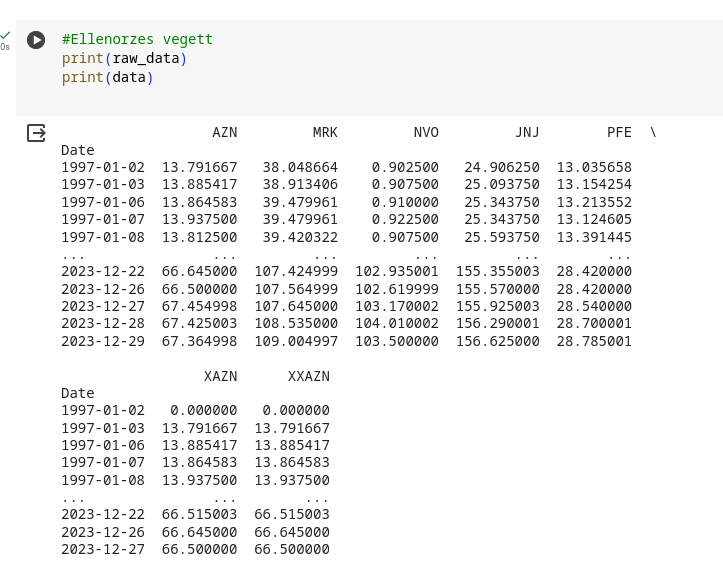
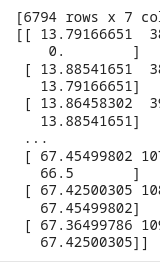
Beszámoló

A dolgozatomban RNN rekurrens hálót tanítottam be pénzügyi (részvény) adatok predikciójára LSTM/Vanilla/GRU modellekkel. Az alapfeladatban 5 különböző gyógyszergyártó cég napi középárfolyamát vettem (magas + alacsony)/2 előállítással 1997 és 2024 között. (Astra Zeneca, MERK, Novo Nordisk, Johnson & Johnson, Pfizer). A napi árát próbálom meg prediktálni az Astra-Zeneca részvényeknek a többi cég aznapi részvényének és az előző napi Astra-Zeneca részvény segítségével (Teacher Forcing) . A rejtett reprezentációban így 5 részvény adatait fogom elkódolni + az előző napi outputot és árat is hozzá veszem.

Lépések:

* A feladat megoldásához az adathalmazt a yahoo finance -rol töltöttem le (yfinance API). A megoldáshoz pytorchot használtam, mert az iterációban manuálisan kellett állítanom át adatokat ami igen nehézkes a magas szintű tensorflowban.
* Az esetlegesen fennálló NA adatokat eldobtam hogy ne okozzon problémát.
* Elsőre is a teacher forcing végrehajtásához 1 nappal elshiftelem az adatokat , majd létrehozok még egy változót, ahová majd vissza feedelem az előző napi outputot ennek segítségével.
* MinMax skálázást alkalmaztam, mivel nem feltételeztem normális eloszlást, illetve a trend komponens le lett vonva (“ki lett laposítva’) a diagram.
* A kezdő tokenben az Astra-Zeneca részvény értékét 0-ra állítottam.
* Leellenőriztem az adat Shape-jét hogy valóban a helyes sorrendben kerülnek e be numpy-ban az értékek.





* Ezek után splitteltem az adatokon 70 % 15 % 15 % felosztással Train, test, és validation halmazokra.
* Ezt folytattam a pytorch beállításaival. Úgy döntöttem hogy Batch-enként számolok, Ehhez létre kellett hoznom egy TimeSeriesDatesetet aminek segítságével Train és test loadereket hoztam létre.
* Ezek után definiáltam 3 féle architektúrát az RNN -nek: VanillaRNN, LSTM és GRU hálókat hoztam létre hogy össze tudjam hasonlítani a teljesítményeket.
* Ezek után betanítom a hálót, majd az eredményeket kiirattam.

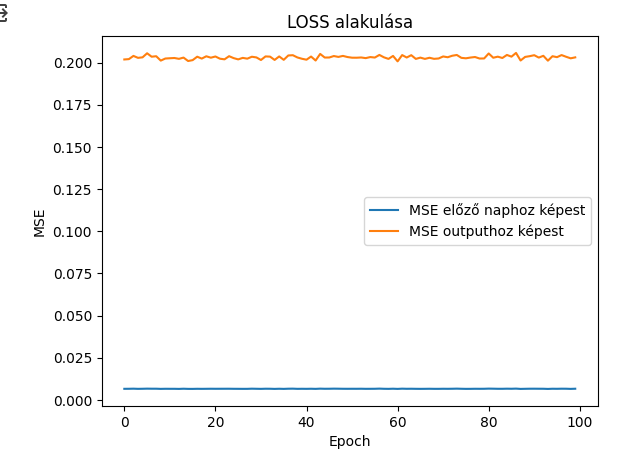
Hiperparaméterek és eredmények:

**1) Vanilla RNN**

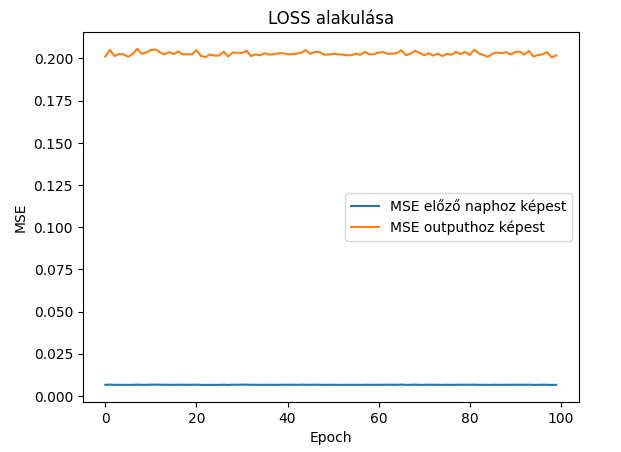
A Vanilla RNN használata során elsőre 1 rejtett reprezentációs réteggel próbálkoztam. A bidirectional parmétert mindenhol Falsera állítottam, hiszen idősoros adatokat használok. A Vanilla RNN rosszabbul teljsesített, ami várható volt, hiszen nincsen “graient highway”, így a hátsó paraméterek egy idő után nehezebben taníthatóak (főleg ilyen extrém esetben ahol majdnem 5 ezer hosszú az adatsor )

A hiperparaméterek megválasztásánál a learning rate-t 0.001 – re választottam meg, epochok számát 100 -ra illetve a regularizaciós paramétert alacsonyra 0.000001 re. Optimizernek mindenhol az Adam optimizert választottam, hogy legyen benne momentum és SGD illetve az amsgradot is belevettem.

Ilyen architektúrában látható hogy az MSE nem tud 20% alá lemenni.



A learning rate megemelésével (0.01) és a lambda paraméter 0 -ra állításával sem javul a modell.

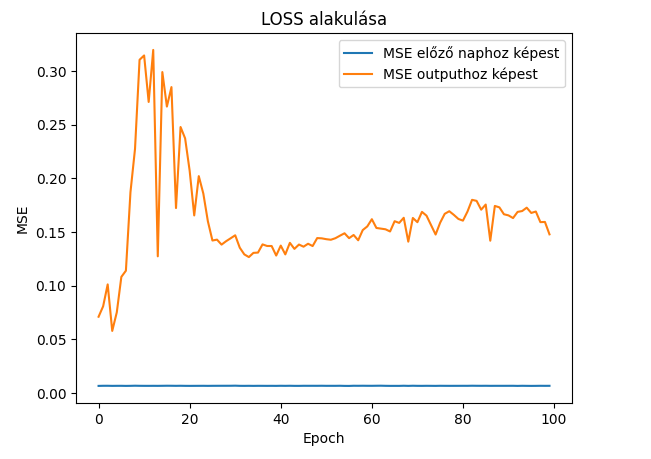


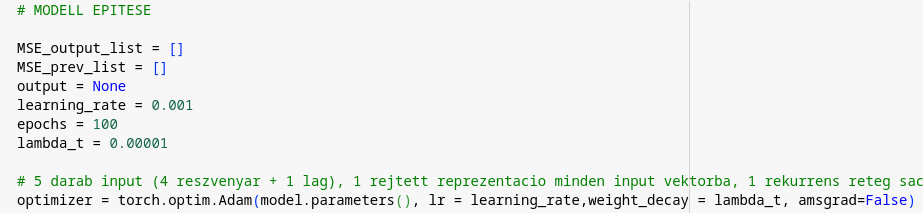
Megemelhetnénk a hidden layerek számát esetleg komplexxebb mintákat tudunk megtanulni, de az elhaló gradiens problémat ez nem fogja megoldani.

**2) LSTM**

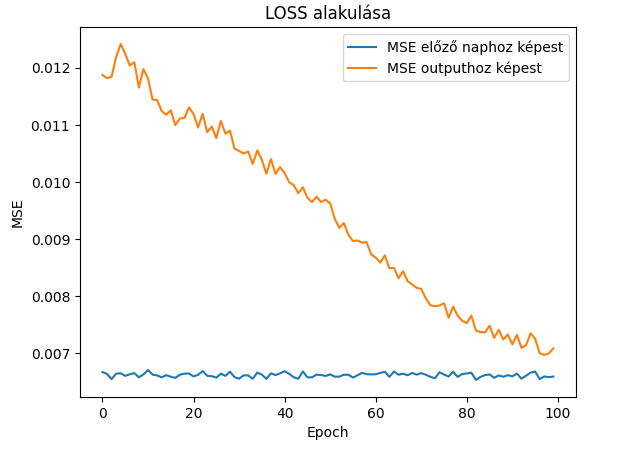
Az LSTM modell az előző problémát hivatott megoldani. Bele van építve a gradient highway, és számon van tartva egy cell state amibol meg tudjuk tanulni hogy mennyit “felejtsünk” az előző állapotból hogy a gradiens visszaterjesztésnél a nagyságrend nem változzon, így taníthatóvá váljanak a paramétereink.

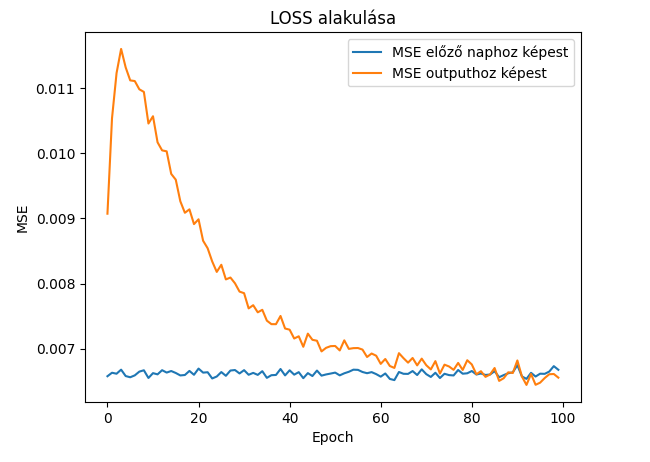
Az eddigi paraméterezést megtartva látszik hogy sokkal jobban teljesít a modell, viszont Magas learning rate mellett illetve magas regularizációs paraméternél nem konvergál.



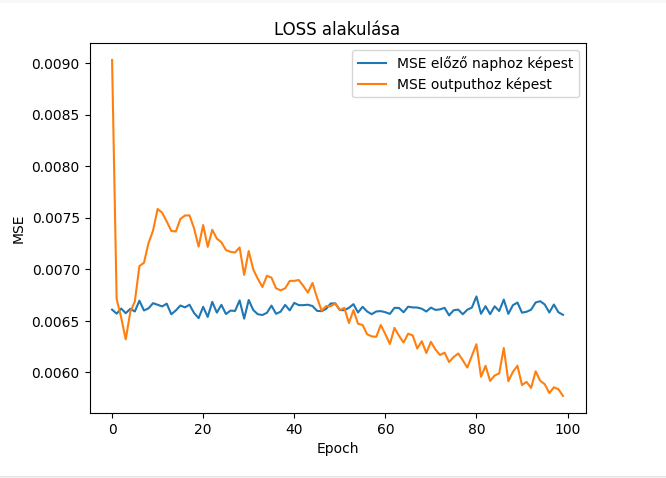
Amint levettem a learning rate-t, és az amsgradot is True -ra állítottam már konvergált a benchmark felé.

Amint még lejjebb vettem a learning rate-em illetve a lambda paramétert sokkal kisebbre vettem, sokkal jobb eredményeket ért el a modell.



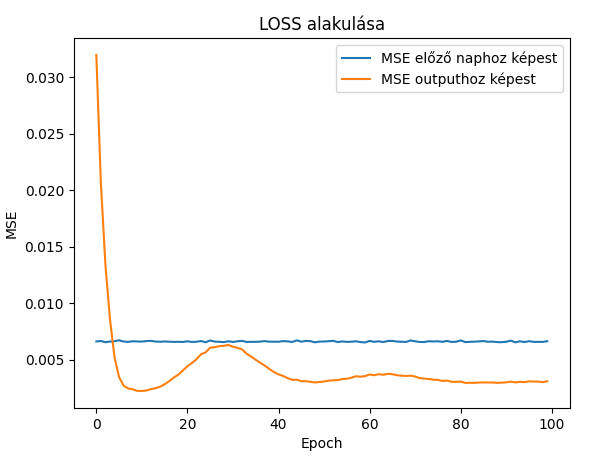


Kettő réteget használva, és a learning rate-et kicsin hagyva, stabilan bekonvergálunk az előző napi ár megismétlésének hibája alá. Hasonló eredményeket mutatva mint az egyrétegű modellnél.

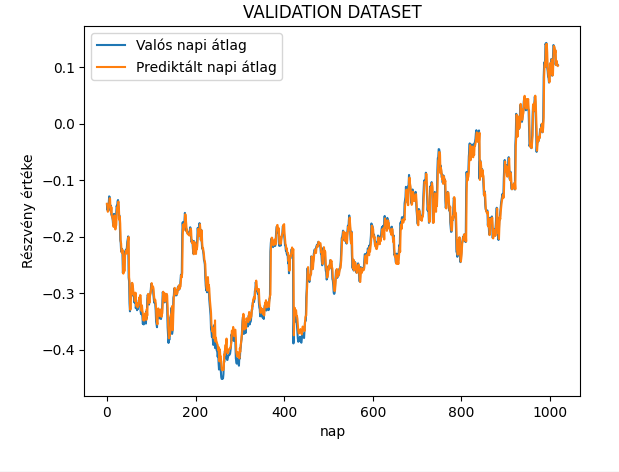
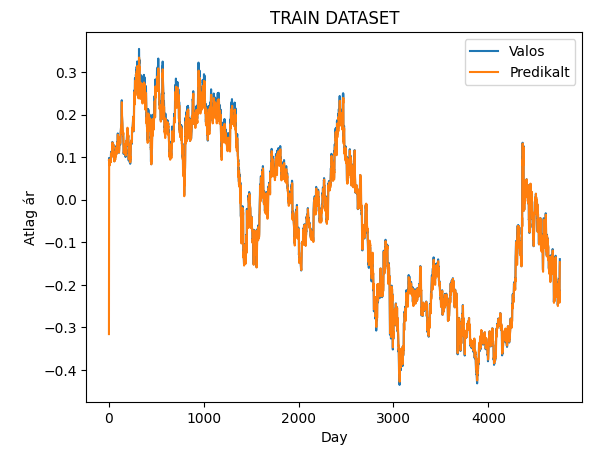


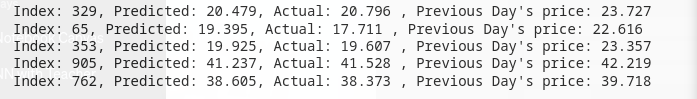
**GRU**

Ezek után kiprobáltam még a GRU modellt ami az LSTM egy javított változata, ahol egy külön cell state és rejtett reprezentáció helyett egy rejtet reprezentációval dolgozik, és megoldja a “túl rövid memória” problémát.



Eredmények





Az eredmények túltanulásra adnak gyanút, viszont a validation dataseten is ugyanolyan jól műkodik a modell. Kiprinteltem pár értéket, predikciót és megelőző napi értéket, ahol láthatjuk hogy jobban becsül a modell mintha csak az előző napi eredményt vennénk. Tehát a betanítás sikeres volt.