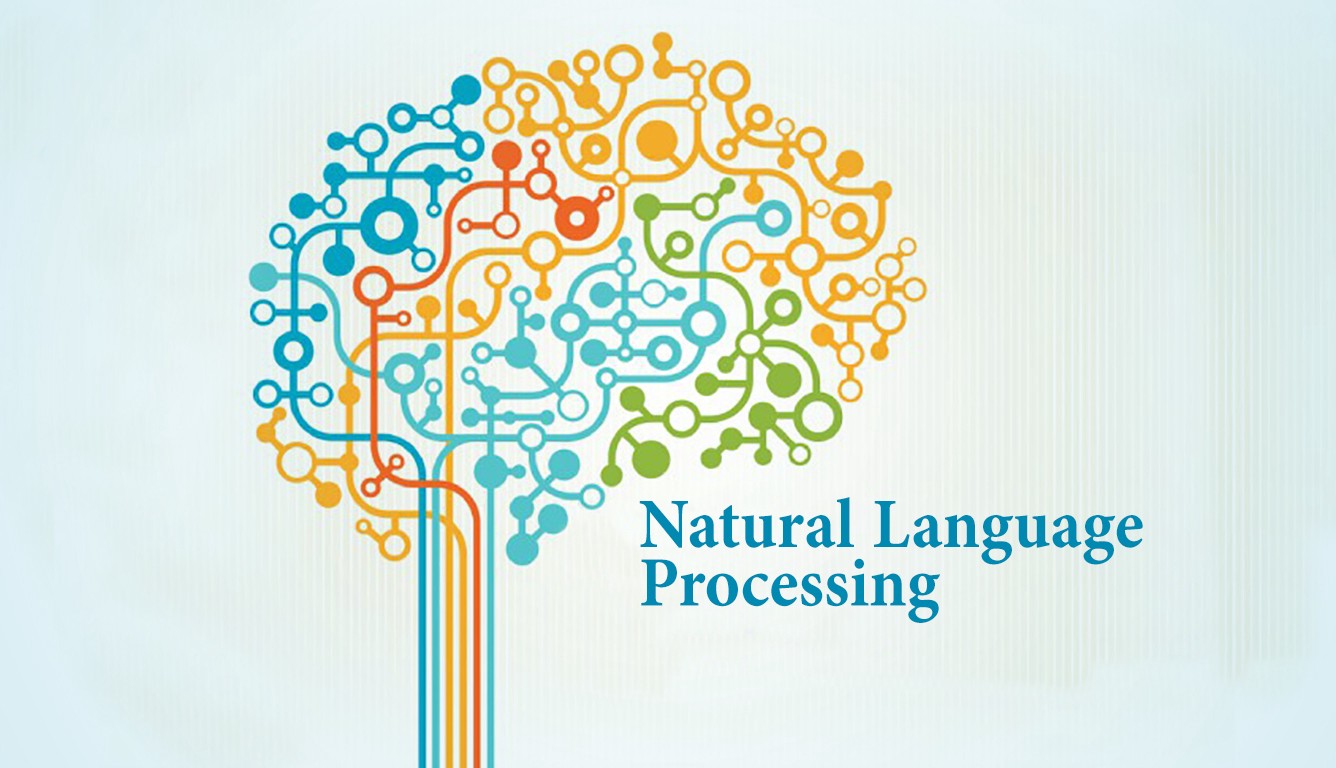
**گزارش پروژه پایانی پردازش زبان های طبیعی**



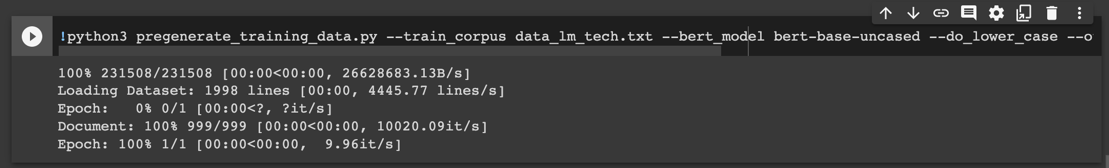
**نگین درخشان ، کیوان بوشهری**

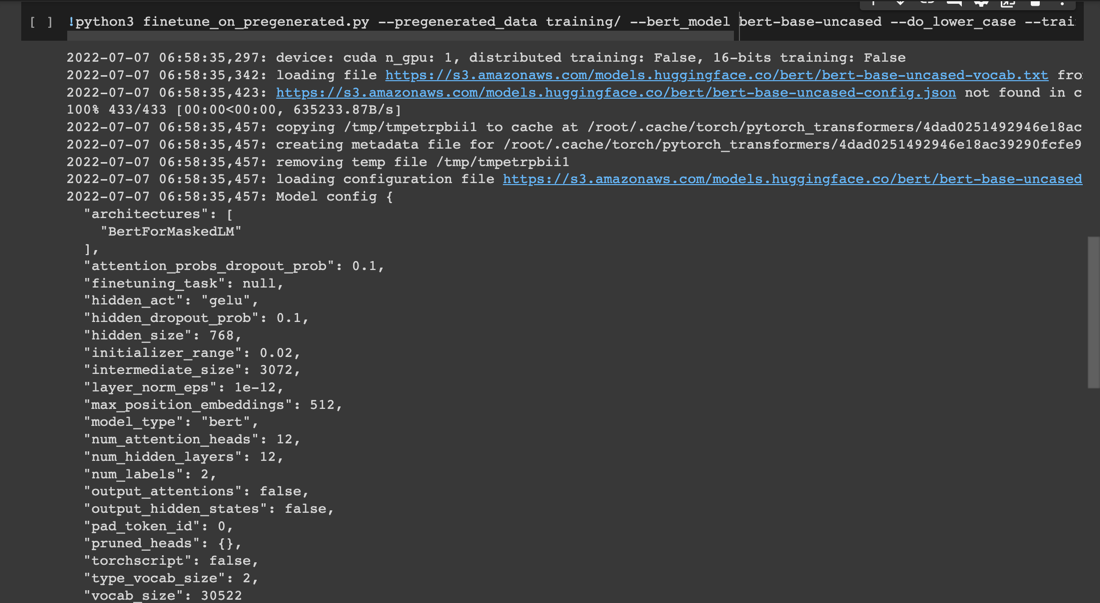
**راهنما : رضا قهرمانی**

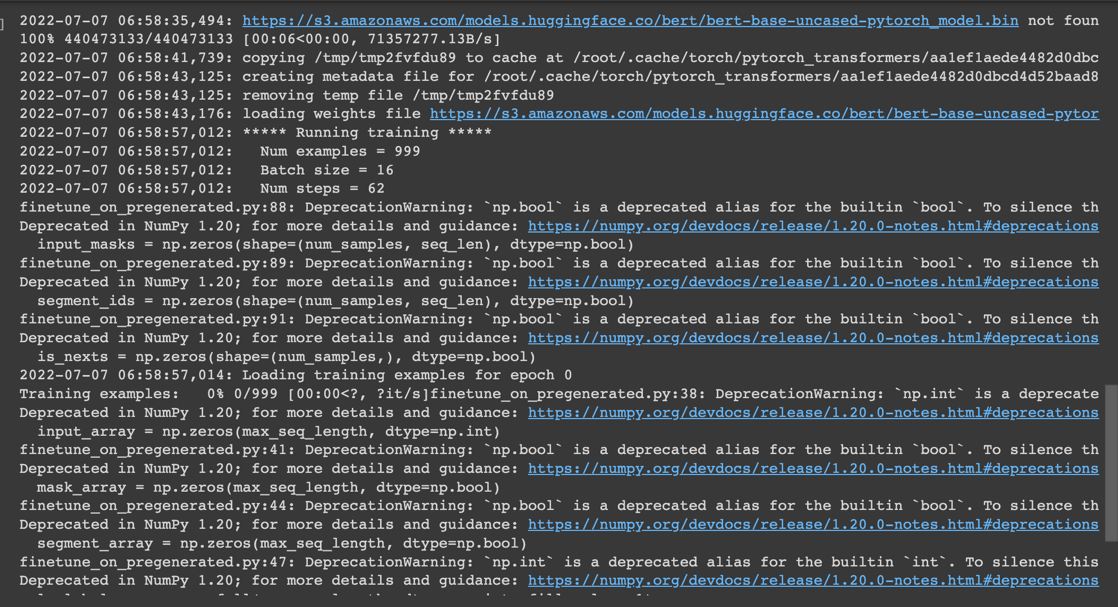
**استاد درس : دکتر مینایی**

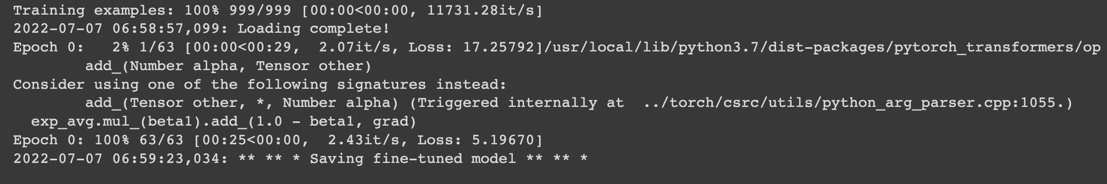
**بخش اول : تولید جملات**

در این بخش برای تولید جملات از مدل bert استفاده کرده ایم. و برای هر دسته از دیتایمان مدل bert را finetune کردیم.تنظیم دقیق یک تکنیک رایج برای انتقال یادگیری است. مدل هدف همه طرح‌های مدل را با پارامترهایشان از مدل منبع به جز لایه خروجی کپی می‌کند و این پارامترها را بر اساس مجموعه داده هدف تنظیم می‌کند. در مقابل، لایه خروجی مدل هدف باید از ابتدا آموزش داده شود.

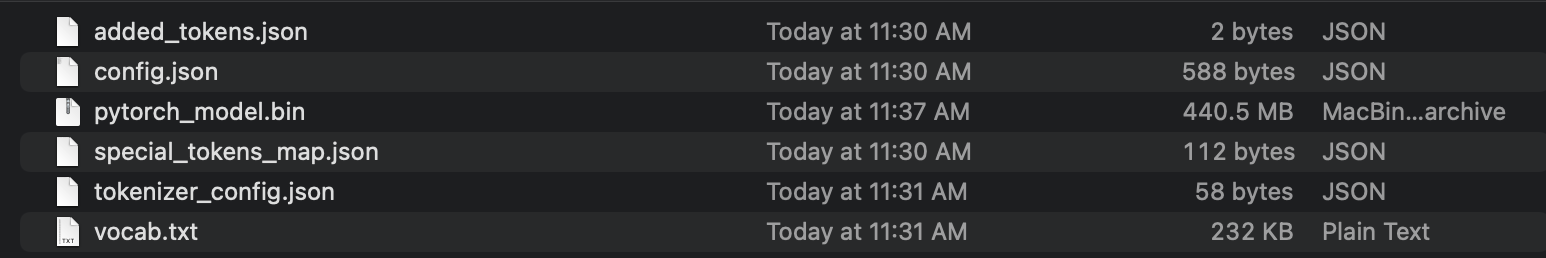




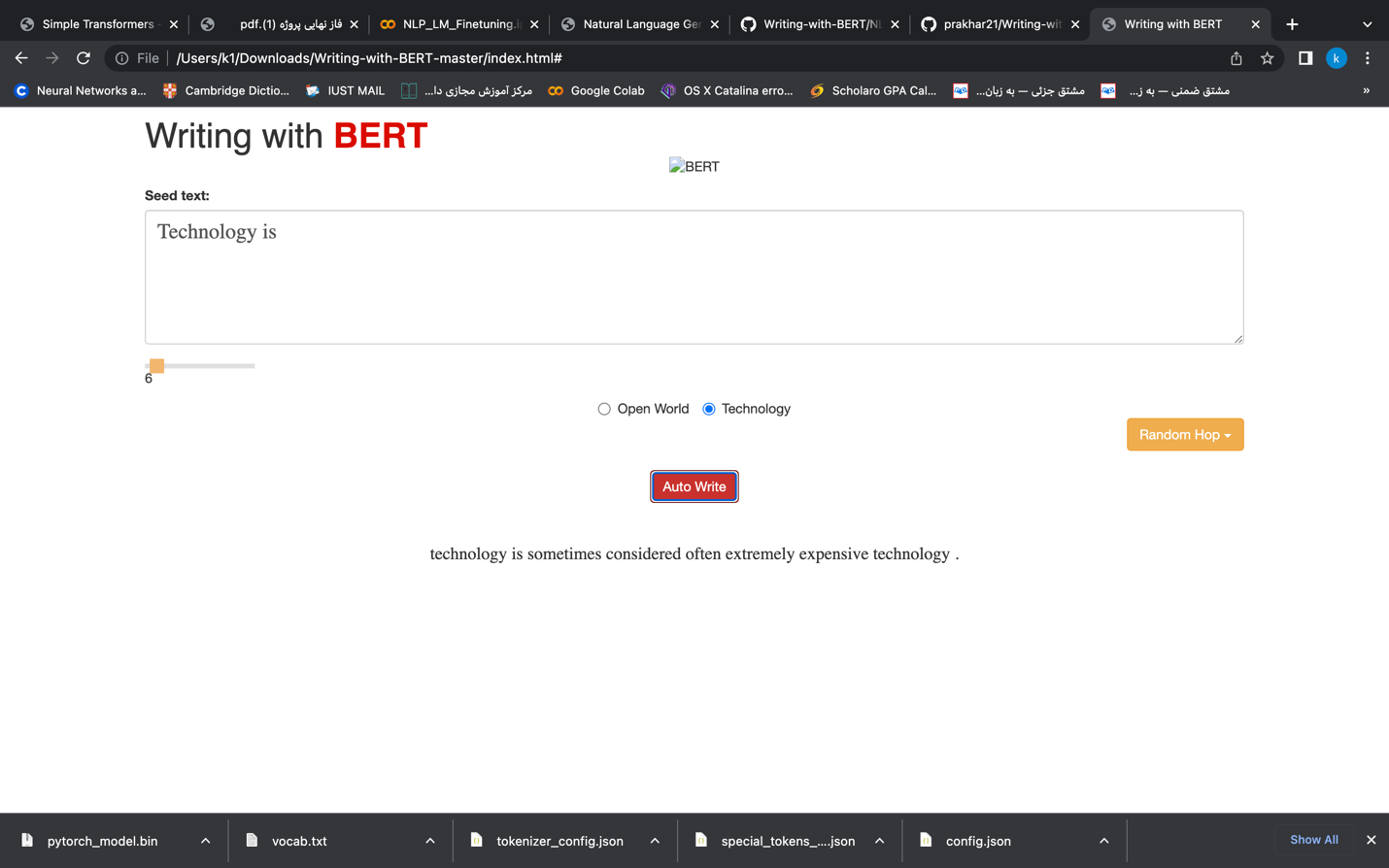




نتیجه نهایی برای مدل تکنولوژی در فولدر finetuned\_lm:

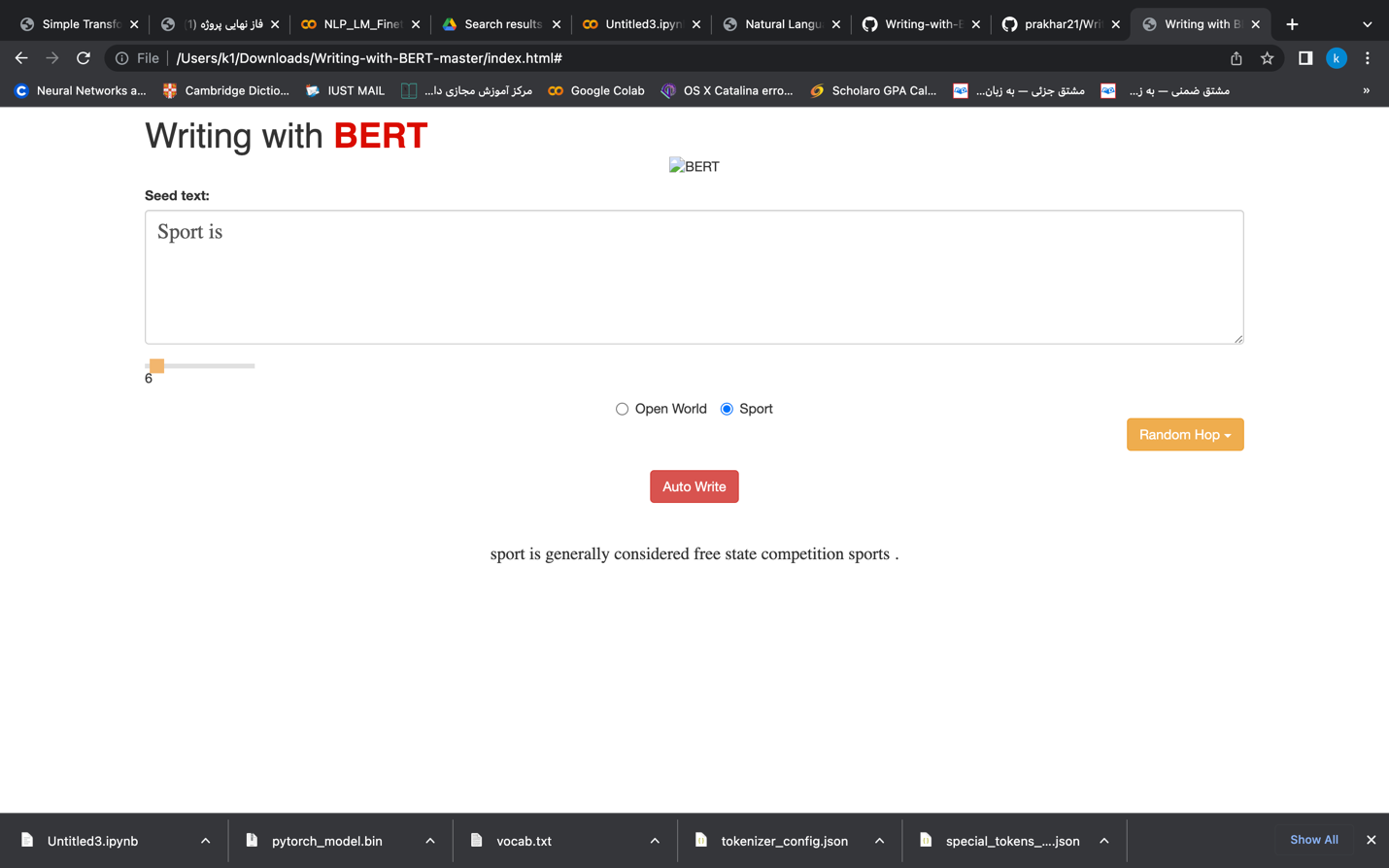


حال برای تولید چند جمله در زمینه تکنولوژی از مدلمان استفاده می کنیم:



مشاهده می شود که جمله تولید شده مربوط به زمینه علم و فن آوری است که نشان دهنده یادگیری موفقیت آمیز مدل ما است.

همین کار را برای دسته ورزشی تکرار کردیم:

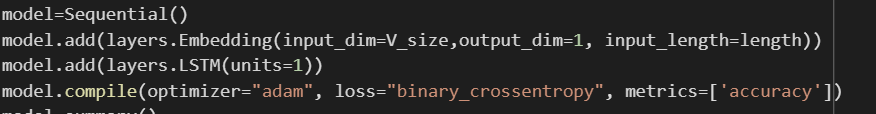


**نکته مهم: زمانی که مدل bert را finetuned می کردیم زمان زیادی برای آموزش لازم بود به این جهت از gpu کلب استفاده کردیم.**

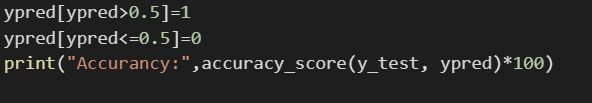
**بخش دوم:شبکه با معماری ساده**

در این قسمت ابتدا جملات تولید شده در قسمت های قبلی را که در دو دستهtest , train هستند را خوانده و آرایه های xtrain, ytrain, xtest, ytest را تشکیل دادیم. در قدم بعدی با استفاده از tokenizer جملات آرایه x را token کرده و تبدیل به دنباله(sequences) کردیم و دنباله را به آرایه ورودی شبکه عصبی تبدیل کردیم.

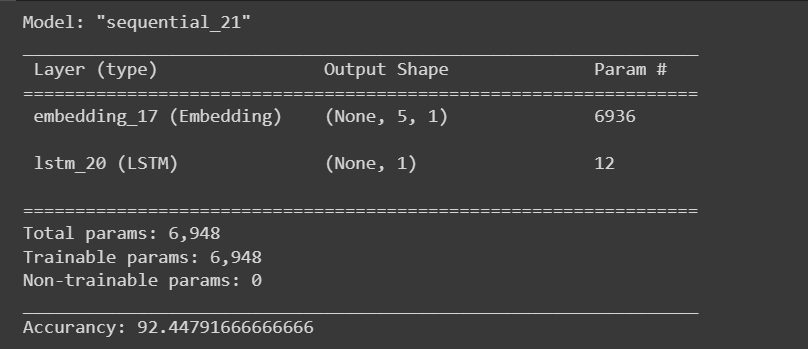
پس از تولید ورودی مناسب، شبکه عصبی را ساختیم. ابتدا مدل را به صورت Sequential تعریف کرده و یک شبکه lstm به آن اضافه کردیم. در ادامه مدل را کامپایل کرده و روی داده های train آموزش دادیم.



در ادامه با استفاده از predict و داده تست، داده پیشبینی شده را بدست آوردیم. این داده مقدار بین 0 و 1 دارد که با توجه به 0 یا 1 بودن کلاس ها با تابع sigmoid کلاس پیش بینی شده را بدست اوردیم. در نهایت با استفاده از مقدار کلاس واقعی مقدار دقت را بدست آوردیم.



شبکه ساخته شده و مقدار دقت این شبکه روی داده تست به صورت زیر است:



**بخش سوم: شبکه با معماری پیچیده تر**

در این قسمت با استفاده از مدل roberta که pretrained می باشد و بر پایه transformers است مدل را با دیتاست خودمان آموزش دادیم:

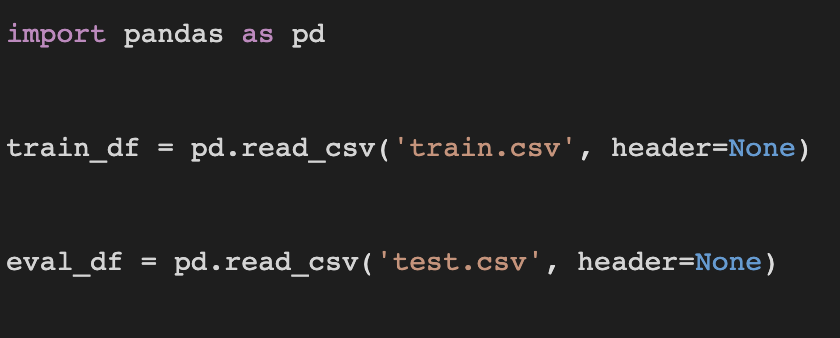
[RoBERTa - Hugging Face](https://huggingface.co/docs/transformers/model_doc/roberta)

برای این کار از library به اسم simpletransformers استفاده کردیم:

[Simple Transformers](https://simpletransformers.ai/)



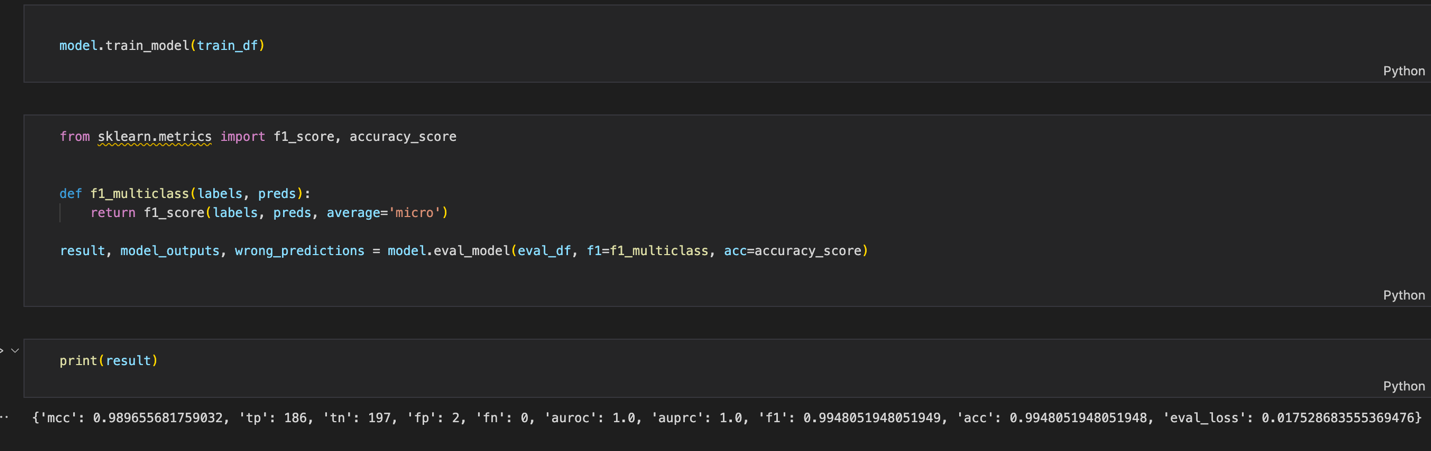
در مرحله بعدی دیتا فریم های آموزش و آزمون که به نسبت ۸۰/۲۰ split کرده بودیم را به کد معرفی می کنیم:



در مرحله بعدی مدل را آموزش می دهیم:



سپس از مدل خود ارزیابی به عمل می آوریم:



همانطور که انتظار می رفت عملکرد مدل بر پایه transformers به طور قابل توجهی از مدل بر پایه lstm بهتر است و نتیجه بهتری دارد که در ادامه این دو ساختار را مقایسه و نتیجه گیری می کنیم.

**نتیجه گیری:**

همانطور که بحث شد، ترانسفورماتورها سریعتر از مدل های مبتنی بر RNN هستند، زیرا تمام ورودی یک بار وارد می شود. آموزش LSTM در مقایسه با شبکه های ترانسفورماتور سخت تر است، زیرا تعداد پارامترها در شبکه های LSTM بسیار بیشتر است. علاوه بر این، انجام یادگیری انتقالی در شبکه های LSTM غیرممکن است. ترانسفورماتورها در حال حاضر شبکه ای پیشرفته برای مدل های seq2seq هستند. از این رو، ما با این واقعیت نتیجه می گیریم که شبکه های ترانسفورماتور بهترین دقت را ارائه می دهند و همچنین با پیچیدگی و هزینه محاسباتی کمتری همراه هستند.

**منابع :**

# [Why are LSTMs struggling to matchup with Transformers?](https://medium.com/analytics-vidhya/why-are-lstms-struggling-to-matchup-with-transformers-a1cc5b2557e3)

# [Natural Language Generation using BERT](https://medium.com/intel-student-ambassadors/natural-language-generation-using-bert-df6d863c3f52)

* [Simple Transformers — Multi-Class Text Classification with BERT, RoBERTa, XLNet, XLM, and DistilBERT](https://medium.com/swlh/simple-transformers-multi-class-text-classification-with-bert-roberta-xlnet-xlm-and-8b585000ce3a)