بسمه تعالی



دانشکده مهندسی کامپیوتر

**یادگیری عمیق**

نام استاد: دکتر محمدی

تمرین دوازدهم

آرمان حیدری

شماره دانشجویی: **97521252**

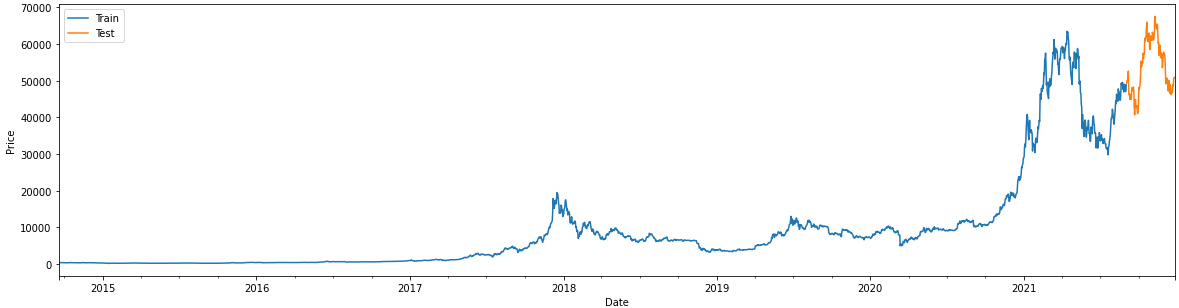
آذر **1400**

# پاسخ سوال اول

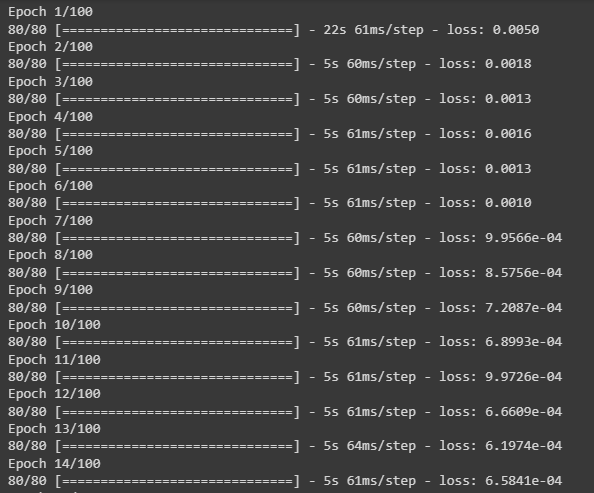
مراحل مختلف این سوال در نوتبوک HW12.ipynb ([لینک گوگل کولب](https://colab.research.google.com/drive/12mgSl1qLFCV_p1ce-St1d5_ydiMq76zp?usp=sharing))، قسمت question 1 پیاده سازی شده است.

توضیحات مرحله به مرحله ی کد، بین سلول های نوتبوک داده شده اند. که شامل لود کردن دیتاست و نمایش قسمت آموزشی و آزمایشی، پیش پردازش داده، تعریف مدل و آموزش آن و در نهایت دیدن نتیجه روی داده های آزمیشی است.

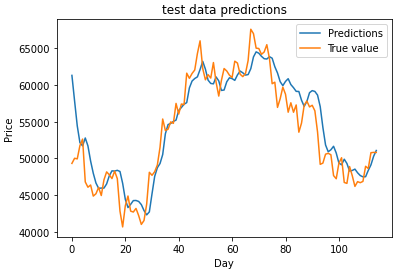
نمودار داده های آموزشی و آزمایشی به این صورت است:



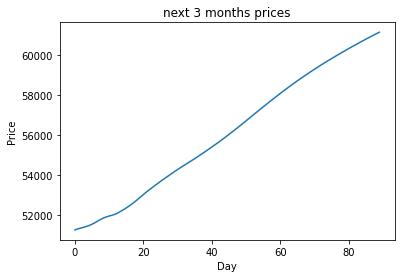
در اینجا چون معیار دقیقی برای accuracy تعریف نمیکنیم و طبیعتا نمیتواند قیمت را با دقت 4 رقم اعشار پیشبینی کند و به همین دلیل نتایج accuracy غیر منطقی خواهند بود، با همان سنجیدن loss و روند کاهشی مطلوبی که دارد میفهمیم که مدل به خوبی آموزش دیده است:



و نتیجه پیشبینی مدل برای 115 روز داده تست هم در مقایسه با داده های واقعی به این صورت بود:



که میبینیم نتایج معقولی است و شکل کلی حرکت قیمت را تا حد خوبی پیشبینی کرده است. برای سه ماه آینده هم با استفاده از مدل پیشبینی میکنیم که به این صورت به دست آمد:



میبینیم که نمودار غیرواقعی به نظر میرسد. به دلیل این است که این شبکه برای داده unsupervised چندان مناسب نیست و نتیجه ای که در هر روز بر اساس 60 روز گذشته گرفته فقط بر اساس قیمت هایی است که داریم و همواره عددی کمی بیشتر از قبل به دست می آید و نوسانات را پیشبینی نمیکند.

## به نظر شما افزایش یا کاهش متغیر تعریف شده در مرحله تهیه داده مورد نیاز برای آموزش مدل یعنی تعداد داده های گذشته برای پیشبینی داده های مشخص چه مزایا یا معایبی دارد؟ شرح دهید.

این متغیر که در کد من به اسم PAST\_VALUES تعریف شده است، پارامتر مهمی در این مسئله است. چون هرچقدر مقدار آن را کمتر کنیم، درواقع ویژگی هایی کمتری را برای پیشبینی در اختیار مدل قرار داده ایم و مزیتش قطعا سرعت بیشتر مدل خواهد بود. چون ابعاد ورودی به لایه LSTM کاش میابد و به خصوص چون در شبکه های بازگشتی محاسبات ناچارا ترتیبی هستند و نه موازی، این تغییر محسوس خواهد بود. عیب اصلی کم کردن این عدد هم قطعا کم شدن دقت (افزایش ضرر) خواهد بود. چون به هرحال به مدل داده کمتری را داده ایم.

با زیاد کردن این پارامتر، سرعت شبکه کاهش میابد و دقت هم بیشتر می شود. البته باید دقت داشته باشیم که خیلی زیاد کردن آن حتی میتواند در برخی مسائل مدل را به اشتباه هم بیندازد. چون مثلا قیمت 1 سال پیش بیت کوین احتمالا اهمیت چندانی برای پیشبینی قیمت فردا نخواهد داشت. و همچنین زیادی بزرگ شدن شبکه میتواند باعن پدیده gradient vanishing شود، که البته به علت استفاده ما از LSTM به جای simple RNN این قضیه کمتر خواهد بود.

## منابع

<https://seaborn.pydata.org/generated/seaborn.lineplot.html>

<https://compucademy.net/getting-stock-data-using-python-and-yfinance/>

# پاسخ سوال دوم

مراحل مختلف این سوال در نوتبوک HW12.ipynb ([لینک گوگل کولب](https://colab.research.google.com/drive/12mgSl1qLFCV_p1ce-St1d5_ydiMq76zp?usp=sharing))، قسمت question 2 پیاده سازی شده است.

در ابتدا برای ذخیره کردن مدل، google drive را mount میکنیم تا وزن ها و سایر موارد مدل ها را پس از هر epoch، در یک فایل my\_model.h5 داخل دایرکتوری DL\_HW12 ذخیره کنیم. این ذخیره کردن را با استفاده از تابع .save() که در کراس برای مدل ها پیاده سازی شده است انجام داده ایم. برای خواندن مدل ذخیره شده هم، در هنگام تست کردن و بعضا اگر آموزش نصفه ماند و خواستیم ادامه دهیم، میتوانیم از تابع load\_model پیاده سازی شده در keras.models استفاده کنیم.

در این مسئله با مدل sequence to sequence سروکار داریم. و درواقع با یک لایه GRU به عنوان many to one و یک یا چند لایه GRU به عنوان one to many استفاده میکنیم. تعداد واحدهای هر لایه میتواند هر عددی باشد اما طبق مثال کلاسی حدودا 64 و 128 و 256 را امتحان میکنیم تا ببینیم کدام نتایج بهتری دارد.

با پیش‌پردازش روی داده های ورودی آن ها را به صورت one hot coding در می‌آوریم. داده تست را هم 10 تا 10 تا جدا کرده و به همین صورت در میآوریم تا بتوانیم به مدل تعریف شده بدهیم و خروجی بگیریم. توابع مختلف در نوتبوک توضیح داده شده اند. هایپرپارامترهای مسئله را به این صورت تعریف میکنیم:

optimizer = 'adam'

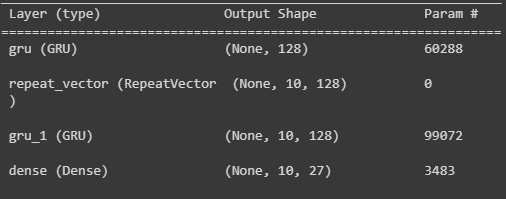
loss = 'categorical\_crossentropy'

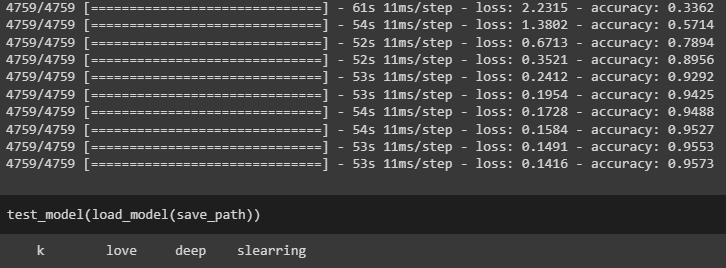
epochs = 10

batch\_size=32

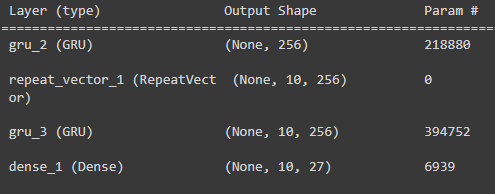
save\_path = '/content/gdrive/MyDrive/DL\_HW12/my\_model.h5'

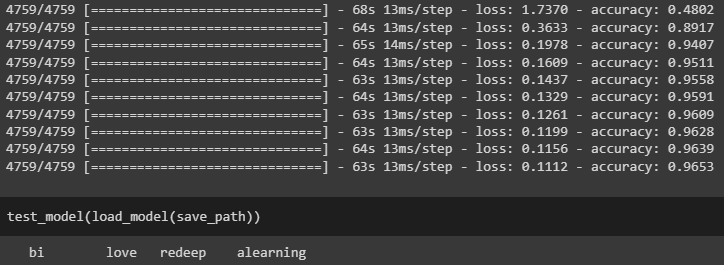
مدل اول به همراه نتیجه روی داده آموزشی و جمله تست:



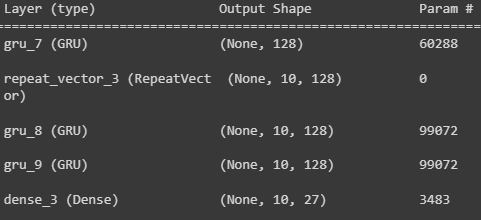


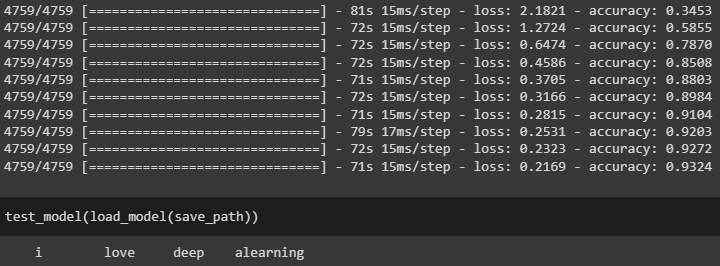
در مدل دوم تعداد unit هارا از 128 به 256 افزایش دادیم. میبینیم که آموزش کمی بهتر انجام شده است که به علت پارامترهای بیشتر مدل طبیعیست. نتیجه روی جمله تست اما تغییر کیفیت چندانی نداشته است:





حالا سعی میکنیم با عمیقتر کردن شبکه به نتایج بهتری برسیم. البته در شبکه های RNN مانند CNN صحبت از صد لایه نمیکنیم و معمولا زیر 5 لایه هستند. لایه dense آخر را هرگز تغییر نمیدهیم چون برای تصمیم گیری نهایی بین 27 کاراکتر (26 حرف انگلیسی+space) است و به همین دلیل تابع فعالسازی softmax دارد. نتایج مدل سوم:





علی رغم این که مدل دیرتر همگرا شده است و حتی از مدل اول هم دقتش روی داده آموزشی کمتر است، به نتیجه ای عالی روی جمله تست میرسد. و استفاده از دولایه با 128 واحد در قسمت one to many به نظر به نتیجه خوبی میرسد.

در مدل چهارم سعی کردم با 64 واحد در 3 لایه GRU مسئله را حل کنم اما نتیجه روی داده آموزشی و تست جالب نبود:

