

TDK dolgozat

Czetin Márk

Kovács Hunor György

2023

Gépi tanulás alapú agilis forecast döntéstámogató rendszer

Czetin Márk

Kovács Hunor György

Széchenyi István Egyetem

Gépészmérnöki, Informatikai és Villamosmérnöki Kar

Mérnökinformatikus szak, BSc 3. év

Témavezető(k):

dr. Bognár Tamás, egyetemi adjunktus

Hajdu Csaba, egyetemi tanársegéd

Kézirat lezárásának dátuma: 2023.11.08

Absztrakt

Gépi tanulás alapú agilis forecast döntéstámogató rendszer

Az elmúlt évek meghatározó eseményei jelentősen megváltoztatták a fogyasztói szokásokat. Ezáltal nagyon megnehezedett a cégek feladata, a vásárlások megbecslése terén, amely ellehetetleníti számukra a megfelelő tervezést. Jelen dolgozat témája egy mesterséges intelligencia fejlesztése, implementálása és használata nagyvállalati környezetben történő gazdasági döntések támogatásához. Elengedhetetlen, hogy a döntéshozók a legalkalmasabb eszközöket használják munkájuk folyamán. Ezek közé tartoznak a különböző mesterséges intelligenciák alkalmazása is, amiket egyre több cég használ a folyamatok gyorsabbá tételére, hatékonyságuk maximalizálására és költségeik minimalizálására. A kutatás egy igényfelméréssel kezdődött, amiben a vállalat egy gazdasági szakembere segítségével feltérképezésre került a cég működése, a folyamatok, amelyeket a fejlesztett rendszer érinteni fog, és a szoftver felé támasztott igények. A program célja, hogy tudjon olyan, vagy jobb eredményeket nyújtani a rendelkezésre álló információk felhasználásával, amelyeket emberi erőforrással csak hosszú idő alatt lenne képes a cég előállítani. Egy mesterséges intelligencia hatékonyabban képes elvégezni ilyen előrejelzéseket historikus információkra támaszkodva, mint emberek, hagyományos módszereket alkalmazva. Az adatokat a cég szolgáltatja a kutatáshoz, amely elengedhetetlen a gépi tanulási módszereket használó szoftver elkészüléséhez. Egy LSTM architektúrára épülő modell alkalmazása lehetőséget ad az adatállomány szekvenciális feldolgozására. A projekt még az elején tart, ahol jelenleg a prioritás a piac kutatáson és a megvalósíthatóságon van. Az első mérföldkő, hogy a vállalat több üzletágából először egyen, és azon belül is, egy megadott terméken működjön az előrejelzés megfelelően. A kívánt eredmények elérése után, a kutatás következő lépcsőfoka az, hogy a további üzletágakon, és akár a vállalat egyéb országokban elhelyezkedő telephelyein is képesek legyenek a megvalósított megoldást alkalmazni.

Köszönetnyilvánítás

Hálával tartozunk felkészítő tanárainknak, Hajdu Csaba, és dr. Bognár Tamás Tanár Úrnak áldozatos munkájukért amit a dolgozat elkészülése során tanusítottak. Észrevételeik, motiválásuk, és szakmai tapasztalatuk nagyon sokat hozzatett mind tartalmi, mind stilisztikai szempontból a kutatás minőségéhez. Valamint szeretnénk megköszönni, a partner vállalatnak és külön köszönet a kapcsolattartó személynek, aki segített betekintést nyerni a vállalat működésébe. Köszönjük szépen a lehetőséget nekik és örömmel várjuk a további együttműködést a jövőben is.

Tartalomjegyzék

1. Bevezetés.....	2
2. A vállalati környezet	4
2.1 A vállalat felépítése	4
2.2 Agilis működés a vállalatban belül	5
2.3 Forecast a vállalatban belül	5
3. Szakirodalmi áttekintés	9
3.1 Neurális hálózat alapú rendszerek	9
3.2 A választott architektúra ismertetése	11
3.2.1 A választott rekurrens modell	11
3.2.2 A LSTM modell neuronjai	13
3.2.3 Az LSTM modell kapujai.....	13
4. Az előrejelzéshez fejlesztett szoftver	15
4.1 A szoftver felé támasztott igények	15
4.2 Szoftver architektúra.....	17
5. Az eredmények.....	19
5.1 A tanítás folyamata	19
5.2 Az előrejelzés pontossága.....	22
6. Konklúzió	27
6.1 Jövőbeli munka	27
ÁBRAJEGYZÉK	30
TÁBLÁZATJEGYZÉK.....	31
IRODALOMJEGYZÉK.....	32

1. Bevezetés

A nagyvállalatok effektív működéséhez elengedhetetlen a megfelelő tervezés. Ez ad lehetőséget arra, hogy az ellátási lánc minden pontja zökkenőmentesen legyen képes együttműködni. Így a cégek pontos képet kapnak arról, hogy milyen eladásra számíthatnak egyes értékesítési csatornákon a konkrét termékeik esetén, amely a láncnak a legvége az értékesítésből származó bevétel beérkezése, ami így mindent befolyásol, ami felette van, ezáltal a gyártást, szállítást, a marketinget, az értékesítést és a pénzügyet, tehát minden lépést, ami tervezést igényel. A felsorolt folyamatok konstrukciójához szüksége van a cégnek olyan számszerű adatokra, amelyek alapján ők is számszerűen képesek megbecsülni az eladásokat, akár értékben, mennyiségben, vagy meg tudják állapítani a gyárba rendelendő csomagolóanyagot például. Ez a számszerű adat a forecast, amely egy előrejelzés, amit a cégben dolgozó személyek becsülnek meg a piacról, termelési környezetből, kapacitásaikból becsült, rendelkezésre álló tudásukkal és előző éves adatokkal. Ez alapvetően is helyet ad az emberi hibáknak, mint a pontatlanság, fáradtság, érzelmi befolyásoltság vagy a szubjektív vélemény túlsúlyossága. Ezen kívül, a becslés folyamatában jelentős szerepet játszik a dolgozók tapasztalata. Azonban az elmúlt pár év szélsőséges piaci helyzeteire nem alkalmazható sem az előtte lévő években gyűjtött adat, sem tapasztalat, hiszen az igények és a trendek kritikusan megváltoztak kereskedelemben az elmúlt idők globális eseményei, valamint az amúgy is egyre hektikusabb piaci dinamika hatására. Ahhoz, hogy a cég újra megfelelően tudjon tervezni az aktuális piaci igényeknek megfelelően, agilisabban kell működnie. Ez azt jelenti, hogy nem feltétlen tudja a vállalat tartani magát ahhoz a forecasthoz, amit egy éve becsültek meg, majd esetleg minimálisan módosítottak időközben, hogyha az év folyamán jelentősen megváltoztak a piaci körülmények. Erre viszont jelenleg nem képesek, mert egy forecast megállapítása jelentős időbe és erőforrásba kerül, így, ha az idő múlásával ki is derül, hogy új tervet kell készíteniük és követniük, már nem tehetik meg, mert túl sok idő és erőforrás lenne azt kialakítani.

A kutatás célja, hogy olyan döntéstámogató rendszer (DTR) készüljön el általa, ami megoldja ezt a problémát. Tehát, a cég legyen képes agilisán forecastot előállítani, amely lehetőséget ad nekik új, alternatív stratégiákat kialakítani anélkül, hogy ebbe újabb jelentős időt, illetve emberi erőforrást fektetnének, sőt, egy, előre előkészített alternatív stratégiára való átálláshoz nagyon gyorsan tudjon forecastot módosítani. Ennek vállalatirányításon belüli támogatását, leképezését készítjük. A szoftver elkészítéséhez gépi tanulós módszerek kerültek felhasználásra. Ezen

belül is, egy Long short-term memory architektúrára épülő modell került kialakításra, amely lehetővé tette, hogy az adatokat szekvenciális sorrendben dolgozza fel a rendszer, így az képes volt tanulni az információk időbeliségéből is. Ehhez a tanításhoz nagy mennyiségű adatra volt szükség, amelyet a cég szolgáltatott a kutatás folyamán. Az elkészült mesterséges intelligencia segítségével és használatával a vállalat képes lesz újra pontos és ezáltal hasznos forecastot megállapítani, ezen a gyorsan változó piacon is. Ez pedig lehetővé teszi számukra, hogy egy olyan tervet kövessenek, amely megfelel az aktuális igényeknek és ezáltal hatékonyabban, gyorsabban és profitábilisabban működjenek.

2. A vállalati környezet

2.1 A vállalat felépítése

A kutatás által érintett vállalat üzletág alapon működik, ami azt jelenti, hogy különböző divíziókra osztható. Összesen 7 ilyen fő divízió létezik, melyek a következők:

- Csokoládé készítmények
- Konyhai készítmények
- Bébiételek
- Állateledek
- Kávé készítmények
- Fizetős termék automaták
- Gabonapehely készítmények

A kutatás folyamán fejlesztett szoftver jelenleg a magyarországi gabonapehely divízióval (MGD) foglalkozik. A gabonapehely divízió (GD) egy vegyes vállalat egy vállalatcsoporton belül, amely egy amerikai és egy európai vállalat együttműködéséből jött létre. Ebben konszernben a termékeket az amerikai cég, míg az arculatot az európai adta. Ezáltal megkülönböztetünk egy külsős és egy belsős együttműködést. A külsős együttműködés egy Európa szintű hálózat, amelynek tagja Magyarország, Csehország, Lengyelország és Szlovákia. A belsős együttműködés pedig a Magyarországon belül működő divíziók hálózata.

Az európai standardot a külsős együttműködés, tehát a közép európai csoport adja a tevékenységekre és az arculatra vonatkozóan. Ebben a klaszterban a termékek bevezetése, tervezése, dizájnolása együttes megegyezés alapján történik. Ez azért nehezíti meg a tervezési folyamatokat, mert a fogyasztói szokások jelentősen eltérőek Lengyelországban, mint például Magyarországon. Így egy termék bevezetése például nagyon nehéz feladat, hiszen szűk az a keresztmetszet, amiben minden ország igényeinek megfelelő termék keletkezhet. Ez olyannyira releváns, hogy bizonyos termékcsoporth Lengyelország forgalmának körülbelül a 40%-át teszi ki, míg ugyanez a termékcsoporth a magyar forgalomnak ez bőven 10%-a alatt van, így nehéz

egy új terméket bevezetni ebbe a csoportba, mert Magyarországon garantáltan kisebb rá az igény, mint Lengyelországban.

A belső együttműködésben a 7 divízió mátrix szervezatként épül fel, tehát a divíziók külön működnek, emellett különféle támogató osztályok közösek, mint a Sales, Human Resources (HR), Information Technology (IT) stb, emiatt az őket támogató vállalatirányítási rendszer is ugyanaz, ugyanazzal a logisztikai modullal. Ez újabb komplexitást idéz elő, mivel nem lehet csak az MGD igényeire specializálva alkalmazni ezeket, muszáj összehangolva használni őket, ami a divíziók külön-külön teljesítésére negatív hatással lehet.

2.2 Agilis működés a vállalaton belül

A GD egy fast-moving consumer goods (FMCG) termék kategória, ami azt jelenti, hogy relatív olcsó és relatív gyakran vásárolt termékek csoportja. Ez magában hordozza azt, hogy egy hiba a tervezésben és kivitelezésben hamar észrevehetővé válik a számokban. Ez előnnyé követhető szigorú monitorozás alkalmazásával, mivel a hibák így elméletileg gyorsan javíthatók. Azonban ez okozza a nehézséget is: egy hirtelen kiderült problémára a döntéshozóknak hirtelen kell reagálniuk. Ez egyrészt azért nehéz, mert, egy új tervet kellene készíteniük, amit azonban nagyon körülményes gyorsan és effektíven megtenni, idő vagy emberi erőforrás hiányában. Másrészt nem is könnyű felismerni ennek szükségességét, ehelyett könnyen adódik a tervek nem teljesüléséért a végrehajtás tervezés „hibáját” keresni. Tovább nehezíti ezt a feladatot a folyamatos együttműködés igénye a vállalaton és a klaszteren belül is.

Ez különösen veszélyes a GD-re nézve, mivel főként 2 termékcsoporthoz van: müzlik, amik a forgalom körülbelül 85%-át teszik ki, és szeletes müzlik, ami a forgalom fennmaradó 15%-át. Ugyanez a szituáció sokkal kevésbé veszélyes a konyhai készítmények divíziójára nézve, mivel ez az üzletág jelentősen több termék kategóriából áll, így, ha az egyik termékcsoporthoz forgalma zuhan egy hiba miatt, az a bevételnek és a portfóliónak egy jelentősen kisebb hányadát teszi ki, valamint könnyen kiegyensúlyozza a forgalmat a többi termék.

2.3 Forecast a vállalaton belül

A forecast az az adatokból álló számszerűsített információ, ami képes előre jelezni a döntéshozóknak az eladások számát a jövőben. Enélkül nem lenne képes a cég tervezni, hiszen nem tudnák előre jelezni a gyárnak, hogy milyen évi termelésre számíthatnak, mennyi csomagolóanyagot rendeljenek az adott hónapokra. Nem lenne lehetőségük felosztani a

marketing költségvetést a divíziók között, havi szinten, termékekre lebontva. Ez képezi az alapját minden tervezésnek a vállalaton belül, és elengedhetetlen a megfelelő és kiszámítható működéshez.

A forecastolás bázis alapon történik. Ez azt jelenti, hogy amikor elkészül, akkor van egy alapszám, amiből kiindul a tervezés, ez a bázis. A tervezés folyamán résztvevő szerepkörök képviselői ezután közös munkával megbecsülik, hogy az olyan paraméterek, mint a termékre szánt promóció, marketing stratégia, sales stratégia, hogyan befolyásolják ezt a bázisszámot. Ez az alapszám, az előző éves valós eladásokból származik, tehát nem a forecastokból. Ennek előnye, hogy egy nagyságrendileg helyes becslést kapunk előző évekből kiindulva. A hátránya viszont pont ugyan ebben mutatkozik meg, mivel olyan alapértékből kell kiindulnia a tervezésnek, ami nem az aktuális piaci helyzetet veszi figyelembe, hanem a múltban lévő. Így, hiába is jön egy olyan piaci változás, ami gyökeresen megváltoztatná ezt a bázisértéket, az egész évre nem tudnak már ezzel az új számmal tervezni, de a hónapokra sem, mivel a havi forecast értékek az évi lebontása hónapokra.

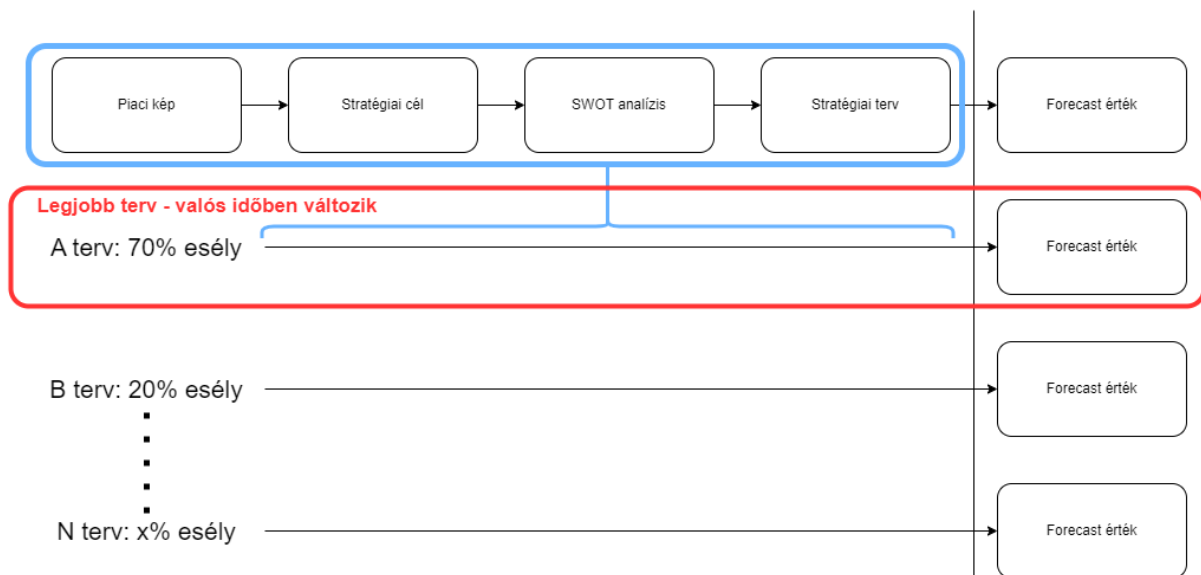
Megkülönböztethető éves és havi forecastolás. Az éves forecastolás egy hónapokon át tartó tervezési folyamat, aminek célja, hogy a GD az egész évre megbecsülje előre a várható eladásokat. Ezt megteszi a klaszteren belül lévő többi ország képviselője is, és feltöltik az informatikai rendszerbe, amin keresztül a gyár is látja ezeket az igényeket, így előre tudnak tervezni, hogy van-e annyi rendelkezésre álló gép, erőforrás, hogy minden ország igényét teljesítsék. Itt is látható, hogy a közös együttműködés az európai csoport többi tagjával negatívumokkal járhat a vállalat számára, hiszen előfordulhat, hogy a kiszámított forecasthoz nem képesek tartani magukat, mert a gyár azt jelzi vissza, hogy nem tud annyi terméket legyártani, mint amennyit kért a cég. Ez újabb forecast tervezését igényli, ami újabb idő és erőforrás költség.

Az éves forecast tervezése május környékén kezdődik. Először ezen, a májustól szeptemberig tartó időszakon előre kell jelezniük az egész éves eladásukat forecast segítségével. Ezután szeptembertől novemberig elkészítésre kerül a havi lebontás erről a becsült évi mennyiségről. Eközben júliusban már eljuttatják a gyárnak az igényt arról, hogy milyen gépfelhasználás várható a termékek terén és mennyi csomagolóanyagra van szükség. Ezt minden egyes termékre meg kell tenni, és azok minden létező kiserelésének variációjára is.

A havi forecastolás folyamata a klaszter helyett csak az MGD-n folyik. Ez egy minden hónapban megtartott meeting sorozat, ahol felülvizsgálják az aktuális hónap és a soron következő 3 hónap forecastjait termékre lebontva és ezen belül is kiszerezésre lebontva. Amennyiben arra a megállapításra jutnak, hogy a korábban megbecsült forecast már nem megfelelő az éppen vizsgált hónapra, azt megváltoztatják. Ezeket a havi forecastokat negyedévente revízionálják, hogy mennyire voltak sikeresek. Egy becsült érték jónak tekinthető a vállalat értékrendje szerint, ha az maximálisan 10%-kal tér el a valós eredménytől. Elfogadhatónak, de már egyáltalán nem jónak, pedig 28%-os eltérésig tekintik.

Amennyiben hirtelen változás történik a piacon, lehet ez a konkurencia egy stratégiai lépése, vagy egy globális esemény, ez új forecastot igényelhet azonnal. A jelenlegi rendszerben csupán arra van lehetőségük, hogy a jelenlegi hónapra megállapított forecastot minimálisan változtassák, azonban teljesen újat egyáltalán nem képesek készíteni.

Ezeket a nehézségeket felismerve a vállalat MGD csapata próbálkozott már egy agilisabb működéshez vezető lépéssel, ami sikeres is volt. Kidolgoztak egy stratégiát (1. Ábra) egy új termék bevezetéséhez, amiben a forecast elkészítéséhez először piaci képet vizsgálnak, ezután elkészítik a stratégiai célt, majd SWOT-analízist végeznek, végül pedig elkészül a stratégiai terv. Ezek segítségével megbecsülik a forecast értékét. Ezt a folyamatot többször megismétlik különböző fiktív gazdasági helyzetekre, így több forecasttal rendelkeznek több szituáció esetére. Ezekhez a különböző tervekhez becsülnék egy valószínűségi értéket is a bekövetkezésükre, aminek segítségével arra is tudnak készülni, hogy melyik terv bekövetkezése valószínűbb, ezáltal milyen irányba várhatóbb a forecast eltolódása. Ez a párhuzamos folyamatos tervezés esélyt adott nekik arra, hogy a termék bevezetésekor tudjanak reagálni a piaci változásokra időben, késlekedés nélkül, és hogy ne legyen akadálya az új forecast készítésének és alkalmazásának az aktuális emberi és időbeli erőforrás, hiszen az új előrejelzésekkel már rendelkeztek azelőtt, hogy szükség lett volna rájuk.



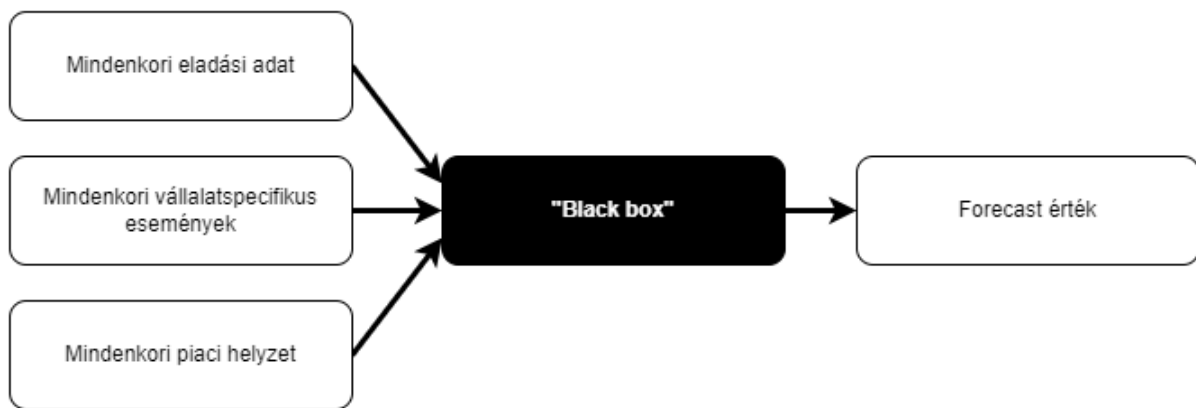
1. Ábra – A cég által tervezett stratégia Forrás: Saját szerkesztés

A stratégia eredményes volt. A termék bevezetése a GD-n belül Magyarországon volt egyedül sikeres, ami a piac gyors lekövetésének köszönhető. A kialakított stratégia egyetlen hátránya, hogy több kezdeti erőforrást és kommunikációt igényel, ami növeli a cégen belüli komplexitást, amit a nagyvállalatok tipikusan próbálnak elkerülni, mivel több hiba lehetőségnek ad helyet. A stratégia sikeressége egyúttal rávilágít a kutatás aktualitására és bebizonyítja, hogy ezen a hektikus piacon a tervezett szoftver által biztosított szolgáltatásra van igény a valós piaci felhasználásban.

3. Szakirodalmi áttekintés

3.1 Neurális hálózat alapú rendszerek

A megvalósítandó funkció és megoldandó probléma jellegéből kifolyólag a kutatás elején egy gépi tanulós megoldás került kiválasztásra [1]. Ez a típusú szoftver több indokból is megfelelő választás egy ilyen feladatra: be tud befogadni input paramétert egy felhasználótól, ami képes leírni az aktuális piaci helyzetet, ezt az inputot fel tudja dolgozni előző éves ismeretek alapján, tehát ismeri a piaci trendeket, ezáltal pedig értelmes számszerű forecast adatot tud adni a felhasználónak. (2. Ábra)



2. Ábra - A rendszer "Black box" alapú működése Forrás: Saját szerkesztés

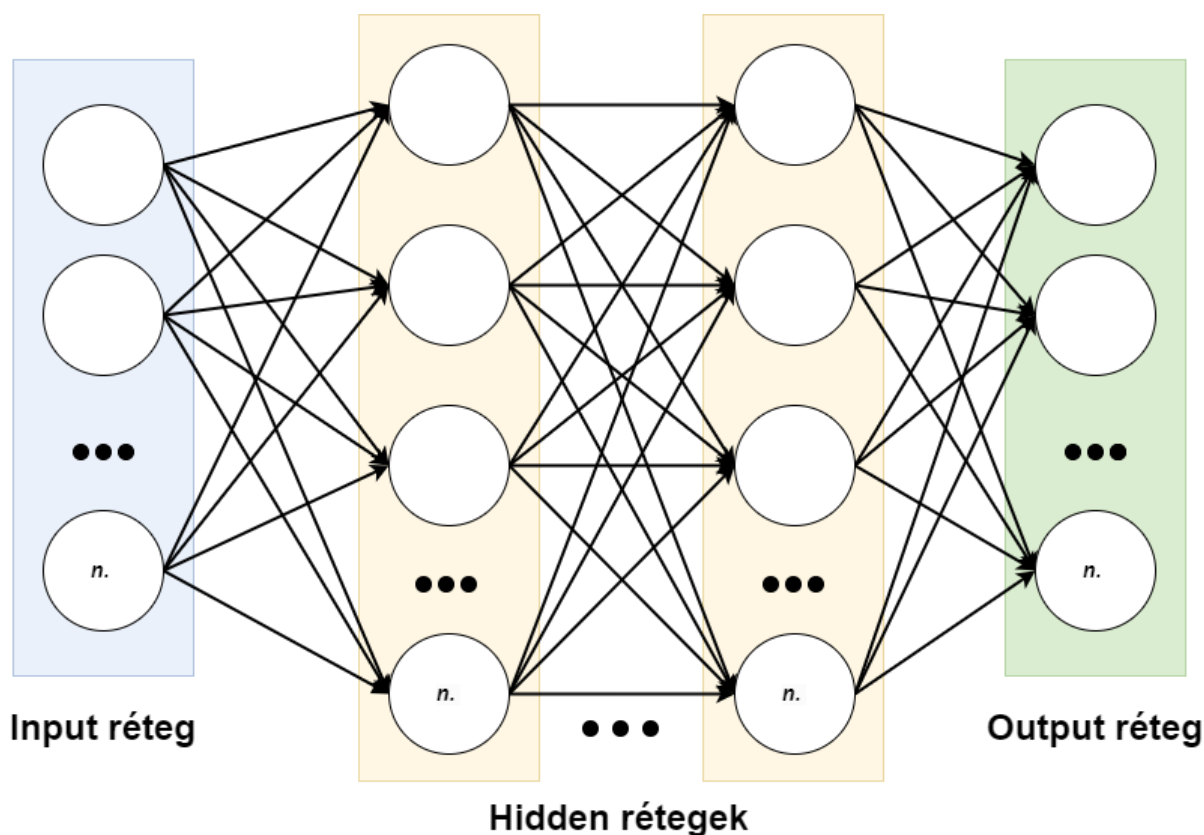
Ahhoz, hogy egy ilyen modell ismerhesse a piaci trendeket és megfelelően tudjon működni, gépi tanulásra van szüksége. Ez azt jelenti, hogy a szoftvernek kialakul egy belső értékelési rendszere, ami képes úgy feldolgozni a bemeneti paramétereket, hogy a kimeneti közelítése az elvárthoz közel legyen.

Egy ilyen modell tanításához nagy mennyiségű adatra van szükség. A céltól függően ez az adat lehet kép, szöveg vagy számszerű adat is akár [2]. A kutatás folyamán az adatokat a vállalat biztosítja, ezek a cégnek évekre visszamenően tárolva vannak az informatikai rendszerükben, az SAP-ban. Számszerű formátumban vannak tárolva, és pontosan szűrhetőek egy adott évre, hónapra, hétre, illetve termékekre és ezen belül kiszűrésre is. A kutatás és a szoftver hatékonysága szempontjából ez nagyon előnyös, mivel minél több információval van ellátva a rendszer, annál pontosabb predikciókra lesz képes.

Ez a tanítás egy mély neurális hálózaton keresztül megy végbe (3. Ábra). Egy ilyen mély neurális háló az architektúrától függően kevesebb-több rétegnyi neuronból áll. A neuronok a hálózatok építőelemei, ezek felelősek az adat ideiglenes tárolásáért, illetve feldolgozásáért. A választott architektúrától függően különböző kommunikáció lehetséges ezek között az adatfeldolgozó egységek között a rétegekben belül. A rétegeket feloszthatjuk 3 fő részre:

- **Input réteg:** a bemeneti adat vektorizálásra kerül és a mélyebb rétegekben fejlődik tovább.
- **Hidden rétegek:** a megkapott vektorizált adatok feldolgozásra kerülnek az LSTM cellák által és kiszámítódik a kimenet.
- **Output réteg:** megkapja a kimeneti adatokat. Ezután inicializálhatja a hiba visszaterjesztést a súlyozások és eltolások változtatásához.

Az input réteg feladata, hogy befogadja a feldolgozandó adatokat, a kutatás esetében az SAP-ból kinyert excel formátumban tárolt adatokat. Ezután tovább küldi az információt a hidden rétegekbe, amik számításokat hajtanak végre a kapott adatokon. A kapott eredmények az output rétegbe kerülnek a hidden rétegekből, ahol ezután készen állnak további felhasználásra, mint végeredmény a bemenő inputra.



3. Ábra - Illusztráció egy mély neurális hálózat architektúrájáról Forrás: [3] nyomán saját szerkesztés

Ha a tanítás sikeres, akkor a szoftvernek képesnek kell lennie egy bemenő adatot feldolgozni a belső számításai segítségével, és olyan kimenetet adnia, ami elvárt tőle. A kutatás esetében, tesztelhető a szoftver tanításának sikeressége, hogyha input adatként megkap egy múltból származó t időponthoz tartozó adatot, majd pedig prognosztizálja a $t + 1$ időponthoz tartozó adatot, ami összevethető a valós $t + 1$ időponthoz tartozó adattal.

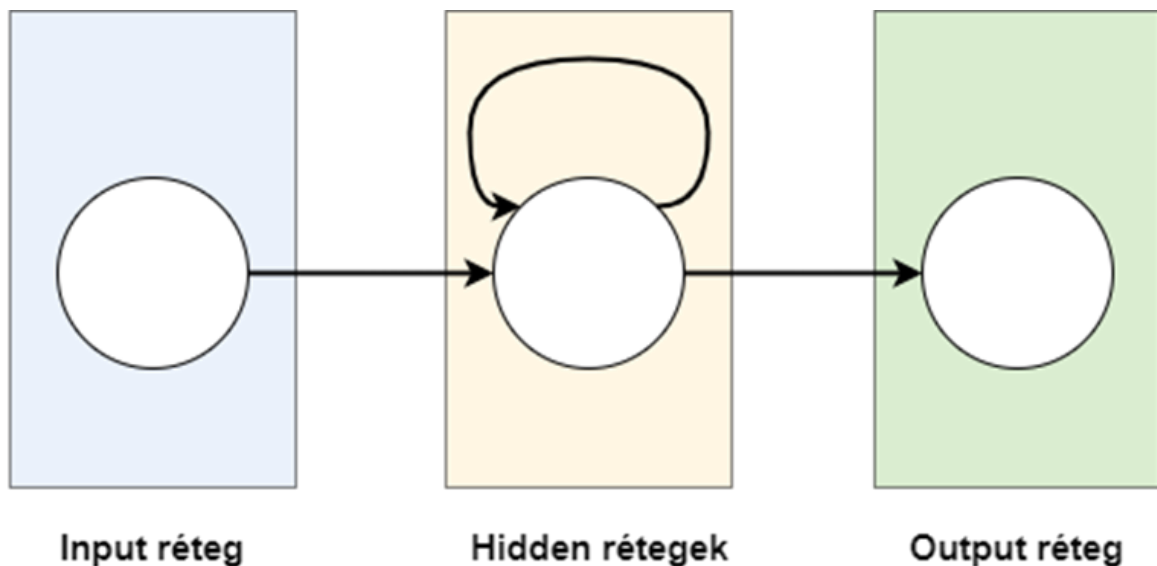
3.2 A választott architektúra ismertetése

A kutatás folyamán készített szoftvernek rendelkeznie kell azzal a képességgel, hogy figyelembe vegye az adatok időrendiségét. Ez azt jelenti, hogy meghatározó szerepe van annak, hogy egy hónap eladásainak adatai a valós sorrendben kerülnek feltanításra, vagy egy kevert sorrendben például. Ez azért jelentőségteljes, mert így tud a modell trendeket felfedezni. Ehhez a választott architektúrának képesnek kell lennie figyelembe venni az előzőleg kezelt adatokat az új adatok feldolgozása közben. Gyakorlatban ez úgy valósul meg, hogy ha egy hónap adatai feldolgozásra kerülnek az első héttől a negyedikig bezárólag, az első hét feldolgozása befolyással lesz a második, harmadik és negyedik hét feldolgozására is.

3.2.1 A választott rekurrens modell

A gépi tanulós modellek esetén ez a működés úgy érhető el, ha egy rekurrens modellt választunk (4. Ábra) [4], [5]. Elsődlegesen ez abban különbözik egy hagyományos neurális háló felépítésétől, hogy a neuronok rendelkeznek még egy önmagukba visszacsatoló éllel, amelynek úgyszint van egy saját súlyozása és eltolása. Ez azt jelenti, hogy a hidden réteg neuronjai rendelkeznek egy memóriával, ami mindig befolyásolja az éppen aktuálisan feldolgozott input adatot. Tehát például a D_{t-2} befolyásolja D_{t-1} -et, ezáltal indirekt módon befolyásolja D_t -t is, hiszen D_{t-1} befolyásolja D_t -t.

Egy adathalmazon való tanítás folyamán a visszacsatoló élek súlyozása és eltolása nem változik. Ez azt eredményezi, hogy a gradiens exponenciális növekedésbe vagy csökkenésbe kezd, attól függően, hogy a súlyozás értéke alacsonyabb vagy magasabb mint 1. Ezt az exponenciális változást felrobbanó vagy elhalványuló gradiens [6] problémának hívják. A gradiens túl nagyra vagy túl kicsire változása ellehetetleníti a backpropagation (BP) algoritmus működését, így a modell tanítását is [7], [8].



4. Ábra - Illusztráció egy rekurrens neurális hálózataarchitektúráról Forrás: [9] nyomán saját szerkesztés

A BP egy olyan gépi tanulási algoritmus, ami a hálózaton belüli súlyozásokat és eltolásokat a hiba visszaterjesztés segítségével közelíti az optimumhoz. Többféle módosított verziója létezik. Az időbeli adatsorok feldolgozására és a rekurrens hálózatok esetén a backpropagation through time (BPTT) nevű modifikált verzió kerül alkalmazásra a legtöbb esetben. Ez annyiban különbözik a hagyományos BP-től, hogy a tanítás időben is szétterített.

Az algoritmus működése a következő módon írható le: miután a bemeneti adatokat feldolgozza a neurális háló, a kapott kimenetet összehasonlítja a valós várt eredménnyel. Ennek a két értéknek a különbsége a hiba. Az algoritmus hátulról előre haladva, tehát az output rétegtől az input réteg irányába visszaterjeszti a hibát. A hiba mérete befolyásolja a súlyozások és eltolások változtatásának mértékét. Ha a hiba visszaterjesztése megtörtént, a neurális hálózat összes súlyozása és eltolása frissítésre kerül. Az értékek megváltoztatásához különböző optimalizációs technikák is alkalmazhatóak, mint például a gradient descend.

A szoftver elkészítéséhez a fentiek ismeretében, nem megfelelő egy hagyományos rekurrens neurális hálózat alkalmazása. A kutatáshoz így egy Long short-term memory (LSTM) architektúra alapú modell került kiválasztásra [10]. Az LSTM architektúra lehetővé teszi az adatok szekvenciális feldolgozását, így lehetőséget adva a trendek megtanulására a modellnek, anélkül, hogy előállna a felrobbanó vagy elhalványuló gradiens problémája.

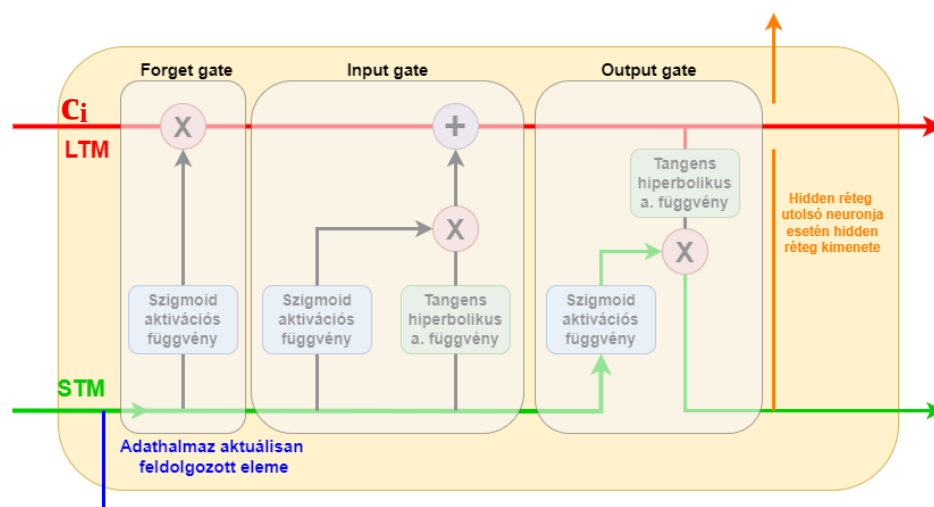
3.2.2 A LSTM modell neuronjai

Az LSTM architektúra fő változtatása a neuronjai felépítésében található. Ezek az átalakított cellák úgy képesek feldolgozni az adatot, hogy nem kezdenek exponenciális változásba, továbbá képesek megőrizni a rekurrens hálók emlékezőképességét, tehát $Dt - n$ -ik érték továbbra is befolyásolja a Dt -ik értéket. Az ilyen viselkedés eléréséhez a neuronok egy él helyett kettőn szállítják az információt, ez a long-term memory (LTM) és a short-term memory (STM). Ezek az élek helyettesítik a visszacsatolást, és ezeken keresztül történik az információ áramlása a neuronok között. Az LTM célja, hogy tárolja a feldolgozott adatok fontosnak ítélt részeit, míg az STM célja, hogy meghatározza melyek a fontos részek, amiket az LTM-nek továbbra is tárolnia kell, és melyek azok az új jelentős információk, amiket hozzá kell adni ezekhez az LTM által tárolt információkhoz.

3.2.3 Az LSTM modell kapujai

Az LSTM architektúra felépítése azon kívül, hogy több élen áramlik az információ, még a neuronok belső felépítésében is különbözik (5. Ábra). Az STM kapuk felhasználásával képes eldönteni, hogy milyen információ kerüljön megtartásra, milyen információ adódjon hozzá az LTM-hez, és ennek az új információnak hány százaléka adódjon hozzá az LTM-hez. Ezek a kapuk a következők:

- Forget gate
- Input gate
- Output gate



5. Ábra - - Az LSTM cella felépítése Forrás: [11] nyomán saját szerkesztés

A forget gate fogadja az adathalmaz aktuálisan bemenő input értékét, az LTM szálon érkező adatot és az STM szálon érkező adatot. Amennyiben a hidden rétegek első rétegébe tartozik ez a neuron, akkor az LTM és az STM értéke 0, mivel a neuronnak nincs megelőző hidden rétegbeli cellája. A forget gate-n belül az aktuálisan bemenő érték és az STM szálon bemenő érték külön-külön megszorozódik a súlyozásukkal, majd összeadásra kerülnek, és végén hozzáadódik az él eltolása. A kapott érték bemenetként szolgál egy szigmoid aktivációs függvénynek, amely ezt az értéket egy $]0,1[$ intervallumba eső számmá alakítja. Ez a kapott érték határozza meg azt, hogy az LTM szálon érkező információ hány százalékát őrizzük meg. Tehát, összeszorozódik a LTM szálon bejövő érték ezzel a százalékos értékkel, ezáltal frissül az LTM értéke.

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f)$$

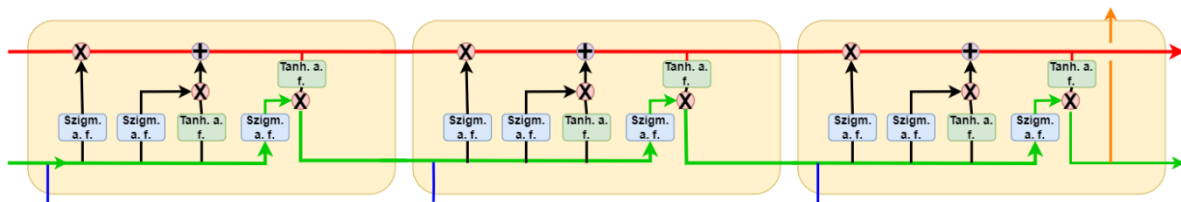
Az input gate fogadja az adathalmaz aktuális bemenő input értékét és az STM szálon érkező értéket. Ezeket felhasználva az input gate kiszámítja, hogy milyen potenciális értéket adjunk hozzá az LTM-hez, és ennek a potenciális értéknek hány százalékát adjuk hozzá. Ehhez a kapu 2 aktivációs függvényrel rendelkezik: a már megismert szigmoid aktivációs függvényvel és egy tangens hiperbolikus aktivációs függvényvel. Előbbi felelős a hozzáadandó érték százalékának kiszámításáért, míg utóbbi a hozzáadandó potenciális érték kiszámításáért. Ezek bemeneti paraméterét úgy képezzük, hogy az aktuális bemeneti érték és az STM szálon bemenő érték külön-külön megszorozódik a súlyozásukkal, majd összeadásra kerülnek, és a végén hozzáadódik az él eltolása. A tangens hiperbolikus aktivációs függvény az inputot egy $] -1,1[$ intervallumba eső számmá alakítja. Ennek a két eredménynek összeszorozásával megkapjuk az LTM-hez hozzáadandó értéket, ami egyúttal az új LTM érték, amit a következő cella forget gate-je és az aktuális output gate fog megkapni, mint bemenet.

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i)$$

Az output gate fogadja az adathalmaz aktuális bemenő input értékét, az STM szálon érkező értéket és az újonnan kiszámolt LTM értékét. Ezeket felhasználva a kapu kiszámítja, hogy milyen érték haladjon tovább az STM szálon. Amennyiben ez a neuron a hidden réteg utolsó rétegében van, ez lesz az output réteg bemenete is, ha nem, akkor a következő LSTM cella forget gate-nek a bemeneti STM szálon futó értéke. Az output gate 2 aktivációs függvényrel rendelkezik hasonlóan az input gate-hez: az első egy szigmoid aktivációs függvény, ami azért felelős, hogy kiszámítsa, hogy a potenciális érték hány százaléka adódjon hozzá az STM-hez.

Ennek bemenete úgy képződik, hogy az aktuális bemeneti érték és az STM szálon bemenő érték külön-külön megszorozódik a súlyozásukkal, majd összeadásra kerülnek, és a végén hozzáadódik az él eltolása. A második aktivációs függvény a tangens hiperbolikus, aminek bemenete az LTM szálon futó új érték. A függvény ezt az inputot egy $]-1,1[$ intervallumba eső számmá alakítja, ami az STM potenciális hozzáadandó értéke. A két aktivációs függvény kimenetét összeszorozva, majd ezt az értéket hozzáadva az eddigi STM-hez megkapjuk az új STM szálon futó értéket. Ha az eredményt számító cella a hidden réteg utolsó neuronja, akkor ez a hidden réteg kimenete is. Amennyiben nem, akkor ez az STM szálon futó érték a következő cella forget gate-nek a bemenete (6. Ábra).

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o)$$



6. Ábra - - Kaszkádosított cellák Forrás: [11] nyomán saját szerkesztés

4. Az előrejelzéshez fejlesztett szoftver

4.1 A szoftver felé támasztott igények

Követelmény, hogy a szoftver be tudjon fogadni egy adathalmazt a tanulás, validálás, és előrejelzés végett. Kell, hogy legyen megfelelő mennyiségű adatunk, és karakterisztikánk, mivel a szoftver csak így tud megfelelő mintákat keresni, hogy később pontos előrejelzéseket hajtson végre. Az adatokat át kell tudnunk alakítani a megfelelő formátumra, ez esetünkben $]-1;1[$ közé eső skálázást, és one-hot encoding alapú kódolást jelent, az eladási adatok mennyiségét, illetve helyét illetően. Az átalakított adatokat tovább kell tudnunk LSTM formátumra alakítani, ez magába foglalja azt, hogy szekvenciális sort készítünk belőlük 3 hét eladásait összefogva, és ezután két egységre bontjuk az adathalmazt, egy tanítási és egy tesztelési blokkra. Ezt követően a két halmazunk alakját LSTM formátumra kell módosítani, ez tartalmazza az adat mennyiségét, azt, hogy hány hét adatát egybefogva és mennyi karakterisztikát használva tanítjuk fel a modellünket. A szoftvernek tudnia kell

paraméterezhetően (batch méret, epoch szám, hidden unit number) feltaníthatónak lennie, tehát a paraméterek módosításával változnia kell a fentebb felsorolt adatoknak is, és vele együtt a tanulás minőségének is. A tanítás közben el kell tudnunk tárolni az adatokat a tanítási és a validációs görbéről. A tanulás végeztével elmentjük a modellt, és grafikonon kirajzoljuk a tanulási és validációs görbe egymáshoz való viszonyát és tendenciáját a használt epoch számok függvényében. A grafikonok lementésre kerülnek, mivel az egymáshoz való relatív viszonyításuk tükrözi a modell tanításának minőségét. A feltanított modellel elvégezzük az előrejelzéseket. Fontos lépés, hogy visszaalakítjuk az adatokat az eredeti skálájuk szerint a]-1;1[-es tartományból, majd ezek az adatok lementésre kerülnek egy külön fájlba. Ezután a valós adatokból és az előrejelzettekből Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE) értékeket [12] számolunk és az így kapott eredmények lementésre kerülnek egy külön fájlba.

A képletekben az $\hat{y}$ az előrejelzett értéke az y -nak, és $\bar{y}$ pedig az átlaga az y -nak

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}|$$

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y})^2$$

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y})^2}$$

$$R^2 = 1 - \frac{\sum (y_i - \hat{y})^2}{\sum (y_i - \bar{y})^2}$$

A kapott előrejelzésekhez hozzárendeljük a dátumokat, amiket előzetesen az adatok jellege miatt levágtunk. Kiíratjuk grafikonon az egész adathalmazt (előrejelzett és valós adatok) az eladás szerint a sorba rendezett dátumok függvényében, amit ezután kép formájában lementünk. Összehasonlítás végett aggregáljuk a valós és előrejelzett adatokat, hogy egymáshoz lehessen hasonlítani őket, majd a kapott eredményt a hónapok függvényében kirajzoljuk egy grafikonon és lementjük kép formájában.

4.2 Szoftver architektúra

Az LSTM alapú előrejelzésért felelős rendszer a Visual Studio Code integrated development environment (IDE) használatával készült Python nyelven. Az IDE-n belül Jupyter Notebook környezetben került fejlesztésre a szoftver, mivel támogatja a cella alapú futtatást, szövegkiemelést, fájl előnézet-megnyitást, kód kiegészítést és egyéb kényelmi funkciókat.

Az adathalmaz egy publikált adatokat tartalmazó Walmart eladási összesítés volt, ami két évet fedett le az üzletlánc eladási számaiból (dollárban), valamint tartalmazott hozzárendelten dátumokat, üzlethelyeket (számokkal kódolva), régiós üzlethelyeket (számokkal kódolva), és azt, hogy Black Friday volt-e az adott nap vagy nem (1. Táblázat). A fentebb említett jellemzőkből alakultak ki a karakterisztikák, ami az alapja volt a tanításnak.

Store	Department	Date	Weekly Sales	IsHoliday
1	1	2010.02.05	24924.5	FALSE
1	1	2010.02.12	46039.49	TRUE
1	1	2010.02.19	41595.55	FALSE
1	1	2010.02.26	19403.54	FALSE
1	1	2010.03.05	21827.9	FALSE

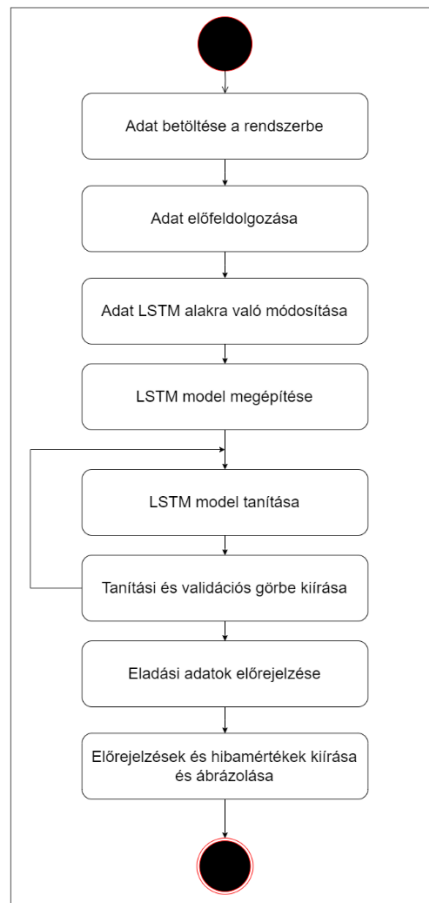
1. Táblázat - Az adatok eredeti jellege Forrás: Saját szerkesztés

A felhasznált technológia a pythonon belül, az adatfeldolgozásra pandas és numpy volt, a különböző megjelenítésekre és diagramokra pedig matplotlib-et használ a rendszer. Az adatok LSTM formájába való átalakításért a Tensorflow [13] felel, valamint magáért a modell felépítéséért a Keras technológia nyújtotta eljárásokat használja a szoftver. A rendszer 6 nagyobb részből áll. Először is a lementett adatot kell betölteni, aztán jön az előfeldolgozás rész, ahol $]-1;1[$ határok közé skálázzuk le a bevételt (2. Táblázat), valamint a nem bináris adatokat átalakítjuk.

Date	Store	Department	Weekly Sales	IsHoliday
2010.02.05	1	1	0.042851	FALSE
2010.02.12	1	1	0.073097	TRUE
2010.02.19	1	1	0.066732	FALSE
2010.02.26	1	1	0.034942	FALSE
2010.03.05	1	1	0.038415	FALSE

2. Táblázat - Az adatok jellege a $]-1;1[$ skálázás után Forrás: Saját szerkesztés

Ezután LSTM formára hozzuk az adatokat, majd felépítjük a modellünket. Miután ez megtörtént, elvégezzük a tanítás és validálás fázist, ami után a rendszer képes lesz előrejelzéseket generálni. A kapott értékekből pedig további hibaértékeket, illetve grafikonokat jeleníthetünk meg, hogy tovább elemezzük a rendszerünk pontosságát és precizitását. (7. Ábra)



7. Ábra - A szoftver architektúrája Forrás: Saját szerkesztés

5. Az eredmények

5.1 A tanítás folyamata

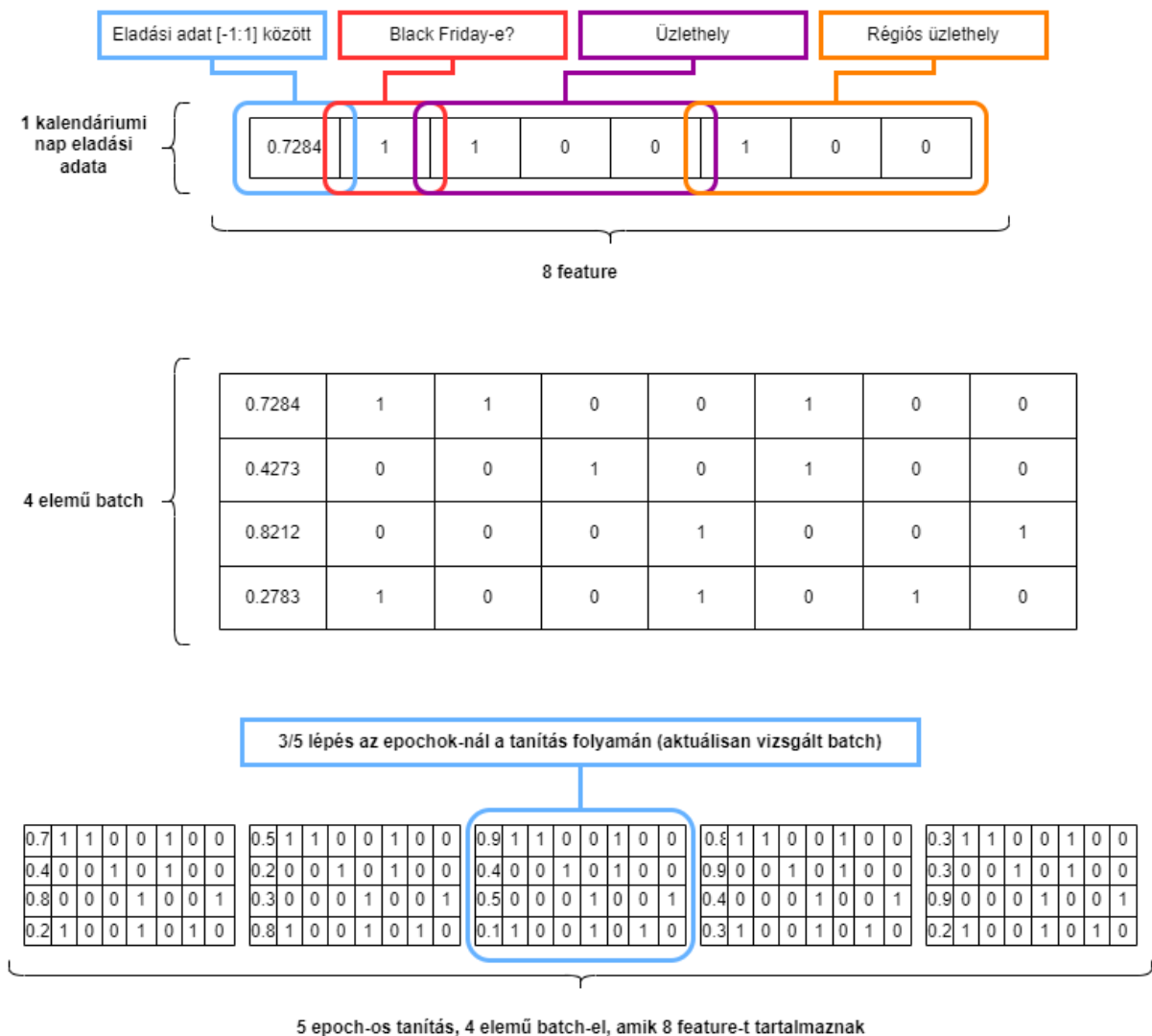
Az elkészült szoftver a fentebb leírtak szerint egy Jupyter Notebookbeli Python program, ami teljes terjedelmében 292 sort tesz ki. Ez tartalmazza az összes feldolgozási műveletet, az adatforrás betöltésétől, egészen az előrejelzés előállításáig. Az adatforrás neve „train.csv”, ez egy 421571 adatsort tartalmazó adatforrás, amit a Walmart publikált kutatási céllal (3. Táblázat). Ez tartalmazza a kalendáriumi napokat, és a hozzájuk tartozó eladási információkat. A szoftvernek az adatforrás ketté választása után az „ismeretlen” napokat kell helyesen előrejelezni.

Adathalmaz neve	Adatok száma	Karakterisztikák	Összesített hetek száma	Magyarázat	Forrás
Tanítási adathalmaz	337253	128	3	Felelős a tanítás elvégzéséért	Walmart adathalmaz
Teszt és egyben validációs adathalmaz	84314	128	3	Felelős az adatok teszteléséért	Walmart adathalmaz

3. Táblázat - Az adathalmazok jellege Forrás: Saját szerkesztés

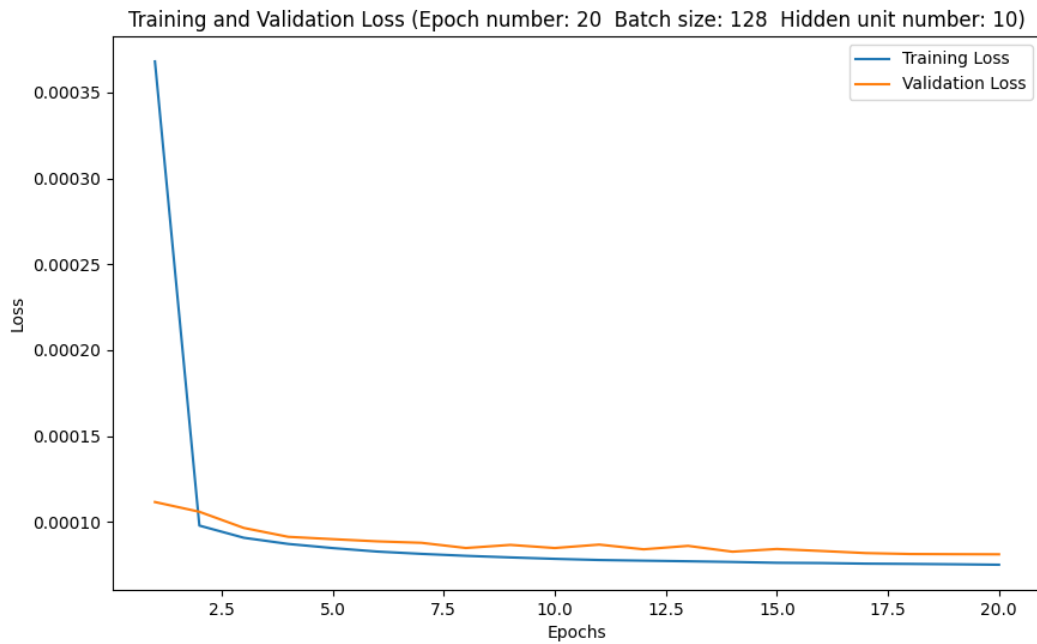
A tanítás fázis sokszor lefuttatásra került, hogy tesztelve legyen a modell számára optimálisnak mondható paraméterezés. A validációs és tanítási görbékre lett meghozva a döntés, hiszen a tanítás ezen szakaszában, ezek tükrözik a legjobban azt, hogy mennyire lesz pontos a későbbi előrejelzés. Minden ehhez szükséges grafikon a fentebb említettek szerint lementésre került. 3 féle paramétert állíthatunk különböző méretekre a tesztek során, ezek a batch size, tehát hogy mekkora adatcsomagok kerülnek feldolgozásra 1 epoch alatt. Az epoch numbert, ami azért felelt, hogy hányszor iteráljon át a tanítási adatbázison a szoftver, és a hidden unit numbert, ami pedig a rejtett karakterisztikák számát határozza meg. Fontos volt, hogy az over fitting-et, és az under fitting-et kiküszöböljük a tanulás során, mivel a paraméterek hibás beállítása esetén mindkettő bekövetkezhet. Karakterisztikából 128 van, ezek tartalmazzák az eladási adatokat,

az üzlet helyét, hogy régiósan melyik üzlethez tartozik (one hot encoding alapú kódolás), és azt, hogy Black Friday akció van-e aznap éppen vagy nem (8. Ábra).



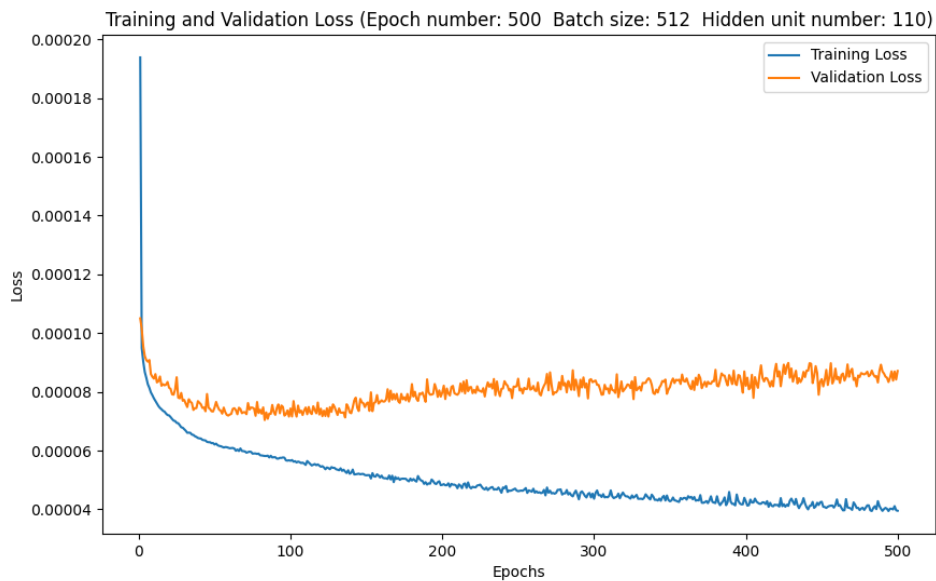
8. Ábra - Összefoglaló ábra a feature, batch, és epoch fogalmak felépítéséről Forrás: Saját szerkesztés

Underfitnek azt hívjuk, amikor a modell túl egyszerű, és nem tud mintázatokat felismerni az adatok között, ilyenkor a tanítási és validációs görbe nagyon ellaposodik és egymáshoz ér. (9. Ábra)



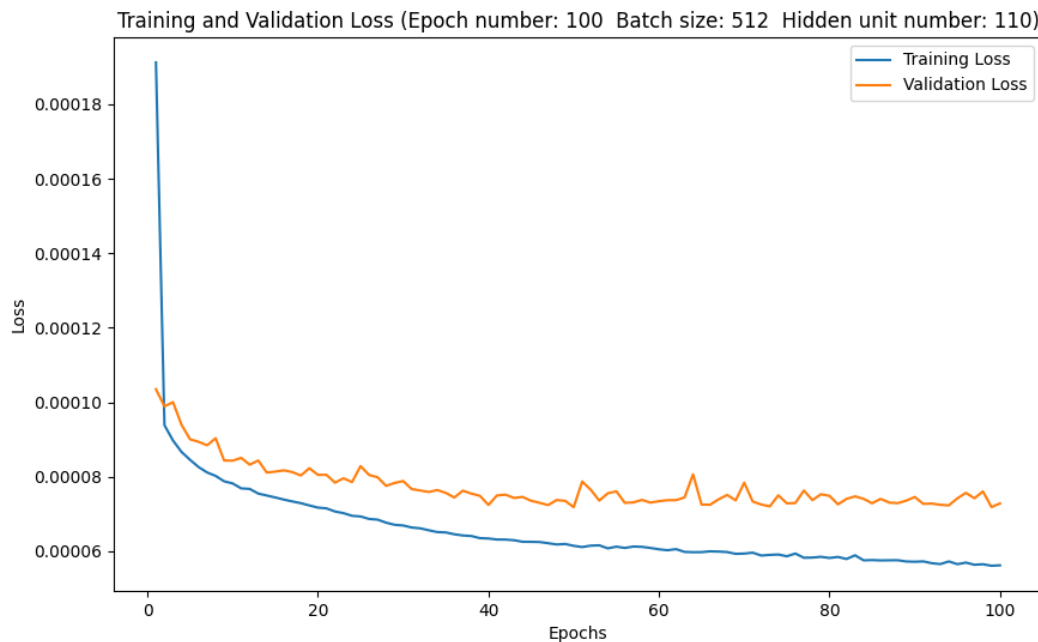
9. Ábra - Példa az underfitting esetére Forrás: Saját szerkesztés

Az overfitting esete pedig, amikor a modell túlságosan megtanulja az adathalmazt, és fals pozitív jeleket kezd el előrejelzni, tehát jó megoldásokat hallucinál a valósághoz képest, ilyenkor a tanítási és validációs görbe eltávolodik egymástól. (10. Ábra)



10. Ábra - Példa az overfitting esetére Forrás: Saját szerkesztés

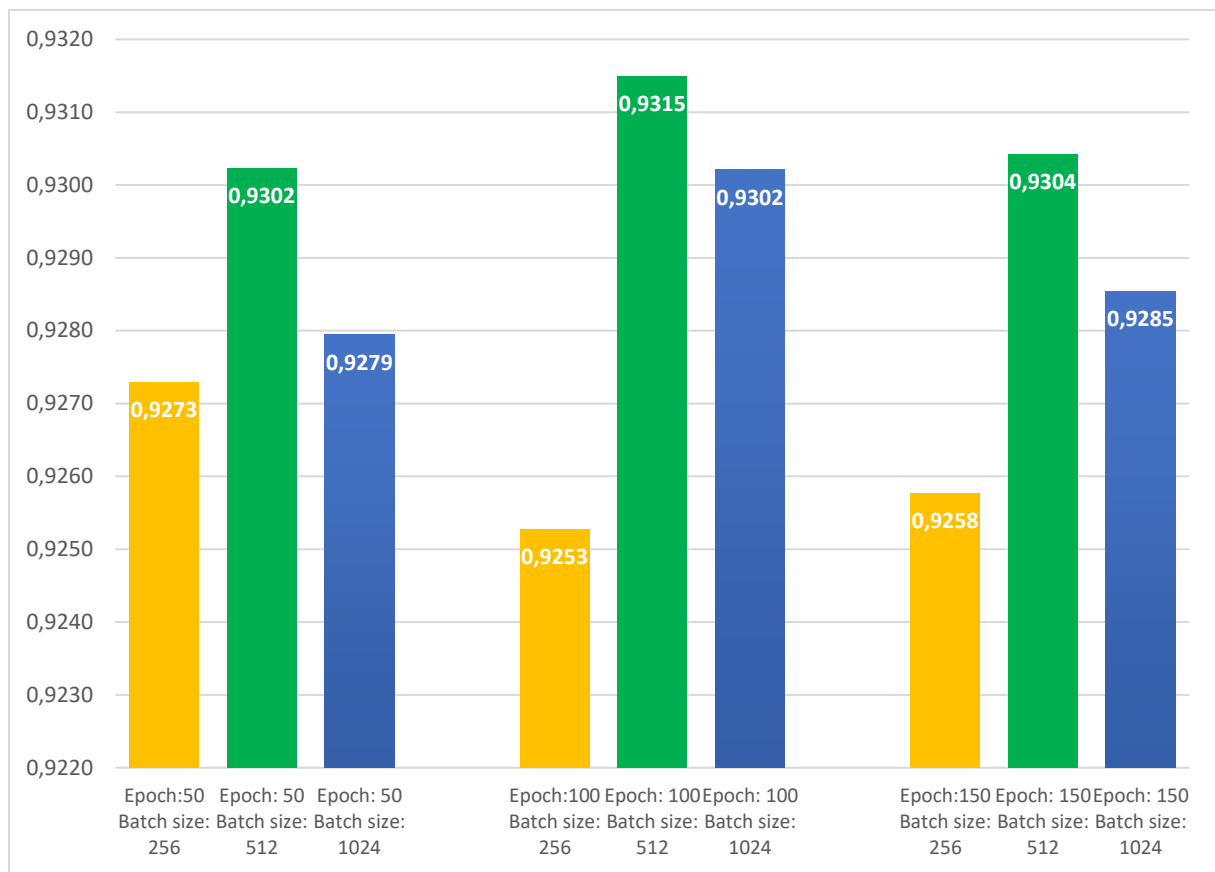
Ezért a tanítás során a hidden unit számra az optimum a 110 körülre esett. A batch size 512-re, és az epoch szám 90-re. Ezt a tanítási és validációs görbe is igazolja, mivel ebben az esetben, a tanítási a validációs görbe alatt halad, jól igazolva a helyes illeszkedés esetét. (11. Ábra)



11. Ábra - Példa a helyes illeszkedésre Forrás: Saját szerkesztés

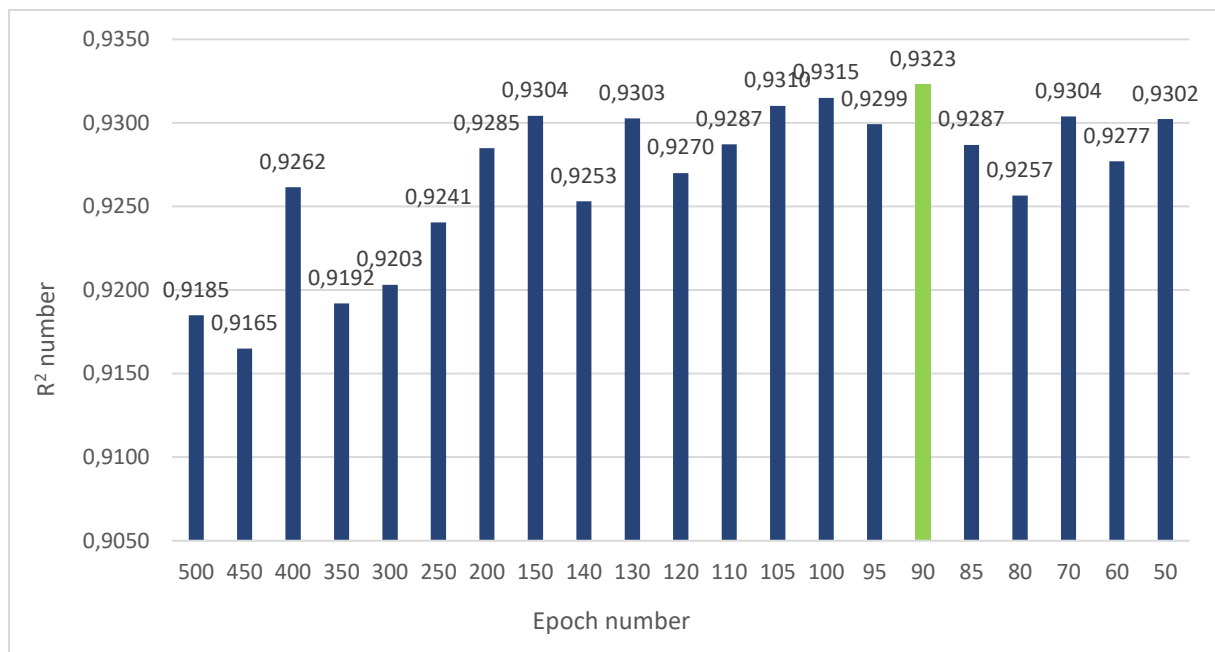
5.2 Az előrejelzés pontossága

Az előrejelzések adatai is a fentebb említett paraméterezés optimális beállítását igazolják, ez az alábbi grafikonról is látszik (12. Ábra). Erőforrás hiány végett keresztteszteléssel lett ellenőrizve az optimális paraméterek nagysága. Kisebb (256-os batch szám) és nagyobb (1024-es batch szám) batch számoknál tapasztalható volt az előrejelzés pontosságának romlása.



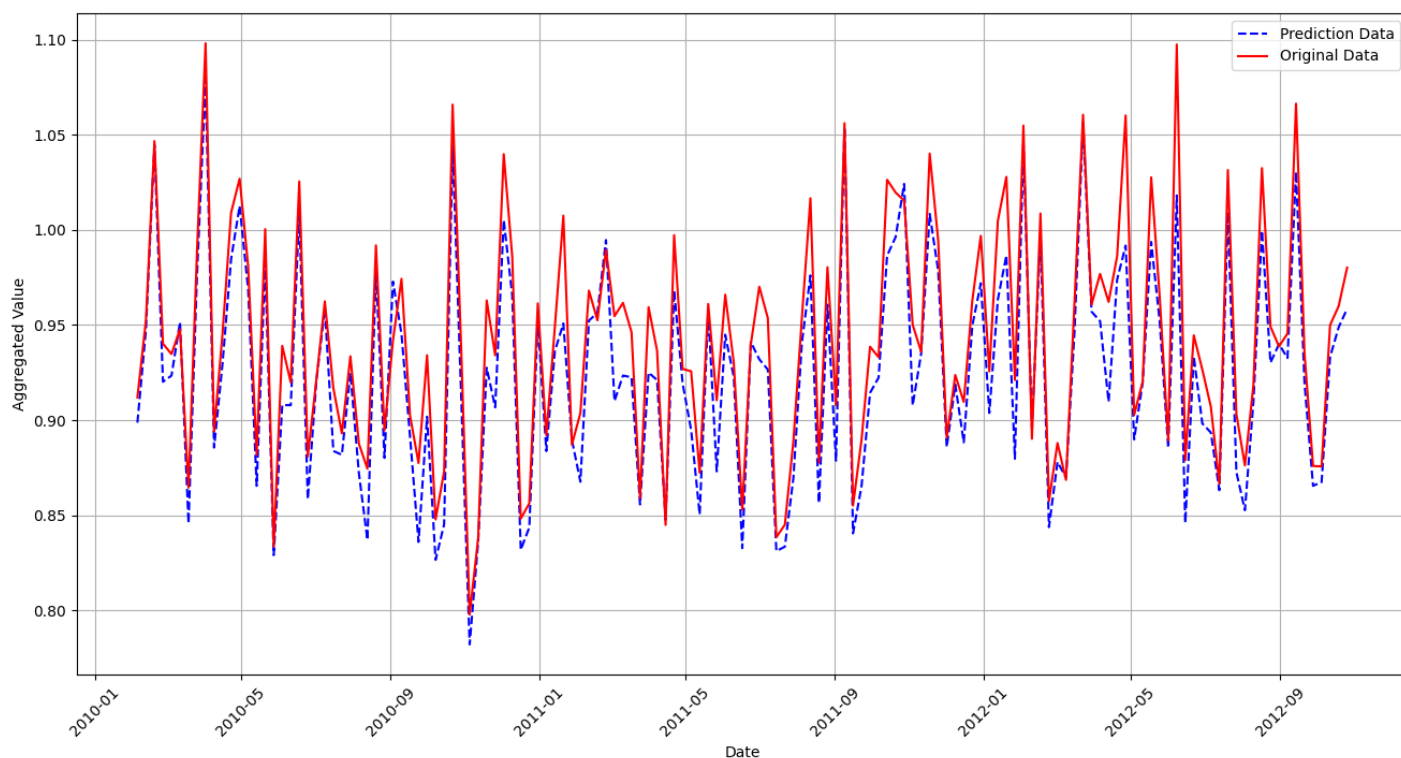
12. Ábra - Összehasonlított tesztek 256-512-1024 batch number és 50-100-150 epoch number esetén Forrás: Saját szerkesztés

Az 512-es batch szám körül, sok tesztet végeztünk, hogy mi is pontosan az az érték, ahol beáll az az optimum, ami fölött a rendszer nem tud jobban teljesíteni. A modell helyes illeszkedését az R^2 számmal határoztuk meg, ami azt jelenti, hogy minél közelebb áll egy eredmény az 1-hez, annál jobb előrejelzéseket várhatunk. (13. Ábra)



13. Ábra - R^2 szám nagysága az epoch számok függvényében (512-es batch és 110 hidden unit number esetén) Forrás: Saját szerkesztés

A 90 epoch-os és 512 batch number-ös tanítás végeredménye elfogadható előrejelzéseket adott eredményül. Ez látszik abból is, hogy az aggregált eredmények grafikonján, az eredeti adat, és az előrejelzett adat görbéje szépen simul egymáshoz, és jól leköveti a tendenciáját (14. Ábra). Érdeemes megjegyezni, hogy az összehasonlítás során, az eredeti adatok gyűjteménye, egy teljesen érintetlen adatforrás, tehát a rendszer még sosem látta, ez az adatsomag teljesen független volt a tanítási adatoktól, és csakis a tesztelésre lett leválasztva a többitől.



14. Ábra - Aggregált előrejelzett adatok vs aggregált eredeti teszt adatok (Epoch number: 90 Batch size: 512 Hidden unit number: 110) Forrás: Saját szerkesztés

Az előrejelzett adatokba mélyebben elemezve, látszik az is, hogy a rendszer az elfogadható értékhatáron belül jelez előre számunkra helyes adatokat. (4. Táblázat)

Date	Prediction Data	Original Data
2010.02.05	3995,029	4593,44
2010.02.05	2018,5092	1728,0001
2010.02.05	149,49971	90,470085
2010.02.05	20159,38	20099,43
2010.02.05	29814,68	27854,352
2010.02.05	13602,572	16132,5
2010.02.05	816,2056	44,52009
2010.02.05	4150,0913	4571,2495
2010.02.05	12540,171	12386,57
2010.02.05	10930,573	8799,859

4. Táblázat - Az előrejelzett adatok és eredeti teszt adatok Forrás: Saját szerkesztés

A számok hatalmas szórása miatt, ezek teljes mértékben elfogadható eredmények, hiszen, ha a teljes eredeti adatsort nézzük, a legkisebb adatunk -1008 volt, a legnagyobb pedig 693099. Ez

azt is jelenti, ha egy olyan adathalmazunk van, ahol a számok szórása sokkal kisebb, akkor az előrejelzett eredmény is sokkal pontosabb lesz.

6. Konklúzió

Mivel a piac hektikus változásainak mértéke várhatóan nem fog csökkenni, sőt, inkább ezen a szinten marad, vagy nőni fog, így a vállalat példáján keresztül kijelenthető, hogy egy alternatív terveket engedélyező stratégia megoldást jelenthet a fentebb ismertetett gazdasági problémára.

Az elért eredmények alátámasztják azt az elképzelést, hogy egy gépi tanulós módszerekkel felépített neurális hálózat alapú rendszer képes kellően jó hibahatáron belüli eredményeket nyújtani, a hagyományos tervezéshez képest relatív 0 emberi idő és erőforrás befektetésével a vállalat szempontjából.

A szoftver teljesítménye közel optimális az elkészített modell esetén, azonban további rekurrens neurális hálózat alapú modellek tervezése és tesztelése szükséges, ahhoz, hogy a kutatás végleges modellje kiválasztásra kerüljön.

6.1 Jövőbeli munka

A kutatás jelenlegi fázisában a prioritás a piac kutatáson, igényfelmérésen és megvalósíthatóságon volt. Ez a kezdeti lépés jelentős időbe telt, azonban kihagyhatatlan és fontos része volt a kutatásnak ahhoz, hogy a szoftver tervezése megfelelően történhessen meg. Amennyiben nem a pontos igényeknek megfelelő rendszer készül el, az jelentős idő és erőforrás-vesztéssel járhat, mert előfordulhat, hogy a rendszert az alapjaitól kezdve kell újra tervezni. Ezért a kutatás technikai része még az elején jár.

A tanulmány következő lépcsőfoka a vállalat által nyújtott valós adatokon tanított, a cég adataira specializált architektúrájú forecast prediktáló mesterséges intelligencia lesz. Ez a jövőbeli szoftver egy alapvető alkotó eleme lesz egy komplett, alternatív stratégiákat alkotó DTR-nek a vállalatban belül. Ez a kész program rendelkezni fog a gépi tanulásból megszerzett súlyozás és eltolás értékekkel, az inputjai pedig az aktuális gazdasági értékek lesznek, mint például az előző havi valós sales adatok.

Miután a rendszer képes forecastokat prediktálni, ezek a forecastok lesznek az inputjai az alternatív stratégia alkotó DTR-nek. Ez a rendszer az előrejelzéseket felhasználva a cégnek konkrét gazdasági terveket ajánl fel, összehasonlítva őket egymással az alapján, hogy a jelenlegi

paramétereknek melyik lenne a legmegfelelőbb választás ökonómiai szempontból. Ezzel lehetővé teszi azt, hogy a piac szélsőséges változásait is képesek legyenek agilisan követni és tudjanak egy olyan alternatív tervre átállni rövid időn belül majd aztán követni, amely a piac jelenlegi állapotának megfelelő. Ennek a rendszernek a technikai működése még a kutatás jövőbeli részét képezi.

A jelenlegi projekt az alapját képezi egy másik rendszernek is, amely hasonlóan a forecast prediktáláshoz, üzgszint eredmények előrejelzését célozza meg. Ez a rendszer egy akciótervező rendszer, amely azt prognosztizálja a vállalatnak, hogy milyen eladásra számíthatnak egy értékesítési csatornán egy termék esetén, ha bevezetnek egy bizonyos akciót, például 2+1-et, vagy egy hagyományos 20%-os akciót bizonyos időre. Jelenleg a cégnek nincs megbízható forrása ezeket az információkat illetően, így nem tudnak megfelelően tervezni ezekkel az akciókkal, így előfordulhat, hogy egy áruházlánc boltjában egy adott termékből nincs annyi, mint amennyi eladás történne, ha lenne. Gazdaságilag ez nagy rizikót hordoz magában, mert azon túl, hogy nem történik eladás, annak is fennáll az esélye, hogy a vásárló a cég hiányzó terméke helyett egy másik vállalat márkáját választja, így lehetséges, hogy a másik termék rendszeres vásárlója lesz a mi cégünk terméke helyett.

Ez a rendszer hasonlóan a forecast előrejelzőhöz, üzgszintén gépi tanulós módszereket fog alkalmazni a múltbeli akciók trendjeinek megtanulására, inputparamétere pedig a bevezetendő akció típusa, mértéke és időintervalluma lesz.

A forecast előrejelző és akciótervező rendszer kritikus továbbfejlesztései közé tartozik még, hogy ne csak egy terméken, hanem egy divízió minden termékén működjön. A kutatás jövőjében a vállalattal együttműködve, a rendszerek kiterjesztésre kerülnek a gabonapelyhek divíziójáról azokra a további divíziókra, melyek adatainak jellege ezt lehetővé teszi. Ezen kívül, a nagyon távoli jövőben, igény esetén a rendszereket a klaszteren belüli országok vállalatainak képviselőivel együttműködve továbbfejlesztésre kerülnek, hogy az ottani igényeknek megfelelően is működhessenek.

Ezek a rendszerek jellegükből adódóan folyamatos karbantartást és fejlesztést igényelnek, hiszen egy olyan rendszer, amelyet utoljára több hónappal vagy évvel ezelőtti adatokkal tanítottak, nem lesz képes az aktuális piaci igényeknek megfelelően működni, így a kutatás jövőbeli részét képezi a rendszer tanításának valamiféle automatizált megoldása, melynek technikai részleteire még nem került sor a kutatás folyamán.

Ahhoz, hogy a cég dolgozói ezeket a szoftvereket megfelelően képesek legyenek használni a mindennapos munka folyamán, szükséges fejleszteni egy graphical user interface (GUI)-t is hozzá, amely áthidalja a felhasználó és a gép közötti kommunikáció nehézségeit. Ennek kellően egyértelműnek kell lennie ahhoz, hogy az olyan felhasználók is magabiztosan tudják használni, akiknek nem céljuk megérteni a szoftver mögöttes működését.

ÁBRAJEGYZÉK

1. Ábra – A cég által tervezett stratégia Forrás: Saját szerkesztés	8
2. Ábra - A rendszer "Black box" alapú működése Forrás: Saját szerkesztés	9
3. Ábra - Illusztráció egy mély neurális hálózat architektúráról Forrás: [3] nyomán saját szerkesztés	10
4. Ábra - Illusztráció egy rekurens neurális hálózatarchitektúráról Forrás: [9] nyomán saját szerkesztés	12
5. Ábra - - Az LSTM cella felépítése Forrás: [11] nyomán saját szerkesztés	13
6. Ábra - - Kaszkádosított cellák Forrás: [11] nyomán saját szerkesztés	15
7. Ábra - A szoftver architektúrája Forrás: Saját szerkesztés.....	18
8. Ábra - Összefoglaló ábra a feature, batch, és epoch fogalmak felépítéséről Forrás: Saját szerkesztés	20
9. Ábra - Példa az underfitting esetére Forrás: Saját szerkesztés	21
10. Ábra - Példa az overfitting esetére Forrás: Saját szerkesztés	21
11. Ábra - Példa a helyes illeszkedésre Forrás: Saját szerkesztés.....	22
12. Ábra - Összehasonlított tesztek 256-512-1024 batch number és 50-100-150 epoch number esetén Forrás: Saját szerkesztés.....	23
13. Ábra - R^2 szám nagysága az epoch számok függvényében (512-es batch és 110 hidden unit number esetén) Forrás: Saját szerkesztés	24
14. Ábra - Aggregált előrejelzett adatok vs aggregált eredeti teszt adatok (Epoch number: 90 Batch size: 512 Hidden unit number: 110) Forrás: Saját szerkesztés	25

TÁBLÁZATJEGYZÉK

1. Táblázat - Az adatok eredeti jellege Forrás: Saját szerkesztés.....	17
2. Táblázat - Az adatok jellege a $]-1;1[$ skálázás után Forrás: Saját szerkesztés	17
3. Táblázat - Az adathalmazok jellege Forrás: Saját szerkesztés	19
4. Táblázat - Az előrejelzett adatok és eredeti teszt adatok Forrás: Saját szerkesztés	25

IRODALOMJEGYZÉK

- [1] J. Schmidhuber, “Annotated History of Modern AI and Deep Learning.” arXiv, Dec. 29, 2022. doi: 10.48550/arXiv.2212.11279.
- [2] J. Heaton, “Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning,” *Genet. Program. Evolvable Mach.*, vol. 19, no. 1, pp. 305–307, Jun. 2018, doi: 10.1007/s10710-017-9314-z.
- [3] R. Aggarwal, H. Ugail, and R. K. Jha, “A deep artificial neural network architecture for mesh free solutions of nonlinear boundary value problems,” *Appl. Intell.*, vol. 52, no. 1, pp. 916–926, Jan. 2022, doi: 10.1007/s10489-021-02474-4.
- [4] Z. Lipton, “A Critical Review of Recurrent Neural Networks for Sequence Learning,” May 2015.
- [5] H. Jaeger, “Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the echo state network approach,” *GMD-Forschungszentrum Informationstechnik* 2002, vol. 5, Jan. 2002.
- [6] S. Hochreiter, “The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions,” *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 6, pp. 107–116, Apr. 1998, doi: 10.1142/S0218488598000094.
- [7] P. J. Werbos, “Backpropagation through time: what it does and how to do it,” *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990, doi: 10.1109/5.58337.
- [8] C. Tallec and Y. Ollivier, “Unbiasing Truncated Backpropagation Through Time.” arXiv, May 23, 2017. doi: 10.48550/arXiv.1705.08209.
- [9] F. Kegong *et al.*, “A LSTM Algorithm Estimating Pseudo Measurements for Aiding INS during GNSS Signal Outages,” *Remote Sens.*, vol. 12, p. 256, Jan. 2020, doi: 10.3390/rs12020256.
- [10] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.
- [11] J. Luo, Z. Zhang, Y. Fu, and F. Rao, “Time series prediction of COVID-19 transmission in America using LSTM and XGBoost algorithms,” *Results Phys.*, vol. 27, p. 104462, Jun. 2021, doi: 10.1016/j.rinp.2021.104462.
- [12] S. Linnainmaa, “Taylor expansion of the accumulated rounding error,” *BIT Numer. Math.*, vol. 16, no. 2, pp. 146–160, Jun. 1976, doi: 10.1007/BF01931367.
- [13] M. Abadi *et al.*, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems.” 2015. Accessed: Nov. 07, 2023. [Online]. Available: <http://download.tensorflow.org/paper/whitepaper2015.pdf>