Eötvös Loránd Tudományegyetem
Természettudományi Kar

Budapesti Corvinus Egyetem
Közgazdaságtudományi Kar

Derivatívák árazása nemparaméteres becslési eljárásokkal

Biztosítási és pénzügyi matematika MSc
Kvantitatív pénzügyek szakirány

Szakdolgozat

Szabó Kristóf

Témavezető:

Badics Milán Csaba

Befektetések és Vállalati Pénzügy Tanszék
Budapesti Corvinus Egyetem

Budapest, 2016

Absztrakt

A dolgozat fő kutatási kérdése, hogy gépi tanulásos módszerek felhasználásával miként árazhatók derivatívák, melynek során bemutatásra kerülnek a Support Vector Regression, az Extreme Learning Machine, Tree Ensemble és a Multilayer Perceptron eljárások. A 2016. január 23. és 2016. február 29. közt gyűjtött, nagyjából 200 000 S&P500 indexre szóló, különböző kötési árfolyamú és lejáratú call opció megfigyelt piaci árát tartalmazó adatbázis segítségével arra a következtetésre jut, hogy a Multilayer Perceptron a legalkalmasabb ezek közül az opció árának pontos megbecslésére. A dolgozat áttekinti a pénzügyi modellek fejlődését, majd a népszerű Heston modell által feltételezett világban bemutatja, hogy a gépi tanulásos módszerek képesek az árazó képlet megtanulására és így a call opció árának nagy pontosságú megbecslésére egy Monte Carlo szimuláció által erre a célra létrehozott adatbázison. A szerző az eredmények alapján arra a következtetésre jut, hogy a nemparaméteres gépi tanulásos módszerek a sztochasztikán alapuló árazó modelleknek méltó kihívói lehetnek.

Köszönetnyilvánítás

Szeretnék ezúton is köszönetet mondani Badics Milánnak a témavezetői munkájáért és remek meglátásaiért, aki nélkül ezen munka ebben a formában és minőségben nem készülhetett volna el. Szeretnék továbbá köszönetet mondani Prof. Dr. Michael Hankenek, aki készségesen elküldte az interneten nem fellelhető írásait, ezzel járulva hozzá a dolgozat színvonalának emeléséhez.

Tartalomjegyzék

Ábrák jegyzéke

A fenti ábrákon és táblázatokon kívül a dolgozat tartalmaz egy DVD-t mellékelve, melyen a felhasznált adatbázis és programkód található, megtartva az eredeti könyvtárszerkezetben.

1) Bevezetés

A XX. század második felének egyik jelentős, ha nem legjelentősebb tudományos vívmánya a mesterséges intelligencia és azon belül a gépi tanulás volt. Mára már annyira életünk részévé vált, hogy bele sem gondolunk, hogy amikor a navigációs szoftverünk útvonalat tervez, amikor a Spotify zenét ajánl nekünk, amikor a Google segítségével keresünk vagy weboldalakat fordítunk le, sőt még akkor is, amikor a Facebook hírek közül „véletlenül" pont a minket érdeklők jelennek meg felül, igazából ezen technológiai lehetőségeket használjuk fel. A sor még nagyon hosszan folytatható lenne.

Mindezen kézzelfogható eredmények mellett rendkívül érdekes, hogy a kvantitatív pénzügyek, azon belül is különösképpen az opcióárazás területén nem sikerült széles körben elterjednie a gépi tanulásos módszereknek, noha az eredmények alapján ez teljesen indokolatlan. Jelen dolgozat célja kettős: Egyrészt szeretné felhívni a figyelmet a gépi tanulás felhasználási lehetőségeire az opcióárazás területén azzal, hogy mind modellkörnyezetben szimulált adatokon, mind valós környezetben piaci adatokon bemutatja a módszer hatékonyságát, amely eredmények alapján a módszer fontos kihívója lehet a területet uraló sztochasztikus pénzügy módszertanának. Másrészt a dolgozat elődeinek munkáját bemutatva, értékelve és azokra nagymértékben építve tesztel és továbbgondol számos tudományos eredményt, továbbá a különböző gépi tanulási módszereket összehasonlítja aktuális adatokon, mellyel segítséget kíván nyújtani a jövő kutatóinak a megfelelő módszer kiválasztásában.

A dolgozat elkészítése során fontos szempont volt, hogy minden, beleértve a véletlenszerűen szimulált adatokat és a véletlen szám generátort alkalmazó modellezési eljárások eredményét is, pontosan reprodukálható legyen. Ezért minden helyen, ahol a véletlen szerepet kapott, a véletlen szám generátor kezdőpontja rögzítésre került, így a későbbiekben a programkódot bármikor lefuttatatva a bemutatott eredmények reprodukálhatóak.

A dolgozat második fejezetében röviden bemutatásra kerül a tudományterületet jelenleg uraló sztochasztikus pénzügy fejlődése, melyből kiderül, hogy a matematikai eszköztár alkalmazhatósága érdekében számos esetben olyan feltételezésekkel élnek a modellezők, melyek a valóságban nem teljesülnek. Bár az utóbbi években ezek a feltételezések egyre közelebb kerültek a valósághoz, ez a számítási probléma jelentős bonyolódásával járt.

A mű harmadik fejezete az eközben a számítástechnikában és a gépi tanulásban végbement fejlődéssel és a különböző módszerek bemutatásával foglalkozik, azok közül kiemelt hangsúlyt fektetve a neurális hálók családjára, elmagyarázva a nemparaméteres becslési eljárások alapelveit. A fejezet foglalkozik azon korlátokkal, amelyek megakadályozták egykor, hogy gépi tanulás párhuzamosan fejlődjön a sztochasztikus pénzüggyel, de mára teljes mértékben megszűntek.

A negyedik fejezet bemutatja a gépi tanulást az opcióárazás területén publikáló korábbi szerzők eredményeit, melyek különböző alaptermékekre szóló, esetenként különböző derivatívák vizsgálatával jutnak gyakran arra a következtetésre, hogy az általuk használt módszer jobb, mint azon sztochasztikus pénzügyekből származó módszer, amelynek eredményeivel összehasonlították sajátjukat. A rész végén táblázatos formában bemutatásra kerül a szerző és a publikáció évszáma, a derivatíva, a felhasznált módszer, illetve az adatbázis alapadatai, melyhez már ezen dolgozat hasonló formában megjelentetett összefoglaló adatai is csatlakoznak.

Az ötödik fejezet a sztochasztikus pénzügyben alkalmazott Heston modellel és a különböző gépi tanulási módszerekkel kapott opcióértéket hasonlítja össze egy olyan modellvilágban, ahol a Heston modell feltevései teljesülnek. Ezen rész fő tanulsága kettős, egyrészt szemlélteti, hogy egy kevés súrlódást tartalmazó rendszerben neurális hálók felhasználásával a valódi opcióérték szinte ugyanolyan hatékonyan megtalálható az alaptermék természetének ismerete nélkül, mint egy olyan modellel, amelybe a modellvilágról fellelhető összes információ be van táplálva. Ezen felül kiderül, hogy a neurális hálók teljesítménye ebben a környezetben felülmúlja a Support Vector Regression, a Tree Ensemble és az Extreme Learning Machine módszerek teljesítményét.

A dolgozat hatodik szerkezeti egysége a valódi adatok beszerzésének módjával, komplikációival, azok megoldásával foglalkozik, elemzi a megszerzett adatokat feldolgozatlan és szűrt formájukban, illetve végigvezeti az olvasót azon lépéseken, melyek a gépi tanulásos módszerekkel kompatibilis adatbázis létrehozásához szükségesek.

A hetedik fejezetben a valós adatokon elért eredmények kerülnek kiértékelésre, emellett a modellvilág és a valóság közti főbb eltérések okozta módszertani problémák és megoldásaik szerepelnek. Ezen részt egy összefoglaló táblázat zárja, mely a különböző gépi tanulási módszerek eredményeit hasonlítja össze a valós adatokon.

A nyolcadik fejezetben áttekintésre kerülnek azon negatív tulajdonságok, amelyeket a neurális hálóval történő becslés során nem sikerült kiküszöbölni. Ezen korlátok áttekintésével teljesebb kép kapható a módszer teljesítményéről. Ezen részben továbbá felvázolásra kerül számos elképzelés, amelyet felhasználva a dolgozat tovább fejleszthető, a becslés pontosabbá tehető, azonban jelen munka keretében még nem kerültek alkalmazásra.

A dolgozat utolsó fejezete az összefoglalás, mely a dolgozat egyes fejezeteinek legfontosabb mondanivalójának felelevenítése után levonja a konklúziót, hogy a gépi tanulásos módszerek alkalmasnak bizonyultak az opciók beárazására az adatbázis jelentős megválogatása nélkül is

2) Opciók árazása sztochasztikus módszerekkel

2.1) Az opció fogalma

Bár az olvasó szinte biztosan tisztában van az opció fogalmával, ennek ellenére nem kerülhető meg az idevágó alapfogalmak gyors bemutatása, hiszen a dolgozat teljes egészében ezzel a témával foglalkozik. Az opció alapvetően felfogható egy biztosításként, melyet az opció kiírója bizonyos összegért (opciós prémium) cserébe ad el az opció vevőjének valamilyen alaptermékre. A call opció lehetőséget ad a vevőnek, hogy egy későbbi időpontban (az opció lejárata) az alaptermék aktuális árától függetlenül bizonyos áron (kötési ár) vehessen az alaptermékből az opciós szerződésben meghatározott darabot az opció kiírójától. A put opció esetében a vevő a kötési árfolyamon egy későbbi időpontban adhat el alapterméket az opció kiírójának a piaci ártól függetlenül. Léteznek amerikai és európai opciók, melyek abban különböznek, hogy az amerikai opciók a lejáratig bármikor, az európai opciók csak lejáratkor hívhatók le. Ezen kívül számos opciófajta létezik, melyeket a kifizetési függvényük és az időpont definiál, amikor lehívhatók.

A call kifizetési függvénye:

$$C = \max(0, S - K)$$

A put kifizetési függvénye:

$$P = \max(0, K - S)$$

A dolgozat a szimulált adatbázis esetében európai call opciókkal, még a valódi adatok esetében amerikai call opciókkal foglalkozik. Az opciók két alapvető felhasználási módja a kockázatok fedezése és a spekuláció, melyek közül az utóbbi terület kerül a fókuszba.

2.2) A Black-Scholes modell

Az opcióárazás területén sok modell van, de szinte az összes fellelhető ilyen modell gondolati hátterét Fisher Black és Myron Scholes (1973) munkája adja, melyben a szerzők egy részvényre szóló európai call beárazásával foglalkoztak egy modellkörnyezetet alapul véve. A modellkörnyezetben az alábbi feltételek teljesültek:

a) *A rövidtávú kamatláb ismert és időben változatlan.*

b) *A részvény geometriai Brown-mozgást követ konstans volatilitással, így a részvény árfolyamának eloszlása lognormális minden időpillanatban.*

c) *A részvény nem fizet osztalékot vagy egyéb járadékot.*

d) *Az opció európai típusú.*

e) *Nem merül fel tranzakciós költség a részvény megvásárlásakor vagy eladásakor.*

f) *Bármekkora pénzmennyiség kölcsönvehető a rövidtávú kamatláb mellett.*

g) *A rövidre eladás (shortolás) engedélyezett.*

(Black & Scholes, 1973, p. 640)

A szerzők a modellvilágban egy önfinanszírozó replikáló portfóliót hoztak létre, mely minden pillanatban ugyanannyit ért, mint az opció, mely gondolat azóta is felhasználásra kerül az opcióárazásban. Az opció ára ekkor a replikáló portfólió opció kiírásakor fennálló értéke, ami a modellvilágban az alábbi képlet szerint („a Black-Scholes képlet") alakul:

$$C(S,t) = N(d_1)S - N(d_2)Ke^{-r(T-t)}$$

ahol N a normális eloszlás eloszlásfüggvénye, r a rövidtávú kamatláb, S a részvény árfolyama, K a kötési árfolyam, T a lejárat időpontja, t a mostani időpont, d_1 és d_2 az alábbiak szerint alakul:

$$d_1 = \frac{1}{\sigma\sqrt{T-t}}\left[\ln\left(\frac{S}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)(T-t)\right], d_2 = d_1 - \sigma\sqrt{T-t}$$

ahol szigma az alaptermék árfolyamát meghatározó geometriai Brown-mozgás volatilitás paramétere (Black & Scholes, 1973, p. 644).

2.3) Az implicit volatilitás felület

A Black-Scholes képlet szerint az input paraméterek ismeretében meghatározható az opció ára, melyek közül a piacon egyedül a volatilitás paraméter nem figyelhető meg. Megfigyelhető ugyanakkor az opció piaci ára, így a végeredmény ismeretében megadható azon volatilitás érték, amely mellett a piacon megfigyelt opcióértéket adja a Black-Scholes képlet. Ezt a volatilitás értéket nevezzük implicit volatilitásnak. A Black-Scholes világban az implicit

volatilitás értéke kötési ártól és a lejáratig hátralévő időtől független, azonban ez nagyon messze áll a valóságtól.

Az implicit volatilitásról széles körben ismert jelenség, hogy időben nem állandó, ezen felül köztudott, hogy az azonos időpontra szóló opciók esetében implicit az volatilitás minimuma a forward árhoz egy közel eső kötési árnál van, melytől távolodva az implicit volatilitás értéke általában egyre nő. Ezen jelenség az implicit volatilitás mosoly elnevezést kapta.

Az opciós piacokon adott alaptermékre szóló számos kötési árfolyam és lejárati idő mellett folyik a kereskedés, így minden létező (lejáratig hátralévő idő, kötési árfolyam) párra kiszámolható az implicit volatilitás, melyet ábrázolva egy háromdimenziós felületté áll össze, melynek x és y tengelyén a hátralévő idő és a kötési árfolyam, z tengelyén pedig az implicit volatilitás szerepel.

Fengler (2010, pp. 4-6) művében az alábbi megfigyeléseket teszi:

- *Minél hosszabb lejáratú az opció, az implicit volatilitás mosoly annál simább, meredeksége kisebbé válik*
- *Az implicit volatilitás mosoly piaci zavarok idején időszakosan aszimetrikussá válhat.*
- *Erős negatív korreláció mutatható ki az alaptermék és az at the money implicit volatilitás között.*

A Black-Scholes modell számos, szinte összes alapfeltételezése sérül a való világban. Az implicit volatilitás egyértelművé tette, hogy a Black-Scholes képlet egyszerű formájában nem alkalmas az opcióárazásra, melynek hatására kétféle irány indult fejlődésnek. Az egyik irány megpróbálta a Black-Scholes modellből visszaszámolt implicit volatilitást megbecsülni különböző idősor elemzés és egyéb területről származó módszerekkel, ide tartozik például a GARCH modellezés. A másik irányvonal pedig a Black-Scholes modell egyes kiindulási pontjainak megváltoztatása, melyek közül az alaptermék adott volatilitású geometriai Brown-mozgásának más mozgásra való cserélése a legnépszerűbb.

2.4) A Heston modell

Miután az empirikus megfigyelések azt mutatták, hogy a volatilitás szintje időben változékony, a következő logikus lépés számos kutató számára a konstans volatilitást egy sztochasztikus folyamattal meghajtott mennyiségre való lecserélése volt. Ezek a modellek lettek a

sztochasztikus volatilitás modellek, ide tartozik többek között a Hull-White modell (1987), a Stein-Stein modell (1991) és a dolgozat számára kiemelten fontos Heston modell (1993) is.

A Heston modell dinamikái az alábbi módon alakulnak:

$$dS_t = \mu S_t dt + \sqrt{v_t} S_t dW_{1,t}$$

$$dv_t = k(\theta - v_t)dt + \sigma\sqrt{v_t} dW_{2,t}$$

$$dW_{1,t} dW_{2,t} = \rho dt$$

Ahol a betűk a következő jelentéssel bírnak:

- S_t és v_t a részvényár és a volatilitás adott időpontbeli értékei
- $W_{1,t}$ és $W_{2,t}$ két Wiener folyamat, melyek közt a korreláció ρ
- μ a Black-Scholes modellhez hasonlóan részvényár drift paramétere
- k a az átlaghoz való visszahúzás erősségének paramétere
- θ a volatilitás hosszú távú átlaga
- σ a volatilitást generáló folyamat volatilitása

(Heston, 1993, pp. 328-329)

A modell különlegessége abban rejlik, hogy a volatilitást meghajtó folyamat egy Cox-Ingersoll-Ross (CIR) folyamat (Cox, et al., 1985), aminek segítségével a volatilitás pozitív, továbbá $2k\theta > \sigma^2$ feltétel teljesülése esetén az sosem éri el a nullát (Mikhailov & Nögel, 2003), így a volatilitás mindig pozitív marad.

A Heston modell számos tulajdonsága ismert, az előnyök a következők:

1. *Európai call opció esetében létezik zárt alakú megoldása*
2. *Képes leírni a részvényárfolyam tulajdonságait akkor is, ha az eloszlása nem lognormális*
3. *Képes a tapasztalt implicit volatilitás felülethez hasonló volatilitás felület létrehozására.*

4. *A részvényár és a volatilitás korrelálhatnak egymással negatívan.*

Hátrányai pedig a következőképpen alakulnak:

1. *Nehéz megtalálni a sztochasztikus modell kalibrálásához szükséges megfelelő paramétereket.*
2. *A Heston modell által visszaadott árak érzékenyek a paraméterek megválasztására, ezért a pontosság nagyban függ a kalibrációtól. (Mikhailov & Nögel, 2003)*
3. *Rövid lejáratok esetében nem képes a volatilitás mosoly piacon megfigyelt mértékig történő reprodukálására. (Mikhailov & Nögel, 2003)*

(Yang, 2013, p. 31)

A Heston modell tehát könnyen tekinthető a Black-Scholes modell javított, ugyanakkor bonyolultabb változatának, azonban még mindig nem volt képes a piacon megfigyelhető minden jelenség reprodukálásra.

2.5) Opciók árazása a Heston modell után

A Heston modell sem volt képes minden piacon látott jelenség okára fényt deríteni és azokat a modellkeretében reprodukálni, ezért az igény valami általánosabbra, akárcsak a Black-Scholes modell kapcsán, ismét fellépett. A modellező eszköztárában ezúttal megjelentek az ugró (Lévy) folyamatok, hogy az alaptermékek áralakulását okozó geometriai Brown-mozgásokat ezzel az általánosabb mozgással lehessen helyettesíteni.

Definíció szerint egy X(t) folyamatot Lévy folyamatnak nevezünk, ha teljesíti az alábbi három kritériumot:

Független növekményű, vagyis:

$$X(t_0), X(t_1) - X(t_0), \dots, X(t_n) - X(t_{n-1}) \text{ függetlenek}$$

Stacionárius növekményű, vagyis:

$$X(t+h) - X(t) \sim X(s+h) - X(s)$$

Sztochasztikusan folytonos, vagyis:

$$\forall \varepsilon > 0: \lim_{h \to 0} P(|X(t+h) - X(t)| \geq \varepsilon) = 0$$

(Márkus, 2013, p. 1)

Amint az látható, így az alaptermék áralakulását leíró folyamat már nagyon általános, mindössze három feltételt köt ki. Feltételezhető, hogy amennyiben valamelyik feltétel empirikusan mégsem lenne igaz, úgy valószínűleg a modell helyébe egy még általánosabb modell lépne, melyből a sérülő feltétel elhagyásra kerülne. Ilyen feltétel lehet például akár az első, mivel egyes kutatók szerint létezik a csordaszellem jelensége (Teh & de Bondt, 1997), illetve amennyiben teljesülne, a technikai elemzés is teljesen értelmetlen lenne. Jelen dolgozatnak nem célja bemutatni az ugró folyamatok segítségével történő árazást, sem annak alapfeltételeit igazolni vagy cáfolni. A pénzügyi modellezésben történt fejlődés azért került részletezésre, hogy bemutassa ennek menetét. Kezdetben voltak piaci adatok, majd alkottak egy egyszerű modellt, amelyről úgy tűnt, leírja azokat. A részletesebb vizsgálatok során kiderült, hogy a modell mégsem mindig olyan eredményeket ad vissza, ami a valóságban megfigyelhető, ezért csináltak egy új modellt, melyben a régi modell bizonyos feltevései elhagyásra kerültek, az alaptermék áralakulása általánosabbá vált. Később ez a folyamat folytatódott, egyre kevesebb feltevést, általánosabb modelleket és bonyolultabb kalibrációt és nagyobb számítási igényt hagyva maga után. A modell jóságának mértéke pedig kizárólag a piaci adatokhoz való illeszkedéstől függött. Ebből a szempontból tekintve a dolgozat témáját adó nemparaméteres modellezés nem csupán egy másik tudományterületről átszivárgott módszertan, hanem szükségszerűen az opcióárazáshoz felhasznált sztochasztikus modellek következő modellje a sorban, melyben még kevesebb feltevéssel élünk.

3) A gépi tanulás

Míg a számítógép megépítésének idején szoba méretű és felettébb lassú volt, az elmúlt évtizedek során mind hardware, mind szoftver tekintetében hatalmas fejlődésen ment át. A kezdetben hozzáférhetetlen és hadi célokat szolgáló monstrum mára már szinte mindenki számára hozzáférhetővé vált, így manapság senki nem lepődik meg azon, ha egy átlagos vállalat pénzügyi döntéseiben aktívan használja az Excel célérték keresését, termelését valamilyen operációkutatásban használt szoftverrel optimalizálja vagy egy tőzsdei kereskedő egész egyszerűen leolvassa az implied volatility pontos értékét a Bloomberg kijelzőjéről. Az előbbiekben közös, hogy mind numerikus szélsőérték-keresést igénylő feladatok, melyek 40 évvel ezelőtt mindenképpen lassabbak, de lehet, hogy kivitelezhetetlenek lettek volna praktikus szempontokat nézve.

Miközben ezen példák jók annak szemléltetésére, hogy a számítási kapacitás növekedése hogyan tette könnyebbé az elvégzendő megszokott feladatokat, ezzel párhuzamosan elindult egy másik irány is, mely ezen kapacitást kihasználva igyekezett olyan algoritmusokat létrehozni, melyek képesek gyorsan, önállóan véleményt alkotni. Bár az elsődleges motivációt valószínűleg a hidegháború jelentette, a fejlődés nem állt meg és a gépi tanulás legújabb fajtáját képviselő deep learning algoritmusok segítségével 2016-ban a számítógép végül legyőzze az embert a GO társasjátékban, mely az egyik legbonyolultabb társasjátékként van számon tartva.

Ezen irány párhuzamos fejlődése az opcióárazás módszertanában akadémiai szinten nem hozott jelentős változást. A korai időkben numerikus eszközök hiányában a valószínűség számítás vált az elfogadott és domináns irányzattá, hiszen a nemparaméteres számítási lehetőségek negyven évvel ezelőtt eszköz és így kutatás hiányában széles körben kivitelezhetetlenek voltak. Ahogy a számítógépek egyre könnyebben hozzáférhetőek lettek, egyre több ember kezdte el úgy érezni, hogy az ebben rejlő potenciált opcióárazás területén is ki lehet aknázni, ezen írásokkal és publikációkkal foglalkozik a 3. fejezet. Sajnos ezen irányzatnak nem sikerült mainstreammé válnia, amit az is jól mutat, hogy a Journal of Computational Intelligence in Finance (korábban NeuroVest Journal) hasonló témájú folyóirat 1999-ben[1] megszűnt és a benne valaha ebben publikált cikkek nehezen vagy egyáltalán nem elérhetőek.

[1] Az egyetlen általam talált dokumentum a www.lchr.com/a/18/1b/JCIFIndex.pdf címen érhető el, mely szerint 1999-ben volt az utolsó szám.

A gépi tanulást leggyakrabban kétféleképpen osztályozzák. Az első szempont a feladat tulajdonsága, melyhez fel akarják használni, mely szerint a cél lehet egy kívánt folytonos érték visszaadása (regresszorok) vagy az adatpontok kategóriákba sorolása (osztályozók). Az osztályozók számos feladatra alkalmasak a pénzügyi csalások felderítésétől a kézzel írott számjegyfelismerésen keresztül a betegségdiagnosztikáig, még a regresszorokhoz nehezebb emblematikus felhasználási területeket csatolni, de nagyon gyakoriak egy egyedi termék, például ingatlan árának becslésénél, illetve várt részvényárfolyamok előrejelzésénél.

A másik lehetséges csoportba sorolás aszerint történik, hogy a tanulás során milyen információt kap a rendszer. Ennek egyik fajtájában nincs kitüntetett célváltozó, mindössze a hasonló adatok megtalálása a cél, melyet felügyelet nélküli tanításnak (unsupervised learning) neveznek. Ennek a csoportnak egy jó példája a klaszterezés. A tanítás másik formája a felügyelt tanítás (supervised learning), melynek során a tanítandó rendszerrel közlésre kerül az inputváltozók feldolgozása után visszaadott célváltozó kívánt értéke, mely egyaránt lehet kategóriaváltozó vagy egy érték. Ebben az esetben a gépi tanulás célja a megadott célváltozók leképzéséhez szükséges módszerek megtalálása, melyet más és más algoritmusok különbözőképpen oldanak meg. A dolgozatban kizárólag felügyelt tanításos módszerek kerülnek elő.

A tanítás során nagyon gyakran használt technika, hogy az adatbázist három különböző részre bontják, az első részt tanító mintának, a második részt validációs mintának, a harmadikat pedig tesztmintának nevezik. Az algoritmus tanulása során csak a tesztmintán tanul, majd az aktuális modell minőségének kiértékelésére annak validációs adatbázison elért teljesítménye alapján történik, ezzel megakadályozva, hogy a rendszer túltanuljon. A túltanulás jellemző szemléltető példája annak felvázolása, hogy amennyiben létezik néhány pontunk két dimenzióban és az adott X érték alapján az y értékét szeretnénk megjósolni, akkor az azokat összekötő n dimenziós polinom mindig nulla hibával dolgozik, azonban mégsem feltételezhető, hogy az általánosságban tükrözné a két változó közti kapcsolatot. Az adatok egy részének visszatartásával (a validációs minta) a túltanulási folyamat megállítható akár első vagy másodfokú polinomnál, ahol is a validációs mintán számolt hiba elkezd nőni, így kapva meg a jól általánosító modellt. Utoljára a teszt mintán kiértékelve megállapítható milyen a modell teljesítménye olyan adatokon, amivel még sosem találkozott.

A kiértékelés során a modell validációja valamilyen hiba definiálásával és mérésével történik, minél kisebb a kiszámolt hiba értéke, annál közelebb áll a kapott eredmény a kívánt eredményhez, még teljes egyezés esetén a hiba értéke nulla. Számos hiba értelmezés elterjedt,

a leggyakoribbak a helytelen csoportba sorolások száma osztályozóknál, még regresszoroknál a négyzetes hibaösszeg és az abszolút eltérések összege a gyakori. A különböző hibák más és más előnyökkel és hátrányokkal rendelkeznek, általában nem ugyanazon modellparaméterek mellett minimálisak, így gyakran a felhasználás céljához illeszkedés a fontos kiválasztási szempont. A dolgozatban a jósolt és a valódi opciós ár közti hiba alatt az átlagos négyzetes eltérést kell érteni, vagyis:

$$HIBA = \frac{\sum_{i=1}^{n}\left(y_{valódi} - y_{jósolt}\right)^2}{n}$$

3.1) A neurális hálók

A bemutatott módszerek közül a neurális hálók kapják a kiemelt figyelmet, melynek oka egyrészt sokféleségük és így rugalmasságuk, másrészt a bennük rejlő potenciál, végső soron pedig azon eredményük, hogy az itt tesztelt adatokon teljesítményük túltesz a másik három módszer által elért teljesítményen.

A neurális hálók alapgondolata, hogy az idegsejtek tanulmányozásának segítségével szerettek volna létrehozni egy komplex rendszert, ami az agyi viselkedést szimulálva igyekszik eredményeket elérni. A módszer kifejlődéséhez vezető utat főleg Warren McCulloch és Walter Pitts neuron modellje (1943) és Donald Hebb tanulással kapcsolatos publikációja (1949) nyitotta meg, mellyel megszülethetett a neurális hálók alapegységét alkotó tanuló neuron, a perceptron (Rosenblatt, 1958). A neurális hálók fejlődése ezek után nem sokkal nagyban lelassult, miután kiderült, hogy a perceptron nem képes még egy *kizáró vagy (XOR)* kapcsolatot sem megtanulni, felhasználási területe mindössze lineárisan szeparálható osztályozási feladatokra korlátozódik (Minsky & Papert, 1969). A neurális hálók tudományos reneszánszát a hiba visszaterjesztéses algorimus (back-propagation) kifejlesztése (Rumelhart, et al., 1986) és köztudatba meggyökeresedése hozta meg, mely segítségével akár többrétegű neurális hálók is taníthatóvá váltak, melynek segítségével az osztályozási feladatokban addig létező korlátokat sikerült áttörni. (Altrichter, et al., 2006) áttekintése alapján.

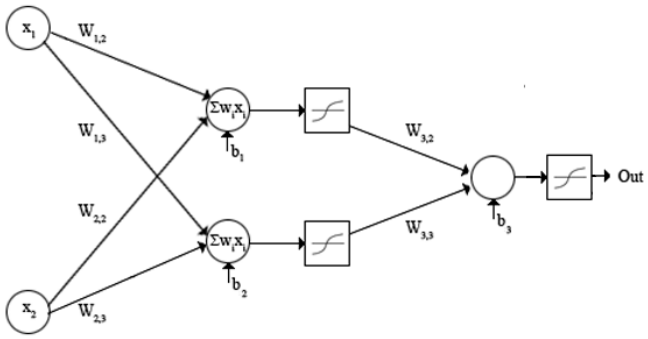
Azóta sokféle formájuk kialakult, melyek közül a legfontosabbak a konvolúciós hálók (Convolutional Neural Network), melyek a képfelismerés területén elterjedtek, a visszacsatolásos neurális hálózatok (Recurrent Neural Network) és a többrétegű perceptronok (Multilayer Perceptron), melyeket a dolgozat részletesebben is bemutat és egyrétegű formáját felhasználja. Ezen kívül különösen elterjedt a Radial Basis Function módszer használata is,

mely ugyan némileg más elven működik, de minden ilyen elméletben bizonyítottan megfeleltethető egy Multilayer Perceptronnak (Maruyama, et al., 1992), ezért, bár néhány szerző inkább ezen módszert részesíti előnyben, a dolgozat mégis kizárólag a többrétegű perceptronok módszertanát használja fel.

3.2) A Multilayer Perceptron

Az eljárás lényege, hogy a többdimenziós input adatpont átalakítások sorával különböző változásokon megy át, az egyes átalakítás után létrejött terméket nevezzük az adott rétegben felvett értéknek. Eszerint egy ilyen eljárásban létezik legalább egy bemeneti (input adatokat tartalmazó) réteg, egy kimeneti (célértéke(ke)t tartalmazó) réteg és legalább egy, ezen dolgozat esetében pontosan egy közbülső, „rejtett" réteg. A rejtett réteg nevével ellentétben nem láthatatlan, értékei megfigyelhetők, mindössze közvetlenül ezen információ nem fontos a probléma szempontjából, ezért általában rejtve marad.

Az átalakítások során egy következő rétegben lévő elem mindig a korábbi rétegekben szereplő adatokból alakul ki, azokat bizonyos súllyal megszorozva, egy konstanst hozzáadva és egy aktivációs függvénynek nevezett függvénnyel kiértékelve, mely általában sigmoid vagy tanh. A rejtett rétegekben konvenció szerint általában ugyanolyan számú elem szerepel, bár ez konvenció kérdése. A rejtett rétegekben szereplő egységek száma a modell egyik legfontosabb paramétere, mely a hálót létrehozó egyén választását tükrözi, egyike azon paramétereknek, melyeket a validációs technikával határozhatunk meg.



1. ábra: Egyrétegű Multilayer Perceptron sematikus ábrája.
Forrás: (Matlab Geeks, 2011)

Ugyanez képlettel egyrétegű perceptron háló esetében:

$$y_t = s\left(\beta_0 + \sum_{i=1}^{q} \beta_i * h_{i,t}\right) + \varepsilon_t$$

$$h_{i,t} = g\left(\alpha_{i0} + \sum_{i=1}^{n} \alpha_{ij} * x_{j,t}\right)$$

Ahol s és g függvények a modell meghatározásánál megadott aktivációs függvények, α tömb a bementi és a rejtett réteg közti súlyokat tartalmazza, β a rejtett és a kimeneti réteg közti súlyok mátrixa, ε pedig egy véletlen hiba nulla várható értékkel (Gencay & Salih, 2003).

A neurális hálók tanításának fő célja, hogy az input és várt output adatok alapján meghatározásra kerüljenek azon α és β súlyok, melyek segítségével a modell minél pontosabban becsüli meg a megfelelő kimenetet. A megfelelő súlyok megtalálásához tanító algoritmusokat használnak, melyek célja, hogy numerikus szélsőérték-keresés alapján az eltérés hibáját minimalizálva egyre közelebb kerülnek a minimumhoz. A megtalált minimum azonban a módszer jellegéből fakadóan nem feltétlenül globális minimum, lehet, hogy csak lokális, ezért a szélsőérték-keresés sikerességéhez ugyanazon paraméterbeállítások mellett a kezdeti véletlen súlyok különböző randomizált értékekről történő elindítása szükséges.

A neurális hálók esetében a beállítandó fő paraméter a rejtett rétegben található neuronok száma, ugyanakkor számos kisebb paraméter is létezik. Ilyen az aktivációs függvény, a tanító algoritmus és annak beállított paraméterei, ennek megállításához választott kritérium. A tanításhoz használt főbb algoritmusok a Back-Propagation, A Levenberg- Marquardt és a Quasi-Newton algoritmusok, melyek felhasználását ezen dolgozat a későbbiekben mint technikai részletet a szükséges mértékig közöl, de mivel ezen ismeretek részletes átadása a dolgozat keretein jóval túlmutat és nem kimondottan fontosak az eredmények értékeléséhez, ezen tanító algoritmusok részletes leírásai és összehasonlításai szükségszerűen a dolgozatból kimaradnak.

A módszer felhasználása melletti egyik legnagyobb érvet Cybenko (1989) és Hornik (1989) szolgáltatta azzal, hogy munkájukban bebizonyították, hogy a lineáris és nemlineáris folytonos függvények majdnem minden fajtája tetszőleges pontosságig közelíthető egyetlen rejtett réteget tartalmazó MLP-vel, amennyiben mind az inputváltozók, mind a célváltozók korlátosak.

Későbbi vizsgálatok során kiderült, hogy a MLP által adott előrejelzés javítható, ha nem csak egy, hanem több MLP-ből kapott előrejelzés értékének számtani közepét vesszük (Crone, et al., 2011).

3.3) Support Vector Regression (Tartóvektor regresszió)

Az Support Vector Regression (továbbiakban SVR) a Support Vector Machine elnevezésű osztályozó módszer regressziós feladatokra kifejezett változata, melyben adott egy inputváltozókat és célváltozót tartalmazó minta és általában nemlineáris regressziós előrejelzés a feladat a minta alapján. Vagyis feltételezzük, hogy az összefüggés:

$$output_i = f(x_i) + \delta, \qquad \delta \sim N(0, \sigma^2)$$

A feladat megoldásához első lépésben az SVR egy felsőbb dimenziós térbe képezi le az input adatokat, ahol azok már lineárisan korrelálnak a célváltozóval. Képlettel:

$$f(x) = \big(v * \Phi(x)\big) + b$$

ahol v egy súlyvektor, b egy konstans, Φ(x) pedig a magas dimenziós térbe képző függvény.

Ekkor a veszteségfüggvény a következőképp definiálható:

$$L_\varepsilon(f(x), q) = \begin{cases} |f(x) - q| - \varepsilon \;\; ha \; |f(x) - q| \geq \varepsilon \\ 0 \; egyébként \end{cases}$$

ahol az ε paraméter egy pontosságot szabályozó paraméter, melyet a felhasználó állít be. A feladat megoldásához az alábbi rizikófüggvény minimalizálására van szükség:

$$R(C) = C * \frac{1}{n} * \sum_{i=1}^{n} L_\varepsilon(f(x_i), q_i) + \frac{1}{2}|w|^2$$

melynek utolsó tagja egy regularizációs tag. C szintén egy felhasználó által megadandó együttható, ami egyfajta regularizációt beállító paraméter. (Lu, et al., 2009).

A módszer tehát egy felhasználótól kapott ε és C paraméter és egy tanító adatbázis alapján tanul, ezen két paramétert pedig gyakran a lehetséges kombinációk empirikus teljesítménye alapján választják ki, a becslést a korábban említett validációs mintán lefuttatva

3.4) Extreme Learning Machine (Extrém tanulógép)

Az Extreme Learning Machine (továbbiakban ELM) egy újfajta becslési módszer abban a tekintetben, hogy az utóbbi 10 évben fejlődött ki, még más tekintetben a régi módszerek, leginkább a neurális hálók egyfajta általánosított változatának tekinthető. A módszer alapötlete az, hogy a neurális hálót ne iteratív tanítási algoritmusokkal tanítsuk, hanem számos paramétert

randomizáljunk, így zárt alakban visszaszámolhatók legyenek adott véletlen beállítás mellett a legjobb hálózat paraméterei.

Adott tanító adatbázis és rejtett réteg kimeneti funkció (lehet például a szokásos sigmoid):

$$\{(x_i, t_i) | \, x_i \in R^d, t_i \in R^m, i = 1,2, \dots N\} \;\; és \;\; G(a, b, x)$$

és beállított L rejtett rétegbeli neuron esetén az ELM modell elkészítésének 3 fázisa van:

1. (a_i, b_i) rejtett réteghez tartozó paraméterek véletlenszerűen megválasztásra kerülnek. (i=1,2,…L)

2. Kiszámításra kerülnek a rejtett réteg kimeneti értékei: $H = \begin{Bmatrix} h(x_1) \\ . \\ h(x_N \end{Bmatrix}$

3. Az output súly β változók kiszámításra kerülnek úgy, hogy minimalizálják a $\|H\beta - T\|_p$ $és$ $\|\beta\|_q$ értékeket, ahol p és q nem feltétlenül ugyanaz a szám.

(Huang, 2015)

A gyakorlatban tehát egészen hasonlóan működik, mint az SVR, a felhasználó megad két paramétert, a rejtett rétegben szereplő neuronok számát, illetve egy másik technikai paramétert, esetleg az aktivációs G függvényt is, majd a betanított hálózattal megbecsüli az eddig nem látott adatbázist és megméri a hibát. A módszer előnye, hogy a dolgozatban bemutatásra került gépi tanulásos módszerekhez képest meglehetősen gyors, miközben a neurális hálók számos pozitív tulajdonságával bír.

3.5) Tree Ensemble (Döntési fa együttes)

Egy ilyen modell különböző regressziós és klasszifikációs döntési fák által adott eredmények összegét jelenti, formálisan az alábbi módon írható fel:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i) \, , \;\; f_k \in F$$

ahol K a döntési fák száma, f pedig egy függvény az F függvénytérben, ahol F minden lehetséges klasszifikációs és regressziós döntési fának a halmaza. Ezután a korábbiakban megismert módon egy hibaérték kerül meghatározásra a felhasználó céljai szerint, mely

különböző módszerekkel minimalizálásra kerül, így találva meg a megfelelő paramétereket (xgboost developers, 2016).

A dolgozatban a tree boosting hibaminimalizálást használtam, illetve ez három paramétere mentén került optimalizálásra, melyek a mélység, a tanulási ráta és az estimator paraméterek voltak. A mélység paraméter a becsléshez használt döntési fák maximális mélységét határozza meg, a tanulási ráta a minimalizáló algoritmus egy paramétere, mely nagyjából a többi algoritmus tanulási rátájához hasonló módon működik, az estimator paraméter pedig a becsléshez használt döntési fák számát határozza meg. Az algoritmusokról részletesebb információk megtalálhatók a fent hivatkozott forrásban, a dolgozat szempontjából azonban nem fontos ennél mélyebben foglalkozni a módszerrel, mivel amint az majd a későbbiekben látható, az opciók árazása terén a módszer nem ad versenyképes eredményeket a másik három algoritmushoz képest. A döntési fa együttes mindazonáltal véleményem szerint az utóbbi évek egyik legkedveltebb becslési módszerévé nőtte ki magát különösen társadalomtudományi területen, ezért ennek gyengébb eredménye a dolgozat szempontjából fontos. A módszernek ugyanakkor vitathatatlan előnye a gyorsasága, mely nagyságrendileg az ELM-ek sebességéhez hasonlít.

A dolgozatban szereplő mind a négy módszer esetében célszerű mind az input, mind a célváltozókat 0,5 várható értékűvé és a [0;1] intervallumba átskálázni, mivel ezzel a tanulás gyorsasága a tapasztalat szerint növelhető, egyes módszereknél pedig ez a célváltozó esetén elengedhetetlen, hiszen a végeredmény is csak ezen tartományba tud leképződni.

4) Neurális hálók az opcióárazásban

Az alapgondolat, miszerint az informatika fejlődéséből származó új becslési lehetőségek előremozdíthatják a derivatívák árazását, már jóval a számítástechnikai teljesítmény 2000-es évekbeli robbanásszerű növekedése előtt megjelent. A legkorábbi általánosan ismert ilyen tanulmány Mary Malliaris és Linda Salchenberger (1993) „A Neural Network Model for Estimating Option Prices" munkája, melyben több, mint 20 évvel ezelőtt az OEX indexre mint alaptermékre kiírt opciók lehetséges új árazási módszerét vizsgálta. A terület úttörői közül nagyon nehéz lenne kihagyni a James M. Hutchinson, Andrew W. Lo, Tomaso Poggio szerzőhármast, akik cikkükben (1994) az S&P 500 alaptermékre szóló opciókat vizsgálták.

Ők és a későbbi szerzők általánosságban inkább egyetértettek abban, hogy a gépi tanulás és különösen a neurális hálók alkalmasabbak az opciók beárazására, mint az addig gyakran használt módszerek, azonban itt a hasonlóság általában véget is ér. A tanulmányok különböztek a felhasznált alaptermék, gépi tanulási módszer, adatmennyiség, lefedett időintervallum és néha még a derivatíva típusa szerint is. A következőkben bemutatásra kerülnek azon publikációk, melyek olyan jelentős észrevételt tettek vagy módszertani javaslatot közöltek, melyek a későbbiekben érdekesnek vagy hasznosnak bizonyultak és amelyek egy része megfogadásra került jelen tanulmány elkészítése során.

Az idők során három olyan fontos kérdéskör is felmerült, melyek módszertani szemszögből nagyon sarkallatosak, ugyanakkor nem igazán sikerült rájuk kielégítő választ találni. Az első és talán legfontosabb ilyen az a kérdés, hogy ha az opció árát szeretné valaki megtudni adott inputok mellett, akkor mit is kell megbecsülnie. A legtriviálisabb lehetőség, amelyet a korábban hivatkozott cikkben Malliaris-Salchenberger (1993) szerzőpáros is választott, hogy egyenesen az opció ára kerül megbecslésre. Ezen lehetőség mellett számos szerzőnél, például Hutchinsonéknál (1994) is felmerült, hogy az opció árának és az opció kötési árának hányadosát kellene megbecsülni, mely álláspont legalább annyira elterjedt, mint az érték egyenes becslése. A harmadik lehetőség a Black-Scholes féle modellből származó valamely paraméter becslése, amely lehet az implicit volatilitás (Malliaris & Salchenberger, 1996), vagy a N1 és N2 segédparaméter a Black-Scholes képletben (Mitra, 2012). Nem túl szigorúan tekintve ide tartoznak a hibrid neurális hálók is, amik a Black-Scholes ártól való eltérést igyekeznek megbecsülni (Blynski & Faseruk, 2006), (Lajbcygier & Connor, 1997). A módszer elméleti előnye, hogy segítségével „súghatunk" valamilyen függvényformát a neurális hálónak, így akár azt hatékonyabbá is tehetjük. Hátránya, hogy így a becslés a segéd-célváltozó és a becsült

segéd-célváltozó közti hibát minimalizálja, amiről kiderülhet, hogy nem ugyanazon neurális háló paraméterek mellett minimális, amely paraméterek mellett a célváltozó és a becsült célváltozó hibája lenne minimális.

A második ilyen fontos kérdés, hogy milyen inputváltozókkal szeretnénk a célt becsülni. Alapvetően abban majdnem egyetértés van, hogy csak a Black-Scholes képlet által felhasznált változók fontosak az opció értékének meghatározásakor, mindössze néhány cikkben jelennek meg egyéb fantáziadús változók, Malliaris (1993) úttörő cikkében többek között az opció és az alaptermék késleltetett árát is felhasználta magyarázó változóként. Számos szerző, különösen azok, akik a célváltozónak az opció ár és a kötési ár hányadosát választották, előszeretettel használja magyarázó változóként az alaptermék árfolyamának és a kötési árnak a hányadosát, ez azonban nem kizárólagos, ez számos esetben az opció árát közvetlenül becsülve is előfordul (Yao, et al., 2000). Ha az opciók nem ugyanarra a lejáratra vonatkoznak, akkor a lejáratig hátralévő idő kihagyhatatlannak tűnik a neurális háló input adatai közül, számos szerző (Hutchinson, et al., 1994), (Garcia & Gencay, 2000) ezen két adatnál meg is áll. Akik nem így tettek, ott fontos volt továbbá valamilyen volatilitás nagyság (legyen az historikus vagy implicit), különösen hosszú távú opciók esetén pedig valamilyen kamatláb nagysága.

A harmadik nagy kérdéskör az adatbázisra fókuszál, mégpedig hogy hogyan is kellene bánni az adatokkal, milyen alapterméket érdemes választani, érdemesebb egyben megbecsülni az egész adatbázist vagy érdemesebb az adatokat valamely szempont (lejárat, árfolyam/kötési ár arány) szerint csoportokra bontani és külön becsülni őket? Általános érvényű válasz ezen kérdésekben sincs, de a tanulmányokból számos érdekes eredmény közölhető.

Az Anders-Korn-Schmitt (1998) szerzőhármas arra a megállapításra jutott, hogy alapterméknek csak magasabb árú részvény vagy index választható, mivel ebben ez esetben a piacon a kerekítés miatt elvesztett információk kevésbé roncsolják az opció értékének pontosságát.

Malliarisék már hivatkozott legelső cikkükben (1993) egy olyan érdekes javaslattal álltak elő, hogy az olyan call opciókat, ahol az alaptermék forward ára nagyobb, mint a kötési árfolyam (in the money) és azokat az opciókat, ahol a forward ár kisebb, mint a kötési árfolyam (out of the money), különítsük el egymástól és a két adatbázist különállóként kezelve kerüljenek beárazásra. A tanács gyakorlati szempontból igen értékes, hiszen a két csoportban az opcióárak teljesen más nagyságrendű skálán mozognak, így a két csoportot együtt kezelve a kis opcióárak között a neurális háló szükségképpen nem tesz nagy különbséget, inkább a magas és az alacsony árak elszeparálására törekszik az árbecslés során. Ugyanakkor nem szabad átesni a ló túlsó

oldalára és túl sok kis csoportot képezni, azokat pedig egyenként tanítani, mivel ilyenkor a minta annyira lecsökkenhet, hogy emiatt a becslések pontatlanná válnak, általánosító képességük és így pontosságuk romlik (Mitra, 2012) szerint. Mitra cikkének tanulsága, hogy amennyiben minden negyedévben külön tanított egy-egy neurális hálót az adott negyedéves adatokon, hogy a negyedéves adatok másik felét megjósolja vele, nem ért el olyan jó eredményt, mintha egységesen kezelte az adatbázist. Tizenhárom negyedévre bontotta az adatbázist és ez kevesebb, mint 30 000 adat mellett túl soknak bizonyult.

Egyes szerzők pedig úgy gondolják, hogy az amúgy az opcióárazáskor az egyéb modellek szerint is bizonytalanul árazható, de pont emiatt rendkívül érdekes opciókkal inkább nem foglalkoznak és a neurális hálók erejét, csak bizonyos tartományon demonstrálják. (Anders, et al., 1998, p. 8) például az alábbi kritériumokat tartalmazza, melyeket más, általa a cikkben feltüntetett korábbi kutatókkal egyetértésben alkalmaz:

Az adatbázisból eltávolításra kerül azon call opció, amelyre az alábbi feltételek legalább egyike igaz:

- *Az opció ára 10 pontnál kisebb.*
- *15-nél kevesebb nap van hátra a lejáratáig.*
- *Az európai call opció alsó korlátja megsérült. ($C < S - K * e^{-r*t}$)*
- *Az opció nagyon „In The Money" vagy „Out of The Money" (S/K <0,85 vagy S/K >1,15)*

A korábban említett három fő kérdéskörön kívül is számos érdekes megfigyelés született. Az optimális rejtett neuronok számának meghatározásával kapcsolatban nem született elterjedt konszenzusos álláspont, sokkal inkább a próbálgatásos módszer jellemző, bár néhány szerző próbálkozott munkájában ennek legalább intervallum szintű meghatározásával (Yao & Tan, 2001), (Teräsvirta, et al., 1993), (White, 1989). Ez valamennyire talán köszönhető annak is, hogy Swanson és White (1995) munkájukban bemutatták, hogy az általánosan elfogadott Schwarz információs kritérium neurális hálózatok esetében nem a jól teljesítő modelleket választja ki, ezért a megfelelő szerkezetű neurális háló kiválasztása rendkívül bonyolult feladat lehet.

Gencay és Qi (2001) különböző regularizációs módszerek lehetőségeivel foglalkoztak és így tovább tudták javítani a becslés pontosságát, akárcsak Garcia és Gencay (2000), akik ugyanezt

bizonyos homogenitási tulajdonságokat kikényszerítve tették meg. A becslés megbízhatósága is fontos probléma, ezen a területen kiemelkedő Lajbcygier és Connor (1997) a gépi neurális hálók esetében a konfidencia intervallum megbecslésének lehetőségével foglalkozott és módszert ad erre, míg Amilon (2003) ugyanerre a problémára bootstrap módszerrel meghatározott megoldást adott.

Az alkalmazott módszerek tekintetében elterjedtek a Radial Basis Function és a Multilayer Perceptronok, illetve rendkívül érdekes megközelítést adott Zaheer Ahmed Dindar (2004) dolgozatában, melyben a neurális hálók optimalizálásának céljából természetből vett algoritmusokkal, mint a „Genetic Algorithm" vagy a „Particle Swarm Optimalization" is foglalkozik, melyekkel jó eredményeket ért el.

Ezen dolgozatban az opció ára közvetlenül kerül megbecslésre, amelynek fő oka, hogy megfelelő inputadat mennyiség mellett a neurális háló a várakozások szerint képes önállóan megtalálni a komplexebb függvényformákat is (amely várakozás a későbbiekben beigazolódik). Az adatbázis két részre bontásával a korábbiakban bemutatottak szerint pedig a különböző moneynessű opciók különböző nagyságrendű áraik külön becsülhetők, így nagyobb pontosság érhető el. A bemenő input adatok közt megtalálható a lejáratig hátralévő idő, az S&P500 Index adott pillanatbeli értéke és az adott pillanatbeli VIX index értéke. A VIX index ilyen felhasználására tudomásom szerint még nem volt példa az opcióárazással foglalkozó szakirodalomban, ennek ellenére mivel a VIX index az S&P500 opcióinak implicit volatilitásaiból számolt széles körben elterjedt index (Chicago Board Options Exchange, 2015), ezért potenciálisan információt tartalmazhat a piaci volatilitás általános szintjéről, akárcsak a szimulált esetben a Heston-modellben szereplő volatilitás érték. A korábbiak szerint lehetőség lett volna az implicit volatilitások neurális hálóval való becslésére és abból származni a becsült opciók árát, azonban mivel a dolgozat nem ezen utat követi, ez tűnt a legjobb becslések nélkül kapott legjobb elérhető implicit volatilitás mutatónak. Amennyiben a VIX mégsem lenne szignifikáns magyarázó változó, a neurális hálók az adott nagyméretű adatbázis mellett képesek ezen zavaró tényezőt figyelmen kívül hagyni. Az eredmények azonban azt mutatják, hogy ezen változó bevétele után az inputváltozók közé az átlagos négyzetes hiba nagyjából felére csökkent, ezért a VIX magyarázó változóként történő felhasználása indokolt. A másik magyarázatra szoruló döntés a valamilyen kamatláb nagyság kimaradása az inputok közül, melyet a rövid lejáratig hátralévő idő (4 hónapnál kevesebb) és a kamatlábak általánosan alacsony szintje indokol, mivel ezek miatt a diszkontálás szerepe nem jelentős és így nem torzítja a végeredményt nagymértékben.

A valódi adatokat felhasználva opciók értékét megbecslő eddig említett szerzők cikkei táblázatos formában is összefoglalásra kerültek, mely még kiegészült (Kelly, 1994), (Broadie, et al., 1996), (Hanke, 1999), (Hanke, 1999), (Andreou, et al., 2008), (Saxena, 2008), (Verma, et al., 2014) és ezen dolgozat adataival:

Szerző(k)	Évszám	Alaptermék	Derivatíva	Felhasznált módszer(ek)
Malliaris, Salchenberger	1993	OEX (S&P 100)	call	MLP
Hutchinson, Lo, Poggio	1994	S&P500	future call	RBF, MLP
Kelly	1994	IBM, Chrysler,GM, Merck részvények	put	neurális háló (valószínűleg MLP)
Malliaris, Salchenberger	1996	OEX (S&P 100)	call és put	MLP
Anders, Korn, Schmitt	1996	DAX	call	MLP
Broadie et al.	1996	OEX (S&P 100)	call és put	kernel és spline függvények
Hanke	1999	DAX	call	MLP
Hanke	1999	DAX	call	MLP
Yao, Li, Tan	2000	Nikkei 225	call	MLP
Garcia, Gencay	2000	S&P500	call	MLP homogenitással
Gencay, Qi	2001	S&P500	call	MLP, regularizáció
Amilon	2001	OMX index (svéd)	call	MLP
Gencay, Salih	2003	S&P500	call	MLP, regularizáció
Dindar	2004	South African Foreign Exchange	call és put	MLP, RBF, SPO, Genetic Algorithm, CNNI, DLCN, CCN
Blynski, Faseruk	2006	OEX (S&P 100)	call	MLP
Andreou, Charalambous, Martzoukos	2008	S&P500	call	MLP
Saxena	2008	S&P CNX Nifty Index	call és put	MLP
Mitra	2012	S&P CNX Nifty Index	call	MLP
Verma, Srivastava, Das	2014	S&P CNX Nifty Index	put	klaszterezés, SVR
Szabo	2016	S&P500	call	Tree Ensemble, ELM, SVR, MLP

1. táblázat: Módszertani összefoglaló táblázat a korábban a témában publikáló szerzők műveiről, kiegészítve ezen dolgozattal.
Forrás: Saját gyűjtés

Szerző(k)	Évszám	Adatbázis információk	Valós adatbázis mérete
Malliaris, Salchenberger	1993	1990. január 1. - 1990. június 30.	?
Hutchinson, Lo, Poggio	1994	1987. január - 1991. december	~62 460
Kelly	1994	1993. október 1. - 1994. április 13. , napi záróárak	1 369
Malliaris, Salchenberger	1996	1992. január 1. - 1992. december 30., napi záróadatok	nincs információ, de az adatbázis jellege miatt nem túl nagy
Anders, Korn, Schmitt	1996	1994. január- 1994. december, Intraday	13 676
Broadie et al.	1996	1984. január 3. - 1990. március 30. , napi záróadatok	napi záróárak miatt valószínűleg közepes mennyiségű
Hanke	1999	1994. március - 1995. augusztus, Intraday	ábrák és intraday alapján valószínűleg nagyon sok, 50000+
Hanke	1999	1994. április - 1995. február, Intraday	ábrák és intraday alapján valószínűleg nagyon sok, 50000+
Yao, Li, Tan	2000	1995. január 4. - 1995. december 29.	17 790
Garcia, Gencay	2000	1987. január - 1994. október	26 770
Gencay, Qi	2001	1988. január - 1994. október	nem közli, becslésem alapján 23 000 -35 000
Amilon	2001	1997. június - 1998. március és 1998. június - 1999. március , bid-ask párok	18 832
Gencay, Salih	2003	1988. január - 1993. december	23 160
Dindar	2004	2001. január- 2003. december	?
Blynski, Faseruk	2006	1986. január - 1993. június	64 280
Andreou, Charalambous, Martzoukos	2008	1998. január - 2001. augusztus	76 401
Saxena	2008	2005. november 1. - 2007. január 25.	?
Mitra	2012	2008. július 1.- 2011. június 30.	29 724
Verma, Srivastava, Das	2014	2012. január 23. - 2014 január 8.	22 840
Szabo	2016	2016. január 23. - 2016. február 29. , 1 perces intraday	195 261

2. táblázat: Felhasznált adatbázisokat összehasonlító táblázat a korábbi kutatások alapján, kiegészítve ezen dolgozattal. Forrás: Saját gyűjtés

5) Gépi tanulás a Heston világban

A korábbi fejezetekben bemutatásra kerültek azon faktorok, amiket figyelembe véve a sztochasztika módszertanával leírt világ nem egyezik meg a valósággal, de egyre közelebb és közelebb kerül ahhoz. Ezen felül bemutatásra került számos olyan publikáció, mely szerint a nemparaméteres becslési eljárások eredményei jobban leírták a valóságot, mint az általuk választott sztochasztikából származtatott modell eredményei. Innen egyenes út vezet a következő kérdéshez: Vajon hogyan alakulna a két módszer teljesítménye, ha a valóság tényleg olyan lenne, ahogy a sztochasztikus modellek azt feltételezik? Mi lenne, ha az alaptermék árfolyamának mozgása tényleg leírható lenne végig ugyanazokkal az egyenletekkel? Akkor vajon mennyire közelítenék meg a különböző nemparaméteres becslési eljárások eredményei a valóságot? Korábban már említésre került, hogy a neurális hálók elméletben képesek megtanulni bármilyen folytonos függvényt, illetve kutatásokból tudjuk, hogy a Black-Scholes modell árazó képletét tényleg meg tudták tanulni (Hutchinson, et al., 1994).

Mindenezen bíztató eredmények mellett sem lehet elmenni az mellett, hogy a Black-Scholes modell egy elég egyszerű világban áraz, árazó képlete sem túl bonyolult, ezért a neurális hálók univerzális közelítő képessége ezzel igazából nincs letesztelve ezen a területen, mert a feladat, amit megoldottak, nem volt elég bonyolult. Ezzel szemben a korábban bemutatott Heston modell sokkal bonyolultabb, de a következőkben bemutatott empirikus teszt alátámasztja, hogy a neurális hálók ezt is különösebb problémák nélkül megtanulják.

5.1) Az alaptermék áralakulásának diszkretizálása

Első lépésben létrehozásra került az alaptermék és a volatilitás árfolyamadata, melyhez a Heston modellben szereplő differenciálegyenleteket diszkretizálni kellett, melyhez (Gong, 2014) forrást használtam fel. A folytonos és diszkrét egyenletek a következők voltak:

$$dS_t = \mu S_t dt + \sqrt{v_t} S_t dW_{1,t}$$

$$dv_t = k(\theta - v_t)dt + \sigma\sqrt{v_t} dW_{2,t}$$

$$dW_{1,t} dW_{2,t} = \rho dt$$

*** diszkretizálva és logaritmizálva***

$$\ln S_{t+\Delta t} = \ln S_t + \left(r - \frac{1}{2}v_t\right)\Delta t + \sqrt{v_t}\sqrt{\Delta t}\,\epsilon_{S,t+1}$$

$$\ln v_{t+\Delta t} = \ln v_t + \frac{1}{v_t}(k(\theta - v_t) - \frac{1}{2}\sigma^2)\Delta t + \sigma\frac{1}{\sqrt{v_t}}\sqrt{\Delta t}\,\epsilon_{v,t+1}$$

$$\rho = corr(\epsilon_{S,t+1}, \epsilon_{v,t+1})$$

$$\epsilon_{v,t+1} = \rho\,\epsilon_{S,t+1} + \sqrt{1-\rho^2}\,\epsilon_{t+1}$$

ahol ϵ_{t+1} független $\epsilon_{v,t+1}$ és $\epsilon_{S,t+1}$ változóktól

5.2) A világállapotok előállítása

A szimuláció során szintén a (Gong, 2014) forrásban található paramétereket használtam fel. Az opció lejáratáig kezdetben 50 nap volt hátra, majd minden egyes lépésben változott a világ állapota az alábbiak szerint:

- A hátralévő napok száma eggyel csökkent
- A diszkretizált Heston egyenletek alapján és egy véletlenszám generátor felhasználásával az alaptermék árfolyama vagy nőtt vagy csökkent.
- A diszkretizált Heston egyenletek alapján és egy az előbbi véletlenszám generátorral kívánt módon korreláló véletlenszám generátor felhasználásával a volatilitás szintje vagy nőtt vagy csökkent.

A fenti módszerrel az útvonalat 500-szor legenerálva összesen 51*500=25500 adatpontot kaptunk, melyek a világ egy-egy lehetséges állapotát tükrözik.

5.3) Az adott világállapotban az európai call opció értékének megállapítása

A szimuláció fő célja annak demonstrálása, hogy a neurális hálók képesek megtanulni a Heston modell árazó képletét és azt felhasználva beárazni egy olyan adatbázist, amelyet korábban még nem láttak. Ahhoz, hogy ez hitelt érdemlően bizonyítani lehessen, a szimuláció semmilyen formában nem használja fel a Heston modell zárt alakú árazó képletét, Monte Carlo szimulációkkal áraz mindent. Itt meg kell jegyezni, hogy amennyiben mégis a pontos Heston opcióérték kerülne felhasználásra, akkor is zajt kellene hozzáadni az így kiszámolt pontos

árakhoz, hogy egy valószerű szimulációt és robosztus árazóformulát adjon az adott gépi tanulásos algoritmus és ne tanuljon túl az adatokon.

Az opcióárak kiszámításához első lépésben szükség van a kockázatsemleges folyamatok dinamikájára, melyek (Gong, 2014) alapján az alábbi módon írhatók fel:

$$dlnS_t = \left(r - \frac{1}{2}v_t\right)dt + \sqrt{v_t}dW_{1,t}^*$$

$$dv_t = k^*(\theta^* - v_t)dt + \sigma\sqrt{v_t}dW_{2,t}^*$$

$$dW_{1,t}^* dW_{2,t}^* = \rho dt$$

$$k^* = k + \lambda \text{ és } \theta^* = \frac{k * \theta}{k + \lambda}$$

*** diszkretizálva és logaritmizálva ***

$$\ln S_{t+\Delta t} = \ln S_t + \left(r - \frac{1}{2}v_t\right)\Delta t + \sqrt{v_t}\sqrt{\Delta t}\in_{S,t+1}$$

$$\ln v_{t+\Delta t} = \ln v_t + \frac{1}{v_t}(k^*(\theta^* - v_t) - \frac{1}{2}\sigma^2)\Delta t + \sigma\frac{1}{\sqrt{v_t}}\sqrt{\Delta t}\in_{v,t+1}$$

$$\rho = corr(\in_{S,t+1}, \in_{v,t+1})$$

$$\in_{v,t+1} = \rho * \in_{S,t+1} + \sqrt{1-\rho^2}\in_{t+1}$$

ahol $\in_{t+1}$ független $\in_{v,t+1}$ és $\in_{S,t+1}$ változóktól

A kockázatsemleges valószínűség felhasználásával mind a 25500 pontban Monte Carlo szimuláció segítségével az opció beárazásra került (melyből a t=0 eset nyilván nem annyira izgalmas), minden egyes alkalommal félmillió szimulációt végezve és az antitetikus változók módszerét felhasználva, hogy ezzel a végeredmény szórása még csökkentjen. A félmilliós szám mögött meghúzódó alapelv elsősorban az volt, hogy fontos a lehető legkisebb szórású végeredmény elérése, de közben a számítási idő maradjon egy elfogadható tartományban. A kettő szempont végül a félmilliós választást eredményezte.

A félmillió szimuláció erre a célra tervezett számítógép hiányában[2] önmagában még nem lenne sok, de 25500-szor azt elvégezve, egyenként átlagosan körülbelül 25 napnyi útvonalat legenerálva már önmagában kihívást jelentő feladat. A probléma feloldását az jelentette, hogy Python2.7 programnyelven a fenti probléma a numpy könyvtár felhasználásával vektorizálva lett implementálva, így a félmillió útvonal leszimulálása rendkívül felgyorsítható. Mindezen gyorsítás eredményeként egy pont árazása nagyjából 5 másodperc alatt végezhető el, a teljes adatbázis pedig nagyjából 40 óra alatt árazható be. Ez sok időnek tűnhet, mely a félmilliós paraméter választás eredménye, de mivel ezt csak egyszer kell lefuttatni az adatbázis létrehozásához, így a kisebb hiba megéri a hosszúra nyúló futási időt.

A Monte-Carlo szimuláció esetén az opció értékének becslési hibája a hátralévő napok számától nagyban függött, 50 nap múlva lejáró opciók esetén a becsült opcióár becsült varianciája $1{,}9*10^{-6}$, még 1 nap múlva lejáró opciók esetében $5{,}04*10^{-8}$ volt.

5.4) Az adatbázis szétválasztása tanító, validációs és teszt adatbázisra

Ahogy az a gépi tanulás alapelveinél bemutatásra került, a kapott adatbázist szét kell választani három részre. A szimuláció során ez az arány [0.4 : 0.4 : 0.2] volt, mely az általános szokással ellentétben nem véletlenszerűen lett kiválasztva, hanem a különböző útvonalak különböző csoportokba kerültek, biztosítva azt, hogy a validációs halmazba ne kerüljön olyan adatpont, amelynek egy későbbi értéke a tanító halmazban van, elkerülendő a későbbi információ felhasználását a becslésben. Ugyanígy a teszt adatbázisban csak olyan pontok szerepelnek, melyeknek későbbi értékei nem szerepelnek sem a tanító, sem a validációs halmazban.

A gépi tanulásos fejezetben említettek szerint az adatokat ezután a [0;1] intervallumba kellett transzformálni 0,5 várható értékkel. Ez relatíve könnyen megtehető, amennyiben az adatbázis eloszlása ismert, azonban jelenleg mindössze a tanító adatbázis eloszlása alapján kell ezt a feladatot véghezvinni. Még a 0,5 várható értékűvé transzformálás általában nem probléma, mert ha nem pontosan 0,5 lesz a várható érték a validációs halmazon, az nem akadályozza a becslési eljárást, a [0;1] intervallumból való kilógás különösen a célváltozó (jelenleg az opció ára) esetében komoly gondot jelent. A probléma úgy orvosolható, hogy a tanító adatbázis az eredetileg [0;1] intervallumba kódolása helyett a [0,05 ;0.95] intervallumba lesz kódolva egy biztonsági faktor segítségével, így a tanító adatbázis maximumánál nagyobb vagy minimumánál kisebb adatpontok még mindig beleférhetnek a [0;1] intervallumba. A másik

[2] A számításhoz használt számítógép egy TOSHIBA L50-B-1MC

módszer, hogy azon validációs és teszt halmazba eső pontok, melyek ilyen szélsőértéket tartalmaznak, outlierként eltávolításra kerülnek és ezekben az esetekben a gépi tanulásos eszközök használata elmarad. Fontos megemlíteni, hogy ez csak akkor követhető stratégia, ha a problémás értékek nem a célváltozóban merülnek fel, hiszen ezek értékét a tanító adatbázis elemzésekor még nem ismert. A szimuláció során az intervallum összenyomásának módszere került felhasználásra, mivel így nem volt szükség egyetlen adatpont eltávolítására sem.

A transzformáció során az alábbi lépések kerültek végrehajtásra minden változó (alaptermék árfolyama, volatilitás szintje, hátralévő idő és az opció Monte Carlo alapján becsült ára) esetében külön-külön.

1. A változó értékéből kivonásra került a változó tanító adatbázison kiszámolt átlaga, így 0 várható értékűvé téve azt ezen adatbázison.

2. Kiszámolásra került ezután a 0-tól való eltérés maximuma (a legnagyobb abszolút érték), melyet később még egy biztonsági faktorral megszorozva kialakult a szám, amivel a változó értékeket elosztva azok minden esetben a [-0,5;0,5] intervallumba estek. (a validációs és tesztmintában is).

3. Ezek után 0,5-öt hozzáadva az értékekhez az adatok a [0;1] intervallumba estek 0,5 várható értékkel.

A transzformált opcióár varianciája a tesztmintán 0,00577 volt, tehát ha ismert lenne pontosan ezen változó átlaga, akkor ezzel becsülve ekkora átlagos négyzetes hibát kapnánk. Ezt figyelembe véve érdemes a gépi tanulásos módszerek eredményét tekinteni.

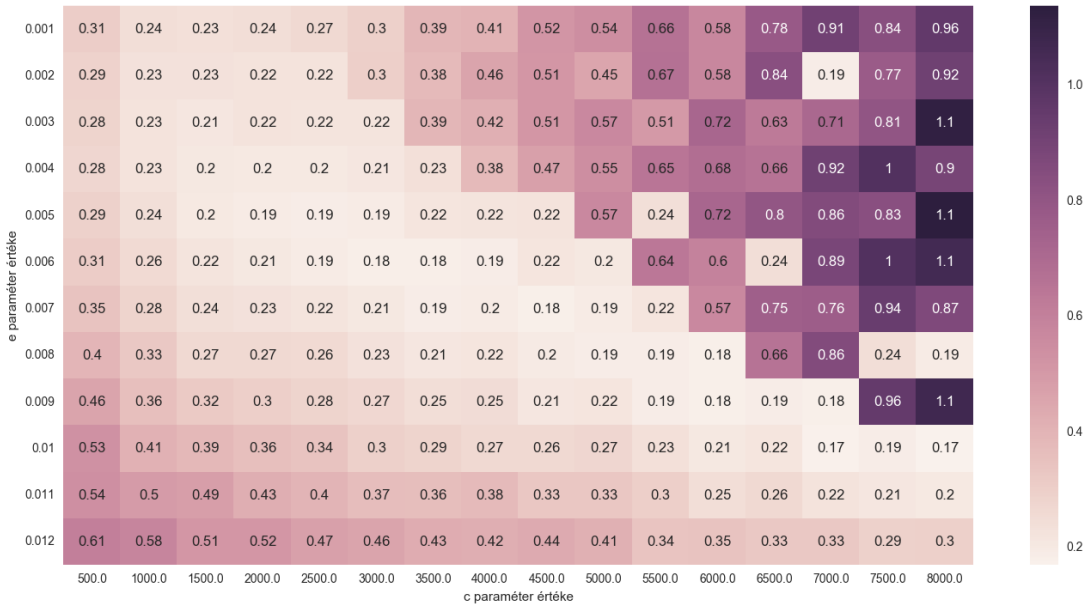
5.5) Az SVRek, ELMek és TE-k tanítása

A kapott tanító adatbázis alapján került sor a négy, második fejezetben taglalt, gépi tanulásos módszer tanítására, melynek a túltanulását a validációs minta akadályozta meg minden esetben. A négy módszer közül a korábban is említettek szerint az egyrétegű perceptron teljesített legjobban, azonban a leírást a másik három módszer eredménye vezeti, hogy így adjanak összehasonlítási alapot.

Az első két módszer abban a technikai szempontban hasonlít, hogy mindkét esetben kettő paramétert kell optimalizálni, amit általában a paraméterpárok számos kombinációját kipróbálva és a validációs halmazon a legjobb eredményt adó megtartásával történik. Kipróbálásra kerültek tehát a különböző paraméterpárok és az általuk a transzformált validációs

adatbázison számolt négyzetes hibaösszeg hőtérkép formájában megjelenítésre került. Fontos megjegyezni, hogy az eredeti és a transzformált adatbázis egy konstans hozzáadásával és egy konstansszoros szorzással alakítható át egymásba, ezért a teljesítmény bármelyiken mérhető, számítási könnyebbség miatt azonban érdemesebb a transzformáltat választani.

Az SVR általában nagy számítási igény mellett megbízhatóan jól teljesít[3], a két paraméterét (e és C) széles skálán végigfuttatva („grid search") a validációs adatbázison, amely megnevezés az átskálázott validációs adatbázisra utal a következőkben[4], az alábbi eredmény született:



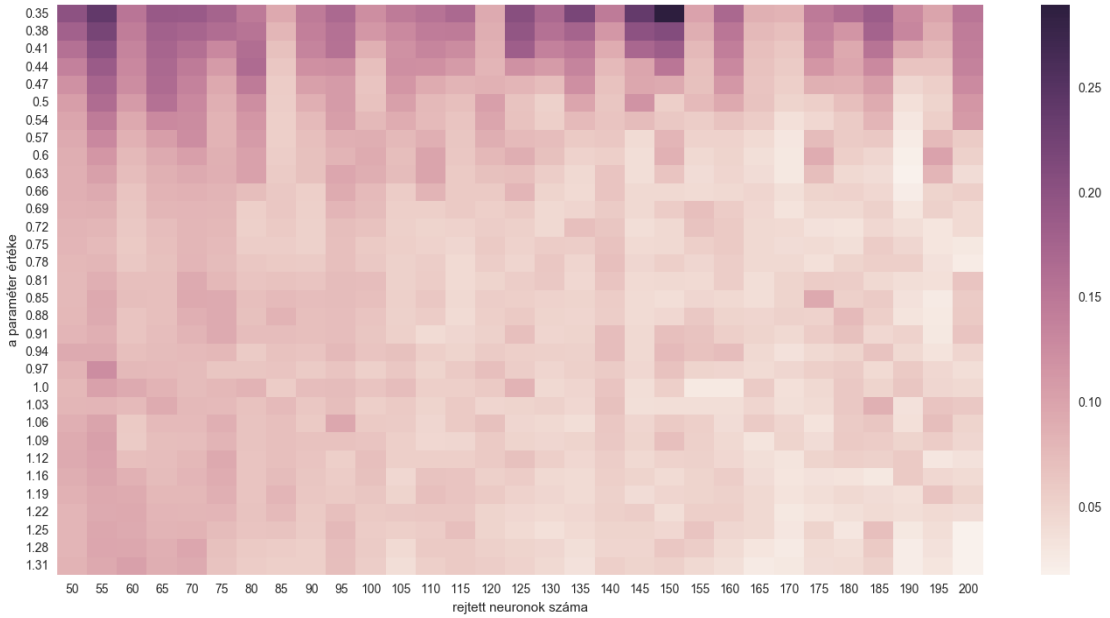
2. ábra: A 10 000 validációs adaton mért négyzetes hibaösszeg különböző C és e paraméterek mellett.
Forrás: Saját számítás

Amint az az ábrán látható, a legjobb eredményt e=0.01 és C=8000 környéki beállításokkal érte el, melynél a 10 000 transzformált validációs adat együttes négyzetes hibája 0,17 volt, amely nem kizárt, hogy a C és e növelését folytató stratégia hatására esetleg ennél lejjebb csúszna. Ezen eredmény elérésére adott C és e paraméter mellett 87 másodpercre volt szükség. Érdekes megfigyelés azonban, hogy amennyiben olyan beálltást választunk, amely valamelyest eltér ettől és gyengébb eredményt ad vissza, például (e=0.007, C=8000) pontot, akkor a futási idő már 1200 másodpercre változik. Ezek alapján erősen kétségbe vonható, hogy a gyakorlatban ennél nagyobb felbontású és így pontosabb C és e beállításokkal rendelkező hőtérkép

[3] Kernelfüggvényként az alapértelmezett RBF kernel került beállításra
[4] a skálázási arány 24,05 volt

elkészíthető lenne praktikus időn belül. Ezen megfontolások alapján az SVR módszerrel kapható validációs hiba minimumát legalább 0,1-re teszem.

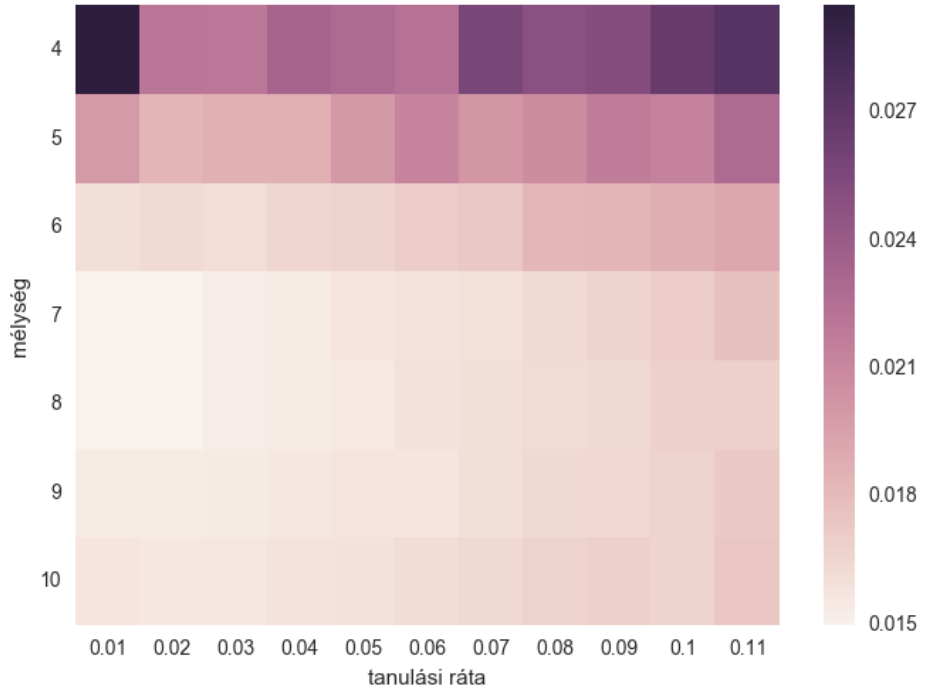
Az ELMekre a korábban említett véletlenszerűség miatt a hőtérkép egyáltalán nem lenne ilyen egyenletes, ezért a vizsgált paraméterpárok esetében mindig 10 véletlenszerű indítás közül a legjobb eredményeit tartalmazza a. hőtérkép, mely még így is erősen véletlenszerűnek hat. Ugyanakkor egyértelműen kisebb hibákkal dolgozik átlagosan, mint az SVR, ami egy elég érdekes felfedezés. A 10 000 validációs adatponton elért legkisebb együttes hibák négyzetösszege mindössze 0,01822 volt egyetlen ELM-t felhasználva. Ennél sokkal meglepőbb, hogy az egész paraméterkombináció kiszámolásához mindössze 4312 másodpercre volt szüksége, ami jóval rövidebb, mint a MLP vagy az SVR esetében. További érdekesség, hogy még az SVR esetén a legjobb hálózat létrehozásához 87 másodpercre volt szükség, a rosszabbakéhoz pedig nagyságrendileg több idő, az ELM minden egyes megjelenített (legjobb a tízből) hálózatot másfél másodpercnél kevesebb idő alatt határozott meg. A jövő kutatási kérdései közé tartozik, de úgy gondolom, hogy a későbbiekben bemutatott több neurális hálózatot felhasználó becsléssel és a paraméterek sűrűbb vizsgálatával 0,01 körüli négyzetes hibaösszeg is talán elérhető lenne.



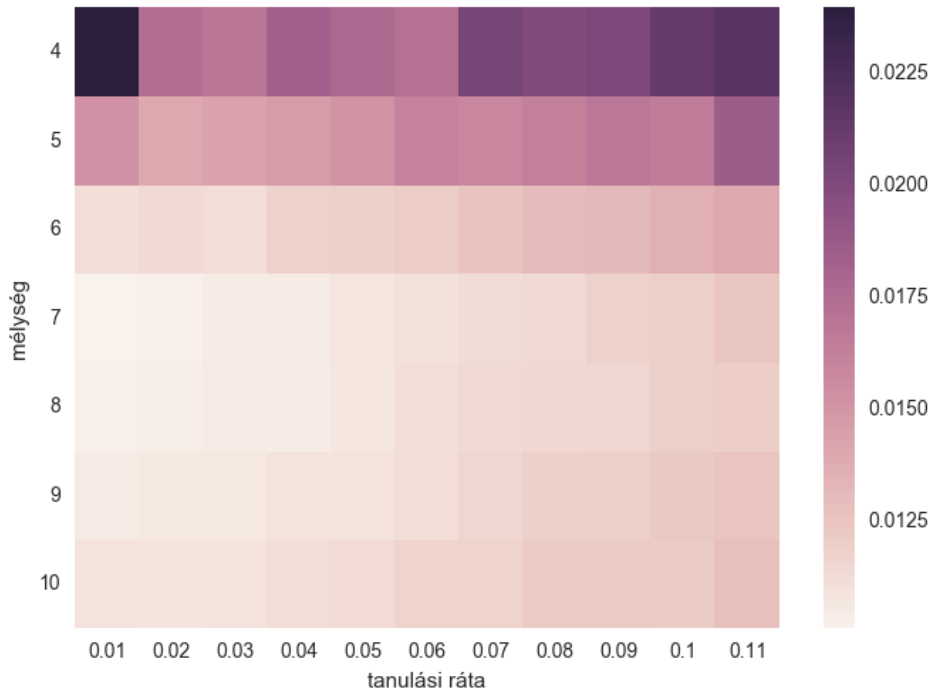
3. ábra: A validációs mintán mért négyzetes hibaösszeg
a rejtett neuronok számának és az a paraméter függvényében.
Forrás: Saját számítás

A Tree Ensemble módszerben az említettek szerint 3 paraméter került optimalizálásra. A döntési fa együttes módszer nagy előnye, hogy a szimulált adatbázison eredménye robosztus,

azon paraméterek mellett, amelyekkel a validációs mintán jó eredményt mutatott, megbízhatóan jó eredményeket adott vissza. A 10 legjobb validációs hibát produkáló modell esetében a teszt adatbázison mért négyzetes hibaösszegek szintén közel a legkisebbek voltak a teszt adatbázison különböző paraméterek mellett mért négyzetes hibaösszegek közül. Ez nem jelenti azt, hogy ezen tulajdonsága valódi adatok mellett is megmarad.



4. ábra: A validációs adatokon mért átlagos négyzetes hiba 10 000-szerese a tanulási ráta és a mélység paraméterek függvényében a legjobb estimator beállítást használva
Forrás: Saját számítás



5. ábra: A teszt adatokon mért négyzetes hibaösszeg a tanulási ráta és a mélység paraméterek függvényében a legjobb estimator beállítást használva
Forrás: Saját számítás

A technikai részleteket illetően az SVR-ek a Python 2.7 scikit-learn.svm.SVR csomagjával, a Tree Enseble-k a python 2.7 XGBoost csomagjával kerültek kiszámításra, még az ELM-ek kiszámítása a http://www.ntu.edu.sg/home/egbhuang/elm_codes.html ELM kutatói honlapról származó, David Lambert által létrehozott Python 2.7 verzióra írt programkódjával történt.

5.6) A MLP-k tanítása

A MLP rendkívül nagy sikernek örvend az utóbbi időben, ennek köszönhetően számos implementáció született, így a dolgozat elkészítésekor az egyik legnagyobb kihívás ezek közül a legjobb megtalálása volt. Az alapvető dilemma a következő: A kutató vagy egyedül megírja a saját programcsomagját, mellyel számol, mely hihetetlenül nagy időráfordítást igényel, különös tekintettel amennyiben a sebesség is fontos számára, márpedig itt nagyon fontos, vagy választ egyet a már megírt programcsomagok közül. Utóbbi esetben teljesen kiszolgáltatott a csomagban szereplő lassú számítás, egyéb kódbeli vagy implementációs hibának, melynek végeredményeképp kutatásának eredménye valamilyen mértékig számára is kétséges.

Ezen tanulmány elkészítésekor a középutat választottam. A Python 2.7 numpy könyvtárában megírásra került egy egyrétegű perceptron kódja vektorizálva a megfelelő back-propagation tanító algoritmussal együtt szintén vektorizálva, mely így kimagaslóan gyors volt és az általa

adott eredmények nem voltak kétségesek. A back-propagation legnagyobb problémája alapvetően, hogy nála sokkal gyorsabb tanító algoritmusok is vannak, de előnye, hogy nem olyan nehezen implementálható. Ezen hálózat segítségével a megfelelő program már megtalálható volt, mindössze ellenőrizni kellett, hogy rendelkezik-e az alábbi kritériumokkal:

1. Python 2.7 által támogatott
2. Implementálva van benne a fejlettebb tanítási algoritmusok minél nagyobb köre
3. Sebességre és pontosságra a back-propagation tanítási beállítás mellett az eredetileg megírt pontosságot és nem sokkal lassabb sebességet produkálja.
4. Támogatja a modern neurális hálókat, melyek a rejtett rétegekben is tartalmaznak konstans hozzáadási lehetőséget, mely régen nem számított bevett eljárásnak.

A fenti szempontok alapján a Python 2.7 NeuPy csomagja bizonyult a legjobbnak. A tanítóalgoritmusok tekintetében érdekes megfigyelés, hogy a gyakran használt Levenberg-Marquardt tanító algoritmus teljesítményét messze felülmúlta a Quasi-Newton, melyekről részletesebben ezen dolgozatban nem esik szó, de fontos technikai részlet, így megemlítésre került.

A MLP-ok esetében a legfontosabb meghatározandó paraméter a rejtett neuronok száma, minél nagyobb ezen szám, a rendszer annál komplexebb függvények közelítésére képes, viszont így sokkal nagyobb az esély, hogy a rendszer az általánosítás helyett inkább a tanítómintát közelíti túlságosan, így egyéb, addig nem látott esetekben rontva a becslés pontosságát. A MLP tanítása során a tanító algoritmus egyetlen iterációja során úgy módosítja a súlyokat, hogy azok kisebb hibát adjanak az addiginál a tanítóminta esetében, mellyel együtt egy ideig a validációs adatbázis hibája is csökken, egy idő után azonban az algoritmus egyre inkább a tanítóminta sajátosságaira koncentrál, mely a validációs hiba növekedéséhez vezet. Az tekinthető tehát adott paraméter esetén a legjobb hálózatnak, amelyen a validációs hiba minimuma a legkisebb. A valóságban ezen probléma sokkal jelentősebb és korábbi iterációknál bekövetkezik, mint a szimuláció során, hiszen itt az adtok a Monte Carlo szimulációból származó véletlen hibát tartalmazzák csak, miközben a valóságban ennél kevésbé egységes adatbázis adott.

A korábbiakban említettek szerint adott rejtett neuron szám mellett is lényeges szerepe van a véletlennek a létrejött háló teljesítményében, ezért többféle véletlen beállítás mellett is elvégezve a számításokat az összes ilyen hálózat elmentésre került. Az adatokból az látszik,

hogy ahogyan 1 neurontól eljutunk a 4 neuronig a legjobb neurális hálók hibaátlaga meredeken csökken, majd egészen 10 rejtett neuronig a hiba tovább csökken, innentől pedig a neuronok növelése során a tanításhoz szükséges idő érezhető módon növekszik, de jelentős hatékonyságnövelés nem társul hozzá. A 10 rejtett neuront tartalmazó legjobb 5 hálózat transzformált validációs adatbázison mért négyzetes hibaösszege 0,008255 volt, mellyel felülmúlta a másik két módszer eredményét.

A neurális hálók esetében azonban jó ötlet lehet nem csak a legjobb hálókkal becsülni, hanem a korábban már hivatkozott (Crone, et al., 2011) szerint a legjobb valahány átlagát használni, mellyel így az eredmény tovább javítható. Ezen hipotézis a szimulált adatbázison letesztelésre került és tényleg működőképes, a legjobb n neurális háló transzformált validációs adatbázison mért négyzetes hibaösszege a következőképp alakult:

Felhasznált legjobb neurális hálók száma	Négyzetes hibaösszeg
1	0,006897
2	0,006525
3	0,006877
4	0,006122
5	0,005941
6	0,006243
7	0,006397

3. táblázat: A validációs adatbázison mért négyzetes hibaösszeg csökkenése a becsléshez felhasznált legjobb MLP-k számának növekedésével.
Forrás: Saját számítás

Összességében tehát a transzformált validációs adatbázison elért legjobb eredmény egyrétegű MLP esetében 0,006 volt, mellyel felülmúlta a másik két módszer eredményét.

5.7) A módszerek kiértékelése a transzformált teszt adatbázison

A validációs minta alapján a legjobb modellek végső próbájaként segítségükkel megbecslésre került az addig még a programmal soha nem érintkező transzformált teszt adatbázis célváltozója a transzformált teszt adatbázis inputváltozói alapján, mely alapján az alábbi eredmények születtek:

Becslési eljárás	Négyzetes hibaösszeg a szimulált teszt adatbázison
Ismert átlaggal becslés	57,7
naív Black-Scholes	6,7
SVR	0,1286
ELM	0,0117
Tree Ensemble	0,0103
MLP	0,0039

4. táblázat: Az egyes módszerek teljesítménye a négyzetes hibaösszeg alapján, ennek 10000-ed része az átlagos négyzetes hiba.
Forrás: Saját számítás

A táblázatban ismert átlaggal való becslés azt jelenti, hogy ha pontosan ismernénk a transzformált teszt adatbázison az opcióár átlagát, akkor minden opcióértéket azzal becsülve 57,7 lenne a négyzetes hibaösszeg. Természetesen ez nem ismert, így ez egyfajta alsó becslése az ilyen módon kapható hibaösszegnek. A naív Black-Scholes sor azt jelenti, hogy amennyiben a Heston világban a Black-Scholes képlet alapján kerül megbecslésre az opciók ára a volatilitás adott időpontbéli szintjét, mint konstans volatilitást használva, akkor a négyzetes hibaösszeg 6,7 lesz a transzformált teszt adatbázison. Fontos kiemelni, hogy ez a Heston-modell általam megválasztott paraméterei esetén lett kiszámolva, más paraméterek valószínűleg nagyban más eredményt adnak ezen sorban.

A táblázatot elemezve és a becslő algoritmusokat összehasonlítva az látszik, hogy amennyiben az adatok mindhárom mintacsoportban nagyon hasonlóak, úgy az SVR teljesítménye lényegesen lemarad az ELM-től és a Tree Ensemble-től, aminél azonban az MLP még sokkal pontosabban becsüli az opció értékét. A valóságban azonban a tanító, a validációs és teszt mintában az adatok lehetnek különböző eloszlásúak, lehetnek az adatsorban strukturális törések, ezért nem érdemes túlságosan általános következtetéseket levonni ezen eredményekből.

6) Az adatok

Az alaptermék tekintetében számos termék szóba jöhet, alapvetően az adatok historikus beszerezhetősége és a likviditás a legfontosabb, hogy minél több kereskedett opció adata elérhető legyen. A korábbi fejezetekben már bemutatásra került számos tanulmány, amik ezen tulajdonságokat szem előtt tartva az S&P500 indexet választották alapterméknek. További előnye, hogy a VIX index a várakozás szerint információval szolgálhat a volatilitás aktuális szintjéről, melyet ezen alaptermék alapján határoznak meg.

Fontos megemlíteni, hogy a VIX index számos S&P500 opció implicit volatilitásából kerül kiszámításra, ezért amennyiben egy opció árának meghatározásakor figyelembe vesszük azt, akkor figyelembe veszünk annyi extra információt, amennyivel az adott opció árából számolt implicit volatilitás módosítaná a VIX index értékét. Ez két okból nem probléma: Először is ez a módosító hatás nagyon kicsi, így nem nyújt érdemi információt. Másrészt a becslőfüggvény létrehozásának célja általában nem a piacon már megfigyelt áradatok kitalálása, hanem esetenként a piacon nem megfigyelhető, nem likvid opciók aktuális árának becslése vagy relatív alul és felülárazások megtalálása. Ilyen irányból nézve a VIX index ezen problémája egyáltalán nem fontos.

6.1) Opciós áradatok

A korábban látott módszerek és eredmények empirikus teszteléséhez szükséges egy alaptermék és egy időintervallum megválasztására. Az idősáv megválasztása esetén igazából egyetlen érdemi döntés meghozására van szükség: napon belüli magas frekvenciájú idősor vagy nap napvégi záróárak alapján készüljön-e el az elemzés. Amennyiben ezen döntés meghozásra került, a Bloomberg historikus adattároló időintervalluma már megszabja, mi az a legtágabb időszak, amin az elemzés elvégezhető. Napon belüli adatok esetén ezen időszak jóval rövidebb, ugyanakkor naponta sok adat van eltárolva, mely bőven kompenzál az elvesztett napokért. Természetesen az adatok más helyről való beszerzése is elképzelhető.

Az ezen írásban felhasznált adatbázis 2016. január 23. és február 29. közti időszakban az egy perces napon belüli adatból állt össze, melyek magukba foglalták az eladott opciók árát (1 perces intervallumban a nyitóár, ami gyakran egyetlen opció ára), az egy perces intervallumra lebontott S&P500 historikus árfolyamát (nyitóár) és a VIX index egy perces intervallumokra

lebontott alakulását (szintén nyitóár). A nyitóár választása teljesen önkényes, természetesen záróárak felhasználása érdemben nem befolyásolná az eredményeket.

6.2) Az adatbázis letöltése

Az adatok letöltése a Bloomberg beépített excel függvényei, a Visual Basic és a Python 2.7 segítségével valósult meg.

Első lépésként Visual Basic segítségével előállítása került egy mintamunkafüzet, mely az adott napon lejáró alaptermékek tickerjeit elkészítette a lejárati dátum és sokféle kötési árfolyam segítségével. A kapott ticker listára a Bloomberg beépített BDH() függvényét meghívta, így építve az adott alaptermékre vonatkozó minden adott napon lejáró opcióra adatbázist a 2016. január 23. és február 29. közt történt tranzakcióról az adott excel file-on belül.

Második lépésként a python 2.7 datetime dátumokat kezelő könyvtárának segítségével legenerálásra kerültek minden péntek és minden hónap utolsó napjának dátumjai olyan formában, ahogy azt a minta excel file befogadni képes. Ezen időpontok nem mindegyikében jártak le opciók, mindazonáltal a Bloombergben szereplő megfigyelt opciólejáratok vagy péntekre vagy hó végére estek, ezért inkább minden ilyen dátum végignézésre került programmal a maximális adatbázisméret érdekében.

Harmadik lépésben a Python 2.7 openpyxl excel kezelő csomagja segítségével a minta excel file másolásra és módosításra került a dátumok adatbázisa alapján, minden dátum alapján kialakítva egy minta excel file formátumával megegyező. attól csak a lejárati dátumban eltérő (és így az összes arra a lejárati napra szóló tickert tartalmazó) excel filet. Ezután az összes file Bloomberg kapcsolattal megnyitásra került, így a beépített BDH() függvény segítségével az összes szükséges adat az munkafüzetekbe került.

Negyedik lépésben a Python 2.7 programnyelvben megírt kód segítségével az excel munkafüzetekből a szükséges adatok nyers formájukban kinyerésre kerültek és a különböző excel filekból szerzett adatbázisok egyesítésre kerültek, melyek így a következő formátumú adatbázist alkották:

Kötési időpont	Ticker	Ár
Kötési időpont	Ticker	Ár
….	…	…

Egy-egy munkafüzet és a Bloomberg adatletöltés varázslója segítségével a 2016. január 23. és február 29. közti időszakra 1 perces sűrűséggel letöltésre kerültek az SPX és VIX adatok is, melyek szintén Python 2.7 programban megírt feldolgozóprogram segítségével a következő formátumban került elmentésre:

Időpont 1	SPX árfolyam 1
Időpont 2	SPX árfolyam 2
….	…

Időpont 1	VIX árfolyam 1
Időpont 2	VIX árfolyam 2
….	…

Összesen 204 287 darab opciós áradatot sikerült így lementeni.

6.3) Az adatok tisztítása

Következő lépésben a három adatbázis az időpontot mint azonosítót felhasználva egyesítésre került, mely során 3 872 darab opciós áradatnak nem sikerül vagy SPX vagy VIX értéket találni. Mivel ezen adatok nem képezik jelentős részét az adatbázisnak és egyéb módszertani hibát sem idéz elő kihagyásuk, ezért ezen adatok az adatbázisból eltávolításra kerültek. Így a továbbiakban 200 415 darab adatpont maradt, melyek immáron tartalmazzák az adott időpontban a megfelelő SPX értéket és VIX szintet.

Az adatbázis már majdnem használható formába került ezzel, utolsó lépésként az opciók tickerjei alapján kiszámításra került annak kötési árfolyama, illetve lejárati dátuma, mely segítségével a lejáratig hátralévő idő is meghatározásra került, így az a következő formát öltötte:

Időpont 1.	Opcióár 1.	SPX árfolyam 1.	VIX 1.	Kötési árfolyam 1.	Idő lejáratig 1.
Időpont 2.	Opcióár 2.	SPX árfolyam 2.	VIX 2.	Kötési árfolyam 2.	Idő lejáratig 2.
…	…	…	…	…	…

Az első oszlopban szereplő időpontok később a tanító, validációs és teszt mintára való szétválasztáskor hasznosak, ezen felül további szerepük nincs, a többi oszlopban szereplő adatok viszont a gépi tanulásos módszerek számára fontosak.

Néhány adat eloszlása rendkívül nyugtalanító, hiszen minden adatot a [0;1] intervallumba szükséges hozni, így a nagyon ferde eloszlások nem szerencsések, hiszen azok az adatbázis

érdemi részét egymáshoz nagyon közelre transzformálják. Az inputváltozók esetében az intervallumból való kilógás nem feltétlenül nagy probléma, de a célváltozó (itt az opció ára) esetében ez elfogadhatatlan, hiszen a modell kimenetele a [0,1] intervallumba esik, így ezen értékeket nem képes visszaadni. A jelen adatbázisban pedig pont ezen változóval van az egyik legnagyobb probléma. Fontos kiemelni a szimulációs adatbázisnál már leírtakat, miszerint az opció árának nagysága alapján nem szűrhetünk, hiszen ezen értékeket csak a tanítóminta esetében ismerhetjük. A probléma megoldásához választott kritériumok, amelyek esetén az opció továbbra is az adatbázisban maradt, a következők voltak:

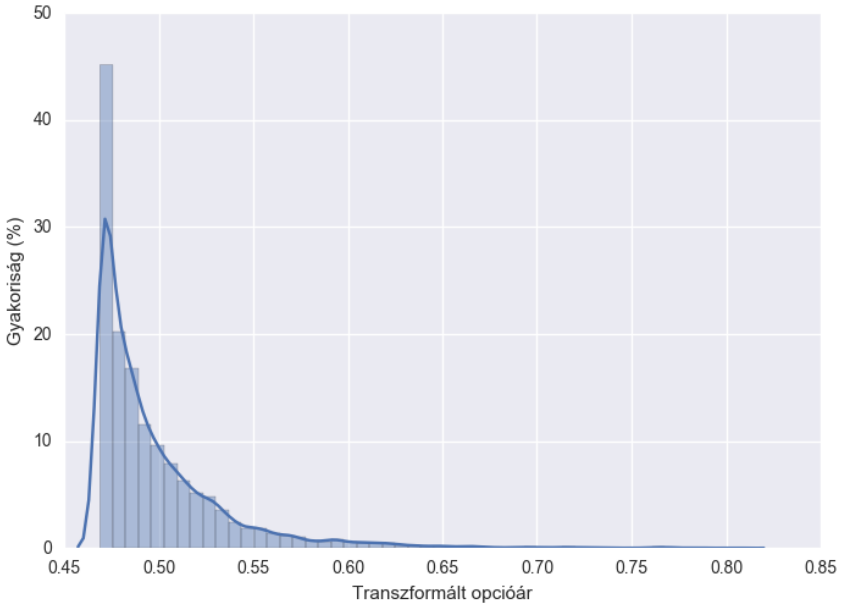
1. 1,1 –nél kisebb S/K arány (továbbiakban moneyness)
2. 0,33 évnél kisebb a lejáratig hátralévő idő (4 hónap)

A fenti kritériumokat használva, illetve az adatbázis több moneyness kategóriába darabolva a probléma nagysága radikálisan csökkenthető, mivel ezek után az adatok az intervallumon sokkal jobban szóródnak, ahogy azt korábbi kutatások is javasolják (Malliaris & Salchenberger, 1993). Fontos kiemelni, hogy az adatok nem valamiféle likviditási vagy egyéb abszolút kritérium alapján kerültek elhagyásra, ahogyan azt számos szerzőnél a korábbi kutatásoknál olvasni lehet. Az ok az, hogy egészen egyszerűen ebben az adatbázisban másféle adatok vannak, így nem illeszkednek. Amennyiben több adat lenne a fenti két tartományban, úgy ezek gond nélkül szerepelhetnének a tisztított adatbázisban.

Az adatbázis ezek után két adatbázisra lett szétbontva moneyness alapján, az „alacsony” adatbázisba a 0,97 vagy az alatti moneyness esetén kerültek opciók és 85 689 adatot tartalmaz. A „magas” adatbázisba 0,97 feletti, de 1,1 alatti moneyness esetén kerültek, mely csoport 109 572 adatpontból áll. (A két adatbázis a következőkben magas és alacsony néven kerül elkülönítésre.)

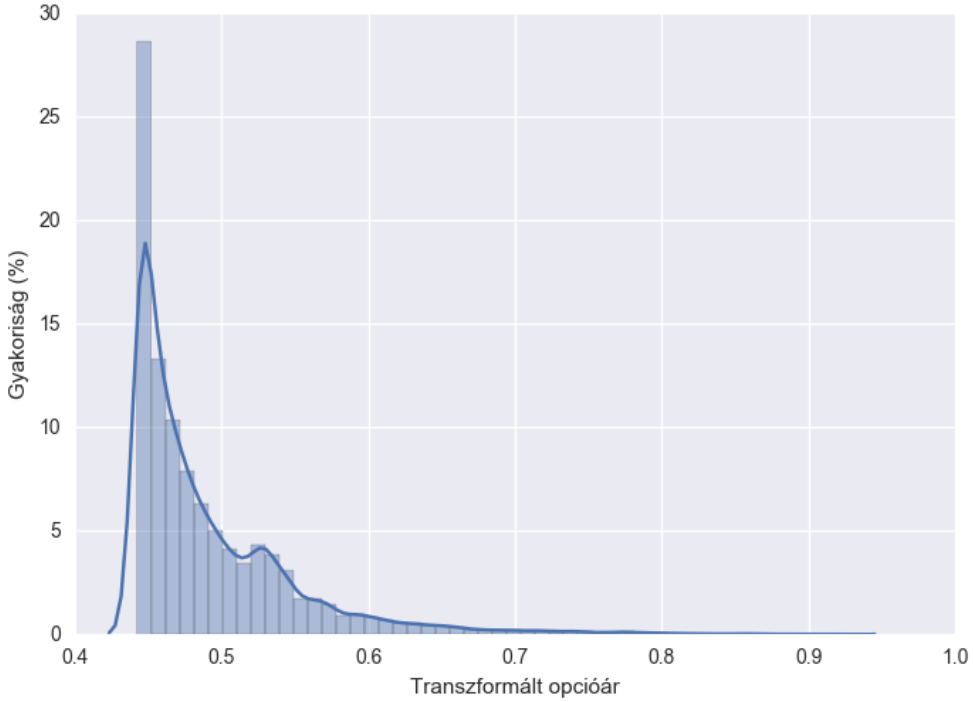
7) Teljesítmény piaci adatokon

A tanítási folyamat első lépésében az adatbázist három részre kellett bontani, melynek első fázisában leválasztásra kerültek azon adatok, amelyek 2016. február 18. vagy az után keletkeztek (18 579 adat az alacsony, 33 886 adat a magas adatbázis esetén), ezek képezték a teszt adatbázist. A tanító és validációs adatbázist a 2016. február 18. előtti adatok alkották, így meggátolva, hogy a teszt kiértékelése során olyan információra támaszkodjon véletlenül az algoritmus, ami akkor még nem állt rendelkezésre. Optimális esetben a tanító és validáló adatbázist is egy dátum mentén lehetne elválasztani, azonban a gyakorlatban a tanító adatbázis adatai kisebb intervallumban mozogtak, mint a validációs adatbázis adatai. Mivel ez megakadályozta, hogy a [0,1] intervallumba történő beskálázás sikeres lehessen a tanító halmazban lévő célváltozó varianciájának nagymértékű csökkenése nélkül, ezért inkább ezek összekeverésre kerültek, majd az adatok 70 százaléka véletlenszerűen a tanítómintába került. Alapvetően amennyiben az alapfeltevés az, hogy az árazó képlet az idő folyamán nem változik, akkor természetesen ez a keverés nem probléma, jelen esetben pedig ez áll fent. Az adatok [0,1] skálába transzformálása és 0,5 várható értékűvé tétele után pedig minden lehetőség adott a gépi tanulásos módszerek felhasználásához.



6. ábra: A transzformált alacsony tesztadatbázis célváltozója, az opció ára[5].
Forrás: Saját számítás

[5] Variancia: 0,00141



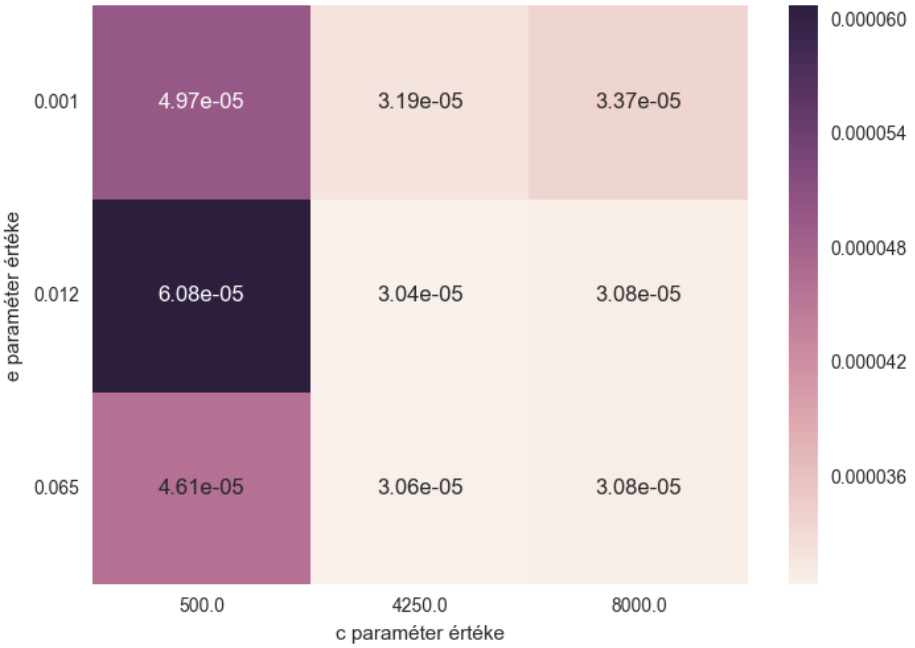
7. ábra: A transzformált magas tesztadatbázis célváltozója, az opció ára[6].
Forrás: Saját számítás

7.1) SVR piaci adatokon

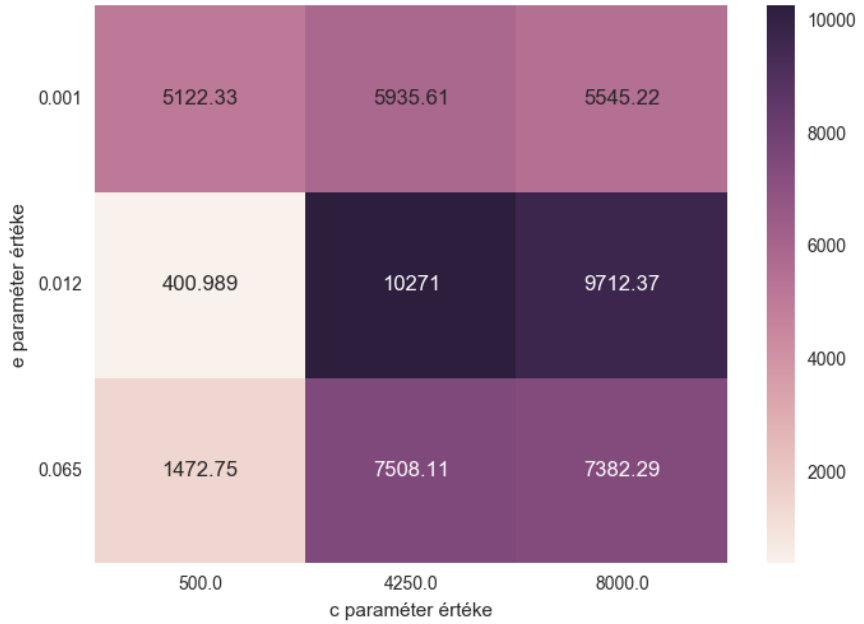
Az SVR tanítása számos meglepetést hozott a piaci adatokon a szimulált környezethez képest. Az első és leglátványosabb ilyen a tanítási idő volt, ugyanis ez rendkívül hosszúra nyúlt, így gyakorlati szempontból mindössze egy 3x3-as négyzet átnézése volt kivitelezhető.

Az alacsony adatbázis esetében a legjobb teljesítményt hozó SVR 10270 másodperc alatt készült el (C=4250, e=0.012) beállítás mellett, mellyel a leghosszabban futóvá vált, ami éppen az ellenkezője annak, ami a szimulált adatbázison volt tapasztalható. Ez jó eséllyel nem a jó vagy rossz illeszkedésnek tudható be, hanem annak, hogy milyen paraméterek melletti SVR adta a legjobb modellt. Ugyanakkor az SVR-ek továbbra is megbízhatóan általánosítanak, a tanító és a validációs minta hibája együtt mozgott, a teszt hiba sem tért el ezektől nagymértékben. A validációs hiba szerint a legjobb becslés a teszthiba szerint is a legjobb becslés lett, a teszt adatbázison mért átlagos négyzetes hibája $2{,}516709*10^{-5}$ lett.

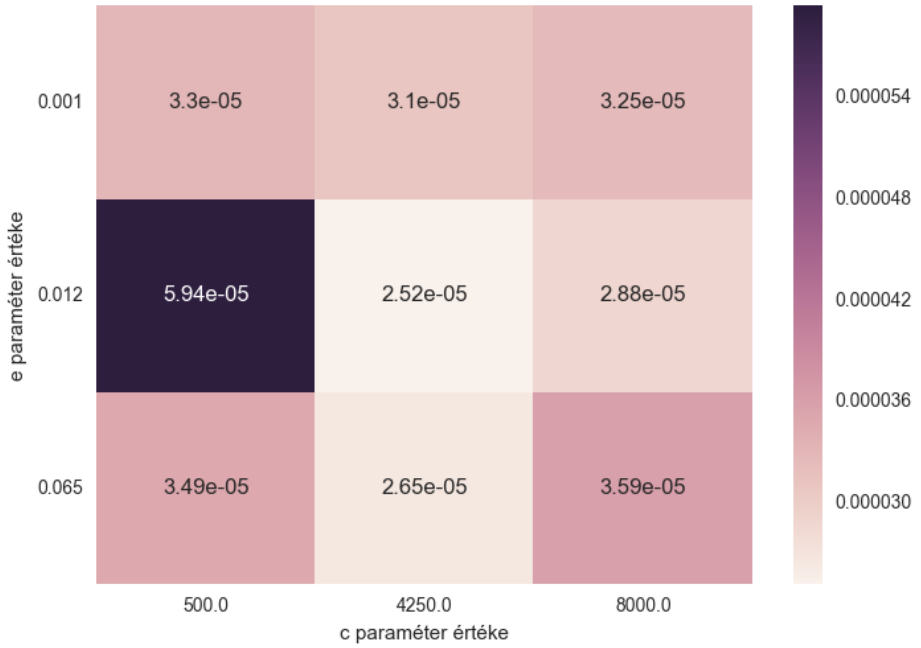
[6] Variancia: 0,00358



8. ábra: A validációs hiba értéke a C és e paraméterek függvényében az alacsony adatbázis esetében.
Forrás: Saját számítás

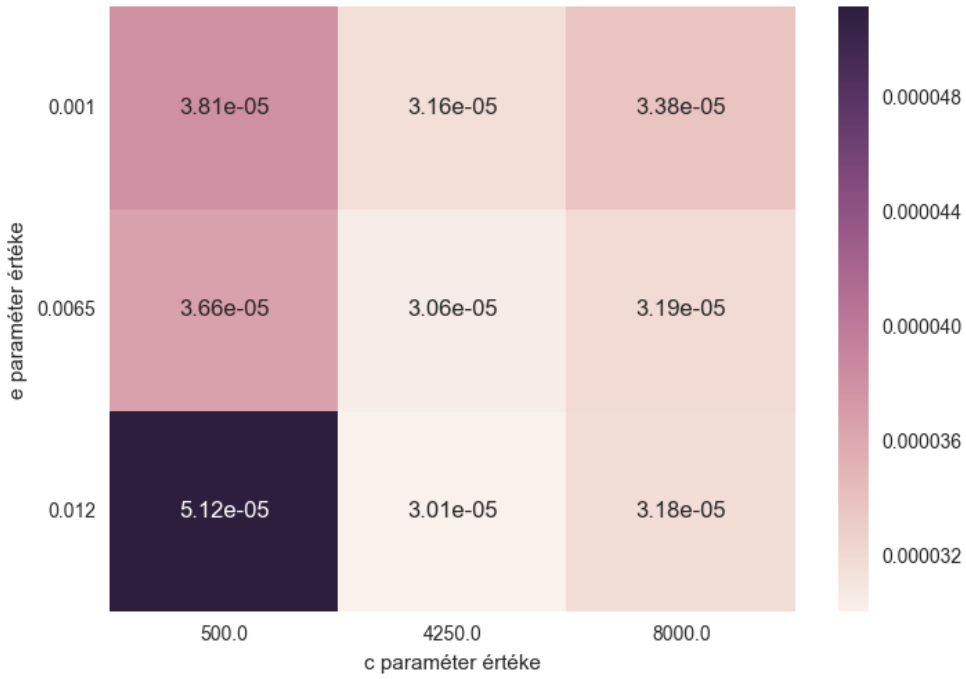


9. ábra: A kiszámításhoz szükséges idő hossza másodpercben a C és e paraméterek függvényében az alacsony adatbázis esetében.
Forrás: Saját számítás

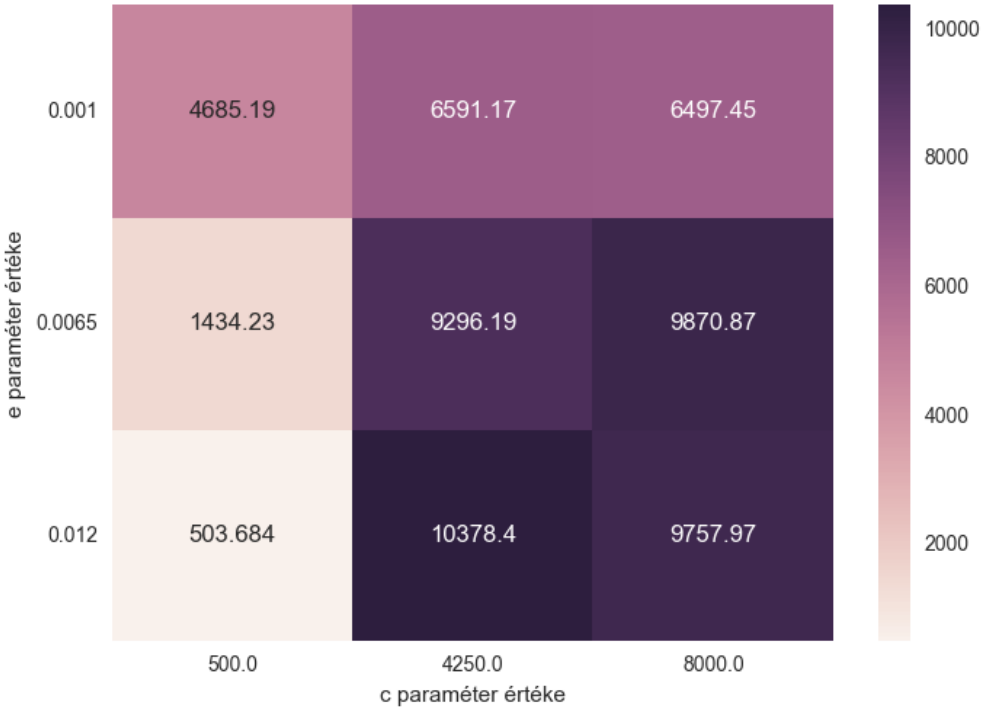


10. ábra: A teszthiba értéke a C és e paraméterek függvényében az alacsony adatbázis esetében.
Forrás: Saját számítás

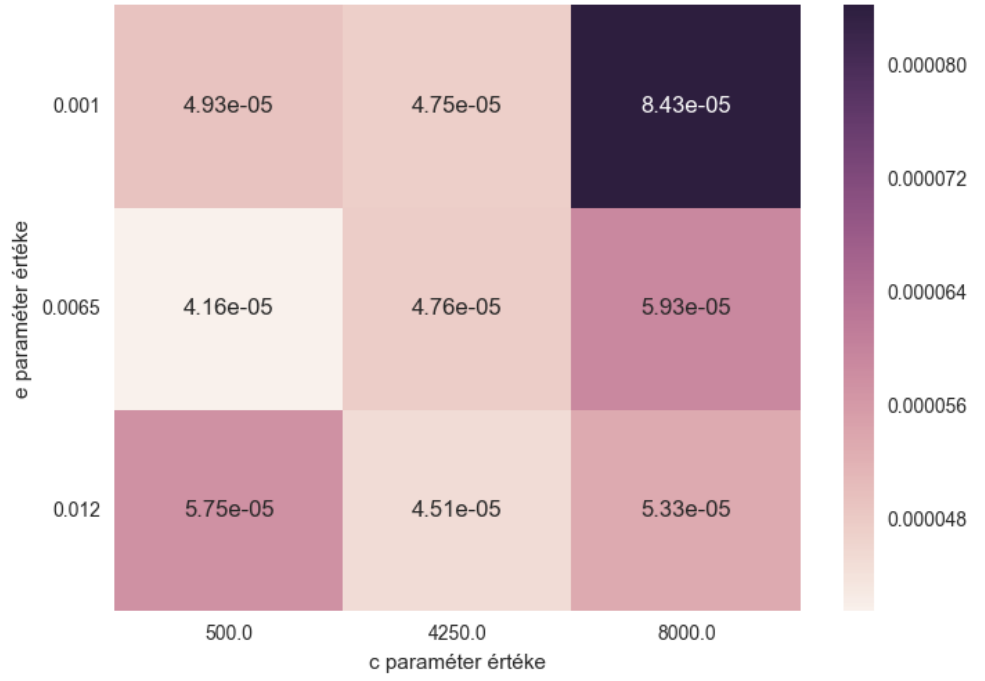
A magas adatbázis esetében hasonló tendenciák érvényesültek, bár a legjobb validációs hiba ($3.0089*10^{-5}$) melletti SVR az előzőkkel megegyező paraméterekkel $4.5104*10^{-5}$ átlagos négyzetes eltérést adott a teszt adatbázison, amellyel nem a legjobb teszthibát adta.



11. ábra: A validációs hiba értéke a C és e paraméterek függvényében a magas adatbázis esetében.
Forrás: Saját számítás



12. ábra: A kiszámításhoz szükséges idő hossza másodpercben a C és e paraméterek függvényében a magas adatbázis esetében.
Forrás: Saját számítás



13. ábra: A teszthiba értéke a C és e paraméterek függvényében a magas adatbázis esetében.
Forrás: Saját számítás

7.2) ELM piaci adatokon

A szimulált adatbázishoz hasonló módon itt is nagyon gyorsan becsült az ELM, bár részben a több adatból fakadóan már néha 3 másodperc fölé is szaladt a 10 véletlen beállítás közül a legjobb validációs hibával rendelkező hálózat. Sajnos ez az egyetlen versenyelőnye a módszernek, ami miatt kitüntetett figyelemnek örvendhet. A probléma, hogy a megtanított hálózatok nem igazán mutatnak olyan tulajdonságot, hogy a tanítóhiba és a validációs hiba túl sokat elárulna a teszt hiba nagyságáról. Másképpen fogalmazva ugyan vannak olyan hálózatok, amelyek teszthibája akár versenyképes is lehetne legalább az SVR hibájával, de ezek nem tűnnek kiválaszthatónak a tanító és validációs hiba alapján. Próbálkoztam azon hálózat kiválasztásával, ahol a tanító és validációs becslés átlagos négyzethibájának összege minimális, hogy így kényszerítsek ki valamiféle általánosítást, de ez ugyanúgy nem volt képes stabilan megtalálni a jó teszthibával rendelkező hálózatokat.

Az alacsony adatbázis esetében a legjobb validációs hibával rendelkező ELM végül $1{,}20832*10^{-4}$ átlagos négyzetes eltérést produkált a tesztmintán, mely az ELMek keverésével és becslésnél átlaguk felhasználásával még valószínűleg javítható lenne, de az élmezőnytől így is egy tizedesjeggyel lemarad.

A magas adatbázis esetében a módszer lemaradása kisebb, amely valószínűleg az adatbázisnak tudható be, a legkisebb teszthiba továbbra sem tűnt megtalálhatónak a tanító és validációs hiba alapján. A legkisebb validációs hibát produkáló ELM átlagos négyzetes hibája a teszt adatbázison $6{,}31835*10^{-5}$ volt.

Az ELM gyengébb eredményei azonban függhetnek az aktivációs függvénytől, itt most sigmoid beállítással került felhasználásra, ezért a fenti eredmények inkább úgy interpretálhatóak, hogy minden igyekezetem mellett ezek voltak a legjobb eredmények, amiket sikerült elérnem, nem feltétlenül a módszer rosszabb, mint a másik három.

7.3) Tree Ensemble piaci adatokon

A pozitív tulajdonságokat előre véve a módszer alapvetően megtartotta kimagasló sebességét, amellyel ezek végére is értünk. A legnagyobb probléma, hogy valódi adatokon a modell elveszti robosztusságát, bár a tanulás során csak a tanító adatbázissal érintkezett az algoritmus, ennek ellenére az addig nem látott validációs minta alapján a teszt adatbázis számára megfelelő paraméterek kiválasztása nem sikerült, akárcsak az ELM esetében.

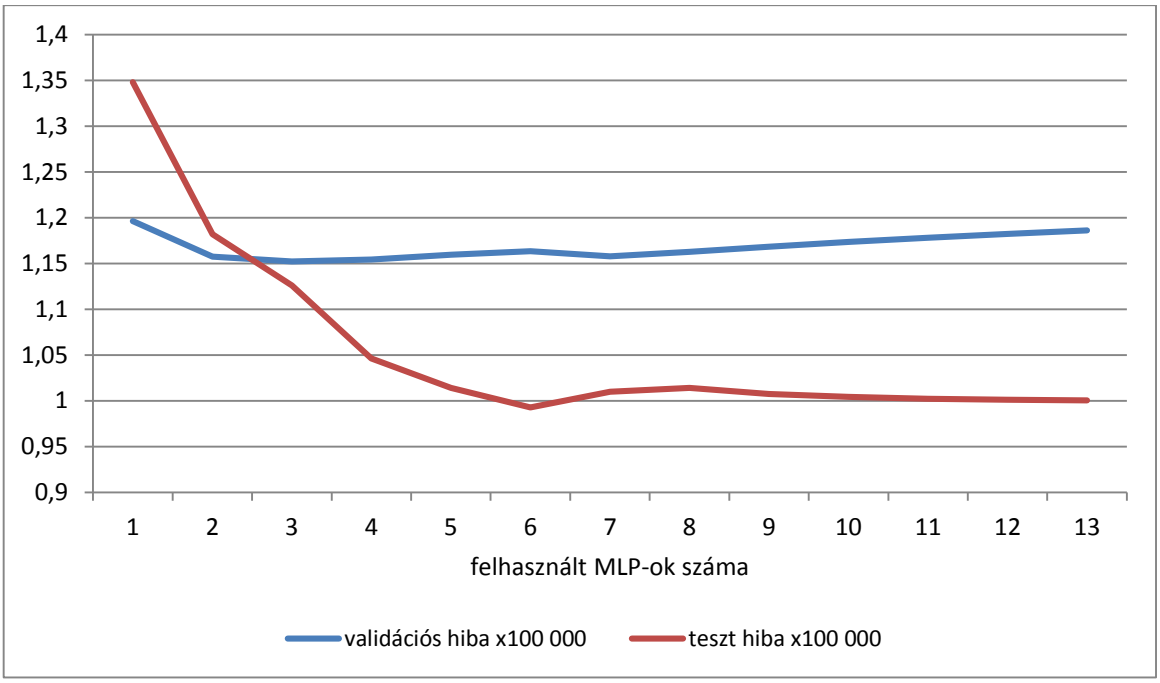
Az eredményei a teszt adathalmazon minden esetben jelentősen elmaradnak a MLP eredményeitől, a pontosságot talán valamilyen regularizáció bevezetésével lehetne orvosolni valamelyest, de a szokásos regularizációk használatával sem sikerült a MLP-vel versenyképes pontosságot elérnem ennél a módszernél. A (transzformált) alacsony adatbázis esetében ez $5{,}15266*10^{-5}$ átlagos négyzetes hibát jelent a tesztmintán, még a magas adatbázis esetében ez $1{,}0844*10^{-4}$ nagyságot jelent.

Alapvetően hasonló eredményeket és tulajdonságokat mutat , mint az ELM, ugyanakkor a regularizációt leszámítva (ami a többi módszer esetén sem került felhasználásra) gyakorlatilag nem maradt olyan kisebb paramétere, aminek változtatásától érdemben javulás remélhető. Használatát opcióárazási célokra ezért nem javaslom.

7.4) MLP piaci adatokon

A piaci adatokra alkalmazott MLP tulajdonságai nagyban hasonlítanak a szimulált adatokon tapasztalható tulajdonságokhoz. A legjobb hálózatokkal elért validációs hiba a rejtett rétegben lévő neuronok számának növekedésével meredeken csökken 5 neuronig, majd lassabban 10 neuronig, innentől pedig a számításhoz szükséges idő elkezd szemmel láthatóan növekedni a teljesítmény érezhető javulása nélkül.

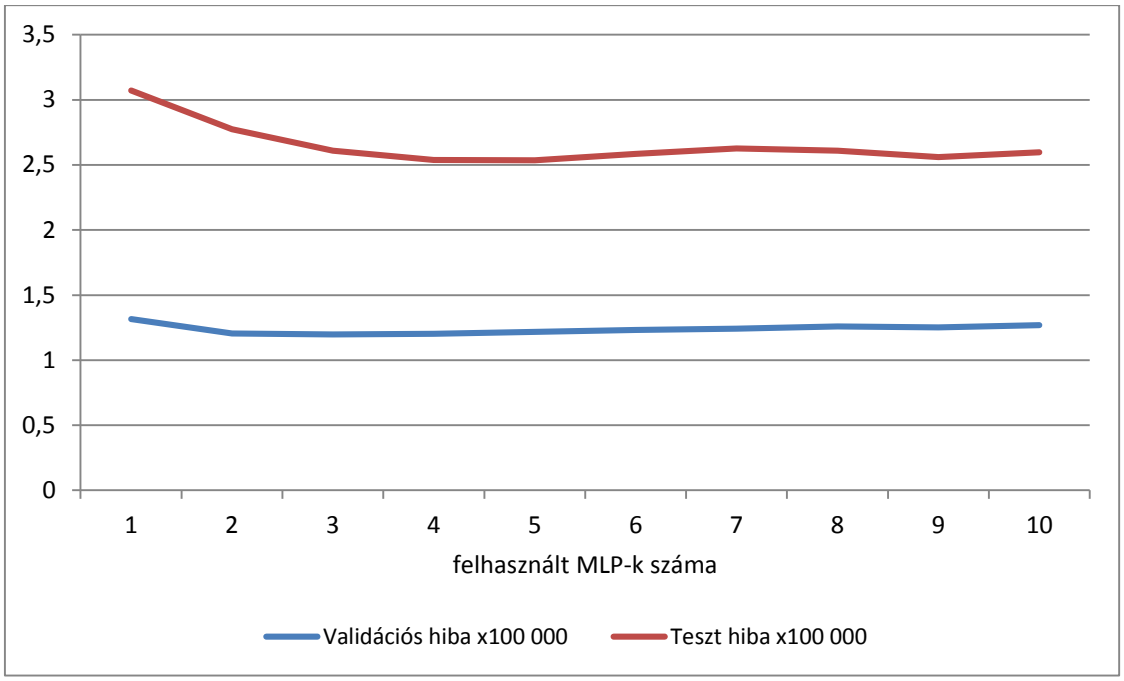
Az alacsony adatbázis esetében az 5-5 darab 1-19 rejtett neuront tartalmazó hálózat mellé 5-10 neuronokat tartalmazó hálózatok kerültek legyártásra, így összesen 146 darab MLP készült el. A legjobb MLP $1{,}19627*10^{-5}$ validációs hiba mellett $1{,}34809*10^{-5}$ átlagos négyzetes hibát ad a teszt adatbázison. A szimulációs adatokon már bemutatott neurális hálók átlagolása módszerrel ez még tovább csökkenthető:



14. ábra: Az alacsony adatbázis esetében a validációs és tesztmintán mért átlagos négyzetes eltérés.
Forrás: Saját számítás

A validációs hiba a legjobb három MLP becslés átlaga esetén veszi fel a minimumát, ebben az esetben ez a teszt adatok esetében is javítja a becslés pontosságát, bár a teszt adatbázis ismeretében megállapítható, hogy több MLP átlagával még jobb eredményt lehetett volna elérni a teszthalmazon. A legjobb validációs hibát ($1{,}15215*10^{-5}$) elérő legjobb 3 MLP-t felhasználó becslés teszthalmazon elért átlagos négyzetes hibaösszege $1{,}12590*10^{-5}$ volt.

A magas adatbázis esetében szintén 1-19 rejtett neuront tartalmazó hálózatokból 5-5 darab készült, azonban ezek eredményeinek megfigyelésével ezúttal csak a 7-10 rejtett neuront tartalmazó MLP-k kerültek nagyobb mennyiségben legenerálásra, így összesen 140 darab MLP-t hozva létre. A magas adatbázis esetén a legjobb validációs hiba $1{,}31448*10^{-5}$ volt, az ehhez tartozó teszthiba pedig $3{,}07061*10^{-5}$ volt. Több MLP felhasználásával és átlagukkal való becsléssel az eredmények így alakulnak:



15. ábra: A magas adatbázis esetében a validációs és tesztmintán mért átlagos négyzetes eltérés.
Forrás: Saját számítás

A validációs hiba a legjobb három MLP-vel való becslés esetén vette fel a minimumát, ami $1{,}197*10^{-5}$ volt, ekkor a teszthiba $2{,}6091*10^{-5}$ értéket vett fel. Ebben az esetben a teszthiba a minimumát a legjobb öt MLP-vel való becslés esetén vette volna fel, tehát a validációs minta alapján kapott legjobb MLP szám ismét kisebb volt az optimálisnál.

Alacsony adatbázis	Hiba a validációs mintán	Hiba a tesztmintán
SVR	$3{,}03793*10^{-5}$	$2{,}51671*10^{-5}$
ELM	$1{,}64569*10^{-5}$	$1{,}20832*10^{\mathbf{-4}}$
Tree Ensemble	$5{,}36516*10^{\mathbf{-6}}$	$5{,}15266*10^{-5}$
MLP	$1{,}15215*10^{-5}$	$1{,}12590*10^{-5}$
Magas adatbázis	Hiba a validációs mintán	Hiba a tesztmintán
SVR	$3{,}0089*10^{-5}$	$4{,}5104*10^{-5}$
ELM	$1{,}8561*10^{-5}$	$6{,}3184*10^{-5}$
Tree Ensemble	$5{,}0958*10^{\mathbf{-6}}$	$1{,}0844*10^{\mathbf{-4}}$
MLP	$1{,}197*10^{-5}$	$2{,}6091*10^{-5}$

5. táblázat: A különböző módszerek teljesítménye a két adatbázison mért hiba szerint[7].
Forrás: Saját számítás

[7] A skálázási intervallum paraméter az alacsony adatbázis esetében 127,58, még a magas adatbázis esetében 411,13 volt.

Amint az eredményekből is látszik, mindkét adatbázis esetén a MLP teljesített a legjobban, melyet az SVR követett, végül az utolsó helyre az ELM vagy a Tree Ensemble szorultak, mivel azok a tesztmintán nem tudták megközelíteni a tanító és validációs mintán mért teljesítményüket.

8) Problémák és további lehetőségek

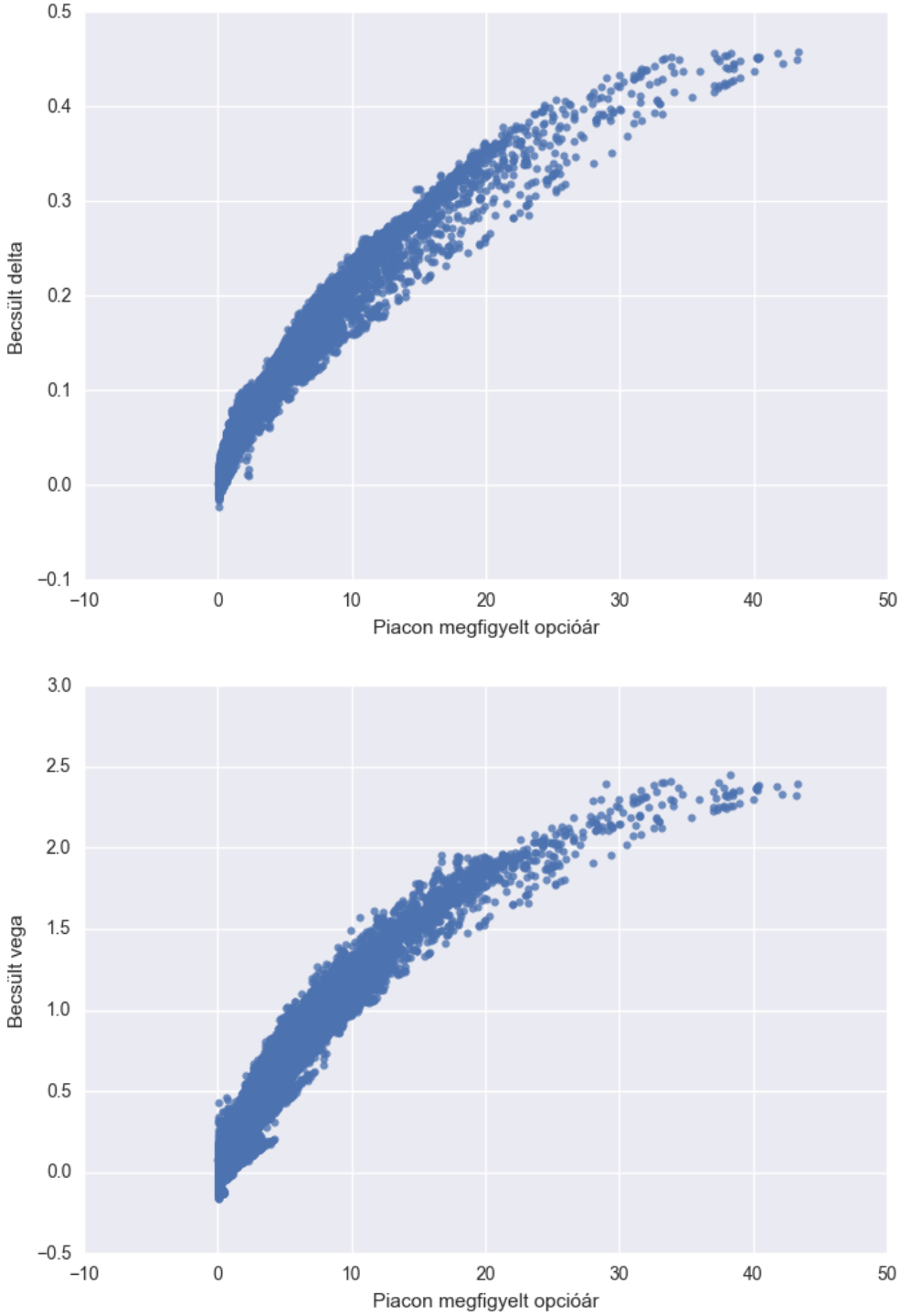
A korábbi fejezetekben az olvasó megismerkedhetett több gépi tanulásos módszerrel, melynek eredményeképp láthatóvá vált, hogy az átlagolt Multilayer Perceptronok segítségével kapott becslés a legkisebb hibával dolgozik. A következőkben az kerül áttekintésre, hogy a dolgozat alapján így konstruált legjobb becslési eljárás hol támadható, illetve hogyan javítható tovább a teljesítménye. A következő eredményeket mindig az eredeti (nem transzformált) adatbázisra kell érteni.

8.1) Problémák

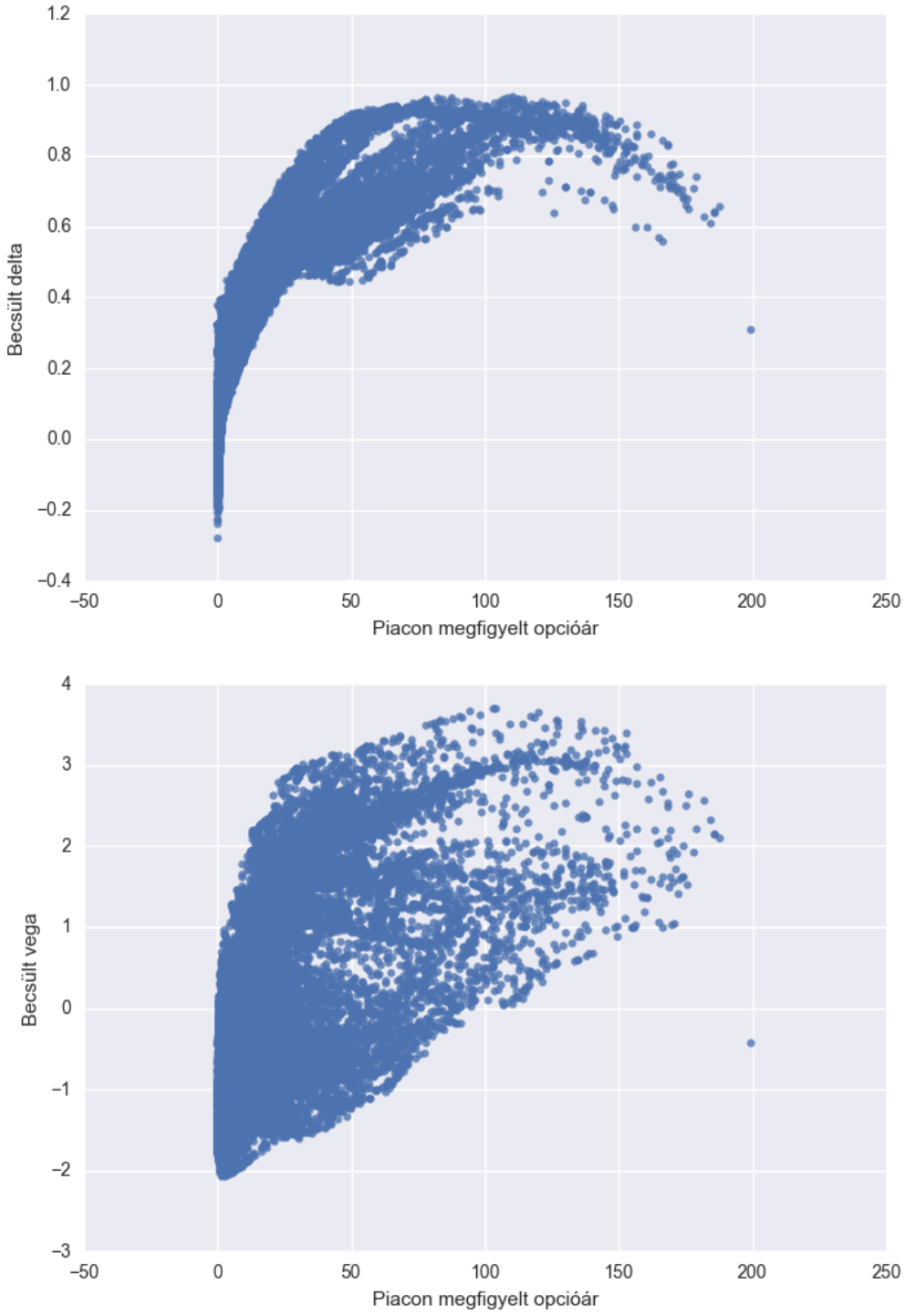
Az első fontos probléma, hogy a hiba nagysága nem függ lényegesen az opció valódi árától. Az alacsony adatbázis esetében az átlagos abszolút eltérés 0,298 , még a magas adatbázis esetében ez 1,646 (dollár) a tesztmintán. Ez azt jelenti, hogy a skálázás során alkalmazott módszer miatt mind az 5 centes, mind a 100 dolláros opciók esetén várhatóan ennyi lesz az eltérés.

A második, talán legsúlyosabb probléma, hogy a létrehozott becslőfüggvénnyel szemben vannak előzetes elvárásaink. Ilyen, hogy az input paraméterek közül mind az alaptermék árfolyamának, mind a volatilitás ceteris paribus növekedésének hatására az opció ára lehetőleg nőjön, de semmiképp se csökkenjen. Ennek oka természetesen nem az eddig ismert árazóképletekből következik, ezen feltételek egyrészt a piacon megfigyelhetők, másrészt a józan ész diktálja őket. Ezen növekményekre a következőkben mint az opció deltája és vegája fogok hivatkozni, noha ezek nem a klasszikus parciális deriváltak, hanem az adott input paraméterek 0,01-el történő elmozdításának hatására az opció értékváltozásának százszorosa (numerikus deriváltak).

Az eredmények sajnos azt mutatják, hogy a gépi tanulásos árazás során a delta néha kismértékben, még a vega gyakran nagymértékben negatívba fordul:



16. ábra: A legjobban teljesítő modell alapján kapott delták és vegák az alacsony teszt adatbázis esetében.
Forrás: Saját számítás



17. ábra: A legjobban teljesítő modell alapján kapott delták és vegák a magas teszt adatbázis esetében.
Forrás: Saját számítás

Amint az látható, a probléma nem ugyanannyira érinti mindkét adatbázist. A delta probléma mindkét adatbázis esetén kizárólag a nagyon alacsony értékű opciók esetében fordul elő, amiket a korábbi szerzők egész egyszerűen kizártak, példaként felhozva a korábban is említett (Anders, et al., 1998, p. 8), ahol a 10 dollár alatti opciók nem képezték vizsgálat tárgyát. Más

szempontból megközelítve az előző pont alapján könnyű felismerni, hogy a becslés nem működik jól ilyen formában kisértékű opciókra, ezért körültekintő ember nem bízna ezen becslés parciális deriváltjaiban sem.

A vega esetében a probléma szintén nem számottevő az alacsony adatbázis esetében, viszont a magas adatbázis esetében számottevő és súlyos. Ahogy az a fenti ábrán is látható, még az 50 dollár feletti opciók esetében is akadnak negatív vegák -1 körüli értékkel, ami nem éppen csak negatív. A Multilayer Perceptron esetén nem tudok olyan módszerről, melynek segítségével bizonyos parciális deriváltak előjele szabályozható lenne. Amint az a korábbiakban említésre került, az alacsony adatbázisban az opcióértéket pontosabban sikerült megbecsülni, véleményem szerint ez az a tényező, ami a vega problémát képes orvosolni.

A harmadik probléma, hogy bár a korábban tárgyaltak szerint az adatbázis moneyness alapján szétválasztásra került két kisebb adatbázisra (alacsony és magas), ez ugyan hasznosnak tűnik az eredmények alapján, de nem különíti el egymástól elég jól az alacsony és a magas árú opciókat. Ez miatt aztán később a [0;1] intervallumra transzformálás során a transzformált opciós árak kevésbé szóródnak és a becslés ezért pontatlanabb lesz. A problémát úgy oldanám fel, hogy a dolgozatban megismert módszerek segítségével egy előfeldolgozó neurális hálót hoznék létre a tanuló és validációs adathalmaz alapján, majd az ezen megtanított neurális hálóval az adatokat két csoportba osztanám alacsony és magas becsült ár szerint. Érdekes kérdés, hogy vajon ezek után tovább kell-e bontani moneyness alapján is és így 4 részmintát kapva vagy ezen bontás nélkül a neurális háló eredményeképp létrejött 2 részmintával dolgozni, ezen kérdésben nem foglalnék állást, le kell mérni, hogy a további bontás miként változtatja a pontosságot. A harmadik probléma megoldásától az előző kettő jelentőségének a csökkenését szintén várom.

8.2) Fejlesztési lehetőségek

A dolgozat elkészítése során a [0;1] intervallumba történő beskálázása lineáris transzformációval történt, azonban ez nem az egyetlen módszer. Az eredeti és a transzformált skála egymásba átvitele történhet nem lineáris függvényekkel, de ide tartozik még az implicit volatilitás, mint célváltozó becslése és a hibrid megközelítés is, ahol a Black-Scholes ártól való eltérés kerül megbecslésre. A különböző megközelítési módszerek másféle hibákat adnak, de érdemes lehet ezen skálázással kísérletezni.

Ahogy a VIX index, mint magyarázó változó bevezetése csökkentette a becslés hibáját, ugyanúgy új jelentős magyarázó változók bevezetésével a becslés még pontosabbá tehető. Korábbi szerzők azt találták, hogy a piaci likviditás (mind az opció, mind az alaptermék esetében) jelentős hatást fejt ki az opcióárakra (Chou, et al., 2011), még az opciós piacon tapasztalt keresleti sokkok szintén befolyásolják az opció árát a nem tökéletes fedezhetőség miatt (Garleanu, et al., 2009).

Jelen mű igyekezett a rendelkezésre álló gépidőhöz mérten a gépi tanulásos módszerek paramétereit a lehető legpontosabban beállítani, azonban jobb gépteljesítménnyel ezek még jobban beállíthatók lennének. Ezen felül érdekes lehet kevésbé fontos paraméterek alapértelmezettől eltérő beállításainak hatását is megvizsgálni, mint az aktivációs- és kernelfüggvények és ezek paraméterei.

Az eredmények szemléletessé tétele érdekében jó lett volna, ha sikerül a valódi adatbázison is valamilyen sztochasztikus pénzügyből ismert modell hibáját megmérni, hogy a különböző algoritmusokat ne csak egymással, hanem a jelenleg jól teljesítő modellel is össze lehessen mérni. Ezen modellek ugyanakkor számos előfeltevéssel élnek, így például az illikvid opciókat az adatbázisból el kellene hagyni, ezért nem sikerült olyan modellt megbecsülnöm, amelynek eredménye a jelen adatbázisokon fair módon összemérhető lenne a gépi tanulásos módszerek teljesítményével.

9) Összefoglalás

A dolgozat fő témáját a gépi tanulásos módszerek felhasználásával történő opcióárazás adta, azon belül is négy módszerre, a Multilayer Perceptronra, a Support Vector Regressionra, a Tree Ensemble és az Extreme Lerning Machine algoritmusokra koncentrált. Ezek közül, mind szimulált, mind piaci adatok esetén a Multilayer Perceptron teljesített a legjobban elfogadható kalkulációs idő mellett.

A bevezetés utáni első két fejezet módszertani áttekintésként szolgált, bemutatta a sztochasztikában, illetve a gépi tanulásban opcióárazásra használható módszereket, illetve bemutatta, hogy utóbbi bonyolult algoritmusok igazából tekinthetők az opcióárazó módszertan következő fejlődési lépcsőjének bizonyos szempontból.

A negyedik fejezetben az olvasó megismerkedhetett azon kutatók munkásságával, akik a gépi tanulásos módszereket a múltban már felhasználták a dolgozat céljához hasonlóan, az általuk tett megfigyelések és felfedezések pedig fontos útmutatásul szolgáltak ezen dolgozat elkészítéséhez.

Az ötödik fejezetben demonstrálásra került, hogy még ha a világ tökéletes is lenne és a sztochasztikus modellek által feltételezett dinamikák teljesülnének, a gépi tanulásos módszerek akkor is jól teljesítenének. A Heston modell dinamikája szerint generált alaptermék árfolyamok és a Heston modell kockázatsemleges dinamikája alapján beárazott opciók értékét a Multi-Layer Perceptron nagy pontossággal sikeresen becsülte meg.

A hatodik fejezet a valós adatok beszerzésének és tisztításának nehézségeivel foglalkozott, bemutatta az adatbázist alkotó adatokat és annak alapvető tulajdonságait, illetve azt moneyness érték szerint alacsony és magas adatbázisra választotta szét, melyek később külön-külön kerültek megbecslésre.

A hetedik fejezet a gépi tanulásos módszerek teljesítményét mutatta be a két piaci adatokat tartalmazó adatbázis felhasználásával, kiderült, hogy az Support Vector Regression általánosító képessége megbízható, még az Extreme Learning Machine és Tree Ensemble módszerek sokkal inkább képesek voltak minden igyekezet ellenére túltanulni a tanító és validációs adatbázist. Mindezek mellett mindhárom algoritmus teljesítményét felülmúlta a Multilayer Perceptron mindkét adatbázison a legkisebb átlagos négyzetes hibát adta a teszt adatbázison, ezzel meglehetősen pontosan becsülte meg az opciók árát mindkét adatbázisban. A fejezet

eredményei alapján a nemparaméteres becslési eljárások fontos versenytársai lehetnek a sztochasztikus pénzügyi modelleknek.

Végezetül, bár a dolgozat számos kérdés esetén megnyugtató válaszokat adott, ugyanakkor számos esetben érdekes új kérdéseket vetett fel. Ilyen például a dolgozatban nem vizsgált más aktivációs függvényekben és paraméterekben rejlő lehetőségek vizsgálata, különböző skálázási eljárások hatása, illetve, hogy érdemes lehet egy olyan sztochasztikus pénzügyi modellt megkonstruálni, amivel az eredmények jól összemérhetők lennének.

Hivatkozások

Altrichter, M. és mtsai., 2006. *Neurális hálózatok.* Budapest: Panem Könyvkiadó Kft..

Amilon, H., 2003. A Neural Network Versus Black-Scholes: A Comparison of Pricing and Hedging Performances. *Journal of Forecasting,* 22(4), pp. 317-335.

Anders, U., Korn, O. & Christian, S., 1998. Improving the pricing of options: a neural network approach. *Journal of Forecasting,* 17(5-6), pp. 369-388.

Andreou, P. C., Charalambous, C. & Martzoukos, S. H., 2008. Pricing and trading European options by combining artificial neural networks and parametric models with implied parameters. *European Journal of Operational Research,* 185. kötet, pp. 1415-1433.

Black, F. & Scholes, M., 1973. The Pricing of Options and Corporate Liabilities. *Journal of Political Economy,* 81(3), pp. 637-54.

Blynski, L. & Faseruk, A., 2006. Comparison of the effectiveness of option price forecasting: Black-Scholes vs. simple and hybrid neural networks. *Journal of Financial Management and Analysis,* 19(2), pp. 46-58.

Broadie, M., Detemple, J., Ghysels, E. & Torrés, O., 1996. *Nonparametric estimation of American options' exercise boundaries and call prices.* [Online]
Available at: https://www0.gsb.columbia.edu/mygsb/faculty/research/pubfiles/1853/1853b.pdf
[Hozzáférés dátuma: 21 március 2016].

Chicago Board Options Exchange, 2015. *The CBOE Volatility Index - VIX.* [Online]
Available at:
http://www.cboe.com/framed/pdfframed.aspx?content=/micro/vix/vixwhite.pdf&section=SECT_MINI_SITE&title=VIX+White+Paper
[Hozzáférés dátuma: 21 március 2016].

Chou, R. K., Chung, S.-L., Hsiao, Y.-J. & Wang, Y.-H., 2011. The Impact of Liquidity on Option Prices. *Journal of Futures Markets,* 31(12), p. 1116–1141.

Cox, J. C., Ingersoll, J. E. & Ross, S. A., 1985. A Theory of the Term Structure of Interest Rates. *Econometrica,* 53. kötet, pp. 385-407.

Crone, S. F., Hibon, M. & Nikolopoulos, K., 2011. Advances in forecasting with neural networks? Empirical evidence from the NN3 competition on time series prediction. *International Journal of Forecasting,* 27. kötet, pp. 635-660.

Cybenko, G., 1989. Approximation by superposition of a sigmoidal Function. *Mathematics of Control, Signals, and Systems,* 2. kötet, pp. 303-314.

Dindar, Z. A., 2004. *Artificial neural networks applied to option pricing.* [Online]
Available at:
http://wiredspace.wits.ac.za/bitstream/handle/10539/181/dissertation.pdf?sequence=1
[Hozzáférés dátuma: 21 március 2016].

Fengler, M. R., 2010. *Option Data and Modeling BSM Implied Volatility.* [Online]
Available at: http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1922441
[Hozzáférés dátuma: 21 március 2016].

Garcia, R. & Gencay, R., 2000. Pricing and hedging derivative securities with neural networks and a homogenity hint. *Journal of Econometrics,* 94. kötet, pp. 93-115.

Garleanu, N., Pedersen, L. H. & Poteshman, A. M., 2009. Demand-Based Option Pricing. *The Review of Financial Studies,* 22(10), pp. 4259-4299.

Gencay, R. & Qi, M., 2001. Pricing and Hedging Derivative Securities with Neural Networks: Bayesian Regularization, Early Stopping and Bagging. *IEEE Transactions on Neural Networks,* 12(4), pp. 726-734.

Gencay, R. & Salih, A., 2003. Degree of Mispricing with the Black-Scholes Model and Nonparametric Cures. *Annals of Economics and Finance,* 4. kötet, pp. 73-101.

Gong, H., 2014. *The Heston Model.* [Online]
Available at: http://www.homepages.ucl.ac.uk/~ucahgon/Heston.pdf
[Hozzáférés dátuma: 21 március 2016].

Hanke, M., 1999. Adaptive Hybrid Neural Network Option Pricing. *Journal of Computational Intelligence in Finance,* szeptember, 5(5), pp. 33-39.

Hanke, M., 1999. Neural Networks vs. Black-Scholes: An Empirical Comparison of the Pricing Accuracy of Two Fundamentally Different Option Pricing Methods. *Journal of Computational Intelligence in Finance,* január, 5(1), pp. 26-34.

Hebb, D. O., 1949. *The Organization of Behaviour.* New York: John Wiley and Sons.

Heston, S. L., 1993. A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options. *Reviews of Financial Studies,* 6. kötet, pp. 327-343.

Hornik, K., Stinchcombe, M. & White, H., 1989. Multilayer feedforward networks are universal approximators. *Neural Networks,* 2. kötet, pp. 359-366.

Huang, G.-B., 2015. *Extreme Learning Machines (ELM) - Filling the Gap between Frank Rosenblatt's Dream and John von Neumann's Puzzle?.* [Online]
Available at: www.ntu.edu.sg/home/egbhuang/pdf/ELM-Tutorial.pdf
[Hozzáférés dátuma: 21 március 2016].

Hull, J. & White, A., 1987. The Pricing of Options on Assets with Stochastic Volatilities. *Journal of Finance,* 42(2), pp. 281-330.

Hutchinson, J. M., Lo, A. W. & Poggio, T., 1994. A Nonparametric Approach to Pricing and Hedging Derivative Securities Via Learning Networks. *The Journal of Finance,* 49(3), pp. 851-889.

Kelly, D. L., 1994. *Valuing and Hedging American Put Options Using Neural Networks.* [Online]
Available at: http://finance.martinsewell.com/option-pricing/Kell94.pdf
[Hozzáférés dátuma: 21 március 2016].

Lajbcygier, P. R. & Connor, J. T., 1997. Improved option pricing using artificial neural networks and bootstrap methods. *International Journal of Neural Systems,* 8(4), pp. 457-471.

Lu, C.-J., Lee, T.-S. & Chiu, C.-C., 2009. Financial time series forecasting using independent component analysis and support vector regression. *Decision Support Systems,* 47. kötet, pp. 115-125.

Malliaris, M. & Salchenberger, L., 1993. A Neural Network Model for Estimating Option Prices. *Applied Intelligence,* 3(3), pp. 193-206.

Malliaris, M. & Salchenberger, L., 1996. Using neural networks to forecast the S&P 100 implied volatility. *Neurocomputing,* 10. kötet, pp. 183-195.

Márkus, L., 2013. *FP1_Lecture20-24a.* [Online]
Available at: http://www.math.elte.hu/probability/markus/FinancialProcesses.htm
[Hozzáférés dátuma: 21 március 2016].

Maruyama, M., Girosi, F. & Poggio, T., 1992. *A Connection between GRBF and MLP.* [Online]
Available at: ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-1291.pdf
[Hozzáférés dátuma: 21 március 2016].

Matlab Geeks, 2011. *Neural Networks – A Multilayer Perceptron in Matlab.* [Online]
Available at: http://matlabgeeks.com/tips-tutorials/neural-networks-a-multilayer-perceptron-in-matlab/
[Hozzáférés dátuma: 21 március 2016].

McCulloch, W. S. & Pitts, W., 1943. A Logical Calculus of Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics,* pp. 115-133.

Mikhailov, S. & Nögel, U., 2003. Heston‟s stochastic volatility model implementation, calibration and some extensions. *WILMOTT magazine,* pp. 74-79.

Minsky, M. & Papert, S., 1969. *Perceptrons.* Cambridge MA: MIT Press.

Mitra, S. K., 2012. An Option Pricing Model That Combines Neural Network Approach and Black Scholes Formula. *Global Journal of Computer Science and Technology,* 12(4), pp. 6-16.

Rosenblatt, F., 1958. The Perceptron: A Probabilistic Model for Information Storage and Organization of the Brain. *Psycol Rev.,* 65. kötet, pp. 386-408.

Rumelhart, D. E., Hinton, G. E. & Williams, R. J., 1986. Learning internal representations by error propagation. In: *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1.* Cambridge, MA, USA: MIT Press, pp. 318-362.

Saxena, A., 2008. *Valuation of S&P CNX Nifty Options: Comparison of Black-Scholes and Hybrid ANN Model.* [Online]
Available at: www2.sas.com/proceedings/forum2008/162-2008.pdf
[Hozzáférés dátuma: 21 március 2016].

Stein, E. M. & Stein, J. C., 1991. Stock Price Distributions with Stochastic Volatility: An Analytic Approach. *The Review of Financial Studies,* 4(4), pp. 727-752.

Swanson, N. R. & White, H., 1995. A model-selection approach to assessing the information in the term structure using linear models and artificial neural networks. *Journal of Business and Economic Statistics,* 13. kötet, pp. 265-275.

Teh, L. L. & de Bondt, W. F., 1997. Herding Behavior and Stock Returns: An Exploratory Investigation. *Swiss Journal of Economics and Statistics,* 133(2), pp. 293-324.

Teräsvirta, T., Lin, C.-F. & Granger, C. W., 1993. Power of the Neural Network Linearity Test. *Journal of Time Series Analysis,* 2. kötet, pp. 209-220.

Verma, N., Srivastava, N. & Das, S., 2014. Forecasting the Price of Call Option Using Support Vector Regression. *IOSR Journal of Mathematics,* 10(6), pp. 38-43.

White, H., 1989. An additional hidden unit test for neglected nonlinearity. *Neural Networks, 1989. IJCNN,* pp. 451-455.

xgboost developers, 2016. *Introduction to Boosted Trees.* [Online]
Available at: https://xgboost.readthedocs.io/en/latest/model.html
[Hozzáférés dátuma: 7 május 2016].

Yang, Y., 2013. *Valuing a European Option with the Heston Model.* [Online]
Available at: http://scholarworks.rit.edu/cgi/viewcontent.cgi?article=5813&context=theses
[Hozzáférés dátuma: 21 március 2016].

Yao, J., Li, Y. & Tan, C. L., 2000. Option price forecasting using neural networks. *Omega,* 28(4), pp. 455-466.

Yao, J. & Tan, C. L., 2001. *Guidelines for Financial Forecasting with Neural Networks.* [Online]
Available at: citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.18.6874&rep=rep1&type=pdf
[Hozzáférés dátuma: 21 március 2016].