به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

درس شبکه‌های عصبی و یادگیری عمیق

تمرین چهارم

پرسش ۱	نام و نام خانوادگی	آرین فیروزی
	شماره دانشجویی	810100196
پرسش ۲	نام و نام خانوادگی	آرمان مجیدی
	شماره دانشجویی	810100205
	مهلت ارسال پاسخ	۱۴۰۱.۰۹.۲۹

فهرست

شکل‌ها

پرسش 1

پرسش 2

جدول‌ها

پرسش 1

پرسش 2

پرسش ۱. تشخیص هرزنامه

۱-۱. مجموعه داده

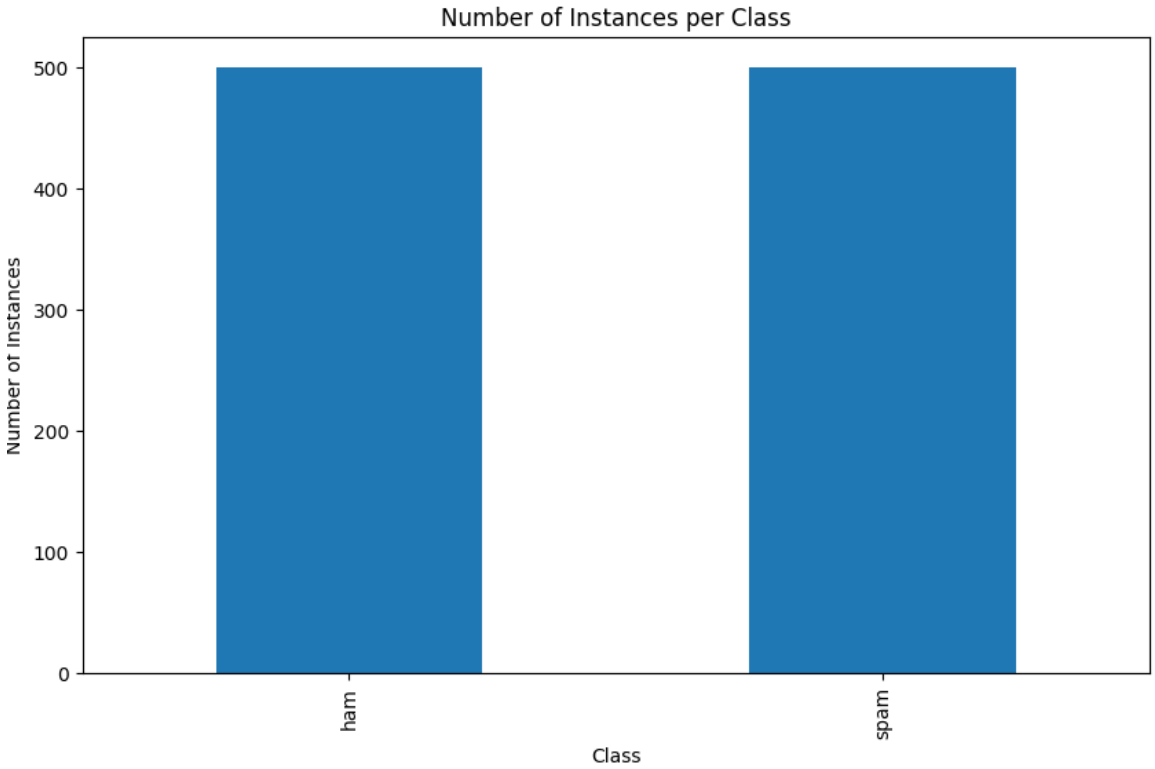
مجموعه داده خواسته شده توسط kagglehub دانلود شد و در پوشه data ذخیره شد. برای خواندن فایل از کتابخانه pandas استفاده کردیم. نمودار مربوط به فراوانی کلاس های این مجموعه در شکل 1.1 قابل مشاهده است که نشان میدهد تعداد دو کلاس باهم برابر بوده و دیتاست بالانسی داریم.

۱-۲. پیش پردازش

پیش پردازش در دو مرحله حذف متن نامربوط از داده ها و حذف stopword ها اعمال شد. برای حذف متن های نامربوط، از pattern های موجود در پایتون استفاده کردیم و برای url، ایمیل و شماره تلفن ها pattern های موجود در دادگان را استخراج کردیم و سپس تمام این پترن ها را ترکیب کرده و هر جا که چنین پترنی وجود داشت، آن قسمت را حذف کردیم. همچنین حروف تکراری پشت سر هم در تابع پاک کردن پترن پاک میشوند.

برای حذف کردن StopWord ها (کلمات پر تکراری که معنی خاصی را نمیرسانند)، از لیست persian stopwords که توسط وحید خرازی و پارسا کمالی پور گردآوری شده است[1] استفاده شد.

[1] https://github.com/kharazi/persian-stopwords



شکل 1.1. تعداد داده های هر کلاس در دیتاست

۱-۳. نمایش ویژگی

برای تبدیل داده های متنی به بردار، از توکنایزر parsBert که توسط HooshvareLab ارائه شده، استفاده کردیم. دادگان قبل از تبدیل به اندازه ی 32 padd شدند و سپس با استفاده از PCA ابعاد ویژگی را به 128 تبدیل کردیم تا خروجی به فرم [1000, 32, 120] تبدیل شوند. کاری که PCA انجام میدهد این است که فضای برداری بزرگتر را طوری به فضای کوچکتر مپ میکند که بیشترین میزان اطلاعات را نگه دارد. درواقع کاری که انجام میدهد به نوعی یافتن تصویر بردار در فضای کوچکتر است.

در پاسخ سوالات پرسیده شده، ParsBesrt تعداد ابعاد بردار را معادل Bert یعنی 768 در نظر گرفته است که این تعداد نشانگر تعداد ویژگی های کلمات است که توسط مدل استخراج شده و در فضای چند بعدی ماتریکس عددی نمایش داده میشوند. این ویژگی ها مربوط به معنی های کلمات هستند که توسط مدل تلاش میکنند یک نمایش عددی از کلمات ارائه دهند. این نمایش عددی بردار تعبیه نام دارد و در آن تلاش میشود کلماتی که از لحاظ معنایی با هم شباهت دارند یا با هم استفاده میشوند، بردار های نزدیک به هم داشته باشند. برای مثال کلماتی مثل "فوتبال"، "ورزش" و "استادیوم" احتمالا بردار های مشابهی دارند.

4-1. ساخت مدل

قبل از ساخت مدل، نیاز بود که label ها به صورت مناسب انکد شوند که با استفاده از labelEncoder به مقادیر عددی تبدیل کردیم و آن را با نسبت های خواسته شده تقسیم کردیم.

برای یافتن بهترین هایپر پارامترها، از gridsearch استفاده کردیم و با ترین کردن مدل به میزان 5 epoch، بهترین دقت را به عنوان بهترین هایپر پارامتر در نظر گرفتیم.

سپس مدل های خواسته شده را با استفاده از توابع کتابخانه Tensorflow و keras ایجاد کردیم. قابل توجه است که لایه Embedding که در مقاله استفاده شده بود، برای ما کاربردی ندارد چرا که به صورت دستی Embedding ها را به دست آورده ایم و نیازی به استفاده دوباره از این لایه نیست.

هایپر پارامتر های خواسته شده برای هر سه مدل به صورت 'batch_size': 8, 'learning_rate': 0.001, 'optimizer': 'Adam' محاسبه شد و با همین پارامتر ها مدل ها را آموزش دادیم. لازم به ذکر است که آموزش تنها به مقدار 10 epoch انجام گرفته است.

در پاسخ سوالات، همانطور که در مقاله اشاره شد، مدل های CNN برای شناسایی کلمات در طبقه بندی خوب عمل میکند، و مدل های LSTM میتوانند سری کلمات و نحوه ی ترکیب آنها در یک جمله را بهتر شناسایی میکند. همچنین CNN بیشتر robust است و سریعتر از LSTM همگرا میشود و نسبت به ورودی زیاد حساس نیست، در مقابل نمیتواند context متن را درک کند و کلمات را به صورت مجزا میبیند. با ترکیب این دو، میتوانیم درک متن LSTM را با Robustness و بهینگی محاسباتی CNN ترکیب کنیم و مدلی با دقت و بالاتر بدست بیاوریم.

5-1. ارزیابی

نتایج در جدول 1.1 آمده اند. این نتایج، همانطور که پیشبینی میشد، نشان میدهند که LSTM و CNN به تنهایی دقت خوبی دارند اما ترکیب این دو نتیجه ی بهتری ارائه میدهد. همچنین اگر از لحاظ زمانی نگاه کنیم متوجه میشویم که LSTM بیشتر از همه زمان برده و CNN سریعتر است.

جدول 1.1 دقت مدل های مختلف، ارزیابی شده توسط معیار های خواسته شده

Method	Accuracy	Precision	Recall	F1-Score	ROC AUC
LSTM	0.9667	0.9730	0.9600	0.9664	0.9667
CNN	0.9533	0.9474	0.9600	0.9536	0.9533
CNN-LSTM	0.9767	0.9735	0.9800	0.9767	0.9767

1-5. امتیازی

روش Bag of Words روشی برای نشان دادن کلمات در علم داده است. این ماتریس از مقادیر عددی برای نشان دادن تعداد کلمات به ازای هر نمونه استفاده میکند. برای مثال همانطور که در تصویر 1.2 مشاهده میشود، مقدار کلمه "دانم" در سطر 2 برابر 1 است، به این معنی که این کلمه یکبار در جمله مربوط استفاده شده است.

	بدنرفع	علویه	فیلسوفي	گرامافون	برنامه	username	سازگار	فیروز	دانم	pin	تقریبی	دوخته	تراز	مس	میشنوم	تشکر	نصاب	درایور	صحبتهات	غني
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
96	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
98	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

شکل 1.2. ماتریس bag of words

برای آموزش مدل های سنتی، از کتابخانه Sklearn مدل های مربوطه استخراج شدند. همینطور به منظور آموزش این مدل ها، نیاز به دادگان ورودی دو بعدی داشتیم که به علت Embedding ایجاد شده برای متد های قبلی، باید داده ها را برای مدل های جدید دو بعدی میکردیم که با استفاده از reshape انجام شد. متد های انتخاب شده برای این کار decision tree, logistic regression, random forest و Extratrees بودند که نتایج به دست آمده در جدول 1.2 قابل مشاهده است.

جدول 2.1 مقایسه مدل های شبکه عصبی با روش های سنتی

Method	ROC AUC	F1-Score	Recall	Precision	Accuracy
LSTM	0.9667	0.9664	0.9600	0.9730	0.9667
CNN	0.9533	0.9536	0.9600	0.9474	0.9533
CNN-LSTM	0.9767	0.9767	0.9800	0.9735	0.9767
Decision tree	0.9033	0.8854	0.9267	0.9055	0.9033
Logistic regression	0.9667	0.9667	0.9667	0.9667	0.9667
Random forest	0.9367	0.9281	0.9467	0.9373	0.9367
Extra trees	0.9467	0.9241	0.9733	0.9481	0.9467

پرسش 2 – پیش‌بینی ارزش نفت

2-1. مقدمه

در این پرسش ما به وسیله چهار روش GRU، LSTM، Bi-LSTM و ARIMA به پیش‌بینی قیمت نفت (CL=F) می‌پردازیم. ابتدا داده را توسط کتابخانه yfinance لود می‌کنیم و در محل کد آن را ذخیره می‌کنیم. سپس، گام‌های تنظیم‌شده توسط تمرین را انجام می‌دهیم. در ابتدا داده را مطابق با هدف تمرین تنظیم می‌کنیم. سپس، سه مدل GRU، LSTM و Bi-LSTM را تنظیم می‌کنیم و آن‌ها را آموزش می‌دهیم. در ادامه مدل ARIMA را بررسی می‌کنیم و به کمک کتابخانه‌های پایتون، اقدام به تشخیص پارامترهای مناسب می‌کنیم. در انتها نیز مدل ARIMA را آموزش می‌دهیم و نتایج بدست آمده را در یک جدول نشان می‌دهیم.

2-2. مجموعه دادگان و آماده‌سازی

اکنون به کمک کتابخانه yfinance دیتا را دانلود می‌کنیم. سپس دیتا را مطابق سایت، تمیز می‌کنیم و آن را در محل jupyter notebook ذخیره می‌کنیم.

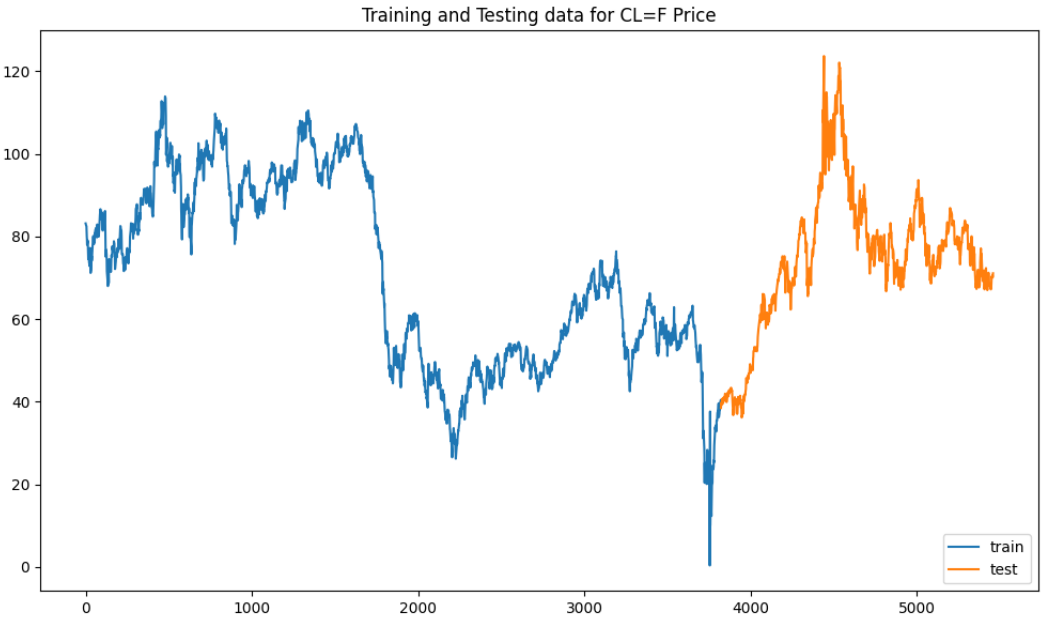
داده‌ای که توسط این کتابخانه ذخیره شود، داده null نخواهد داشت. اما طبق گفته پرسش ستون‌های با حضور داده null حذف خواهند شد. حال، 10% را نیز طبق خواسته پرسش حذف می‌کنیم. سپس، مطابق با مقاله داده‌شده 70% اول ستون Adj Close را به عنوان داده train و 30% باقی‌مانده را به عنوان داده test می‌گزینیم. سپس باید داده را مطابق با خواسته پرسش پنجره‌بندی کنیم. در نتیجه، داده خود را با پنجره‌های به طول 2 و با همپوشانی 50% انتخاب می‌کنیم. اگر بخواهیم نحوه تشکیل داده‌های x و y را با یک مثال توضیح دهیم، می‌توان به این نحو گفت که داده‌های روز 1 و 2 را به عنوان x و داده روز 3 را به عنوان y انتخاب می‌کنیم. در گام (ستون) بعد، داده‌های روز 2 و 3 را به عنوان داده x و داده روز 4 را به عنوان داده y انتخاب می‌کنیم. این روال را تا انتها انجام می‌دهیم. برای نرمالیزه کردن، از Min-Max Normalization برای نرمالیزه کردن داده‌ها استفاده می‌کنیم. تنها داده‌های x را نرمالیزه می‌کنیم و همچنین داده‌های test را به وسیله پارامترهای بدست آمده از داده‌های train نرمالیزه می‌کنیم. سایز داده‌های بدست آمده به شکل زیر می‌باشد:

- x_train: (3822, 2)
- y_train: (3822,)
- x_test: (1638, 2)
- y_test(1638,)

پس از آن، اقدام به رسم هیستوگرام داده Adj Close می‌کنیم. این هیستوگرام در شکل 2-1 نمایش داده شده‌است.

شکل 2-1. هیستوگرام داده **Adj Close**

همچنین، برای شهود بهتر از داده train و test، اقدام به رسم نمودار آن ها می‌کنیم. شکل 2-2 این نمودار را نشان می‌دهد.



شکل 2-2. نمودار داده **Adj Close**

3-2. پیاده‌سازی مدل‌ها

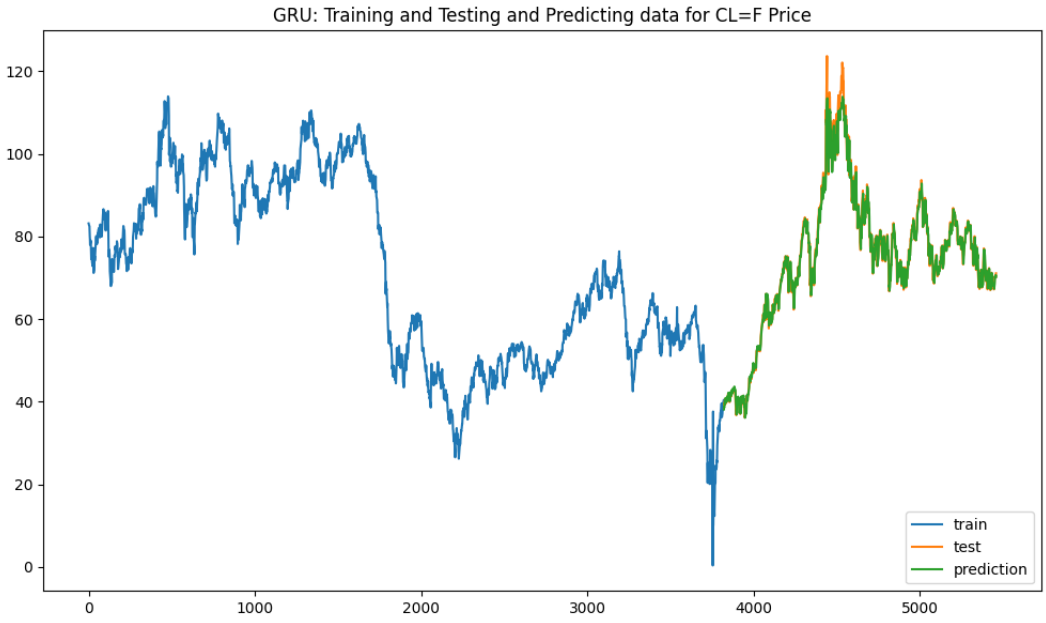
ابتدا به تعریف پارامترها مطابق با مقاله پیشنهادی می‌پردازیم. سپس مدل‌ها را طبق مدل‌های گفته‌شده در مقاله تعیین و تنظیم می‌کنیم. حال نتیجه هر مدل را به صورت مجزا می‌نویسیم:

- مدل GRU

این مدل پس از 50 ایپاک به نتایج زیر می‌رسد:

- loss: 2.2668
- MAE: 1.0298
- MAPE: 3.1066
- R-Squared: 0.9957
- RMSE: 1.4989

همچنین، نمودار مقایسه نتایج پیش‌بینی شده توسط این مدل و همچنین نتایج واقعی در شکل 3-2 آمده‌است.



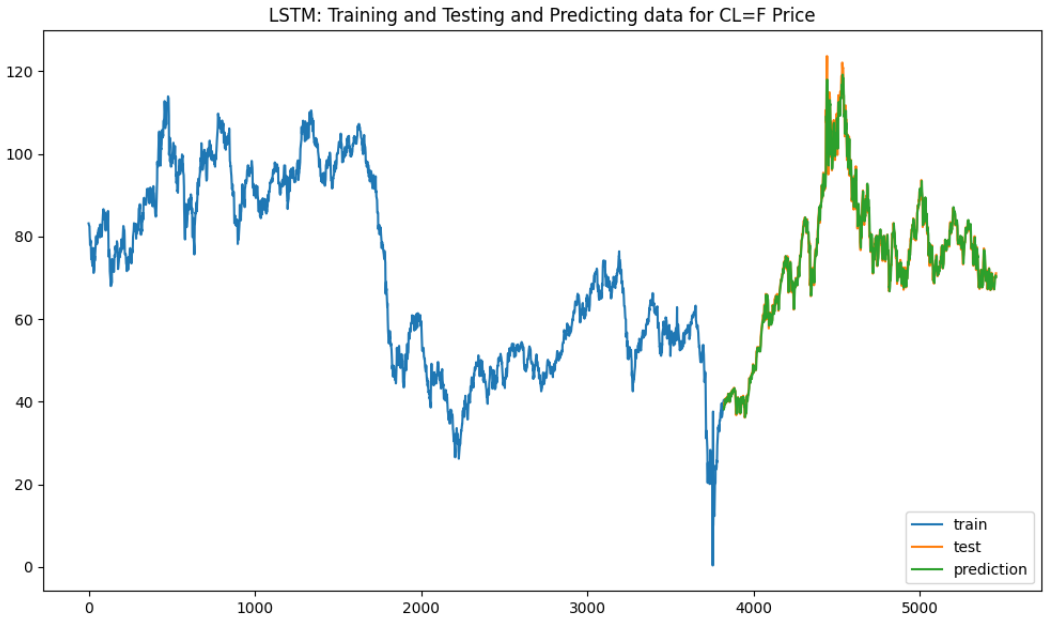
شکل 3-2. نمودار نتایج پیش‌بینی شده و نتایج واقعی مدل **GRU**

- مدل LSTM

این مدل پس از 50 ایپاک به نتایج زیر می‌رسد:

- loss: 2.5331
- MAE: 1.0724
- MAPE: 3.9815
- R-Squared: 0.9952
- RMSE: 1.5897

همچنین، نمودار مقایسه نتایج پیش‌بینی شده توسط این مدل و همچنین نتایج واقعی در شکل 2-4 آمده‌است.



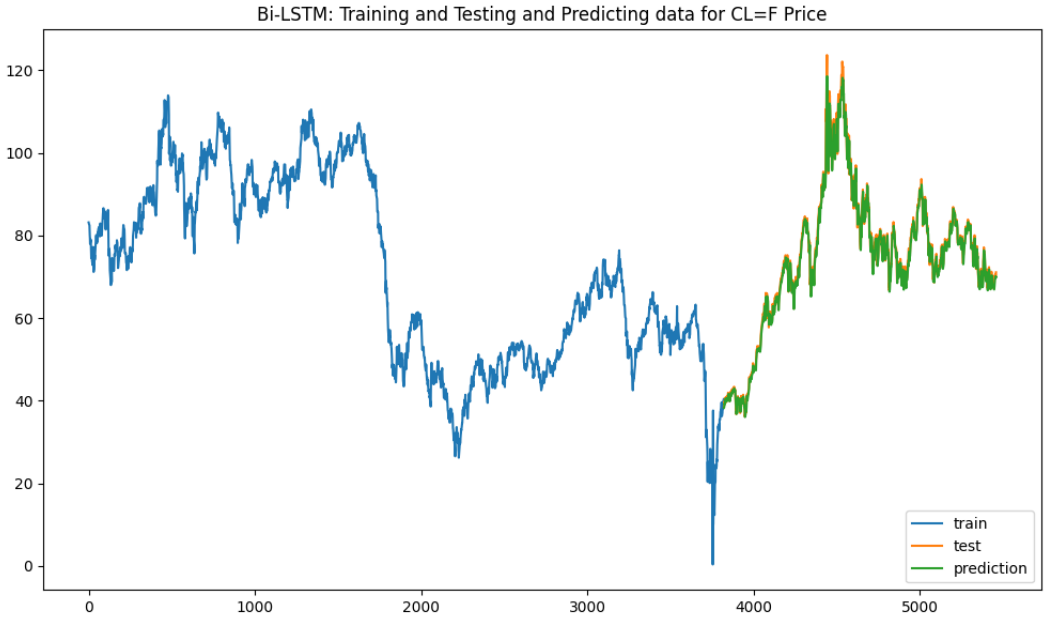
شکل 2-4. نمودار نتایج پیش‌بینی شده و نتایج واقعی مدل **LSTM**

- مدل Bi-LSTM

این مدل پس از 50 ایپاک به نتایج زیر می‌رسد:

  - loss: 1.5631
  - MAE: 0.8425
  - MAPE: 1.8595
  - R-Squared: 0.9969
  - RMSE: 1.2470

همچنین، نمودار مقایسه نتایج پیش‌بینی شده توسط این مدل و همچنین نتایج واقعی در شکل 2-5 آمده‌است.



شکل 2-5. نمودار نتایج پیش‌بینی شده و نتایج واقعی مدل **Bi-LSTM**

2-4. ARIMA

مدل ARIMA برای پیش‌بینی داده‌های زمانی که فصلی نیستند طراحی شده‌است. این مدل تشکیل‌شده از سه بخش می‌باشد:

- AR (Auto Regressive)

  این بخش به بررسی رابطه میان مقدار فعلی و مقادیر قبلی می‌پردازد.
- I (Integrated)

  این بخش به تفاضل‌گیری داده‌ها برای ایستا کردن داده‌ها می‌پردازد.
- MA (Moving Average)

این بخش نیز به بررسی ارتباط میان مقدار فعلی داده و همچنین خطای پیش‌بینی‌های گذشته می‌پردازد.

مدل SARIMA نیز نسخه توسعه‌یافته از ARIMA می‌باشد. این مدل برای داده‌هایی که فصلی هستند نیز کاربرد دارد. به همین دلیل دارای پارمترهایی اضافی برای مدل‌سازی این فصلیت داراست.

چون داده ما قیمت نفت می‌باشد و این داده، داده‌ای فصلی می‌باشد پیشنهاد می‌شود از مدل SARIMA استفاده شود. اما ما طبق خواسته پرسش از مدل ARIMA استفاده می‌کنیم.

از مزایای مدل ARIMA می‌توان موارد زیر را نام برد:

- سادگی و کاربردی بودن

- عدم نیاز به ورودی‌های پیچیده
- توسعه مدل‌های پیچیده‌تر
- توانایی در پیش‌بینی داده‌های ایستا

همچنین، از محدودیت‌های مدل ARIMA می‌توان موارد زیر را نام برد:

- نیاز به ایستاسازی داده‌ها
- نیاز به تنظیم دستی پارامترها
- عدم توانایی در مدل‌سازی روابط پیچیده
- حساسیت به نویز
- ناتوانی در مدل‌سازی داده‌های فصلی

مدل ARIMA سه پارامتر p، d و q دارد که به صورت خلاصه هر کدام از این پارامترها را توصیف می‌کنیم:

- پارامتر p

این پارامتر مربوط به بخش AR می‌باشد. این پارامتر نشان‌دهنده تعداد مقادیر گذشته می‌باشد.

- پارامتر d

این پارامتر مربوط به بخش I می‌باشد. این پارامتر نشان‌دهنده تعداد تفاضل‌گیری است که باید روی داده انجام شود تا داده ما ایستا شود.

- پارامتر q

این پارامتر مربوط به بخش MA می‌باشد. این پارامتر نشان‌دهنده ارتباط میان خطاهای پیش‌بینی گذشته و مقدار فعلی داده‌ها است.

برای بدست آوردن پارامترهای بهینه، از کتابخانه pmdarima استفاده می‌کنیم تا پارامترهای بهینه را بدست آوریم. پس از gridsearch برای بدست آوردن بهترین پارامتر، پارامترهای (p, d, q) به ترتیب مقادیر (3, 1, 0) بدست می‌آید. حال این پارامترها را به کتابخانه statsmodels می‌دهیم و نتایج را بدست می‌آوریم. نتایج به صورت زیر می‌باشد:

- loss (MSE): 1465.57
- MAE: 34.42
- MAPE: 42.72
- R-Squared: -4.07
- RMSE: 38.28

در انتها تمام نتایج را در یک جدول نشان می‌دهیم. جدول 2-1، جمع‌بندی تمام نتایج را نشان می‌دهد.

Method	*MSE*	*MAE*	*RMSE*	*R-squared*	*MAPE(%)*
GRU	2.2668	1.0298	1.4989	0.9957	3.1066
LSTM	2.5531	1.0724	1.5897	0.9952	3.9815
Bi-LSTM	1.5631	0.8425	1.2470	0.9969	1.8595
ARIMA	1465.57	34.42	38.28	-4.07	42.72

جدول 2-1. جمع‌بندی نتایج پرسش 2