim Analyzer hiba 1 (Forrás: Thinkorswim desktop).



3. ábra: Thinkorswim Analyzer hiba 2 (Forrás: Thinkorswim desktop).

A 3. ábra 7 nappal később készült, valóságos adatokkal, és ezen azt látjuk, hogy az árfolyamegyenes 500-tól 620-ig terjedő szakaszán különböző mértékben, de végig veszteséges lett a kötőcsomag. Egy ügyfél számára, aki komolyan veszi ezt a sok munkával előállított, funkciókban gazdag programot, ez egy sok ezer dolláros csalódás is lehet, anélkül, hogy bárki megtérítené a kárát, vagy akár csak elmagyarázná, miért is járt így. Mi tudhatjuk a megoldást: az első görbe a rajzolás időpontjabeli volatilitásértékekkel számolta végig az árfolyamegyenest, márpedig ezek az értékek egy ismert, előre látható esemény előre, statisztikailag kimutatható hatásait figyelembe véve jóval alacsonyabbak lettek.

Összegezve, a piacon lévő legjobbra értékelt asztali opcióskerekedéssel foglalkozó alkalmazás. Számos hasznos funkcióval rendelkezik, átlátható, letisztult felhasználói felület jellemzi. Valós időben tudjuk ábrázolni az opció csomagunkat, gyors, a modellező funkciója a kínálatában lévő összes opcióra működik. Hátránya viszont a volatilitás számítása, mely az óvatlan felhasználót nagy mértékben károsíthatja.

2.5 Volatilitás-jóslás mély neurális hálózat segítségével

A jövőbeli volatilitás megközelítésének egy módjára a mély gépi tanulás (angolul „Deep Learning”) technológiát vizsgáltam meg. A Deep Learning egy gépi tanulási módszer, amely a mesterséges neurális hálózatokon alapul. Mélytanulási architektúrákból több különfélét is megkülönböztetünk, ilyen például: konvolúciós neurális hálózatok, mély neurális hálózatok, rekurrens neurális hálózatok. Ezeknek a hálózatoknak egyre több szerep jut napjainkban, használják őket: számítógépes látásra, időjárás előrejelzésre, szövegfeldolgozásra, szövegalkotásra, anyagfelismerésre, hangfelismerésre, képfelismerésre, orvostudományban és még számos egyéb szakterületen [5].

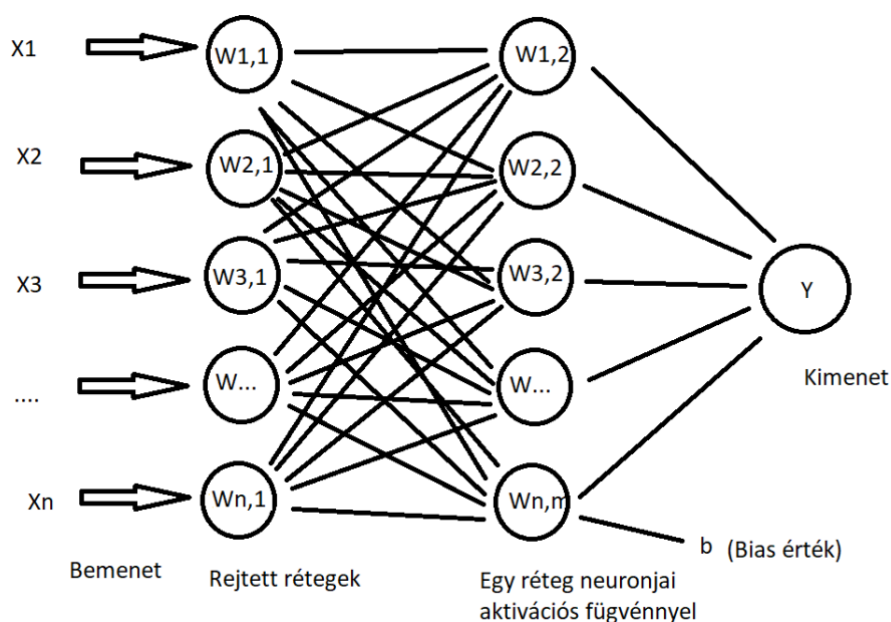
2.5.1 Deep Learning

A Deep Learning a gépi tanulási algoritmusok egy osztálya, mely a bemenő adatokból akár több száz réteg segítségével határoz meg kimenetet [6]. A bemenő adatot azonosítja alsó rétegei segítségével (például képfeldolgozásnál az élek detektálása), majd ezen részletekből határoz meg egy összefüggést, mely alapján eredményt tud prediktálni. A rétegek előrehaladtával határozza meg az emberi szemmel is felismerhető részleteket, (ha egy képről

van szó, akkor az elején csak az éleket detektálja), a későbbi rétegekben már kisebb objektumok is megjelennek.

A „mély” jelző a hálózatban a rétegek számára utal.

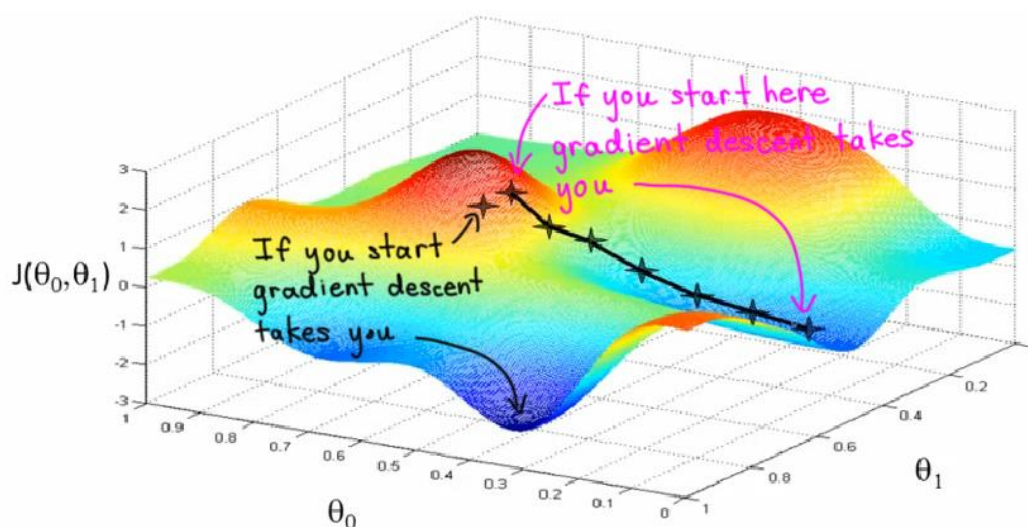
2.5.2 A hálózat felépítése



4. ábra: A neurális hálózat felépítése.

A 4. ábrán egy 2 rejtett réteggel rendelkező x_n bemenetű, 1 kimenetű teljesen összekötött neurális hálózat látható. Az „ X ” jelöli a bemenő adatokat, a volatilitás jóslásnál ezek az adatok a strike ára, részvény ára, opció típusa, lejáratáig hátra lévő napok száma stb. lesz. Az „ Y ” a kimeneti jóslt érték (volatilitás), a „ w ” lesz a súly, amit Gradient Descent eljárással határoz meg tanulás során a hálózat.

2.5.3 Gradient Descent eljárás

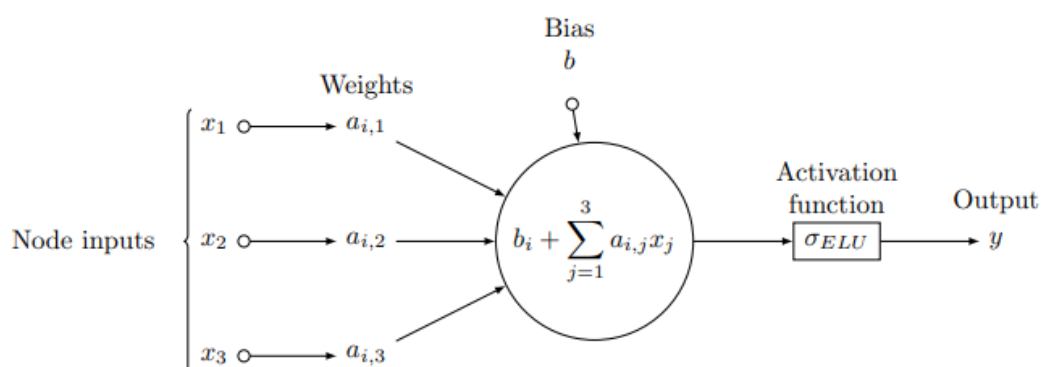


5. ábra: Gradient Descent eljárás.

Az 5. ábrán a Gradient Descent (magyarra fordítva: legmeredekebb süllyedés) 3 dimenzióbeli példája látható [7]. Ez egy optimalizáló iteratív függvény, mely egy differenciálható függvény lokális minimumát képes meghatározni. A függvény úgy működik, hogy minden ciklusban megnézi egy előre meghatározott távolságra az elemeket, majd ahol ez az érték a legkisebb arra lép egyet. J a célfüggvényt reprezentálja az 5. ábrán.

Ez egy működőképes eljárás, viszont vannak hátrányai is, nem garantálja a globális minimum megtalálását, rosszul választott távolság esetén képes „beragadni” egy pontba, ahol egy r sugarú környezetben nem talál a jelenleginél kisebb értéket. Ezt az 5. ábrán is láthatjuk: ebben a kis metszetben a sötét színek jelölik a kis értékeket, ha az eljárás kezdeti pontjának a rózsaszín színnel jelölt „start” gradienst választjuk, akkor más irányba mozdul el és ragad be, mint a fekete nyíllal jelölt pont. Ez természetesen a sugár megválasztásával módosítható.

2.5.4 Neuron felépítése



6. ábra: Neuron felépítése.

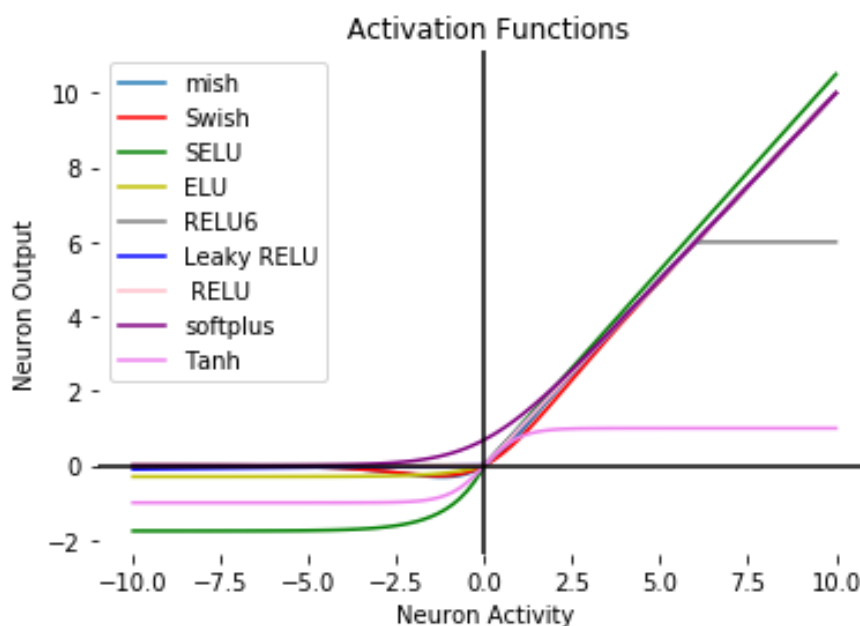
Egy neuron felépítése a 6. ábrán látható. A hálózat által meghatározott súlyokkal a bemenő adatot a neuron összeadja, egy előre meghatározott aktivációs függvény aktiválja, majd a következő réteg neuronjának a bemenő értékére kerül az alábbi neuron kimenő „Y” értéke. A 6. ábrán az x paraméterek a bemenő adatokat, az a változó az indexhelyesen megfeleltethető x bementi adat súlyát jelenti. Az adott csomópont ezeket összegzi, jelen esetben 1-től 3-ig summázza és ezen felül hozzáad egy b bias értéket, majd egy ELU aktivációs függvénnyel normalizálja a kapott értéket, amely az y kimenet lesz.

A bemenő adatok súlyait a 2.5.3 pontban leírt Gradient Descent függvénnyel határozom meg.

2.5.5 Aktivációs függvény

Az aktivációs függvényeknek fontos szerepe van egy mély neurális hálózat felépítésénél, mert ez kiegészíti a hálózat tudását úgy, hogy az adott hálózat összeadás és súlyozás műveleteken kívül magasabb dimenziójú feladatokkal is működjön. Az aktivációs függvény segítségével nem lineáris problémákat is képes lesz megoldani a hálózat: normalizál; ha egy nagyobb érték kerül a hálózatba, akkor azt is -1 és 1 közé normalizálja. A normalizálás minden esetben valós számból valós számot képez. Ez a kritérium elengedhetetlen, mivel így a függvény differenciálható marad.

Aktivációs függvényből több fajta is létezik, ahogy az a 7. ábrán is látszik. Először is nézzük meg az egyik legismertebbet a ReLU-t (Rechilied Linear Unit), mely értéke az alábbi módon adható meg: $f(x) = 0$, ha $x \leq 0$, $f(x) = x$, ha $x > 0$ és $f'(x) = 0$, ha $x \leq 0$, $f'(x) = 1$, ha $x > 0$.



7. ábra: Aktivációs függvények.

Ez az aktivációs függvény számos előnnyel jár, ilyen például a számítási igény minimalizálása, egyszerű differenciálhatóság, monoton növekvő, gyors konvergencia jellemzi és nem jelentkezik olyan probléma, hogy csak kis tartományban változik az értéke. Hátrányai is vannak: ez a függvény nem normalizál, csak rejtett rétegekben használható, nem lineáris, így van töréspontja.

Azonban ezen hátrányok felül is létezik még egy súlyos hibája a ReLU aktivációs függvénynek, melyet más néven dead ReLU-nak is szoktak nevezni, ez a probléma azt jelenti, hogy a neuronok nem fognak aktiválódni és a 0 értékhez tartozó lokális optimumba ragadnak.

Ennek a problémának feloldására két megoldás létezik: az egyik a tanítási ráta csökkentése, a másik módszer a leaky ReLU aktivációs függvény használata.

A leaky ReLU-ről néhány szó: a ReLU aktivációs függvény összes előnyével rendelkezik és e felett a zéró tartományban is ad vissza 0-tól különböző értéket.

A leaky ReLU megadása: $f(x) = 0,001x$, ha $x \leq 0$, $f(x) = x$, ha $x > 0$ és $f'(x) = 0,001$, ha $x \leq 0$, $f'(x) = 1$, ha $x > 0$.

Nézzünk meg egy harmadik aktivációs függvényt is, ez lesz az ún. „softmax”. Ez egy olyan aktivációs függvény, amelyet általában többosztályú klasszifikációnál használják kimeneti aktivációs függvénynek. Bemenetként egy vektort vár el, kimenetként a vektorának

az elemei 0 és 1 közötti elemek lesznek és ennek a kimeneti vektornak a summája 1. Így a kimenetet valószínűség-eloszlásként értelmezhetjük, amely azt mutatja meg, hogy az adott elem milyen valószínűséggel tartozik bele az egyes kategóriákba. A maximum érték indexe a legmagasabb valószínűségű kategória lesz:

$$\text{Softmax}(Z_i) = \frac{\exp(Z_i)}{\sum \exp(Z_i)} \quad (2.5.5)$$

Összegzésként a softmax függvény az általam tervezett feladatra nem megfelelő, mivel én nem a volatilitás kategóriáját szeretném megadni, hanem a pontos értékét szeretném megjósolni. Ezzel szemben a leaky ReLU jól illeszkedik az általam tervezett feladatra.

2.5.6 A megfelelő tanítás és tanító minta

A mély neurális hálózatoknál az egyik legfontosabb kérdés a megfelelő tanítási minta megtalálása és megfelelő formátumba rendezése, hogy a hálózat ez alapján megtanulja a benne rejlő összefüggéseket.

A minta kiválasztásánál törekednünk kell a tisztaságra. A tisztaság az adatoknak a helyessége a lehető legminimálisabb zajjal. Tegyük fel, hogy képfelismerésre tanítom a hálózatot, viszont a bemeneti képek rosszul vannak címkézve, akkor a hálózat téves összefüggéseket fog megtanulni, így a predikció eredménye nagyobb valószínűséggel lesz sikertelen. A tanító mintában a zaj a különleges eseteknek felel meg, ez ellen a megfelelő védekezés az adat tisztítása.

A tanító minta kiegyensúlyozottságára is ügyelnünk kell, mert a hálózat minden összefüggést megtanul. Ezt is egy példával szeretném szemléltetni, amely szintén képfeldolgozásra épül: a hálózatot kutya és macska felismerésére tanítom be. A tanítási képeim 100 darabból állnak, melyből csak kettő macska van, a többi mind kutya. Itt fennáll az a veszély, hogy a hálózat nem a kisebb összefüggéseket tanulja meg, mint például az orr, fül, mancs sajátossága, hanem az adatok kategóriájának méretét, így, ha a hálózat minden képre azt mondja a tanítás során, hogy kutya, akkor 98%-os pontossággal működik, mely a teszt adatokon (abban az esetben, ha itt már beszélhetünk kiegyensúlyozottságról) csak 50%-os pontosság lesz.

A tanító minta lehető legnagyobb számosságára kell törekedni, ez a mély neurális hálózatok egyik alappillére.

Végezetül a tanításról: a tanítás során is követhetünk el hibákat, ilyen hiba a túlillesztés (angolul overfitting) [8]. A túlillesztés akkor jelentkezhet, ha kevés tanítómintánk van vagy a mintához képest sok tanítható paraméterrel dolgozunk. Ilyen esetben a tanítási adatokkal megfelelő eredményt kapunk, viszont a teszt adatokkal használhatatlan lesz a végeredmény.

A túlillesztés jelentkezése a tanítás során rendeltetésszerű, ezért a megoldására alkalmas módszereket is tanulmányozni kell. Ilyen megoldások lehetnek a regularizációs technikák.

A *Dropout regularizációs* technika [9] lényege, hogy a tanítás során véletlenszerűen kiveszünk neuronokat a hálózathoz, ennek köszönhetően a megmaradt neuronok a kivettek pótlására törekszenek; ez a módszer segít a túltanítást elkerülni. A Dropout regularizációs technikát rétegenként is használhatjuk és érdemes együtt alkalmazni a súlycsökkentő technikával, a hatékonyság növelése érdekében.

Ezen felül más regularizációs technika is létezik, ilyenek például az L1 és L2 regularizációs technikák [10].

A másik megoldás a problémára a korai leállítás feltétel bevezetése (angolul Early Stopping), amely a tanítást abban az esetben leállítja, ha a hiba (angolul loss) egy megadott ideig nem változik, esetleg a túlillesztés jelenség elkezdődik [11].

3. Követelmény specifikáció

Egy olyan szoftvert tervezek, amely az embereket közelebb tudja hozni a tőzsde, azon belül is az opciók világa felé. A szoftver fő lényege, hogy a felhasználó saját igénye szerinti optimális opciókat tudjon venni és analizálni az adott portfólióját. A szoftver fő újdonsága az lesz, hogy jövőbeli volatilitás-jóslás alapján számol kockázatanalízist a már meglévő portfóliókra vagy megadott opcióra, könnyű és érthető használatot biztosít a felhasználónak. Több adatot is figyelembe vesz, mint az eddigi opció modellerek (pl.: a Nyereség/Veszteség görbe kirajzolásánál és az opció csomag összeállításánál a volatilitás jelenlegi állását és mozgását is kalkulálja), szűri a tévesen jegyzett adatokat, ennek köszönhetően a témában kevésbé járatos felhasználó is tud biztonságosan opciókkal kereskedni. A szoftver nem képes árfolyam „jóslásra”: nem fogja tudni megmondani, hogy holnap mennyi lesz egy részvény értéke, de képes lesz valós időben számolni, így megadja lehetséges jövőbeli részvényárra vonatkozóan a portfólióink pontos értékét

Így egy olyan alkalmazást valósítok meg, amely megadott részvényopcióra, kontraktusra (1 kontraktus egyenlő 100 darab opcióval), opció vételi árra és megadott jövőbeli dátumra kiszámolja a portfólióink lehetséges értékeit.

A szoftver 2 külső adatforrással rendelkezik: első a historikus, ez tartalmazza jelenleg egy évre visszamenőleg a VXX részvény összes opcióját óránkénti mintavételezéssel, a második pedig a saját portfólióink lesz, amelyet vissza tudunk olvasni és elemezni, a Nyereség/Veszteség görbéjének az alakulását. Ezt jelenleg manuálisan lehet megadni a szoftvernek a tesztelhetőség érdekében, a későbbiekben API-al valósítom meg.

3.1 A szoftver használata, tulajdonságai

Működés: A felhasználó először megadja a (8. ábra) (manuális opció megadás, a későbbiekben API esetében ilyenre nem lesz szükség):

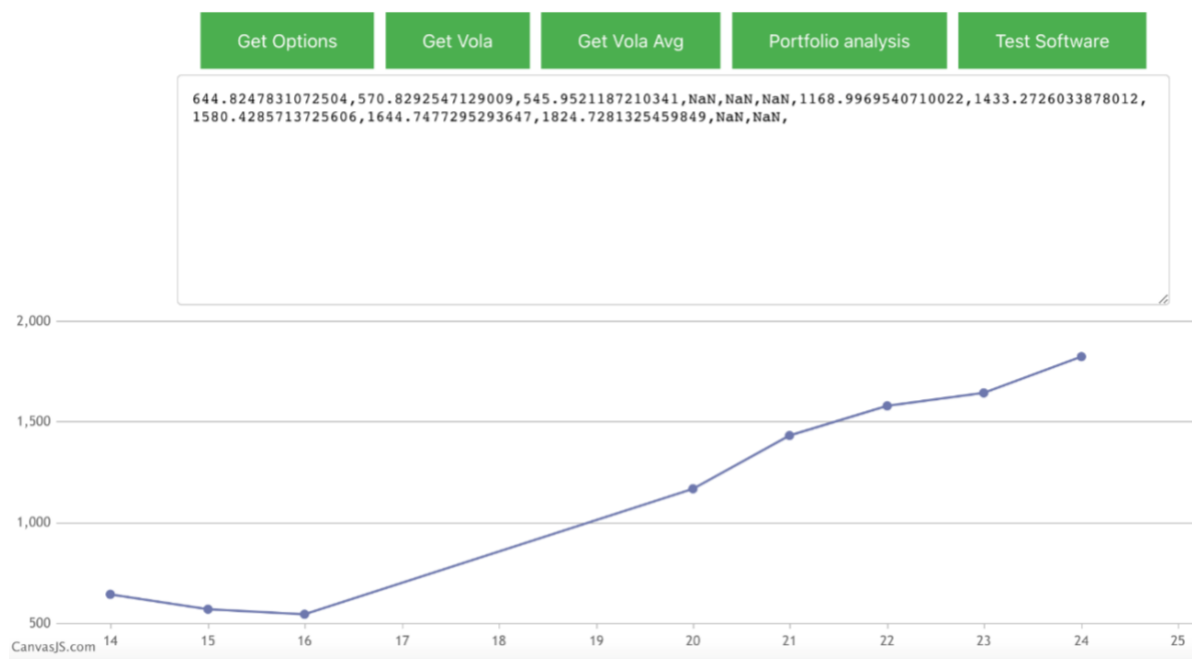
- strike árat (\$)
- opció típusát (C = 'Call', P = 'Put')
- az opció lejáratáig hátralévő napok számát
- moneyness (ITM, ATM, OTM), az opciónak van-e belső értéke
- részvény árát (elemzés pillanatában értve)

Strike:	18
Type:	C
Left Days:	4
Moneyness:	ITM
Underlying Price:	20
X Days later:	2
Trade Price:	3

8. ábra: Bemenő adatok.

- napot, darabszámban kifejezve a jövőbeli elemzés dátumát
- az opció vásárlási árát (\$)
- a kontraktot a teszt üzemben nem szükséges megadni, ezt az értéket a szoftver 1-re állítja be (így 100 darab opciós csomaggal számol),

ezután a szoftver az ezeken a paramétereken keresztül megkapott opciót analizálja adott nappal későbbi dátumra vonatkoztatva. Grafikusan megjeleníti a nyereség/veszteség görbét, amint ez a 9. ábrán is látszik.

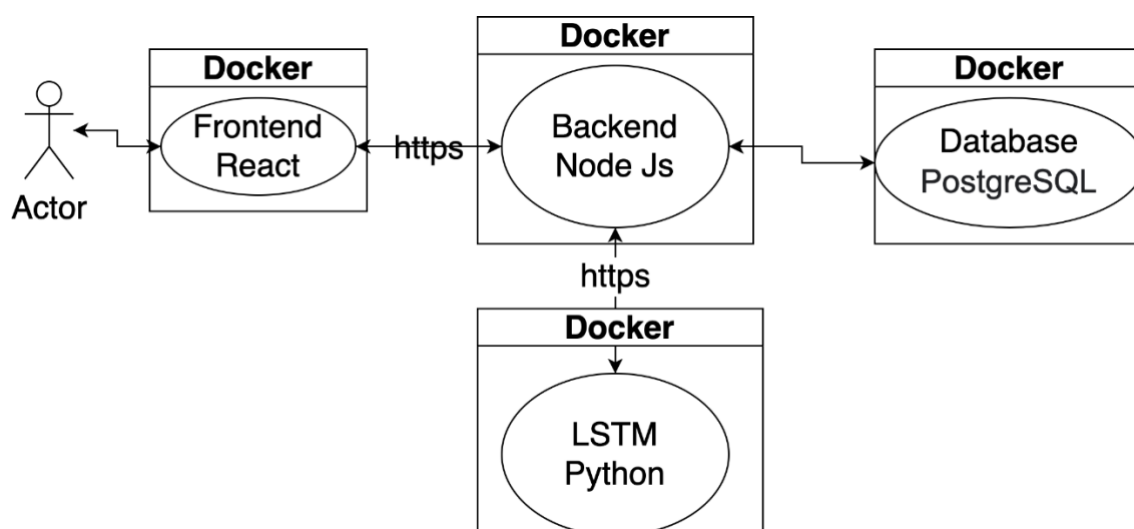


9. ábra: Nyereség/Veszteség görbe grafikus megjelenítése.

A szoftverben az újdonság az lesz, hogy egyedi stratégiával számol pontos volatilitást, amellyel majd megkeresi a kívánt opciót, elemzi saját portfóliónkat. A kezdők számára érthetőbbé teszi a tőzsdepiacra az opció fogalmát, és biztosítja, hogy egy adott határnál ne tudjanak többet veszíteni, az árfolyam-várakozás teljesülése esetén a számított profitot eredményezze a portfólió.

4. Fejlesztői specifikáció

A szoftverem microservice architektúrára épül, négy egységből áll a program, mely a 10. ábrán látható. Az egységeket dockerizáltan valósítottam meg a platformfüggetlenség érdekében [12]. Az első egység a Frontend, ami a megjelenítésért felel, ezen keresztül tud kommunikálni a felhasználó a szoftverrel, a második egység a backend, ahol az üzleti logika lett implementálva. A backend az adatbázissal és egy előre betanított LSTM felépítésű mély neurális hálózat modellel kommunikál, ebből a két egységből kapja a jövőbeli volatilitást, ezt felhasználva számítja ki a nyereség-veszteség alakulását a felhasználói igényeknek megfelelő opciócsomag összerakására [13]. Az LSTM (Long short-term memory) hálózatot histórikus adatokból tanítottam be jövőbeli volatilitás prediktálására, a PostgreSQL adatbázisban pedig ezek a histórikus adatok vannak feltöltve, melyekből megadott feltétel alapján határozza meg a szoftver a jövőbeli volatilitást [14].



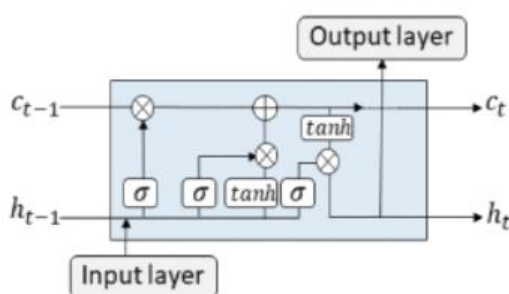
10. ábra: A rendszerterv.

4.1 Python alapú volatilitás-jósló egység

Ennek az egységnek a lefejlesztésére Python programozási nyelvet használok (a Keras könyvtár ez a programozási nyelv biztosítja) Tensorflow, Keras könyvtárak segítségével, mert a piacon a jelenlévő Neurális hálózat fejlesztői környezetek közül ez az egyik élenjáró megoldás. C++ alapokra épül, ennek köszönhetően nagyobb hatékonysággal tudom tanítani a modelletemet. Támogatja a GPU-n való tanítást. A Keras a lehető legtöbb segítséget adja a neurális hálózatok létrehozására a piacon, magas absztrakciós szinttel rendelkezik, ezért is

választottam ezt a környezetet. A volatilitás jóslását LSTM-mel (Long short-term memory) valósítom meg, mely a Rekurrens neurális hálózatnak (RNN) egy továbbfejlesztett variációja, mert az ilyen típusú hálózatok jellemzően idősoros adatok kiértékelésére szolgálnak. Különböző hálózatok vizsgálata és a bemenő adatok alapján az állapítható meg, hogy mivel az opció jegyzések időben egymást követik, összefüggő paraméterekkel rendelkeznek, akkor nem célszerű teljesen összekötött rétegeket alkalmazni [15]. Az egymásutániség figyelembevétele érdekében a visszacsatolással és memóriával rendelkező rekurrens neurális hálózat réteget választottam, mely felépítése a 11. ábrán látható.

A 11. ábrán látható LSTM-cella 2 bemenettel és 2 kimenettel rendelkezik. A C_{t-1} bemenet az előző cella státuszának bemenetét, a C_t pedig az aktuális cella státuszának kimenetét jelenti. H_{t-1} az előző cella kimenetét és H_t az aktuális cella kimenetét jelenti, σ a neurális hálózat rétegét jelöli.



11. ábra: Rekurrens Neurális Hálózat felépítése.

Az LSTM-ek fő komponense a cellastátusz, amely az egész láncon végigmegy, és csak kisebb lineáris változtatások történnek rajta. Az LSTM képes hozzáadni vagy elvenni információt a cellastátusztól, ezt pedig kapuk segítségével teszi. Az LSTM hálózatok felépítése az RNN neurális hálózattól a több kapus, komplex belső architektúrájában tér el. A kapuk az információk átengedését biztosítják egy előre definiált logika alapján. Ezek a kapuk egy szigmoid rétegből és egy pontonkénti szorzásból állnak. Az input layeren bemenő információ lényegi részét számolja az adott réteg, mondhatni egy kis számológépként viselkedik, ennél fogva nem csak adott bemenetet veszi figyelembe a kiértékelésnél a hálózat, hanem a már eddig elraktározott korábbi adatokat is.

A hálózat bemeneti adatai egy szövegfájlból kerülnek betöltésre, aminek a tartalma egy adott részvény (jelen esetben a VXX) egy éves histórikus opciói (rendezett formában és előre

tisztított opciók kerülnek bele, hogy ne vezesse félre az algoritmust) [16]. A VXX a tőzsdén részvénytulajdonosként kereskedhető termék, amelynek az árfolyamát rövidtávon nagyon nagy mértékben a piaci volatilitás mozgatja. Azért ezt a részvényt választottam, mert ezt kevésbé befolyásolják az események, így a volatilitását jobb eredménnyel tudom kiszámolni historikus adatokból.

A historikus adatok a VXX opcióinak 1 éves historikus adatai, napi 6 mintavétellel.

A file soronként opciókat tartalmaz, melyek volatilitására az oszlopokba rendezett paraméterei hatnak:

- Mintavételezés dátuma: ez az a dátum, amikor a pillanatkép készült az alábbi opciókról
- ImpVola: az adott opció mintavételezés kori volatilitása
- Underlying Price: a részvény mintavételezés kori árfolyama (Amerikai dollárban kifejezve)
- Call/Put: amint a 2.1 fejezetben ki lett fejtve, ez az opció típusa
- Strike: az adott opció lehívási ára
- Left Days: az adott opció mintavételezés kori pillanatában hátralévő napok száma (+1, mint a jelenlegi nap, ezt a növelést tőzsdei tapasztalatokkal rendelkezők ajánlása végett alkalmazom)
- ITM, ATM, OTM: a 2.1 leírtak alapján az opció kategóriája 0 ITM, 1 ATM és 2 OTM-nek felel meg
- 1 nap: a mintavételezés kori dátumhoz képest egy nappal ezelőtti (órára pontosan) ugyanannak az opciónak és a jelenleginek az Underlying Price hányadosa
- A 3, 5, 10, 15, 20, 25 nap hasonló, mint az 1 nap mutató, viszont itt 3 és 5 stb. nappal ezelőtti opciókról van szó. Ezek a mutatók azért fontosak, mert így a hálózat segítségével leszünk a nagy piaci hangulat-ingadozások detektálásában, mivel a volatilitás egy félelemindex, ennél fogva annak az opciónak magasabb lesz a volatilitása, ami előtt az árfolyam nagyobb mértékben változott, mint ahol ez a változás alacsonyabb. Fontos azt észrevenni, hogy ez a változás bekövetkezhet egy adott pillanatban, viszont az eredménye (magas volatilitás) akár hetekig fennállhat, így nem elég csak az előző napi opciók árhányadát figyelembe venni. (Például x napon a részvény ára a szokottól eltérően emelkedik, majd x-hez képest y nappal

később elemzünk egy opciót, akkor az y nappal későbbi opció volatilitása relatív magas lehet, viszont ezt a tegnapi ár nem igazolja).

	A	B	C	D	E	G	H	I	J	K	P
1	Date Time	ImpVola	Underlying	Call/Put	Strike	Left Days	ITM ATM	1 nap	3 nap	5 nap	
2	2019.10.08 11:00	0.7123	25.34	0	16.0	4	0	1.0513781	0.9613050	1.0497100248550124	
3	2019.10.08 11:00	0.7184	25.34	1	16.0	4	2	1.0513781	0.9613050	1.0497100248550124	
4	2019.10.08 11:00	0.7123	25.34	0	17.0	4	0	1.0513781	0.9613050	1.0497100248550124	
5	2019.10.08 11:00	0.7184	25.34	1	17.0	4	2	1.0513781	0.9613050	1.0497100248550124	
6	2019.10.08 11:00	0.7123	25.34	0	17.50	4	0	1.0513781	0.9613050	1.0497100248550124	
7	2019.10.08 11:00	0.7184	25.34	1	17.50	4	2	1.0513781	0.9613050	1.0497100248550124	
8	2019.10.08 11:00	0.7123	25.34	0	18.0	4	0	1.0513781	0.9613050	1.0497100248550124	
9	2019.10.08 11:00	0.7184	25.34	1	18.0	4	2	1.0513781	0.9613050	1.0497100248550124	
10	2019.10.08 11:00	0.7123	25.34	0	18.50	4	0	1.0513781	0.9613050	1.0497100248550124	
11	2019.10.08 11:00	0.7184	25.34	1	18.50	4	2	1.0513781	0.9613050	1.0497100248550124	
12	2019.10.08 11:00	0.7123	25.34	0	19.0	4	0	1.0513781	0.9613050	1.0497100248550124	
13	2019.10.08 11:00	0.7184	25.34	1	19.0	4	2	1.0513781	0.9613050	1.0497100248550124	
14	2019.10.08 11:00	0.7123	25.34	0	20.0	4	0	1.0513781	0.9613050	1.0497100248550124	
15	2019.10.08 11:00	0.7184	25.34	1	20.0	4	2	1.0513781	0.9613050	1.0497100248550124	
16	2019.10.08 11:00	0.7123	25.34	0	20.50	4	0	1.0513781	0.9613050	1.0497100248550124	
17	2019.10.08 11:00	0.7184	25.34	1	20.50	4	2	1.0513781	0.9613050	1.0497100248550124	
18	2019.10.08 11:00	0.7123	25.34	0	20.50	4	0	1.0513781	0.9613050	1.0497100248550124	
19	2019.10.08 11:00	0.7184	25.34	1	20.50	4	2	1.0513781	0.9613050	1.0497100248550124	
20	2019.10.08 11:00	0.7123	25.34	0	22.0	4	0	1.0513781	0.9613050	1.0497100248550124	
21	2019.10.08 11:00	0.7184	25.34	1	22.0	4	2	1.0513781	0.9613050	1.0497100248550124	
22	2019.10.08 11:00	0.7123	25.34	0	22.50	4	0	1.0513781	0.9613050	1.0497100248550124	
23	2019.10.08 11:00	0.7184	25.34	1	22.50	4	2	1.0513781	0.9613050	1.0497100248550124	
24	2019.10.08 11:00	0.7602	25.34	0	23.0	4	0	1.0513781	0.9613050	1.0497100248550124	

12. ábra: Histórikus VXX opciók.

4.1.1 Histórikus adatbázis árak

Discountoptiondata [17] adatbázis árak:

- 2019-2020-as összes részvény opciója, ára: 160 USD (6,4Gb) (2021.04.24.)
- 2020-as összes részvény opciója, ára: 125 USD (3,4Gb) (2021.04.24.)

Historicaloptiondata [18] adatbázis árak:

- 1 éves összes opció adat, viszont csak napi végi mintavétellel, ára: 245 USD

IVolatility [19] adatbázis árak:

- 1 éves VXX opció adat, napi 6 mintavétellel (~1Gb), ára: 150 USD (2022.01.25)

A *IVolatility* adatbázisát vásároltam meg, mert a napközbeni adatok opciókkal kereskedő szakemberek tapasztalatai alapján pontosabb eredményt adnak, mint a nyitó és záró adatok.

4.2 Opció feldolgozó

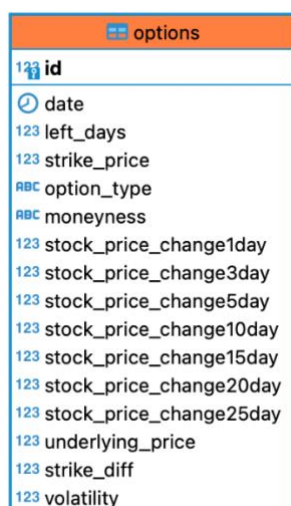
A VXX részvény histórikus adatait előre tisztítom az 5.1 leírtak alapján, erre a célra C# nyelven írtam a szoftvert. Azért ezt a nyelvet választottam, mert magas absztrakciós szinttel rendelkezik, így támogatja a fejlesztést. A modern nyelvek közül az egyik élen járó a piacon, a Microsoft számos úton támogatja a nyelv sikerességét, jól dokumentált, kiváló fordítóprogrammal rendelkezik, a Visual Studio számos funkcióval bír, könnyű a hibakeresés, függőség ellenőrzés, vastag és vékony kliens réteg is hozzákapcsolható az alkalmazáshoz. Windows platformon belül univerzálisnak számít, mivel egy virtuális réteget hoz létre és ezen fut a program, ez a réteg kezeli az erőforrásokat, így kevesebb a hardware általi kockázat felmerülésének az esélye.

Hátránya is természetesen van, a programozási nyelvekhez képest nem a leggyorsabb, a sebességben élen járó nyelvek egyike a C++, mivel itt nincs virtuális réteg, közvetlenül tudja kezelni az erőforrásokat.

Ebben az egységben dolgozom fel a megvásárolt opciós adatbázist, tisztítom, megfelelő formátumba rendezem szükségtelen oszlopok eltávolításával és újak hozzáadásával.

4.3 PostgreSQL Database

Az adatbázist Dockerizált környezetben hoztam létre a kompatibilitás, erőforrás kihasználás érdekében. A tisztított histórikus adatokat ide töltöm be a Backend segítségével a 13. ábrának megfelelően, majd az előre meghatározott statisztika alapján innen keresek hasonló opciókat az elemezni kívántakhoz. Azért ezt a felépítést használom, mert egy portfólió elemzésénél több lekérdezést is kell végeznem az opciók között, így elengedhetetlen a sebességre való törekedés a felhasználó érdekében. Ezen műveletek például szöveg file-ban tárolt adatok esetén akár percekig eltarthatnának. A postgresQL egy modern, megbízható, könnyen kezelhető relációsadatbázis-kezelő rendszer, mely a komplex lekérdezések miatt szükséges a szoftverembe.



13. ábra: DB Diagram.

4.4 Backend Node.js (Typescript)

A backend a szoftver üzleti logikáját tartalmazó egység, mely szintén egy docker konténerben fut. Ez az egység kommunikál az adatbázissal, betanított modellel és a frontenddel. Ebből az egységből hoztam létre a modellt és töltöttem fel adatokkal MikroORM segítségével [20]. A frontendről érkező https Rest API kérést fogadja, értelmezi és megfelelő választ küld a felhasználónak. Adott opcióra kéréseket indít az adatbázis felé, hasonló tulajdonságokkal rendelkező opciókat keres, majd ezek volatilitásának az átlagát felelteti meg jövőbeli volatilitásnak és ezt felhasználva küldi vissza a megfelelő profit/loss értékeket a frontendre. A Python service egység is szintén volatilitást jósol, viszont ott a modell prediktálja a volatilitást. Azért ezt a szoftverrendszert választottam, mivel jól dokumentált, stabil használatot biztosít és egyszerű a váltás a frontend-backend fejlesztés, továbbfejlesztés között a JavaScript ökoszisztémának köszönhetően.

4.5 Frontend React (Typescript)

Ez az egység szorosan kapcsolatban áll a felhasználóval, a megjelenítésért, kommunikációért felel. Rest API-al kommunikál a backenddel, a válaszban kapott profit/loss adatokat grafikusán jeleníti meg. Azért ezt a JavaScript könyvtárat választottam, mert modern, jól dokumentált, folyamatosan fejlesztett a Meta által. Átlátható, jó kinézetű webappok fejlesztését teszi lehetővé. A függvényt a canvasJS segítségével jelenítem meg grafikusán.

5. A szoftver megvalósítása

Az alábbi fejezetekben a szoftver implementálásának lépéseit írom le. A probléma elméleti síkon való felvázolása és az irodalomkutatás során szerzett tudás alapján valósítottam meg a szoftveremet.

5.1 Histórikus adatbázist feldolgozó egység

A 4.2 pontban leírt opció-feldolgozó egységgel a megvásárolt adatot feldolgoztam. Ezt az eljárást csak új adatok beszerzése esetén szükséges újra végrehajtani. Az adatbázist az 5.2 pontban leírtak alapján tisztítottam, ezen felül a nyitó és záró adatokat töröltem belőle, majd 12. ábrán látható formába rendeztem, kiszámoltam a lejárat napok számát, meghatároztam minden egyes opcióra, hogy melyik kategóriába esik (ITM, ATM, OTM), minden opcióra, a tőle 1, 3, 5, 10, 15, 20 és 25 nappal korábbi (órára pontosan) opciónak mennyi a részvényár (Underlying Price) hányada. Az adatbázis első opcióinak nem lenne ilyen értéke (mivel az első adat előtt nincs opció), ennél fogva az első 25 napnyi opciót töröltem.

5.2 Az adattisztítás folyamatáról

Az árjegyzők néha hamis adatokat is visznek fel a rendszerbe, amelyek hibás statisztikát eredményezhetnek, ezeket az adatsorokat kiszűrtem és eltávolítottam az adatbázisból. Példaként szemléltetem a 14. ábrán a BIDU részvény néhány hibásan jegyzett opcióját.

Részvény neve: BIDU
 Ára: 89.68 USD
 Opció típusa: Call

Volatilitás				Ask	Lejárat	Strike
	96.54%	77.90%	.28	0	4.90	27 MAR 20 101
	100.55%	78.80%	.27	0	4.90	27 MAR 20 102
	104.46%	79.61%	.26	0	4.90	27 MAR 20 103
	107.79%	80.45%	.25	0	4.85	27 MAR 20 104
	77.15%	88.50%	.14	0	1.75	27 MAR 20 105
	114.14%	81.94%	.24	0	4.75	27 MAR 20 106
	119.17%	82.31%	.24	0	4.90	27 MAR 20 107

14. ábra: Hibásan jegyzett opciók (Forrás: Thinkorswim desktop, 2020.03.16.).

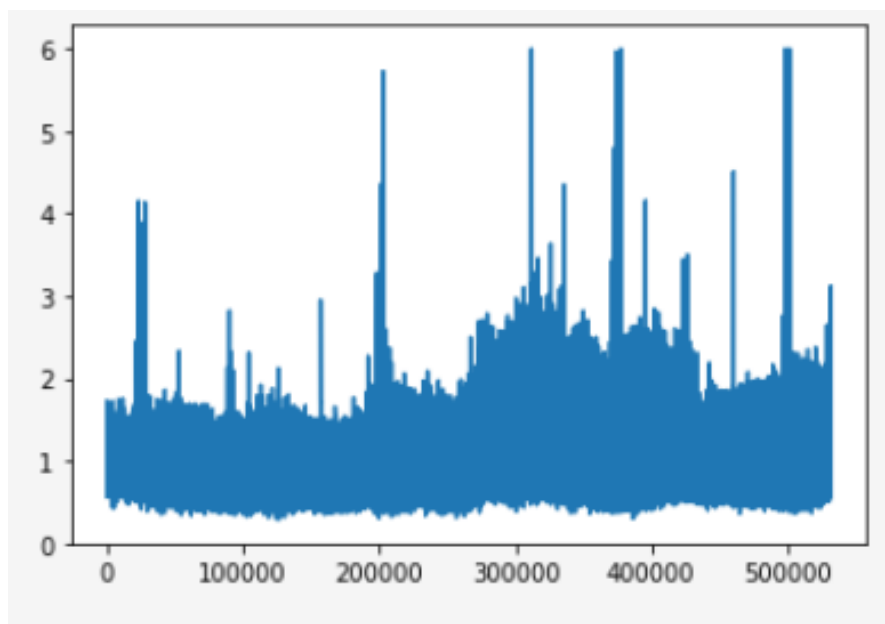
A kijelölt opciók ezt jól szemléltetik, nem éri meg ugyanannyiért vagy drágábban Call opciót venni egy részvényre, ha egy másik opciónak kisebb a lehívási ára. Ebben az esetben a pirossal bekeretezett opciók kerülnek kiszűrésre. Ezen felül arbitrázs lehetőségeket keresek, historikus adatokban előforduló jelenség, mely kereskedési időben átlag felhasználó számára nem létezik, ennek következtében ezeket az adatokat is hibásnak tekintem és törlöm.

5.3 Mély neurális hálózat megvalósítása volatilitás jóslására

A 4.1 fejezetben leírt módszerek alkalmazásával létrehoztam egy rekurrens visszacsatolással és memóriával rendelkező mély neurális hálózatot (LSTM). A pandas csomag segítségével hajtottam végre adatmanipulációs lépéseket (file beolvasás, oszlopok törlése, file tartalmának megjelenítése). Az adathalmaz 14 különböző paraméter (részvény ár, strike, call/put, lejáratig hátra lévő napok száma stb.) óránkénti mintavételezéseit tartalmazza 2020-ban, összesen 531711 rekordot (a megvásárolt adathalmazban 5,5 millió rekord volt, viszont az adat tisztítással ez a szám csökkent).

A matplotlib könyvtár segítségével szemléltetem a volatilitás alakulását az egész adathalmazon.

A 15. ábrán jól látható, miért van szükség az előző napok részvényár hányadára, mivel a volatilitás a hirtelen árfolyam esés/növekedésre való reakcióját „tartja” néhány napig/hétig.



15. ábra: Volatilitás alakulása az egész adathalmazon.

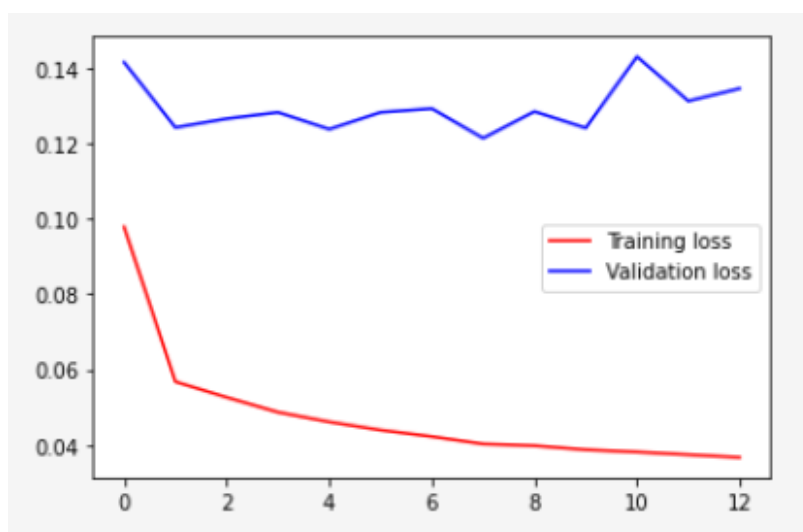
Ezen adatok alapján egy olyan modellt készítettem, amely az elmúlt napok megfigyelései alapján képes jövőbeli volatilitás becslésére.

Az adatokat 3 részhalmazzra bontottam. Az első volt a tanító, a második a validációs és végül a teszt adathalmaz. Megírtam egy generátor függvényt, mely segítségével feltöltöttem a 3 adathalmazt.

SimplaRNN Keras könyvtárat használtam, *Gradient descent* eljárással állítom be a hálózat súlyait (w). Alkalmazom a tanítás során a korai megállás feltételt: ha a loss érték adott iteráción át nem csökken, esetleg nő, akkor a túltanítás elkerülése miatt leállítom a tanítást.

Jupyter lab-ban dolgoztam, a tanítás a saját notebookomon történt, egy RTX 2070 videokártyán (Intel Core i7 10750H CPU, 16GB DDR4 RAM, Windows 11), mely jelentősen gyorsította a folyamatot [21].

Tapasztalat: A hálózat megfelelően működik, viszont a kapott eredmények jelen esetben még a statisztikai becsléshez képest pontatlanok, ezért az adatok további tisztítására, kiegyensúlyozására lesz szükségem a kellően pontos jövőbeli volatilitások végett, esetlegesen hálózat módosítást is alkalmazok. A 16. ábrán is látható, hogy míg a tanítási adathalmaz hibái szigorúan monoton csökkennek, a validációs adathalmazé ezzel szemben nem konvergálnak az optimum irányába.



16. ábra: LSTM hálózat kiértékelése: a hiba az iterációk függvényében.

5.4 Statisztika alapú volatilitás-becslés megvalósítása

A volatilitás előrejelzéséhez vegyesen kell alkalmazni a piac mozgásából egyéb következtetési módokon leszűrt, megalkotott várakozásokat és a volatilitásról készített statisztikákat. Maguk a statisztikák úgy készülnek, hogy feltárjuk az összes, volatilitást befolyásoló tényezőt és megkeressük a hozzájuk legjobban illő statisztikai modellt.

A megvalósítási nyelveket, környezetet a 4.2 pontban határoztam meg. A statisztika alapú volatilitás-jóslást úgy valósítottam meg, hogy a teljes histórikus adatbázisból keresek hasonló opciókat a kívánt jövőbeli opcióhoz, majd a kapott opciók volatilitásának veszem az átlagát.

A jövőbeli volatilitás meghatározása úgy történik, hogy a portfóliónk egy adott opciójának csökkentem a hátralévő napok számát annyival, amennyi nappal a jövőbeli volatilitásra vagyok kíváncsi - a részvény pontos árát természetesen nem tudom -, de mivel a kockázat-analízis lényege, hogy a lehető legtöbb lehetséges részvényárra számolja ki a portfóliónk értékét, így lépcsőzetesen kiszámolom egy opció lehető legtöbb részvényárra a hasonló opciójának a volatilitását.

A hasonló opció fogalmán pontosan azt értem, hogy a lejárat napok száma, az opció típusa, 1, 3 és 5 nappal ezelőtti részvény-árváltozás, strike és részvényár-arány stb. az általam meghatározott intervallumba esnek.

Részletesebben:

A lejáratig hátralévő időt napokban számítjuk, és a következő statisztikai időszakokat képezzük:

- 1 vagy 2 nap,
- 3-5 nap
- 6-9 nap
- 10-12 nap stb. (lejáratig hátravan)

A vételi vagy eladási opció megkülönböztetése egyértelmű.

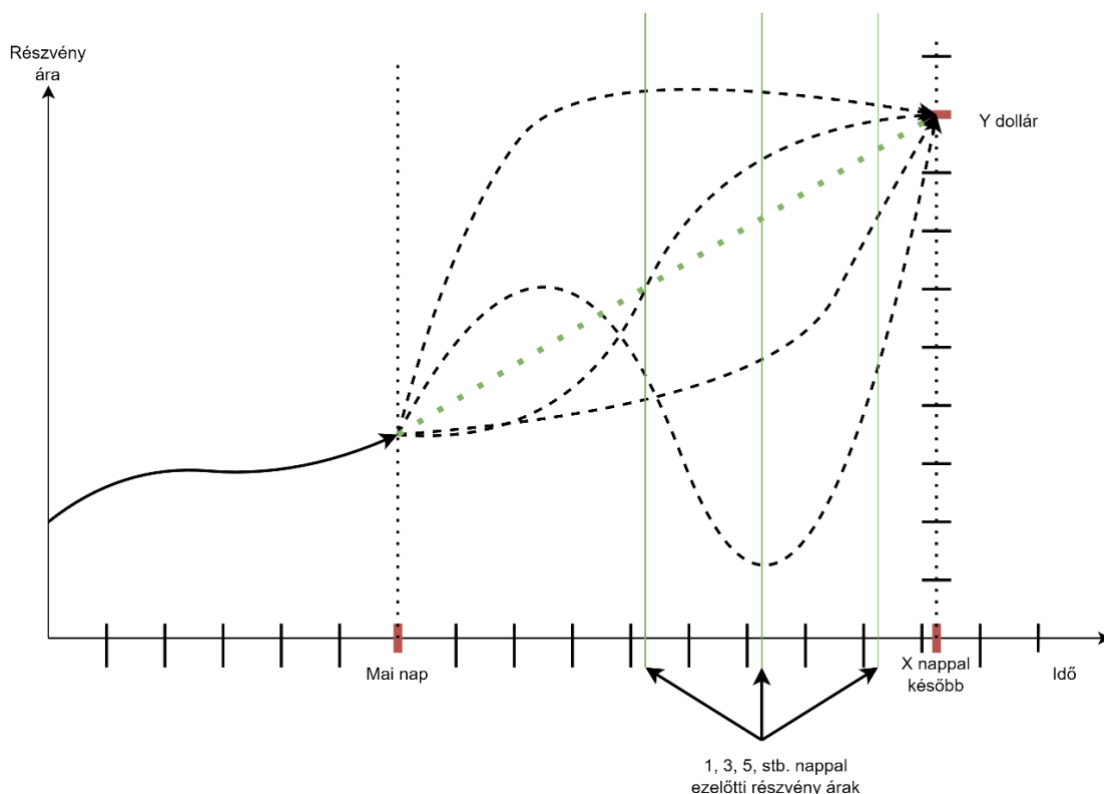
A lejáratig való kifizetés időpontját és összegét nyilvános adatbázisból be kell vinni a programba. Ilyen kifizetések az osztalékok, kamatok. Ezeket több weblapon jól tagolva nyilvánosságra hozzák.

Az előre látható események legfontosabbika a „jelentés”, azaz egy cég negyedéves nyereség-gyorsjelentése. Ennek időpontját ugyancsak nyilvános adatbázisból meg lehet tudni,

be lehet vinni a program adatbázisába. Mivel jelentős különbség van egy jelentősen túlfutó és egy, az előtte kifutó lejáratbeli árazások között, ezért ezen események időpontja mentén „NO” és „PO” kategóriákba soroljuk az összes opciót: NO-ba azokat, amelyek határidejéig nem lesz jelentés, PO-ba a többit. Jelen esetben a VXX-nél ilyen események nincsenek, mivel ez egy speciális termék, ahogy ezt a 4.1 pontban a bemenő adatoknál megfogalmaztam.

Ezen mutatók közül az 1, 3, 5 stb. nappal ezelőtti árhányadra térnék ki, mivel, ha nem egy nappal előre szeretnénk portfóliót elemezni, akkor ezen adatok is jövőbeliek lesznek.

Ahogy a 17. ábrán is látható, ha a mai napon szeretnénk egy jövőbeli volatilitást meghatározni x nappal későbbre, akkor az 1, 3, 5 stb. mutatók jövőbeli részvényárak hányadai lehetnek. Ezt a problémát a szoftver jelenleg úgy kezeli, hogy a jelenlegi részvényárat és az x nappal későbbi adott részvényárat egy lineáris egyenessel összeköti, majd arányosan határozza meg az ezelőtti árakat. Ez a módszer néhány hetes becslésre megfelelően működik, viszont ha több héttel/hónappal előre szükséges elemezni a portfóliót, akkor a két részvényár közötti alakulásra metrikákat fogok bevezetni, melyekkel minimum, átlag és maximum volatilitást-becslést lesz képes a szoftver prediktálni.



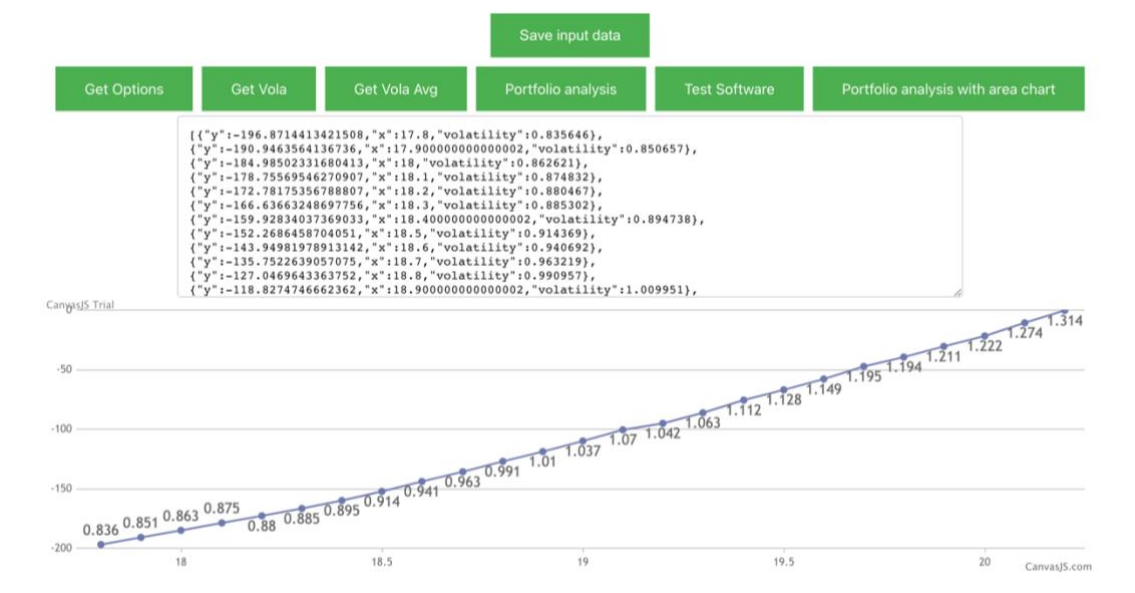
17. ábra: Opció jövőbeli volatilitásának meghatározása.

Az üzleti logikát a backendben valósítottam meg, a leírtak alapján kérdezek le adatokat az adatbázisból. Az adatbázist a backend építi fel. Létrehoztam egy entity-t, majd MikroORM migrációval megcsináltam az adattáblát. MikroORM Seederrel kötegenként beolvastam, majd feltöltöttem az adatbázisba az adatokat.

Jelenleg 5 API-t valósítok meg a backenden, az elsővel megadott paraméterekhez hasonló opciókat kérdezek le, ahogy ez a 18. ábrán is látható. Erre a lekérdezésre azért van szükség, mert így tudom ellenőrizni a szoftver helyességét, finomra hangolva az adatbázist, lekérdezési statisztikát.

Strike:	18	Stock Price 1 Day Before:	19.5	Stock Price 1 Day change (%):	0.15	Underlying Price	1
Type:	C	Stock Price 3 Day Before:	19	Stock Price 3 Day change (%):	0.15	X Days later	1
Left Days:	4	Stock Price 5 Day Before:	18.5	Stock Price 5 Day change (%):	0.15	Future Underlying Price:	1
Moneyness:	ITM	Stock Price 10 Day Before:	18	Stock Price 10 Day change (%):	0.15	Calc	
Underlying Price:	20	Stock Price 15 Day Before:	18	Stock Price 15 Day change (%):	0.15		
X Days later:	2	Stock Price 20 Day Before:	18	Stock Price 20 Day change (%):	0.15		
Trade Price:	2.5	Stock Price 25 Day Before:	18	Stock Price 25 Day change (%):	0.15		

18. ábra: Get Options API.



19. ábra: Portfólió elemzése.

A 2. API hasonlít az elsőhöz, azzal a különbséggel, hogy itt nem az egész opciót adja vissza a lekérdezés, hanem csak a volatilitásait. A 3. ezeknek veszi az átlagát. Az 5. API-val tudom elemezni a portfóliót, a 19. ábrán láthatók szerint.

Az ábrán látható az elemzett portfólió, ez az elemzés 2 nappal későbbi adatokra vonatkozik. A 18. ábrán látható input mezőbe meg kell adni a kívánt opció paramétereit, hány nappal későbbre szeretnénk az elemzést és a szoftverem jelenlegi verziójánál az 1, 3, 5 stb. nappal ezelőtti részvényár-hányad ismert adatait és ennek a hasonlóság eltéréseinek mértékét, mely opcionális, viszont így szemléletesebben látszik ezen mutatók fontossága. A 19. ábrán az x tengely a jövőbeli részvényár lehetséges változásait jelzi, az y pedig ezen árhoz mérten a portfólió alakulását. Az 5. API-val tudom tesztelni a szoftvert, egy előre szétválasztott validációs adatbázissal.

A tesztadatok egy file-ból kerülnek betöltésre és a felépítése megfelel a tanító mintának. Egy adott paramétereknek megfelelően határoz meg a szoftver volatilitást, majd a kapott értéket összeveti az elvárttal. A 20. ábrán látható a tesztadatokra kapott volatilitás pontossága (0 és 100% között), ennek az átlaga: 91.7312274301564% lett, mely pontosabb jelen esetben a neurális hálózat által prediktált volatilitásnál.



20. ábra: Szoftver volatilitás jóslás teszt.

6. Tesztelés

Ebben a fejezetben a rendszer működésének a tesztelése lesz kifejtve, összehasonlítva a piacot meghatározó Thinkorswim desktop alkalmazás kockázat-analizálás funkciójával, rendszer (System) és átvételi (Acceptance) teszteken keresztül.

A teszt 2022. 12. 11-én lett végrehajtva, a VXX221216C00014500 azonosítójú vételi opción, melyet a 21. ábrán lehet látni. Ez egy call típusú opció, 2022.12.16-i lejáratral rendelkezik, \$14,5 a lehívási ára, 75% az implikált volatilitás értéke, jelenleg az opció 1,1 USD-ért vásárolható meg, a VXX részvény ára \$15,30.

VXX							
☒	☒	STK		0	VXX		STOCK
☒	☒	SINGLE	BUY	+1	VXX	16 DEC 22	CALL

STOCK CALL							
.00	0.00%	76.62%	0.00%	.00	-		
1.10	0.00%	75.33%	0.00%	74.96	-	x	

21. ábra: VXX221216C00014500 opció (két részletben).

6.1 Hasonló opciók és volatilitásuk lekérdezése funkció

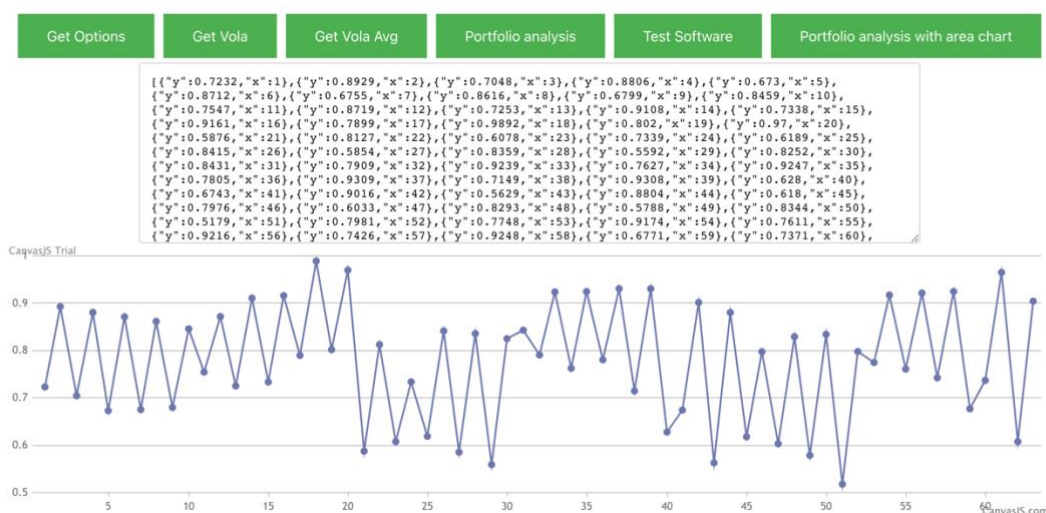
Get Options	Get Vola	Get Vola Avg	Portfolio analysis	Test Software	Portfolio analysis with area chart
<pre>{ "id": "176517", "volatility": 0.7232, "date": "2019-12-02T12:00:00.000Z", "strikePrice": 16.5, "leftDays": 4, "underlyingPrice": 17.23, "strikeDiff": 0.042367962855484644, "optionType": "C", "moneyness": "ATM", "stockPriceChange5Day": 1.0551132884262096, "stockPriceChange3Day": 1.0579314155895987, "stockPriceChange10Day": 0.9919401266551525, "stockPriceChange15Day": 0.9057075873379661, "stockPriceChange20Day": 0.899738903394256, "stockPriceChange25Day": 0.8437806072477962, "id": "176523", "volatility": 0.8929, "date": "2019-12-02T12:00:00.000Z", "strikePrice": 18, "leftDays": 4, "underlyingPrice": 17.23, "strikeDiff": 0.04468949506674402, "optionType": "C", "moneyness": "ATM", "stockPriceChange5Day": 1.0551132884262096, "stockPriceChange3Day": 1.0579314155895987, "stockPriceChange10Day": 0.9919401266551525, "stockPriceChange15Day": 0.9057075873379661, "stockPriceChange20Day": 0.899738903394256, "stockPriceChange25Day": 0.8437806072477962, "id": "178071", "volatility": 0.7048, "date": "2019-12-02T12:00:00.000Z", "strikePrice": 16.5, "leftDays": 4, "underlyingPrice": 17.36, "strikeDiff": 0.0495391705069 }</pre>					

22. ábra: Adott opcióhoz hasonló opciók lekérdezése, grafikus megjelenítése.

A 22. ábrán a VXX221216C00014500 azonosítójú opcióhoz keres a rendszer hasonlókat, majd grafikus megjeleníti a felhasználó számára. Ez a funkció nem része a végleges szoftvernek, viszont a rendszer gazdasági oldalának a tesztelését segíti elő. Ennek a funkciónak köszönhetően manuálisan is meg lehet keresni a hibásan hasonlónak értékelt opciókat, majd a hasonlóság fogalom újratervezésével pontosítani ezen.

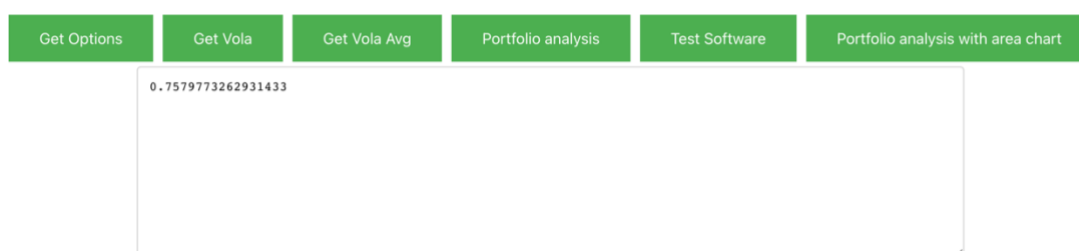
A 23. ábrán egy adott opcióhoz hasonló opciókat jeleníti meg a rendszer, mely szintén a rendszer tervezéséhez szükséges funkció. Viszont a 23. ábrán látható, hogy a backend által

visszaküldött hasonló opciók volatilitását helyesen jeleníti meg a frontend 2 dimenziós koordináta-rendszer segítségével, mely x tengely minden egyes pontja egy adott opciót jelöl, y pedig a hozzá tartozó implikált volatilitását. Ezáltal egy tapasztalt szakmabeli bróker nagyobb valószínűséggel jön rá a rendszer hibájára és annak megoldására.



23. ábra: Hasonló opciók implikált volatilitásának grafikus ábrázolása

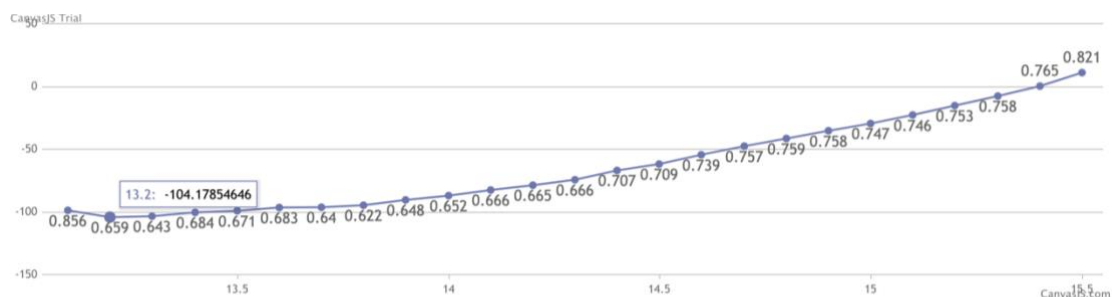
6.2 Hasonló opciók volatilitásainak átlaga funkció tesztelése



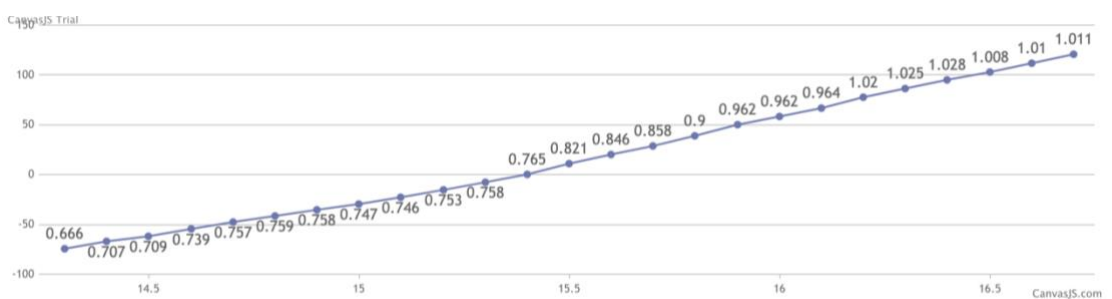
24. ábra: Volatilitás prediktálás eredménye.

A 24. ábrán az említett opciónak a rendszer által prediktált volatilitása látható, mely az elvárt 75%-os implikált volatilitás-eredménytől való eltérése az 1%-os hibához konvergál, amely elfogadható. Az alábbi tesztből a rendszer helyes működésére lehet következtetni, a megfelelően megadott paraméterek alapján a backend segítségével meghatározott volatilitás értékeket kapja meg a frontend és ezt sikeresen megjeleníti a felhasználónak.

6.3 Portfólió elemzésének tesztelése, Thinkorswimmel való összehasonlítása



25. ábra: VXX221216C00014500 opció portfólió analízise 1.



26. ábra: VXX221216C00014500 opció portfólió analízise 2.

A következő teszt a rendszer portfólió-elemzését vizsgálja. A 25. és 26. ábrán szintén a VXX221216C00014500 opció kockázatanalízise látható, a különbség a két ábra között a részvényár-lépték, a 27. ábrán a Thinkorswim által megoldás látható.

A Thinkorswim az implikált volatilitásból opcióár meghatározásra Bjerksund-Stensland modellt használ alapbeállításként, mely az ábrán is látható [22]. Ezen kívül képes Black scholes és Binomial modellel is számolni, melyek néhány hasonló eredményt adnak, jelen grafikonon a maximális eltérés néhány cent, ezzel szemben a rendszer több dolláros eltérést mutat [23].

A rendszer a prediktált implikált volatilitásból opcióárat Black scholes modell által számol.

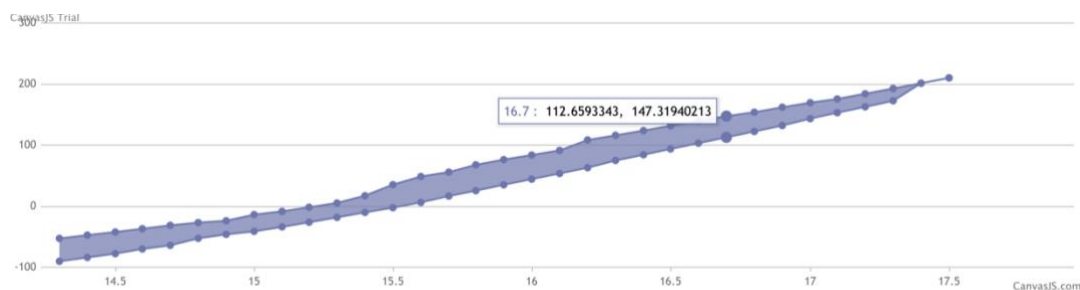


27. ábra: VXX221216C00014500 opció portfólió analízise Thinkorswim által.

Ahogy a 27. ábrán látható a Thinkorswim 16,7 USD részvényárra és 13,2 USD részvényárra, amely között több, mint 21% eltérés van kevesebb, mint 5%-os volatilitás eltérést határozott meg, mely részvényopciókkal kereskedő szakemberek véleménye, tapasztalata által nem elégséges. Ezzel szemben a rendszer által meghatározott eltérés valósághűbb ábrázolást ad a portfólióról, itt az eltérés több, mint 34%.

Amelyben megegyeznek az ábrák, hogy közel 15,4 USD részvényárnál 0%-os a nyereség-veszteség az adott opciónál, melyből arra lehet következtetni, hogy a portfólió elemzés funkció megfelelően működik, csak az implikált volatilitás meghatározásában tér el. Az implikált volatilitás meghatározásának helyessége a 7. fejezetben van részletesebben kifejtve.

6.4 Portfólió elemzése minimum, maximum volatilitás-becsléssel, ársáv használatával



28. ábra: Portfólió számítása minimum, maximum volatilitás prediktálása esetén.

Az implikált volatilitás-mutató szorosan kapcsolódik egy adott részvényár mozgásához, mivel a részvény árának alakulását nem hivatott a rendszer meghatározni, ezért nem mondható teljes bizonyossággal, hogy az adott portfólió egy megadott dátumon pontosan milyen értéket vesz fel. Ehelyett historikus adatok alapján meghatározható egy olyan ársáv, amelyben magas konfidencia intervallummal kijelenthető, hogy a portfólió jövőbeli értéke része lesz a halmaznak. A rendszer ezen funkciója erre a problémára ad megoldást, meghatároz egy minimum és maximum volatilitást és a portfóliót nem egy görbén, hanem egy sávban ábrázolja, ahogy az a 28. ábrán is látható, amelyet egy szakmabeli bróker könnyen előnyére tud fordítani.

6.5 Portfólió betöltése Api segítségével

Option symbol:	VXX221209C00012500
Trade price:	1.68
Calc	

29. ábra: Portfólió betöltése Yahoo Api-ról.

A rendszer újdonsága, hogy a felhasználónak a portfólióját nem manuálisan kell megadni, hanem elég az opció azonosítóját beírnia a 29. ábrán látható beviteli mezőbe és az opció árát, amelyen vásárolta ezt a terméket. A rendszer képes a Yahoo Api-ról lekérdezni az adott opcióhoz szükséges adatokat és feltölteni ezzel a 18. ábrán látható beviteli inputokat. Az Api kérések manuális tesztekkel lettek ellenőrizve, melyek sikeresnek bizonyultak.

7. Kiértékelés

A jelenlegi rendszer 2020-ban rögzített VXX opciókkal számol, viszont gazdasági szempontból ez az év nem tekinthető átlagosnak, így az opció volatilitások is olyan értékeket tartalmazhatnak, melyek a megszokottól eltérnek, ez megnehezíti a pontos jövőbeli prediktálást. Ez a probléma az adatbázis növelésével kiküszöbölhető. A VXX részvényopcióinak histórikus adata az 12. ábrán látható.

A rendszer idáig elért eredményeiből a volatilitás prediktálása a számottevő, mivel a megfelelő jövőbeli volatilitással a nyereség görbe egyértelműen meghatározható.

A statisztika alapú volatilitás jóslás 1 éves histórikus adatok alapján az elvárt eredményeket adja, mely a 20. ábrán látható. A teszteléshez szükséges adatokat egy file-ből olvastam be, mely sorai egy adott opció paramétereit tartalmazta és az ehhez elvárt volatilitást értéket, az ábrán az x tengely sorok számát, az y pedig az elvárt eredménytől való eltérést reprezentálja. Ahogy az ábrán is látható egy-egy adott pontban mutat az elvártnál nagyobb hibát, melyet visszafejtve azt tapasztaltam, hogy ezek akkor fordulnak elő, ha az opció volatilitása a szokványostól többszáz százalékkal eltér, mely a szoftver használatakor nem okoz gondot mivel az ilyen kategóriába eső (többszázszázalékos volatilitást) magas kockázatú termékként kezeli a rendszer és ezt jelzi a felhasználó számára is.

A volatilitás pontossága javítható az adatok tisztításával, hasonló opciók közül a zaj kiszűrésével, nagyobb méretű histórikus adatbázissal és a paraméterek finomra hangolásával.

8. Összegzés

A dolgozat célja, hogy feltárja a piacon lévő opciókkal kereskedhető szoftverek hibáit, majd bemutasson egy olyan újszerű rendszert, amely pontosabb, nem félrevezető a kockázatanalízise. A kutatás újdonsága az egyedi jövőbeli implikált volatilitás számítás, amelyet statisztika és LSTM-modell segítségével határoz meg a rendszer.

A 20. ábra szemlélteti a rendszer pontosságát, amelyből látható, hogy a részvényopció implikált volatilitását befolyásoló egyik tényezője a részvényárfolyam alakulása, mivel a rendszer nem jósolja meg a részvényárfolyamot, statisztikailag nem lehetséges a jövőbeli volatilitás pontos meghatározása, a rendszer annál pontatlanabb eredményt ad, minél távolabbi a megadott elemzés dátuma. Ennek egyik megoldása a minimális és maximális volatilitás meghatározása, így a kimenet nem egy pontos portfólió görbét, hanem egy lehetséges portfólió ársáv fog reprezentálni.

Összegzésként a statisztika alapú volatilitás jóslás az 1. éves historikus adatok alapján az elvárt eredményeket adja, melyek a 18., 19. és 20. ábrán láthatóak. Fontos megjegyezni, hogy a jelenlegi adatbázis a VXX részvény 2020-as opcióit tartalmazza, mely gazdasági megközelítés szempontjából nem mondható átlagos évnak. Ebből azt a következtetést lehet levonni, hogy az adathalmaz bővítése 2020-tól eltérő évekre pontosabb végeredményt eredményezhet.

A volatilitás pontossága javítható az adatok tisztításával, hasonló opciók közül a zaj kiszűrésével, nagyobb méretű historikus adatbázissal és a paraméterek finomra hangolásával.

Gyorsítási lehetőség: az összes lehetséges hasonló opció volatilitásának az átlagát előre összegyűjteni, majd kereséskor csak az opció kategóriáját kell megállapítani, és az alapján visszakeresni az adatbázisból.

Jövőbeli továbbfejlesztési lehetőségként megemlíteném az Alert funkció beállítását, melynek működését tekintve, már a meglévő portfóliót vizsgálja, és ha a részvény egy megadott érték alá vagy fölé emelkedik, akkor a felhasználónak jelez (pl.: email-en, applikáción keresztül). E mellett további terveim között szerepel, hogy megadott intervallumra, maximális kockázatra opciós csomagot válogat össze a szoftver, valós időben szűri a jegyzett opciókat és megfelelő jövőbeli portfólió-analízissel dolgozik.

Többféle végeredményt is adhat a szoftver, annak függvényében, hogy a felhasználó mire szeretné optimalizálni: ha biztonság az elsődleges szempont, akkor kevesebb profit, ha a

nagyobb profit, akkor kevésbé biztonságos portfóliót válogat össze, illetve lehet még ennek egyfajta átlaga is.

A későbbiekben tervezek főleg vastag kliensű (Electron Js), de igény szerint Android vagy iOS-es (React Native) elérést is adni a felhasználóknak, amelyen a program leegyszerűsített fő funkciója lesz elérhető.

Irodalomjegyzék

- [1] R. Affarwal, "Stock index futures and cash market volatility, " Review of Futures Markets, vol.7, pp.290-9, 1988
- [2] G. A. Fontanills, "The Options Course - High Profit & Low Stress Trading Methods," Second Edition.
- [3] S. Abraham, "History of Financial Options - Investopedia," Investopedia, Retrieved June 2, 2014 (May 13, 2010).
- [4] Tdameritrade.com, "Electronic trading platform by TD Ameritrade used to trade financial assets," <https://www.tdameritrade.com/tools-and-platforms/thinkorswim.html> (accessed 11/10/2022).
- [5] H. Schulz, S. Behnke, "Deep Learning," Künstliche Intelligenz, vol. 26, no. 4, pp.357–363, 2012.
- [6] T. Hastie, R. Tibshirani, J. Friedman, "The Elements of Statistical Learning: Data Mining, Inference, and Prediction," Springer, New York, NY, 2009.
- [7] H.B. Curry, (1944). "The Method of Steepest Descent for Non-linear Minimization Problems". Quart. Appl. Math., 1944.
- [8] B. Christian, T. Griffiths, "Chapter 7: Overfitting," Algorithms To Live By: The computer science of human decisions, William Collins, pp. 149–168, 2017.
- [9] J. Xie, Z. Ma, J. Lei, G. Zhang, J.-H. Xue, Z.-H. Tan, J. Guo, "Advanced Dropout: A Model-Free Methodology for Bayesian Dropout Optimization," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.44, no.9, pp.4605-4625, 2022.
- [10] A. Y. Ng, "Feature selection L1 vs. L2 regularization and rotational invariance," Proc. 21st ICML, 2004.
- [11] Y. Yao, L. Rosasco, A. Caponnetto, "On Early Stopping in Gradient Descent Learning," Constructive Approximation, vol. 26, no. 2, pp.289–315, 2007.
- [12] J. Barbier, "It's Here: Docker 1.0, " Docker Inc, 2014. Retrieved September 30, 2019.
- [13] H. Sepp; S. Jürgen, "Long short-term memory," Neural Computation, vol. 9, no. 8: pp.1735–1780, 1997.
- [14] "PostgreSQL: The World's Most Advanced Open Source Relational Database". Retrieved September 21, 2019.

- [15] D. Britz, "Recurrent Neural Network Tutorial, Part 4 – Implementing a GRU/LSTM RNN with Python and Theano – WildML," Wildml.com, 2015. Retrieved May 18, 2016.
- [16] Ivolatility.com US Historical Tick and Intraday Options Data. <https://www.ivolatility.com/data/us-historical-intraday-options-data2.html?smid=.3.1.1.2>. Retrieved October 11, 2022.
- [17] "Historical Option Data is a website that sells historical stock and option data," Discountoptiondata.com, <https://historicaloptiondata.com/>. Retrieved October 11, 2022.
- [18] "Historical Option Data website has been actively providing historical option prices and historical stock prices," Historicaloptiondata.com. Retrieved October 11, 2022.
- [19] "Stock Options Historical Data and Trading Tools," IVolatility.com, <https://www.ivolatility.com/>. Retrieved October 11, 2022.
- [20] "What is Object/Relational Mapping?", Hibernate Overview, JBOSS Hibernate. Retrieved January 27, 2022.
- [21] L. Vu, "Project Jupyter: A Computer Code that Transformed Science," Berkeley Lab Computing Sciences, 2021. Retrieved August 15, 2022.
- [22] P. Bjerksund et al. Managing investment opportunities: From last chance to wait and see strategies Financial Management (1990)
- [23] F. Black, M. Scholes, "The Pricing of Options and Corporate Liabilities," Journal of Political Economy, vol. 81, no. 3, pp.637–654, 1973.

Ábrajegyzék

1. ábra: Opciók bonyolultsága (Forrás: Thinkorswim desktop).	6
2. ábra: Thinkorswim Analyzer hiba 1 (Forrás: Thinkorswim desktop).	13
3. ábra: Thinkorswim Analyzer hiba 2 (Forrás: Thinkorswim desktop).	13
4. ábra: A neurális hálózat felépítése.	15
5. ábra: Gradient Descent eljárás.	16
6. ábra: Neuron felépítése.	17
7. ábra: Aktivációs függvények.	18
8. ábra: Bemenő adatok.	22
9. ábra: Nyereség/Veszteség görbe grafikus megjelenítése.	22
10. ábra: A rendszerterv.	24
11. ábra: Rekurrens Neurális Hálózat felépítése.	25
12. ábra: Histórikus VXX opciók.	27
13. ábra: DB Diagram.	29
14. ábra: Hibásan jegyzett opciók (Forrás: Thinkorswim desktop, 2020.03.16.).	30
15. ábra: Volatilitás alakulása az egész adathalmazon.	31
16. ábra: LSTM hálózat kiértékelése: a hiba az iterációk függvényében.	32
17. ábra: Opció jövőbeli volatilitásának meghatározása.	34
18. ábra: Get Options API.	35
19. ábra: Portfólió elemzése.	35
20. ábra: Szoftver volatilitás jóslás teszt.	36
21. ábra: VXX221216C00014500 opció (két részletben).	37
22. ábra: Adott opcióhoz hasonló opciók lekérdezése, grafikus megjelenítése.	37
23. ábra: Hasonló opciók implikált volatilitásának grafikus ábrázolása.	38
24. ábra: Volatilitás prediktálás eredménye.	38
25. ábra: VXX221216C00014500 opció portfólió analízise 1.	39
26. ábra: VXX221216C00014500 opció portfólió analízise 2.	39
27. ábra: VXX221216C00014500 opció portfólió analízise Thinkorswim által.	40
28. ábra: Portfólió számítása minimum, maximum volatilitás prediktálása esetén.	41
29. ábra: Portfólió betöltése Yahoo Api-ról.	41