به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

درس شبکه‌های عصبی و یادگیری عمیق

تمرین چهارم

نام و نام خانوادگی	محمدامین یوسفی
شماره دانشجویی	810100236
نام و نام خانوادگی	محمدرضا نعمتی
شماره دانشجویی	810100226

فهرست

شکل‌ها و جدول‌ها

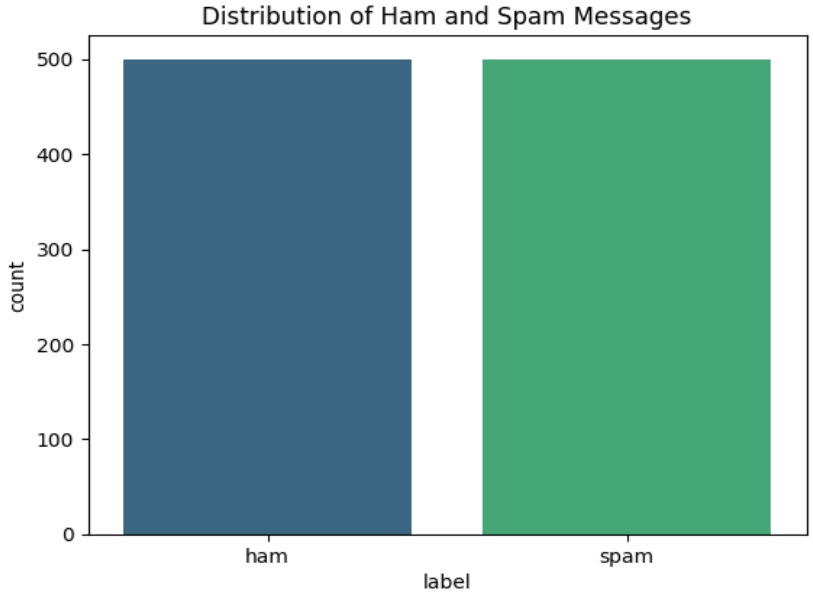
پرسش 1 – تشخی‌ص هرزنامه

1-1. مجموعه داده

ابتدا با مجموعه داده از کگل دریافت شد. در اشکال 1-1 و 1-2 می‌توانید کلاس‌های موجود در ستون Label و تعداد نمونه‌های هر کلاس را مشاهده کنید.

```
Label Distribution:
label
ham     500
spam    500
Name: count, dtype: int64
```

شکل 1-1. کلاس‌های موجود در ستون label



شکل 1-2. تعداد نمونه‌های هر کلاس

1-2. پیش‌پردازش داده‌ها

در قطعه کد زیر می‌توانید پیش‌پردازش‌های انجام شده روی مجموعه داده را مشاهده نمایید. برای نرمال‌سازی متن و همچنین دریافت لیست کلمات توقف از کتابخانه hazm استفاده شده است. بالای هر بخش از این تابع، با استفاده از کامنت، وظیفه آن بخش توضیح داده شده است.

```python
def preprocess_text(text):
    # Remove URLs
    text = re.sub(r'http\S+|www\S+|https\S+', '', text, flags=re.MULTILINE)

    # Remove email addresses
    text = re.sub(r'\S+@\S+\.\S+', '', text)

    # Remove phone numbers (any sequence of 7+ digits)
    text = re.sub(r'\b\d{7,}\b', '', text)

    # Reduce repeated characters (e.g., علی → عععلی)
    text = re.sub(r'(\w)\1{2,}', r'\1', text)

    # Remove usernames
    text = re.sub(r'@[^ ]+', '', text)

    # Remove htmls
    text = re.sub(re.compile('<.*?>') , '', text)

    # Remove hashtags
    text = re.sub(r'#', '', text)

    # Normalize text
    text = normalizer.normalize(text)

    # Tokenize
    tokens = word_tokenize(text)

    # Remove stopwords
    tokens = [word for word in tokens if word not in stopwords]

    # Remove empty tokens
    tokens = [word for word in tokens if word.strip() != '']
    return ' '.join(tokens)
```

3-1. نمایش ویژگی

ParsBert، یک مدل زبانی مبتنی بر BERT برای فارسی، از توکن‌سازی WordPiece برای تبدیل متن به زیرکلمه‌ها و سپس به اندیس‌های عددی استفاده می‌کند. این فرایند شامل نرمال‌سازی متن و افزودن توکن‌های خاص مانند [CLS] و [SEP] است. توکن‌های تولیدشده در مرحله تعبیه‌سازی به بردارهای عددی تبدیل می‌شوند که با ترکیب تعبیه‌های کلمه، موقعیت و نوع توکن، معنای کلمات و ترتیب

آن‌ها را در متن نمایش می‌دهند. این بردارها برای وظایف زبانی نظیر طبقه‌بندی و استخراج ویژگی استفاده می‌شوند.

در اینجا از توکن‌ساز و مدل ParsBert استفاده شد تا بردار تعبیه‌ساز متون ایجاد شود.

- **سوال: ابعاد پیش‌فرض بردار تعبیه در ParsBERT چقدر است؟**

  ابعاد پیش‌فرض بردار تعبیه در ParsBert برابر 768 است.
- **سوال: تعداد ابعاد این بردار بیانگر چیست؟**

این ابعاد نشان‌دهنده تعداد نودهای خروجی در لایه‌های مخفی مدل است و به طور پیش‌فرض برای نمایش ویژگی‌های هر توکن در فضای تعبیه استفاده می‌شود. هر بعد از این بردار نمایانگر یک جنبه از ارتباطات معنایی و زمینه‌ای کلمه است. مثلاً یک بعد ممکن است اطلاعاتی درباره فعل بودن یا اسم بودن توکن ارائه دهد، در حالی که بعد دیگر ممکن است به زمینه‌ای که کلمه در آن استفاده شده (مثلاً علمی، روزمره، یا احساسی) اختصاص یابد.

- **سوال: مفهوم بردار تعبیه را توضیح دهید و بیان کنید کدام کلمات موجود در مجموعه داده ممکن است تعبیه‌ای نزدیک به هم داشته باشند؟**

بردار تعبیه نمایش عددی کلمات یا توکن‌ها در یک فضای برداری است که روابط معنایی و نحوی بین آن‌ها را حفظ می‌کند. این بردار، یک نمایش عددی چگال از کلمات است که مدل به کمک آن، معنای کلمات را در فضای چندبعدی نمایش می‌دهد درParsBert ، این بردارها حاصل ترکیب سه مؤلفه هستند:

- **Token Embeddings :** نمایش معنای اولیه توکن
- **Positional Embeddings** : اطلاعات مربوط به موقعیت توکن در جمله.
- **Segment Embeddings:** تمایز میان بخش‌های مختلف جمله.

کلماتی که معنای مشابه یا کاربرد یکسانی دارند (مانند «زیبا» و «خوشگل» یا «دانش‌آموز» و «محصل») معمولا تعبیه‌های نزدیک به هم دارند. همچنین کلمات در یک زمینه مشابه (مثلا «کتاب» و «کتابخانه») نیز ممکن است بردارهای نزدیک به هم داشته باشند.

1-4. ساخت مدل

در ابتدا داده‌ها را به نسبت‌های گفته شده در صورت پروژه تقسیم می‌کنیم. در شکل 1-3 می‌توانید چگونگی و ابعاد این تقسیم‌بندی را مشاهده نمایید.

```
x_train: torch.Size([560, 120])
y_train: torch.Size([560])
x_val:   torch.Size([140, 120])
y_val:   torch.Size([140])
x_test:  torch.Size([300, 120])
y_test:  torch.Size([300])
```

شکل 1-3. ابعاد هر دسته پس از تقسیم‌بندی داده‌ها

سپس الگوریتم جستجوی حریصانه را برای هر سه مدل CNN ، LSTM و CNN-LSTM در فضای گفته شده اعمال می‌کنیم. بهترین هایپرپارامترها برای هریک از مدل‌ها در جدول 1-1 نمایش داده شده است.

جدول 1-1. بهترین هایپرپارامترها برای هر مدل

مدل	batch_size	learning_rate	optimizers
CNN	64	0.0001	Adam
LSTM	8	0.001	Adam
CNN-LSTM	64	0.0001	Adam

- **سوال: نقاط قوت و ضعف هر یک از مدل‌ها CNN و LSTM چیست؟**
  - مدل CNN به دلیل توانایی بالای خود در استخراج ویژگی‌های مکانی مانند لبه‌ها، بافت‌ها، و الگوها، به‌ویژه در داده‌های تصویری بسیار قدرتمند است. این مدل با استفاده از عملیات کانولوشن وpooling ، تعداد پارامترها را کاهش داده و باعث می‌شود محاسبات سریع‌تر و بهینه‌تر انجام شود. علاوه بر این، CNNنسبت به تغییرات مکانی در داده‌ها مقاوم است و طراحی ساده‌تری نسبت به مدل‌های پیچیده‌تر مانند LSTM دارد. همچنین برای پردازش داده‌ها وbatch های بزرگ به خوبی عمل می‌کند. با این حال، این مدل محدودیت‌هایی نیز دارد؛ از جمله اینکه نمی‌تواند وابستگی‌های بلندمدت یا ترتیب زمانی داده‌ها را مدل‌سازی کند. علاوه بر این، ورودی‌های آن باید طول

ثابت داشته باشند و برای دستیابی به نتایج قابل‌قبول، معمولاً نیاز به حجم زیادی از داده‌ها دارد.

- مدل LSTM به طور خاص برای داده‌های ترتیبی یا زمانی مانند متن، صوت، و سری‌های زمانی طراحی شده است. این مدل با استفاده از سلول‌های حافظه (Memory Cell)و دروازه‌های کنترل، می‌تواند وابستگی‌های بلندمدت و کوتاه‌مدت را به‌خوبی حفظ کند. توانایی LSTM در حفظ اطلاعات کلیدی از داده‌های قبلی، آن را برای حل مشکلات رایج RNN مانند محو شدن یا انفجار گرادیان مناسب کرده است. یکی از ویژگی‌های برجسته این مدل، عدم نیاز به طول ثابت برای ورودی‌ها است، که آن را برای داده‌هایی با طول متغیر ایده‌آل می‌کند. با این حال، LSTMنیز نقاط ضعفی دارد؛ از جمله اینکه به دلیل ساختار پیچیده‌تر، زمان و منابع محاسباتی بیشتری نیاز دارد. همچنین در یادگیری ویژگی‌های محلی) مانند (n-gram و داده‌های مکانی، کارایی کمتری نسبت به مدل‌های تخصصی مانند CNN دارد.

- **سوال: ادغام این دو مدل با چه هدفی انجام میشود؟**

ادغام مدل‌های CNN و LSTM با هدف بهره‌گیری از نقاط قوت هر دو مدل انجام می‌شود و سیستمی توانمند برای مدیریت داده‌های پیچیده ایجاد می‌کند .CNN با استخراج ویژگی‌های محلی، مانند الگوهای n-gram در متن یا ویژگی‌های بصری در تصاویر و ویدئوها، داده‌های مکانی را پردازش می‌کند. سپس این ویژگی‌ها به LSTM منتقل می‌شود تا وابستگی‌های بلندمدت و روابط زمانی یا ترتیبی مدل‌سازی شوند. این ترکیب در کاربردهایی مانند تحلیل ویدئوها، تشخیص حرکات، پردازش متن تصویری، و شناسایی هرزنامه‌ها بسیار کارآمد است، زیرا هر دو جنبه‌ی محلی و ترتیبی داده‌ها را به طور همزمان مدیریت می‌کند. نتیجه این ادغام، دقت بالاتر و عملکرد بهتر در وظایف پیچیده‌ای است که به تحلیل عمیق‌تر داده‌ها نیاز دارند.

1-5. ارزیابی

در اینجا از متریک‌های گفته شده در مقاله استفاده می‌شود.

جدول 1-2. نتایج ارزیابی مدل‌ها

مدل	Accuracy	Precision	Recall	F1-Score	ROC AUC
CNN	**0.976**	**0.976**	**0.976**	**0.976**	**0.976**
LSTM	0.966	0.966	0.966	0.966	0.966
CNN-LSTM	0.966	0.966	0.966	0.966	0.966

1-6. امتیازی

مدل کیسه کلمات متون را به بردارهایی عددی تبدیل می‌کند که نشان‌دهنده تعداد تکرار کلمات در متن هستند، بدون توجه به ترتیب آن‌ها. ابتدا تمام کلمات منحصربه‌فرد مجموعه داده به‌عنوان واژگان جمع‌آوری می‌شوند. سپس برای هر متن، برداری ساخته می‌شود که طول آن برابر با تعداد کلمات واژگان است و مقادیر آن نشان‌دهنده فراوانی هر کلمه در متن است.

این روش ساده است اما محدودیت‌هایی دارد؛ مثل نادیده گرفتن ترتیب و معنای کلمات. برای بهبود، از تکنیک‌هایی مثل TF-IDFیا حذف کلمات غیرمفید استفاده می‌شود. با وجود کاربرد در طبقه‌بندی متون و تحلیل داده‌های اولیه، BoW اغلب با روش‌های پیشرفته‌تری مثل Word2Vec یا BERT جایگزین می‌شود.

مدل‌های انتخاب شده:

- **Support Vector Machine** : SVM یک الگوریتم یادگیری نظارت‌شده است که برای مسائل طبقه‌بندی و رگرسیون استفاده می‌شود. هدف اصلی SVM پیدا کردن یک ابرصفحه است که بتواند داده‌های مختلف را به بهترین شکل ممکن از هم جدا کند. این الگوریتم با استفاده از داده‌های نزدیک‌ترین به ابرصفحه (که به آن‌ها بردارهای پشتیبان گفته می‌شود)، مرز تصمیم‌گیری را تعیین می‌کند. اگر داده‌ها به‌صورت خطی قابل تفکیک نباشند، از هسته‌ها (kernels) مانند RBF یا پلی‌نومیال برای تبدیل فضای ویژگی استفاده می‌کند.

- **Logistic Regression** : یک روش آماری برای پیش‌بینی احتمال یک رویداد دودویی (مانند صفر یا یک) است. این الگوریتم بر اساس رگرسیون خطی عمل می‌کند، اما خروجی آن به کمک تابع لجستیک (سیگموئید) به بازه [0, 1] نگاشت می‌شود. لجستیک رگرسیون برای

مسائل طبقه‌بندی دوکلاسه (مانند پیش‌بینی اسپم یا غیر اسپم بودن ایمیل) بسیار موثر است. همچنین می‌توان آن را برای طبقه‌بندی چندکلاسه با تکنیک‌هایی مثل One-vs-Rest یا Softmax Regression توسعه داد.

- **Random Forest** : یک الگوریتم یادگیری جمعی است که از ترکیب چندین درخت تصمیم‌گیری برای بهبود دقت پیش‌بینی استفاده می‌کند. این مدل با انتخاب تصادفی داده‌ها و ویژگی‌ها در ساخت هر درخت، از بیش‌برازش جلوگیری می‌کند و تنوع مدل را افزایش می‌دهد. پیش‌بینی نهایی از طریق رأی‌گیری اکثریت (در مسائل طبقه‌بندی) یا میانگین‌گیری (در مسائل رگرسیون) صورت می‌گیرد.
- **Multinomial Naive Bayes** : این مدل یک نسخه از الگوریتم Naive Bayes است که برای مسائل طبقه‌بندی چندکلاسه و متون طراحی شده است. فرض اصلی این مدل استقلال شرطی ویژگی‌ها است؛ به این معنی که حضور یا غیاب یک ویژگی (مثلاً یک کلمه) مستقل از دیگر ویژگی‌ها در متن در نظر گرفته می‌شود. Multinomial Naive Bayes بر اساس توزیع چندجمله‌ای عمل می‌کند و برای تحلیل متون، دسته‌بندی ایمیل‌ها و طبقه‌بندی داده‌های گسسته بسیار موثر است.

در جدول 1-3 می‌توانید نتیجه ارزیابی همه مدل‌ها با یکدیگر را بررسی کنید.

جدول 1-3. نتایج ارزیابی مدل‌ها

مدل	Accuracy	Precision	Recall	F1-Score	ROC AUC
Support Vector Machine	0.896	0.901	0.896	0.896	0.974
Logistic Regression	0.943	0.943	0.943	0.943	0.981
Random Forest	0.930	0.930	0.930	0.929	**0.983**
Multinomial Naive Bayes	0.936	0.936	0.936	0.936	0.980
CNN	**0.976**	**0.976**	**0.976**	**0.976**	0.976
LSTM	0.966	0.966	0.966	0.966	0.966
CNN-LSTM	0.966	0.966	0.966	0.966	0.966

نتایج جدول 1-3 نشان می‌دهند که مدل‌های مبتنی بر شبکه عصبی در مقایسه با مدل‌های سنتی یادگیری ماشین عملکرد بهتری دارند. مدل CNN با دقت 0.976، بهترین نتیجه را ارائه داده و تمامی معیارهای دیگر مانند Precision، Recall و F1-Score نیز در بالاترین سطح قرار دارند. مدل‌های LSTM و CNN-LSTM نیز عملکردی مشابه و نزدیک به هم داشته‌اند، اما برخلاف انتظار، CNN-LSTM نتوانسته عملکرد بهتری نسبت به CNN یا LSTM به‌صورت مجزا نشان دهد. این موضوع ممکن است به تنظیمات معماری مدل یا داده‌های استفاده‌شده مرتبط باشد.

در میان مدل‌های سنتی، Logistic Regression با دقت 0.943 بهتر از سایر روش‌ها عمل کرده است، در حالی که Random Forest و Multinomial Naive Bayes نیز عملکرد مناسبی از خود نشان داده‌اند. با این حال، این مدل‌ها همچنان در معیارهای مختلف از مدل‌های شبکه عصبی عقب‌تر هستند.

نکته قابل‌توجه این است که مدل CNN علاوه بر دقت بالا، در تمامی معیارها عملکرد متعادلی داشته و نشان می‌دهد که توانایی شناسایی الگوهای پیچیده و مهم در داده‌ها را دارد. اگرچه انتظار می‌رفت که مدل CNN-LSTM به دلیل ترکیب قدرت CNN در استخراج ویژگی‌های مکانی و LSTM در تحلیل داده‌های ترتیبی، عملکرد بهتری داشته باشد، اما این امر محقق نشده است.

پرسش ۲ – پیش‌بینی ارزش نفت

۲-۲. مجموعه دادگان و آماده‌سازی

ابتدا با استفاده از کتابخانه yfinance دیتای CL=F را از تاریخ 01-01-2010 تا تاریخ 18-12-2024 دریافت می‌کنیم. مجموعه داده CL=F مربوط به داده‌های تاریخی قیمت آتی نفت خام است که در بورس کالای نیویورک (NYMEX) معامله می‌شود. این داده شامل اطلاعات قیمت باز، بالا، پایین، بسته شدن، حجم معاملات و قیمت بسته‌شدن تعدیل‌شده (Adj Close) است.

```
import yfinance as yf

ticker = "CL=F"
start_date = "2010-01-01"
end_date = "2024-12-18"

crude_oil_data = yf.download(ticker, start=start_date,
end=end_date)

crude_oil_data.to_csv("crude_oil_data.csv")
```

Price	Adj Close	Close	High	Low	Open	Volume
Ticker	CL=F	CL=F	CL=F	CL=F	CL=F	CL=F
Date						
2010-01-04	81.510002	81.510002	81.680000	79.629997	79.629997	263542
2010-01-05	81.769997	81.769997	82.000000	80.949997	81.629997	258887
2010-01-06	83.180000	83.180000	83.519997	80.849998	81.430000	370059
2010-01-07	82.660004	82.660004	83.360001	82.260002	83.199997	246632
2010-01-08	82.750000	82.750000	83.470001	81.800003	82.650002	310377

شکل2-1. نمونه‌هایی از دیتای دریافت شده **CL=F**

همانطور که مشاهده می‌شود، روزهایی وجود دارد که برای آنها داده‌ای ثبت نشده است. همچنین یکی از روزها، مقدار Adj Close منفی است که واضحا دیتای اشتباه نیازمند جایگزینی است. علاوه‌بر این روزهای ثبت نشده و یا قیمت منفی، 10 درصد از داده‌ی موجود را نیز به عنوان null قرار می‌دهیم تا بتوانیم آنها را با روش‌های معرفی شده، جایگزین کنیم.

```
crude_oil_data.isnull().sum()
```

```
Price      Ticker
Date                  0
Adj Close  CL=F      1704
Close      CL=F      1704
High       CL=F      1704
Low        CL=F      1704
Open       CL=F      1704
Volume     CL=F      1704
dtype: int64
```

شکل 2-2. تعداد داده‌های **NaN** پس از این مرحله

جایگزینی داده‌های null

در تحلیل داده‌های سری زمانی، مدیریت مقادیر null از اهمیت زیادی برخوردار است، زیرا این مقادیر می‌توانند مدل‌های پیش‌بینی را تحت تاثیر قرار دهند. در زیر به بررسی چهار مورد از روش‌های رایج برای پر کردن این مقادیر میپردازیم.

1. روش **Forward Fill**: در این روش مقادیر null با آخرین مقدار non-null موجود در سری زمانی پر می‌شوند. این روش به دلیل سادگی و حفظ روند زمانی داده‌ها در مواردی که داده‌ها به صورت ترتیبی اهمیت دارند، مانند قیمت سهام یا دمای هوا، بسیار استفاده می‌شود. اما باید توجه داشت که این روش ممکن است در صورت وجود دوره‌های طولانی از مقادیر خالی، اطلاعات نادرستی ارائه دهد.
2. روش **Backward Fill**: در این روش هر مقدار null با نزدیک‌ترین مقدار non-null بعد از آن جایگزین می‌شود. این روش زمانی مفید است که مقادیر آتی در سری زمانی از اهمیت بالایی برخوردار باشند یا داده‌ها به صورت ترتیبی نیازمند اطلاعات آینده برای تکمیل باشند. مانند Forward Fill، این روش نیز به راحتی پیاده‌سازی می‌شود و روند زمانی داده‌ها را حفظ می‌کند. با این حال، در صورتی که مقادیر خالی در بخش‌های ابتدایی داده زیاد باشند، ممکن است نتایج غیرقابل اعتمادی ایجاد شود. علاوه بر این، Backward Fill در داده‌هایی با نوسانات شدید یا وابستگی قوی به مقادیر قبلی، عملکرد بهینه‌ای نخواهد داشت.
3. روش **Linear Interpolation**: با استفاده از خط مستقیم بین دو مقدار not-null مجاور، مقادیر خالی را برآورد می‌کند. این روش با فرض تغییرات یکنواخت در داده‌ها، مقادیر دقیق‌تری نسبت به Forward Fill ارائه می‌دهد. با این حال، اگر داده‌ها دارای نوسانات شدید یا الگوهای پیچیده باشند، ممکن است این روش دقت کافی نداشته باشد.

4. روش **Moving Average**: از میانگین مقادیر مجاور برای پر کردن مقادیر خالی استفاده می‌شود. این روش می‌تواند نویز داده‌ها را کاهش داده و روند کلی را حفظ کند، بنابراین در سری‌های زمانی با نوسانات زیاد کاربرد زیادی دارد. با این حال، بسته به طول بازه window size، ممکن است اطلاعات مهمی از داده‌ها حذف یا کم‌اثر شوند.

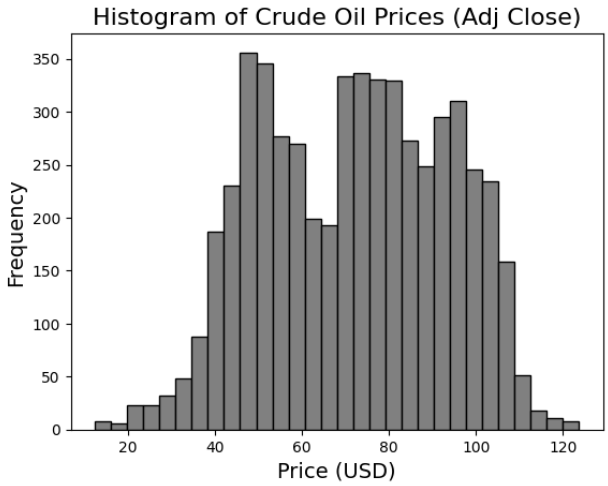
در مجموعه داده CL=F که تغییرات قیمت آتی نفت خام را ثبت می‌کند، حفظ پیوستگی و دقت داده‌ها برای تحلیل و پیش‌بینی اهمیت بالایی دارد. با توجه به ماهیت این داده‌ها که به صورت سری زمانی و وابسته به رویدادهای اقتصادی و بازار عمل می‌کنند، پر کردن مقادیر خالی نیازمند رویکردی است که روند طبیعی تغییرات را تا حد ممکن بازتاب دهد.

در ابتدا از روش Linear Interpolation برای پر کردن مقادیر خالی میان داده‌ها به دلیل طبیعت نسبتا پیوسته تغییرات قیمتی در بازارهای مالی استفاده شد. این روش فرض می‌کند که تغییرات قیمت در بازه‌های زمانی کوتاه معمولا به صورت خطی یا نزدیک به خطی هستند، که با رفتار واقعی بازار نفت خام هم‌خوانی دارد. این روش مقادیر میانی را با دقت خوبی پر کند، اما در مواجهه با داده‌های خالی در ابتدای مجموعه، کارایی لازم را ندارد زیرا اطلاعاتی برای برآورد مقادیر اولیه وجود ندارد. برای این حل این مشکل، Backward Fill به عنوان راهکار به کار گرفته شد. قیمت‌های ابتدایی در این نوع داده‌ها معمولا به عنوان مقادیر پایه برای تحلیل روندهای آتی استفاده می‌شوند. پر کردن این داده‌ها با نزدیک‌ترین مقدار موجود در آینده باعث می‌شود که تحلیل روندها و مقایسه داده‌های اولیه با داده‌های آتی امکان‌پذیر شود، بدون اینکه نیاز به فرضیات پیچیده باشد.

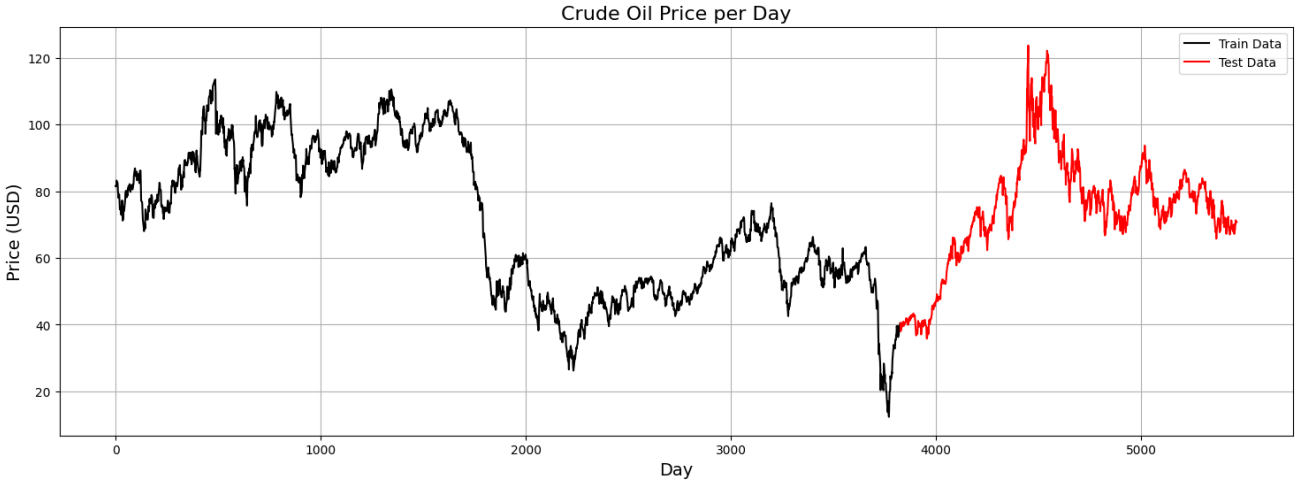
```
crude_oil_interpolated = crude_oil_data.interpolate(method='linear')
crude_oil_interpolated.bfill(inplace=True)
```

تقسیم داده و بررسی توزیع قیمت

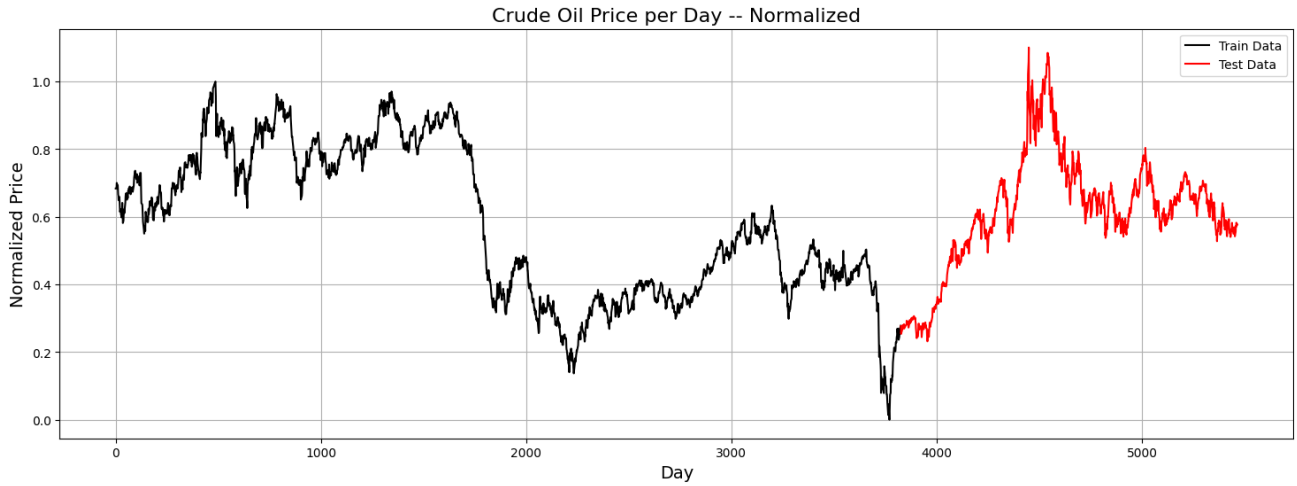
طبق متن مقاله، دیتای زمانی را به نسبت 70-30 به دیتای آموزش و آزمایش تقسیم می‌کنیم. توجه کنید که با توجه به ماهیت دیتا، این تقسیم بندی رندوم نیست و 70 درصد ابتدای سری زمانی به دیتای آموزش و ادامه‌ی آن به دیتای آزمایش اختصاص داده می‌شود. در شکل‌های 2-3 و 2-4، توزیع کلی قیمت و همچنین روند قیمت در کل سری زمانی مشاهده می‌شود. همچنین نمودار قیمت نرمال‌شده بر اساس روش Min-Max Scaling در شکل 2-5 آورده شده است.



شکل 2-3. نمودار هیستوگرام توزیع قیمت



شکل 2-4. نمودار قیمت در هر روز



شکل 2-5. نمودار قیمت نرمال‌شده در هر روز

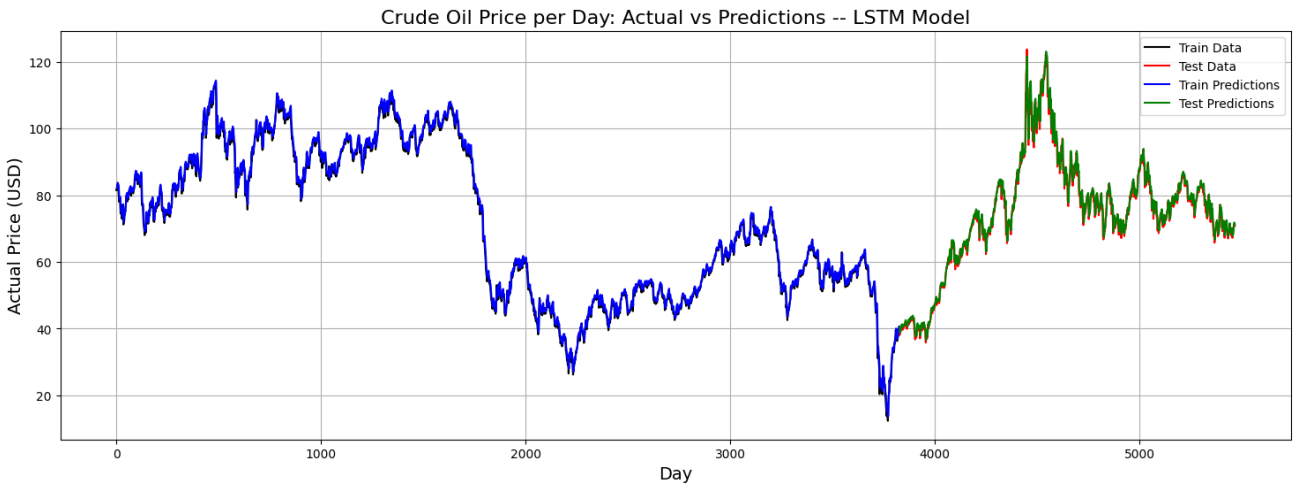
۲-۳: پیاده‌سازی مدل‌ها

برای پیاده‌سازی و آموزش مدل‌های LSTM، GRU و Bi-LSTM از هایپرپارامترهای زیر طبق مقاله استفاده شده است. نتایج پیش‌بینی به همراه مقادیر واقعی برای هر سه مدل پس از آموزش در شکل‌های 2-6 تا 2-8 آورده شده است.

جدول 2-1. هایپرپارامترهای مدل‌های **LSTM**، **GRU** و **Bi-LSTM**

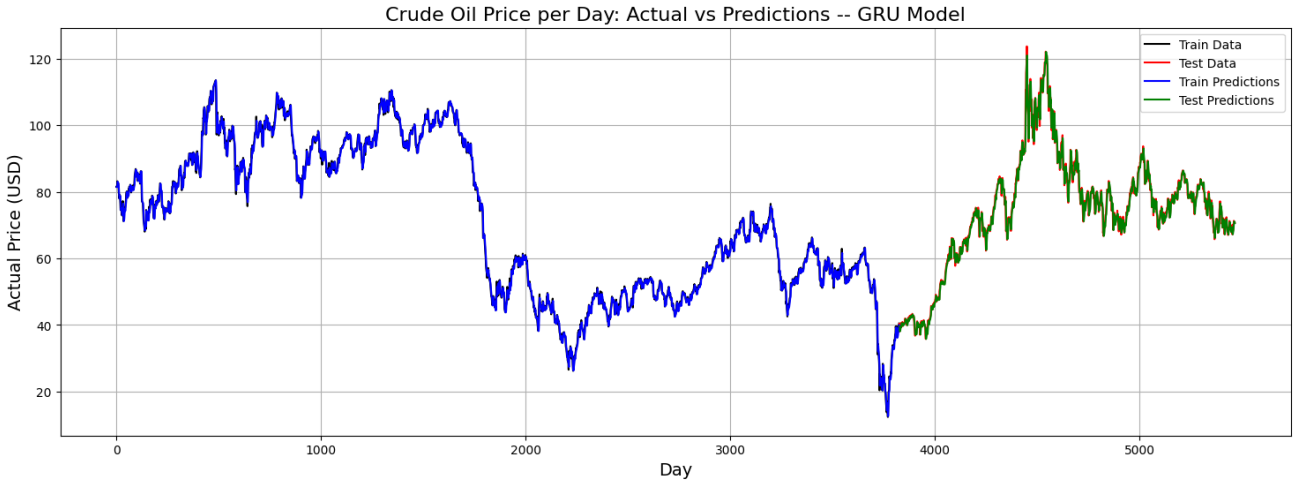
Epochs	50
Batch Size	100
Optimizer	Adam
Learning Rate	0.001
Loss Function	MSE
Metrics	MAPE, R-Squared, RMSE, MAE
Units	512 (LSTM & GRU) – 1024 (Bi-LSTM)

```
model = Sequential([
    LSTM(512, input_shape=(train_sequences.shape[1], 1)),
    Dense(1)])
model.compile(optimizer=Adam(learning_rate=0.001), loss='mean_squared_error')

history = model.fit(train_sequences, train_labels, epochs=50, batch_size=100,
     validation_data=(test_sequences, test_labels), verbose=1)
```



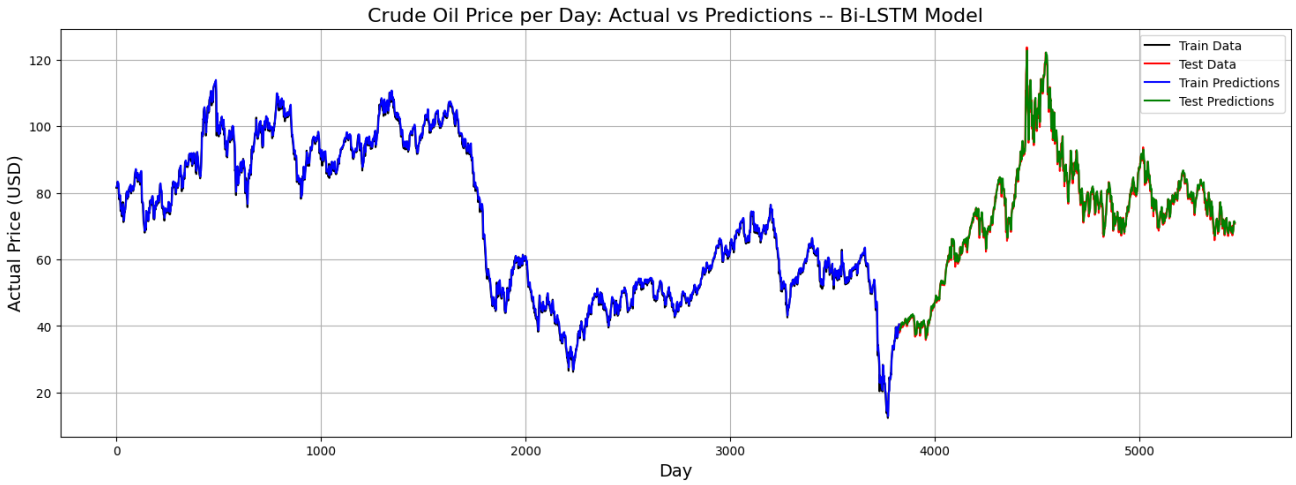
شکل 2-6. نمودار مقادیر واقعی و پیش‌بینی شده توسط مدل **LSTM**

```
gru_model = Sequential([
    GRU(512, input_shape=(train_sequences.shape[1], 1)),
    Dense(1)
])
gru_model.compile(optimizer=Adam(learning_rate=0.001),loss='mean_squared_error')

gru_history = gru_model.fit(train_sequences, train_labels,
                            epochs=50, batch_size=100,
                            validation_data=(test_sequences, test_labels),
                            verbose=1)
```



شکل 2-**7**. نمودار مقادیر واقعی و پیش‌بینی شده توسط مدل **GRU**

```
bi_lstm_model = Sequential([
    Bidirectional(LSTM(1024), input_shape=(train_sequences.shape[1], 1)),
    Dense(1)
])
bi_lstm_model.compile(optimizer=Adam(learning_rate=0.001),
      loss='mean_squared_error')

bi_lstm_history = bi_lstm_model.fit(train_sequences, train_labels,
                                    epochs=50, batch_size=100,
                                    validation_data=(test_sequences, test_labels),
                                    verbose=1)
```



شکل 2-8. نمودار مقادیر واقعی و پیش‌بینی شده توسط مدل **Bi-LSTM**

معیارهای ارزیابی

جدول 2-2. معیارهای استفاده شده برای ارزیابی و فرمول آنها

Metric	Formula	Description
Mean Absolute Error (MAE)	$\frac{1}{n}\sum_{i=1}^{n}\lvert y_i - \hat{y}_i\rvert$	میانگین قدر مطلق اختلاف بین مقادیر واقعی و پیش‌بینی‌شده است. این معیار حساس به بزرگی خطا نیست و به سادگی نشان‌دهنده میزان خطای متوسط است.
Mean Absolute Percentage Error (MAPE)	$\frac{100}{n}\sum_{i=1}^{n}\left\lvert\frac{y_i - \hat{y}_i}{y_i}\right\rvert$	میانگین درصد خطای مطلق است که خطای نسبی را نسبت به مقدار واقعی ارزیابی می‌کند. این معیار برای داده‌هایی با مقادیر نزدیک به صفر مناسب نیست زیرا می‌تواند به مقادیر غیرمنطقی منجر شود.
Root Mean Squared Error (RMSE)	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$	میانگین ریشه دوم مربع خطاهاست و خطاهای بزرگ را بیشتر وزن می‌دهد. بنابراین، در مسائلی که خطاهای بزرگ اهمیت بیشتری دارند، RMSE به‌عنوان معیار مناسب‌تری استفاده می‌شود.
R-Squared (R^2)	$1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$	بخشی از تغییرات داده واقعی را که مدل می‌تواند توضیح دهد نشان می‌دهد. مقدار آن بین 0 و 1 است؛ هر چه به 1 نزدیک‌تر باشد، مدل بهتر عمل کرده است. این معیار به مدل‌های رگرسیون محدود است و تاثیری از واحد مقادیر ندارد.

در جدول 2-3، نتایج عملکرد مدل‌ها در پیش‌بینی دیتای آزمایش آورده شده است.

جدول 2-3. عملکرد مدل‌ها بر روی دیتای آزمایش

Method	MAE	RMSE	R-Squared	MAPE (%)
LSTM	1.3234	1.9269	0.9872	1.755
GRU	1.2227	1.7636	0.9893	1.6073
Bi-LSTM	1.035	1.5465	0.9917	1.3632

مشاهده می‌شود که Bi-LSTM به‌طور کلی بهترین عملکرد را در مقایسه با سایر مدل‌ها نشان داده است. این مدل با داشتن کمترین مقادیر MAE (1.0350) و(1.5465) RMSE، نشان‌دهنده پایین‌ترین خطای مطلق و ریشه مربعی در پیش‌بینی‌هاست. همچنین مقدار R-Squared برابر با 0.9917، نشان می‌دهد که این مدل بخش بیشتری از تغییرات داده‌های واقعی را توضیح می‌دهد. مقدار MAPE نیز 1.3632 درصد است که بیانگر دقت نسبی بالای این مدل در پیش‌بینی قیمت‌ها است.

عملکرد بهتر Bi-LSTM می‌تواند به دلیل ساختار خاص این مدل باشد. Bi-LSTM از دو LSTM استفاده می‌کند که داده‌ها را در دو جهت (پیش‌رو و عقب‌رو) پردازش می‌کنند. این ویژگی به مدل اجازه می‌دهد تا اطلاعات مربوط به وابستگی‌های بلندمدت را هم از گذشته و هم از آینده داده‌ها استخراج کند. در مسائل سری زمانی که روندهای پیچیده و روابط طولانی‌مدت میان نقاط داده وجود دارد، این قابلیت می‌تواند به دقت بیشتری در پیش‌بینی منجر شود. داده‌های سری زمانی CL=F نیز به دلیل نوسانات و تغییرات غیرخطی قیمت‌ها، به مدل‌هایی نیاز دارند که بتوانند چنین وابستگی‌هایی را به‌خوبی درک کنند.

مدل GRU نیز عملکرد مطلوبی داشته و با MAE و RMSE نسبتا پایین (1.2227 و 1.7636) و مقدار R-Squared برابر با 0.9893 نشان‌دهنده دقت بالای آن در پیش‌بینی است. علاوه بر این، مقدار MAPE برابر با 1.6073 درصد است که نشان‌دهنده توانایی قابل‌قبول این مدل در ارزیابی نسبی خطا است، اما همچنان از Bi-LSTM ضعیف‌تر عمل می‌کند.

یکی از معایب Bi-LSTM، کندی آن در آموزش و پیش‌بینی است. این مدل به دلیل پردازش دوطرفه، به منابع محاسباتی بیشتری نیاز دارد که زمان اجرا را افزایش می‌دهد. در مقابل، مدل GRU سریع‌ترین آموزش و پیش‌بینی را داشته است. ساختار ساده‌تر GRU در مقایسه با LSTM و Bi-LSTM باعث کاهش پیچیدگی محاسباتی می‌شود، در حالی که همچنان دقت مناسبی ارائه می‌دهد.

مدل LSTM در مقایسه با دو مدل دیگر عملکرد ضعیف‌تری دارد. با MAE برابر با 1.3234 و RMSE برابر با 1.9269، خطای این مدل بیشتر است. مقدار R-Squared برابر با 0.9872 نیز نشان می‌دهد که این مدل در توضیح تغییرات داده‌ها نسبت به GRU و Bi-LSTM ضعیف‌تر عمل کرده است. علاوه بر این،

مقدار MAPE برابر با 1.7550 درصد است که کمترین دقت نسبی در میان مدل‌ها را نشان می‌دهد. البته باید اشاره کرد که همین مقادیر معیارها نیز بسیار خوب هستند و نشان از یادگیری خوب مدل دارند و در اینجا صرفا در مقایسه با بقیه مدل‌ها، LSTM کمی ضعیف عمل کرده است.

۴-۲: ARIMA

تفاوت ARIMA و SARIMA

مدل‌های ARIMA و SARIMA از مدل‌های کلاسیک و پرکاربرد در تحلیل داده‌های سری زمانی هستند. هر دو به‌طور مشابه از سه مولفه اصلی برای پیش‌بینی داده‌های سری زمانی استفاده می‌کنند، اما تفاوت‌های قابل‌توجهی نیز بین آنها وجود دارد. نخستین تفاوت در این است که مدل ARIMA برای سری‌های زمانی غیر ایستا استفاده می‌شود و به‌طور خاص برای داده‌هایی که روند یا الگوی فصلی ندارند طراحی شده است. این مدل از بخش Integrated (I) برای از بین بردن روندهای زمانی استفاده می‌کند تا داده‌ها ایستا شوند، سپس از اجزای AutoRegressive (AR) و Moving Average (MA) برای مدل‌سازی روابط درون‌سری و پیش‌بینی استفاده می‌کند.

اما مدل SARIMA، که به‌طور خاص برای داده‌های سری زمانی با ویژگی‌های Seasonal طراحی شده است، علاوه بر سه مولفه ذکرشده برای ARIMA، چهار مولفه فصلی اضافه دارد: Seasonal AR، Seasonal I، Seasonal MA و Seasonal Period. این اجزا به SARIMA این امکان را می‌دهند که نوسانات یا الگوهای فصلی در داده‌ها را شبیه‌سازی کرده و پیش‌بینی کند. به‌عنوان مثال، در داده‌هایی که تغییرات قیمت‌ها در فصول مختلف سال مشاهده می‌شود، مدل SARIMA قادر به شبیه‌سازی این تغییرات و پیش‌بینی روند فصلی آنها خواهد بود.

درARIMA فقط یک مجموعه از پارامترها برای مدل‌سازی روابط داده‌ها و پیش‌بینی استفاده می‌شود. در حالی که درSARIMA، پارامترهای اضافی برای مدلسازی داده‌های Seasonal نیاز است که پیچیدگی بیشتری در ساختار مدل ایجاد می‌کند و فرآیند انتخاب بهترین مدل می‌تواند زمان‌برتر و پیچیده‌تر باشد. از سوی دیگر، ARIMA ساختار ساده‌تری دارد و برای داده‌های فاقد ویژگی‌های فصلی یا زمانی که روندهای ساده دارند، مناسب‌تر است.

مزایا و محدودیت‌های ARIMA

مدل ARIMA مزایای زیادی دارد که آن را به یکی از مدل‌های پرکاربرد در تحلیل سری‌های زمانی تبدیل کرده است. از جمله مزایای آن می‌توان به سادگی و قابلیت فهم آسان آن اشاره کرد. ARIMA می‌تواند روندهای زمانی را به‌خوبی مدل‌سازی کرده و پیش‌بینی دقیقی برای سری‌های زمانی بدون

ویژگی‌های فصلی یا پیچیده ارائه دهد. این مدل به‌ویژه در داده‌هایی که تغییرات آنها به‌طور پیوسته و به‌صورت تصادفی هستند، کارایی خوبی دارد و نیازی به فرضیات پیچیده یا داده‌های خارجی ندارد. یکی از اصلی‌ترین محدودیت‌های ARIMA این است که قادر به مدل‌سازی داده‌های فصلی یا دوره‌ای نیست و تنها برای سری‌های زمانی غیر ایستا که فاقد الگوهای فصلی هستند، مناسب است. همچنین، انتخاب پارامترهای مناسب (p، d و q) می‌تواند چالش‌برانگیز باشد و نیاز به آزمون و خطای زیادی دارد. علاوه بر این، مدل ARIMA فرض می‌کند که داده‌ها از یک روند ثابت پیروی می‌کنند و در صورت وجود نوسانات یا تغییرات غیرخطی، ممکن است دقت پیش‌بینی‌ها کاهش یابد.

پارامترهای ARIMA

مدل ARIMA از سه پارامتر اصلی برای مدل‌سازی و پیش‌بینی استفاده می‌کند که به‌طور ریاضی به صورت $ARIMA(p,d,q)$ شناخته می‌شوند. مدل ARIMA با ترکیب این سه پارامتر مدل‌سازی و پیش‌بینی سری‌های زمانی ایستا را انجام می‌دهد.

1. پارامتر **p** (AutoRegressive): این پارامتر نشان‌دهنده تعداد lags متغیر وابسته در مدل است که برای پیش‌بینی مقادیر آینده از آن‌ها استفاده می‌شود. یا به زبانی ساده‌تر، p تعداد مشاهدات گذشته‌ای است که در پیش‌بینی مورد استفاده قرار می‌گیرند. منظور از lags، مقدار تاخیر در استفاده از داده‌های گذشته در مدل‌های پیش‌بینی است. در مدل AR، فرض می‌شود که مقدار جاری سری زمانی به صورت خطی از مقادیر گذشته آن سری زمانی وابسته است.

مفهوم ریاضی

در مدل AR با درجه p، پیش‌بینی مقدار آینده به صورت زیر انجام می‌شود:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t$$

که در آن y_t مقدار جاری سری زمانی، $(\phi_1, \phi_2, \ldots, \phi_p)$ ضرایب AR و ϵ_t خطای مدل است.

2. پارامتر **d** (Integrated): نشان‌دهنده تعداد دفعات تفاضل‌گیری لازم برای تبدیل داده‌ها به یک سری زمانی ایستا یا stationary است. داده‌های سری زمانی ممکن است روند یا فصلی داشته باشند که به‌طور طبیعی ایستا نیستند. تفاضل‌گیری به معنای محاسبه تغییرات بین مقادیر متوالی سری زمانی است تا روند یا فصول از داده‌ها حذف شود.

مفهوم ریاضی

برای ایجاد ایستایی در سری زمانی، معمولا از تفاضل‌گیری به‌صورت زیر استفاده می‌شود:

$$\Delta y_t = y_t - y_{t-1}$$

اگر سری زمانی هنوز ایستا نشد، این فرآیند تفاضل‌گیری ممکن است چندین بار تکرار شود.

3. پارامتر **q** (Moving Average): این پارامتر تعداد lags خطاهای پیش‌بینی است که برای بهبود پیش‌بینی‌های آینده در نظر گرفته می‌شود. در مدل MA، پیش‌بینی آینده تنها به خطاهای پیش‌بینی‌های گذشته وابسته است، نه مقادیر واقعی سری زمانی. این خطاها از نوسانات تصادفی در سری زمانی ناشی می‌شوند و مدل MA سعی می‌کند این خطاها را برای بهبود پیش‌بینی‌ها لحاظ کند.

مفهوم ریاضی

در مدل MA با درجه q، پیش‌بینی به صورت زیر می‌شود:

$$y_t = \mu + \epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2} + \cdots + \theta_q\epsilon_{t-q}$$

که در آن μ میانگین سری زمانی، ϵ_t خطا در زمان t، و $(\theta_1, \theta_2, \ldots, \theta_q)$ ضرایب MA هستند.

پارامترهای بهینه ARIMA برای دیتای CL=F

در این بخش با استفاده از کتابخانه pmdarima، بهترین ضرایب را پیدا می‌کنیم. این کتابخانه شامل الگوریتم‌هایی مانند auto_arima است که به‌طور خودکار بهترین پارامترهای مدل ARIMA را برای داده‌ها پیدا می‌کند تا نیازی به تنظیم دستی پارامترها نباشد. الگوریتم auto_arima بر اساس معیار AIC بهترین پارامترها را انتخاب میکند. AIC (Akaike Information Criterion) یک معیار آماری است که برای ارزیابی کیفیت مدل‌های آماری مختلف استفاده می‌شود. AIC به‌ویژه برای مدل‌های رگرسیونی و سری‌های زمانی کاربرد دارد. هدف AIC این است که مدلی را انتخاب کنیم که هم بهترین fitting را به داده‌ها داشته باشد و هم پیچیدگی مدل را در نظر بگیرد تا از overfitting جلوگیری شود. به عبارت دیگر، AIC مدلی را انتخاب می‌کند که به‌طور بهینه تعادلی بین دقت پیش‌بینی و پیچیدگی مدل برقرار کند.

```
auto_arima_model = auto_arima(
    arima_train_data,
    start_p=1, start_q=1,
    max_p=5, max_q=5,  # Range for p and q
    d=None,            # Automatically determine 'd'
    seasonal=False,
    trace=True,
    error_action='ignore',
    suppress_warnings=True,
    stepwise=True
)
```

خروجی این الگوریتم در شکل 2-9 آورده شده است.

```
Performing stepwise search to minimize aic
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=10759.642, Time=0.91 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=10791.942, Time=0.35 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=10761.854, Time=0.22 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=10764.258, Time=0.32 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=10790.391, Time=0.06 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=10760.723, Time=0.59 sec
 ARIMA(1,1,2)(0,0,0)[0] intercept   : AIC=10760.642, Time=0.92 sec
 ARIMA(0,1,2)(0,0,0)[0] intercept   : AIC=10758.751, Time=0.77 sec
 ARIMA(0,1,3)(0,0,0)[0] intercept   : AIC=10760.609, Time=1.39 sec
 ARIMA(1,1,3)(0,0,0)[0] intercept   : AIC=10762.724, Time=1.32 sec
 ARIMA(0,1,2)(0,0,0)[0]             : AIC=10757.104, Time=0.39 sec
 ARIMA(0,1,1)(0,0,0)[0]             : AIC=10762.642, Time=0.17 sec
 ARIMA(1,1,2)(0,0,0)[0]             : AIC=10758.990, Time=0.34 sec
 ARIMA(0,1,3)(0,0,0)[0]             : AIC=10758.957, Time=0.51 sec
 ARIMA(1,1,1)(0,0,0)[0]             : AIC=10757.985, Time=0.33 sec
 ARIMA(1,1,3)(0,0,0)[0]             : AIC=10761.074, Time=0.59 sec

Best model:  ARIMA(0,1,2)(0,0,0)[0]
Total fit time: 9.201 seconds
Best ARIMA parameters: (0, 1, 2)
```

شکل 2-9. خروجی الگوریتم **auto_arima** برای پیدا کردن بهترین پارامتر های **ARIMA**

همانطور که دیده می‌شود، مقادیر p=0، d=1 و q=2 انتخاب شده است.

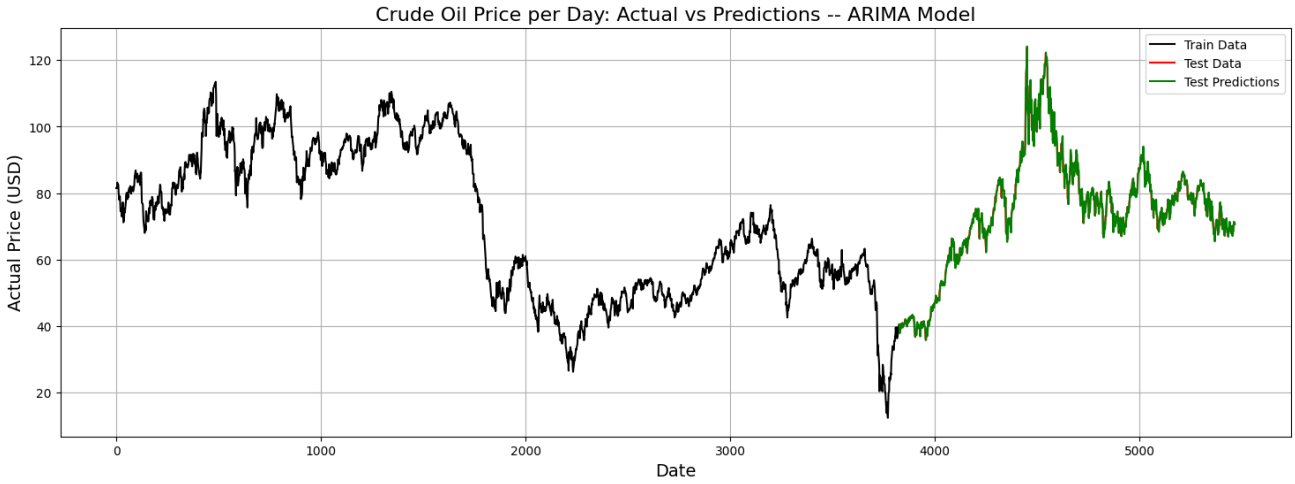
- مقدار p=0 یعنی هیچ تاثیری از مقادیر گذشته (AutoRegressive) در مدل وجود ندارد.
- مقدار d=1 یعنی داده‌ها یکبار تفاضل‌گیری شده‌اند تا ایستا شوند.
- مقدار q=2 یعنی مدل شامل تاثیر خطاهای پیش‌بینی در دو زمان قبل است.

اجرای مدل و مقایسه

برای پیش‌بینی مقادیر دیتای آزمایش، از روش rolling forecast استفاده می‌شود. در این روش پیش‌بینی‌ها به‌طور تدریجی و با اضافه کردن هر نقطه جدید به داده‌های قبلی انجام می‌شود. در این فرآیند، ابتدا داده‌های آموزشی به‌عنوان history داده‌ها در نظر گرفته و به مدل داده می‌شود. سپس برای هر داده در داده‌های آزمایش، مدل ARIMA با استفاده از داده‌های تاریخچه تا آن نقطه fit می‌شود و پیش‌بینی برای یک گام آینده انجام می‌شود. پیش‌بینی حاصل ذخیره و سپس داده‌ی واقعی آن نقطه از داده‌های آزمایش به history داده‌ها اضافه می‌شود تا برای پیش‌بینی نقطه بعدی مورد استفاده قرار گیرد. Fit کردن مجدد مدل برای هر داده جدید به دلیل تغییرات احتمالی در روندها و الگوهای سری‌های زمانی ضروری است. در سری‌های زمانی، وابستگی‌ها و روندها ممکن است با گذشت زمان تغییر کنند و fit کردن مجدد مدل به‌طور گام به گام این امکان را می‌دهد که مدل روندها و الگوهای جدید را شبیه‌سازی کند. به‌ویژه در سری‌های زمانی که داده‌ها نوسان زیادی دارند، پیش‌بینی‌های مدل ممکن است با تغییرات ناگهانی یا روندهای جدید دچار خطا شوند. با به‌روزرسانی مدل به‌صورت تدریجی، مدل می‌تواند به‌طور پیوسته خود

را با شرایط جدید تطبیق دهد و پیش‌بینی‌های دقیق‌تری ارائه دهد. نتیجه پیش‌بینی و مقادیر واقعی داده‌های آموزشی در شکل 2-10 آورده شده است.

```
rolling_predictions = []
history = list(arima_train_data)

for actual in arima_test_data:
    model = ARIMA(history, order=auto_arima_model.order)
    model_fit = model.fit()

    forecast = model_fit.forecast(steps=1)[0]
    rolling_predictions.append(forecast)

    history.append(actual)
```



شکل 2-10. نمودار مقادیر واقعی و پیش‌بینی شده توسط مدل **ARIMA(0,1,2)**

جدول 2-4. عملکرد هر چهار مدل بر روی دیتای آزمایش

Method	MAE	RMSE	R-Squared	MAPE (%)
LSTM	1.3234	1.9269	0.9872	1.755
GRU	1.2227	1.7636	0.9893	1.6073
Bi-LSTM	1.035	1.5465	0.9917	1.3632
ARIMA (0,1,2)	0.8758	1.3780	0.9934	1.1539

مدل ARIMA (0,1,2) بهترین عملکرد را در پیش‌بینی داده های آزمایش نشان داده است. با MAE برابر با 0.8758، RMSE برابر با 1.3780، و R-Squared برابر با 0.9934، این مدل کمترین خطا و بالاترین توانایی در توضیح تغییرات داده‌ها را ارائه داده است. همچنین، مقدار MAPE برابر با 1.1539

درصد نشان‌دهنده دقت بالای پیش‌بینی‌هاست. یکی از دلایل این عملکرد برجسته می‌تواند استفاده از rolling forecast باشد که در آن مدل به‌طور پیوسته و در بازه‌های زمانی مختلف به‌روزرسانی می‌شود. این روش به مدل اجازه می‌دهد تا به‌طور مداوم به داده‌های جدید واکنش نشان دهد و پیش‌بینی‌های دقیق‌تری ارائه کند.

برای مقایسه با مقاله، با نتایج شکل 2-11 که بر روی دیتا SPOT است مقایسه می‌کنیم.

Table 7. MAE, RMSE, R-Squared, and MAPE of Crude Palm Oil Price in Rotterdam (SPOT)

Method	MAE	RMSE	R-squared	MAPE (%)
•LSTM	52.1434	74.6622	0.9431	4.2732
•**LSTM** [34]	**436.2500**	**578.0000**	**0.9000**	**2.8500**
•GRU	37.5861	57.7891	0.9659	3.0493
•Bi-LSTM	55.4324	77.5116	0.9387	4.5689
•**ARIMA (1,1,5)** [14]	**14.0376**	**43.5947**	**0.9846**	**2.4500**
•**ARIMA (2,2,2)** [14]	**14.4271**	**45.4282**	**0.9833**	**2.5126**
•Simple RNN [11]	38.5600	50.8500	0.3900	3.8100
•SVR [13]	279.4392	343.3406	0.0460	34.9876

شکل 2-11. یکی از جدول های نتایج در مقاله

در ابتدا باید گفت که معیار های MAE و RMSE برای مقایسه بین دو دیتاست مختلف مناسب نیستند. چون بستگی به مقیاس داده ها دارند و مقیاس داده‌های مقاله و پروژه انجام شده متفاوت است.

اما برای مقایسه با R-Squared و MAPE که به مقیاس داده‌ها حساس نیستند، مشاهده می‌شود که در هر سه مدل LSTM، GRU و Bi-LSTM، مدل‌های اجرا شده در این پروژه و با دیتای CL=F نتایج بسیار بهتری داشته اند و پیش‌بینی دقیق‌تری توانسته‌اند انجام دهند.

مدل‌های ARIMA ای که در مقاله به آن رفرنس داده شده است نیز از مدل پیاده‌سازی شده در این پروژه عملکرد ضعیفتری دارند ولی این اختلاف کم است و چندان بزرگ نیست. که این میتواند قدرت مدل ARIMA در صورت پیدا کردن پارامترهای خوب برای آن را نشان دهد.

همچنین سایر مدل‌های رفرنس شده درون جدول نیز عملکرد بسیار ضعیفی دارند و اصلا جای بررسی ندارند.