دانشگاه لرستان

دانشکده فنی و مهندسی

رشته مهندسی کامپیوتر

عنوان

پیش بینی قیمت سهام

نگارش

آتوسا طغیانی

استاد راهنما

دکتر محمد باقر دولتشاهی

مرداد ماه ۱۴۰۰

دانشگاه لرستان

دانشکده فنی و مهندسی

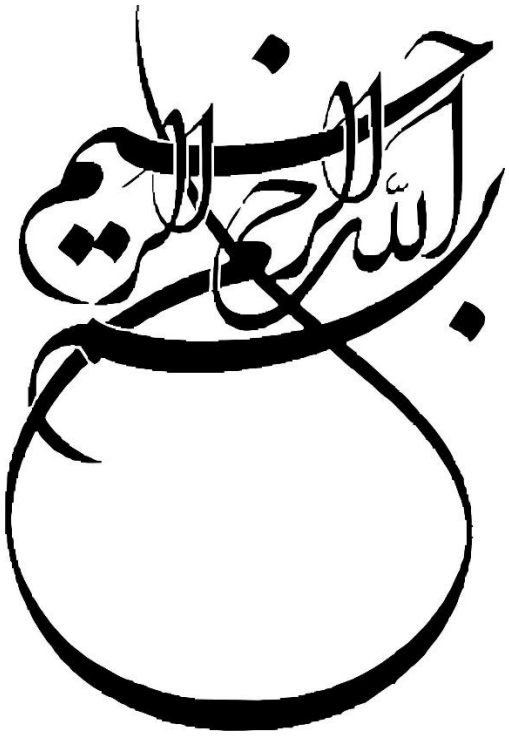
رشته مهندسی کامپیوتر

عنوان: پیش بینی قیمت سهام

نگارش: آتوسا طغیانی

استاد راهنما: دکتر محمد باقر دولتشاهی

تاریخ و امضا


بسم الله الرحمن الرحيم

سپاسگزاری

بدین وسیله از زحمات و تلاش بی‌دریغ استاد محترم جناب دکتر دولتشاهی و خانواده عزیزم صمیمانه سپاسگزاری می‌نمایم.

تقدیم به پدر و مادر عزیز و مهربانم که در سختی‌ها و دشواری‌های زندگی همواره یاوری دلسوز، فداکار و پشتیبانی محکم و مطمئن برایم بوده‌اند.

چکیده

پیش‌بینی دقیق بازده بازار سهام به دلیل ماهیت ناپایدار کاری بسیار پیچیده است و برای محاسبه قیمت بلند مدت سهام نیاز به یک الگوریتم قوی دارد. با معرفی هوش مصنوعی و استفاده کردن از روش‌های یادگیری عمیق بسیاری از مشکلات پیش‌بینی با موفقیت حل شده است. در این پژوهش از شبکه عصبی حافظه کوتاه مدت (LSTM) ، که در تجزیه و تحلیل سری‌های زمانی داده‌ها خوب است، برای پیش‌بینی قیمت سهام FAANG استفاده می‌شود. این سیستم در مقایسه با الگوریتم‌های پیش بینی قیمت سهام در حال حاضر نتایج دقیقی را ارائه می دهد.

واژگان کلیدی: شبکه عصبی حافظه کوتاه مدت (LSTM)، سهام FAANG ، پیش‌بینی قیمت سهام

فهرست شکل‌ها

فهرست جدول‌ها

فهرست مطالب

عنوان **صفحه**

فصل اول: مقدمات

۱-۱. مقدمه

بازار سهام ماهیتی پویا، غیرقابل پیش‌بینی و غیر خطی دارد. پیش‌بینی قیمت سهام یک کار چالش برانگیز است زیرا به عوامل مختلفی از جمله شرایط سیاسی، اقتصاد جهانی، گزارشات مالی شرکت، عملکرد و ... بستگی دارد. بنابراین، برای به حداکثر رساندن سود و به حداقل رساندن زیان، استفاده از تکنیک‌هایی برای پیش‌بینی ارزش سهام بسیار مفید است.

پیش‌بینی بهتر قیمت سهام یک مرجع کلیدی برای استراتژی معاملاتی بهتر و تصمیم گیری توسط سرمایه گذاران عادی و کارشناسان مالی است. جدا از پیش بینی جهت قیمت سهام، پیش‌بینی جهت بازار سهام یکی از مسائل مهم در مطالعات تحلیل مالی اخیر محسوب می‌شود.

به طور سنتی ، دو رویکرد اصلی برای پیش‌بینی قیمت سهام ارائه شده است:

- روش تجزیه و تحلیل فنی: از قیمت تاریخی سهام مانند قیمت بسته شدن و افتتاح ، حجم معامله شده ، مقادیر نزدیک و غیره سهام برای پیش بینی قیمت آتی سهام استفاده می‌کند.
- نوع دوم تحلیل کیفی است: که بر اساس عوامل خارجی مانند مشخصات شرکت، وضعیت بازار، عوامل سیاسی و اقتصادی، رسانه‌های اجتماعی و حتی وبلاگ‌ها توسط تحلیلگر اقتصادی انجام می‌شود.

امروزه از تکنیک‌های پیشرفته هوشمند مبتنی بر تجزیه و تحلیل فنی یا بنیادی برای پیش‌بینی قیمت سهام استفاده می شود. به ویژه، برای تجزیه و تحلیل بازار سهام، اندازه داده‌ها عظیم و همچنین غیرخطی است. برای بررسی این انواع، مدل‌های داده‌ای کارآمد لازم است که بتواند الگوهای پنهان و روابط پیچیده را در این مجموعه داده بزرگ شناسایی کند. بر اساس تحقیقات ثابت شده است که تکنیک های یادگیری ماشین در این زمینه نسبت به روش‌های گذشته کارایی را ۶۰-۸۶ درصد بهبود می‌بخشد.

اکثر کارهای قبلی در این زمینه از الگوریتم‌های کلاسیک مانند رگرسیون خطی، تئوری پیاده‌روی تصادفی (RWT) ، همگرایی/ واگرایی میانگین متحرک (MACD) استفاده می‌کردند.

در سال‌های اخیر، به دلیل پیشرفت فناوری‌های GPU ، قدرت محاسبات بسیار بهبود یافته است. مدل‌های شبکه عصبی مربوط به یادگیری عمیق برای بسیاری از برنامه‌های موفق در زمینه‌های مختلف همراه با حجم زیادی از آموزش داده‌ها تسریع شده و در دسترس است.

این موفقیت به ما در حل و مقابله با بسیاری از برنامه‌های پیش‌بینی کمک می‌کند. همانطور که می‌دانیم قیمت سهام، داده‌های سری زمانی است و می‌توان از آن برای الگوگیری و شناسایی روندها با استفاده از مدل‌های یادگیری عمیق استفاده کرد. شاید این روندها آنقدر پیچیده باشند که برای انسان یا سایر فرآیندهای رایانه‌ای معمولی قابل درک نباشد.

پژوهش‌های اخیر نشان می‌دهد که پیش‌بینی بازار سهام را می‌توان با استفاده از یادگیری ماشین افزایش داد. تکنیک‌هایی مانند ماشین بردار پشتیبانی (SVM)، جنگل تصادفی (RF). برخی از تکنیک‌های مبتنی بر شبکه‌های عصبی مانند شبکه عصبی مصنوعی (ANN)، شبکه عصبی کانولوشنال (CNN)، شبکه عصبی مکرر (RNN) و شبکه های عصبی عمیق مانند حافظه کوتاه مدت بلند (LSTM) نیز نتایج امیدوار کننده ای را نشان داده‌اند.

ANN قادر به یافتن ویژگی‌های پنهان از طریق یک فرآیند خودآموزی است. این‌ها تقریب خوبی هستند و قادرند رابطه ورودی و خروجی یک مجموعه داده پیچیده بسیار بزرگ را بیابند. ANN از نظر مشخصات مدل در مقایسه با مدل‌های پارامتری قوی هستند که باعث می‌شود اغلب در پیش‌بینی قیمت سهام و مشتقات مالی استفاده شود. با این حال ، یکی از اشکالات ANN این است که وقتی مدل شبکه عصبی با مشاهدات موجود بیش از حد مجهز شود، کارایی پیش‌بینی نمونه‌های ناشناخته به سرعت کاهش می‌یابد. به عبارت دیگر، اطلاعات سهام پر سر و صدا ممکن است ANN را به یک مدل پیچیده هدایت کند که ممکن است منجر به مشکل پردازش بیش از حد شود.

یکی دیگر از روش‌های غالب در پیش‌بینی جهت بازار سهام، رویکردهای مبتنی بر SVM است. از آنجا که SVM از اصل به حداقل رساندن ریسک ساختاری استفاده می‌کند، اغلب به نتایج بهتری دست می‌یابد. تحقیقات نشان می‌دهد که SVM در پیش‌بینی جهت آینده بازار سهام از ANN بهتر عمل می‌کند.

یک اشکال عمده SVM برای پیش‌بینی این است که متغیرهای ورودی ابعاد بالا دارند، از صدها تا هزاران. ذخیره متغیرها به حافظه و زمان محاسبه زیادی نیاز دارد و بازار سهام شامل چندین صدها سهام است که منجر به ابعاد بالای متغیرها می‌شود. بنابراین، انجام کاهش ابعاد برای بدست آوردن نمایشی کارآمد و متمایز قبل از طبقه بندی اهمیت بسزایی دارد.

از دیگر روش‌های پیش‌بینی سهام، شبکه عصبی مکرر (RNN) است که یک شبکه عصبی مصنوعی مناسب برای حل مسائل سری زمانی است. ارتباطات بین نورون‌های آن یک حلقه جهت‌دار را تشکیل می‌دهد و آن را قادر می‌سازد تا مانند رفتارهای زمان، پویا عمل کند.

RNN در حال حاضر در پردازش زبان طبیعی، پردازش داده‌های صوتی و ... استفاده می‌شود و نتایج خوبی را به همراه دارد. حافظه RNN اصلی به دلیل تعداد لایه‌های پیچیده پس از بازگشت‌های متعدد، تأثیر آن را کاهش می‌دهد.

بنابراین، مفهوم شبکه عصبی حافظه کوتاه مدت (LSTM) معرفی می‌شود. این یک مدل RNN خاص و مهم است که می‌تواند مقادیر بلند مدت یا کوتاه مدت را حفظ کند و به صورت انعطاف پذیر به شبکه عصبی اجازه دهد تا اطلاعات لازم را حفظ کند. شبکه‌های عصبی LSTM برای ساخت مدل‌های پیش بینی قیمت سهام مناسب این پژوهش است.

۲-۱. تعریف مساله

هدف این پروژه پیش‌بینی قیمت سهام FAANG (Facebook, Apple, Amazon, Netflix, Google) با استفاده از یادگیری ماشین است. در این کار از تکنیک LSTM برای پیش بینی قیمت سهام استفاده شده است. داده‌های مالی که برای پیش‌بینی قیمت سهام استفاده می‌شوند شامل Open, High, Low and Close هستند که برای ایجاد متغیرهای جدید که به عنوان ورودی مدل استفاده می‌شوند، استفاده می‌شود. مدل‌ها با استفاده از شاخص استراتژیک استاندارد RMSE ارزیابی می‌شوند که مقدار پایین این شاخص نشان می‌دهد که مدل‌ها در پیش بینی قیمت سهام کارآمد هستند. سپس، یک داشبورد برای دیدن نتایج با استفاده از dash ایجاد می‌کنیم.

۳-۱. انگیزش ناشی از پایان نامه

هوش مصنوعی (AI) برای ایفای نقش اساسی در برنامه‌های روزمره زندگی ما چه در برنامه‌های محیط خانه مانند دستیار صوتی الکسا و چه در برنامه‌های مالی مانند تجارت، پیشرفتی است به سوی عصر جدیدی از فناوری. این پروژه شامل کاربرد هوش مصنوعی بر روی داده‌های مالی است که به عنوان معامله الگوریتمیک شناخته می‌شود.

سیستم‌های تجاری خودکار شامل استفاده از سیستم‌های هوش مصنوعی پیچیده برای تصمیم گیری تجاری بسیار سریع مانند خرید، نگهداری یا فروش است. این شامل معاملات با فرکانس بالا یا HFT است تا میلیون ها تجارت در روز انجام شود.

یادگیری ماشین زیر مجموعه ای از هوش مصنوعی است و به طور کلی راه حل‌هایی را ارائه می‌دهد که بدون تجربه برنامه ریزی صریح از تجربه یاد می‌گیرند. به عبارت ساده ، فقط مدل‌های یادگیری ماشین انتخاب شده و با داده‌ها تغذیه می‌شوند ، سپس مدل به طور خودکار پارامترهای خود را تنظیم کرده و نتیجه آن را بهبود می‌بخشد.

۴-۱. ساختار پایان نامه

در ادامه به بررسی مفاهیمی که در پروژه از ان استفاده شده است می‌پردازیم و سپس شروع به برنامه نویسی پروژه قدم به قدم می‌نمائیم و در انتها داشبوردی را برای نمایش نتایج به صورت وب سایت می‌سازیم.

فصل دوم: مروری بر مفاهیم

۱-۲. سخت افزار و نرم افزار :

مشخصات سخت افزار

سیستم‌های تجاری خودکار ممکن است بسیار پیچیده به نظر برسند اما به سخت افزار بسیار کمی نیاز دارند، برای این پژوهش فقط به یک کامپیوتر خوب با ویرایشگر خوب نیاز هست و نیاز چندانی به سیستمی با سخت افزار فوق العاده نیست!

مشخصات نرم افزار

پیاده سازی این پروژه با زبان python 3.9.0 انجام شده است و برای برنامه‌نویس به یک python ide قوی و خوب احتیاج هست. ما در اینجا از Visual Studio Code استفاده کرده‌ایم و برای راحتی بیشتر از jupyter notebook نیز استفاده شده است.

۲-۲. FAANG:

- **FB Inc.:**

فیس‌بوک ، شرکت فناوری چند ملیتی آمریکایی مستقر در Menlo Park، کالیفرنیا است. در سال ۲۰۰۴ با نام TheFacebook توسط مارک زاکربرگ، ادواردو ساورین، اندرو مک کولوم، داستین مسکوویتز و کریس هیوز که هم اتاقی و دانشجو بودن در کالج هاروارد تأسیس شد.

- **AAPL Inc.:**

اپل یک شرکت فناوری چند ملیتی آمریکایی است که در زمینه لوازم الکترونیکی مصرفی، نرم افزار رایانه و خدمات آنلاین تخصص دارد. اپل از نظر درآمد بزرگترین شرکت فناوری جهان و از ژانویه ۲۰۲۱ با ارزش‌ترین شرکت جهان است.

- **AMZN Inc.:**

امازون یک شرکت فناوری چند ملیتی آمریکایی است که بر تجارت الکترونیکی، رایانش ابری، پخش دیجیتال و هوش مصنوعی تمرکز دارد. این شرکت به همراه گوگل، اپل، مایکروسافت و فیس بوک یکی از پنج شرکت بزرگ در صنعت فناوری اطلاعات ایالات متحده است.

- **NFLX Inc.:**

نتفلیکس یک پلتفرم و شرکت تولید محتوای فوق العاده آمریکایی است که دفتر مرکزی آن در Los Gatos، California است. Netflix در سال ۱۹۹۷ توسط Reed Hastings و Marc Randolph در Scotts Valley کالیفرنیا تأسیس شد.

- **GOOG Inc.:**

گوگل یک شرکت فناوری چند ملیتی آمریکایی است که در خدمات و محصولات مرتبط با اینترنت تخصص دارد که شامل فناوری‌های تبلیغات آنلاین، موتورهای جستجو، محاسبات ابری، نرم افزار و سخت افزار است.

۳-۲. کتابخانه‌ها

برای نصب کتابخانه‌ها در پایتون از pip استفاده می‌شود. PIP در واقع یک ابزار خط فرمان است که بسته‌های PyPI را با یک دستور ساده نصب، حذف و یا نصب مجدد می‌کند. بعد از مطمئن شدن از نصب pip بر روی سیستم به راحتی می‌توان کتابخانه‌ها را با وارد کردن دستور زیر در ترمینال نصب کرد.

```
pip install نام کتابخانه مورد نظر
```

- **Pandas:**

یک ابزار تجزیه و تحلیل داده‌ها است. اوپن سورس، سریع، قدرتمند، انعطاف پذیر و آسان برای استفاده است که بر روی زبان برنامه نویسی پایتون ساخته شده است و در اینجا برای خواندن و پردازش داده‌ها استفاده می‌کنیم.

- **Pandas-datareader:**

دسترسی به اطلاعات از راه دور برای pandasها. برای نسخه های مختلف pandasها کار می‌کند. در اینجا برای خواندن داده‌ها از سایت stooq استفاده می‌شود.

- **Datetime:**

کلاس‌هایی را برای دستکاری تاریخ و زمان ارائه می‌دهد.

- **Math:**

این ماژول دسترسی به توابع ریاضی تعریف شده توسط استاندارد C را فراهم می‌کند.

- **Numpy:**

پکیج اساسی محاسبات علمی با پایتون است. در اینجا نیز برای محاسبات ریاضی استفاده می‌شود.

- **Matplotlib:**

یک کتابخانه جامع برای ایجاد تجسمات استاتیک ، متحرک و تعاملی در پایتون است. برای رسم نمودارها و چارت‌ها استفاده می‌شود.

- **Seaborn:**

یک کتابخانه تجسم داده پایتون است که بر اساس matplotlib ساخته شده است. این یک رابط سطح بالا برای ترسیم گرافیک‌های آماری جذاب و آموزنده ارائه می‌دهد. در اینجا برای رسم نمودار heatmap استفاده شده است.

- **Yfinance:**

پکیجی برای ارائه روشی قابل اعتماد، موضوعی و Pythonic برای بارگیری داده‌های بازار تاریخی از Yahoo! دارایی، مالیه، سرمایه گذاری. در اینجا برای دریافت دیتاست‌های داده‌های سهام شرکت‌ها از yahoo استفاده می‌شود.

- **Yahoo_fin:**

قیمت‌های سهام (روزانه/ هفتگی/ ماهانه)، قیمت‌های لحظه‌ای، داده‌های اساسی، جریان‌های نقدی، اطلاعات تحلیلگر، قیمت‌های فعلی ارزهای رمزنگاری شده ، سابقه سود و موارد دیگر را با می‌توان با yahoo_fin به‌راحتی دریافت کرد.

- **Sklearn.preprocessing:**

این پکیج چندین توابع مفید و کلاس‌های ترانسفورماتور را برای تغییر بردارهای ویژگی خام به نمایشی ارائه می‌دهد که برای برآورد کننده‌های پایین دست مناسب‌تر است.

- **keras.layers & keras.models:**

Keras یک API است که برای انسان طراحی شده است، نه ماشین. Keras بهترین شیوه‌ها را برای کاهش بار شناختی دنبال می کند: API های سازگار و ساده ارائه می‌دهد، تعداد اقدامات کاربر مورد نیاز برای موارد استفاده معمول را به حداقل می‌رساند و پیام‌های خطای واضح و قابل اجرا را ارائه می‌دهد. باید خاطر نشان کرد برای استفاده از Keras باید قبل از ان کتابخانه TensorFlow را نصب کنیم.

- **Dash:**

یک فریمورک پایتون اوپن سورس است که برای ساخت برنامه‌های کاربردی وب تحلیلی استفاده می‌شود. این یک کتابخانه قدرتمند است که توسعه برنامه‌های کاربردی مبتنی بر داده را ساده می‌کند. ساخته شده براساس Plotly.js، React و Flask است .عناصر UI مدرن مانند dropdowns، sliders و نمودارها را مستقیماً به کد

پایتون پیوند می‌دهد. برنامه های Dash شامل یک سرور Flask است که با اجزای React با استفاده از بسته‌های JSON بر روی درخواست های HTTP ارتباط برقرار می‌کند. برنامه های کاربردی Dash صرفاً با پایتون نوشته می‌شوند ، بنابراین نیازی به HTML یا JavaScript نیست.

- **Dash-bootstrap-components:**

اجزای مضمون Bootstrap برای استفاده در Plotly Dash. یک کتابخانه از اجزای بوت استرپ برای استفاده با Plotly Dash است که ساخت برنامه‌های dash با طراحی منظم با طرح بندی پیچیده و پاسخگو را آسان تر می‌کند.

- **Dash-html-components:**

اجزا html برای dash.

- **Plotly:**

یک کتابخانه تجسم اوپن سورس و منبع تعاملی برای پایتون است. برای رسم چارت‌ها در dash مورد استفاده قرار می‌گیرد.

۲-۴. بررسی مفاهیم:

۲-۴-۱. Neural networks:

شبکه‌های عصبی زیر مجموعه‌ای از یادگیری ماشین هستند و در قلب الگوریتم‌های یادگیری عمیق قرار دارند. نام و ساختار آن‌ها از مغز انسان الهام گرفته شده است و از راهی که نورون‌های بیولوژیکی به یکدیگر نشان می‌دهند تقلید می‌کند. یک شبکه عصبی سعی می‌کند تابعی را بیاموزد که ویژگی‌های ورودی را با پیش‌بینی‌های خروجی ترسیم می‌کند. این شامل شبکه‌ای از نورون‌ها است که هر یک از آن‌ها مجموع وزنی ورودی‌ها را نشان می‌دهد. خروجی‌های عصبی در توابع فعال سازی قرار می‌گیرند. شبکه شامل لایه‌هایی از نورون‌ها است که روی هم چیده شده‌اند و نورون‌ها بین لایه‌های جداگانه به طور کامل به هم متصل شده‌اند.

۲-۴-۲. Long Short-Term Memory (LSTM):

حافظه کوتاه مدت بلند (LSTM) یک معماری شبکه عصبی تکراری مصنوعی (RNN) است که در زمینه یادگیری عمیق استفاده می‌شود. برخلاف شبکه‌های عصبی استاندارد پیشرو ، LSTM دارای اتصالات بازخورد است. می‌تواند نه تنها نقاط داده (مانند تصاویر) ، بلکه کل توالی داده (مانند گفتار یا فیلم) را پردازش کند. به عنوان مثال، LSTM برای وظایفی مانند تقسیم بندی نشده(unsegmented)، تشخیص دست خط ، تشخیص گفتار و تشخیص ناهنجاری در ترافیک شبکه یا IDS (سیستم های تشخیص نفوذ) قابل اجرا است.

۲-۴-۳. داشبورد:

Dash یک چارچوب منبع باز برای ایجاد رابط‌های تجسم داده است. این کتابخانه در سال ۲۰۱۷ به عنوان کتابخانه پایتون منتشر شد و شامل پیاده سازی‌هایی برای R و جولیا شد. Dash به دانشمندان داده کمک می‌کند تا بدون نیاز به دانش پیشرفته توسعه وب، برنامه‌های کاربردی وب تحلیلی بسازند.

سه فناوری هسته اصلی Dash را تشکیل می‌دهند:

- Flask عملکرد وب سرور را تأمین می‌کند.
- React.js رابط کاربری صفحه وب را ارائه می‌دهد.
- Plotly.js نمودارهای مورد استفاده در برنامه را تولید می‌کند.

Plotly ، یک شرکت مستقر در کانادا، Dash را ایجاد کرده و از توسعه آن پشتیبانی می‌کند. Plotly (شرکت) Dash منبع باز و آن را تحت مجوز MIT منتشر کرد ، بنابراین می‌توان بدون هیچ هزینه ای از Dash استفاده کرد. داشبوردی که در اینجا می‌سازیم روی سرور Heroku به طور رایگان اجرا می‌شود.

۴-۴-۲. Heroku:

Heroku یک پلتفرم ابری مبتنی بر سرویس (PaaS) است. توسعه دهندگان از Heroku برای استقرار، مدیریت و مقیاس گذاری برنامه‌های مدرن استفاده می‌کنند. Heroku پلتفرمی زیبا، انعطاف پذیر و آسان برای استفاده است. Heroku به طور کامل مدیریت می‌شود و به توسعه دهندگان این آزادی را می‌دهد که بر محصول اصلی خود بدون توجه به حفظ سرورها، سخت افزار یا زیرساختها تمرکز کنند. تجربه Heroku خدمات ، ابزارها، گردش کار و پشتیبانی چند گانه را ارائه می‌دهد.

فصل سوم: برنامه نویسی

۱-۳. شروع برنامه نویسی:

همانطور که در ابتدا گفته شد می‌خواهیم سهام شرکت‌های فیسبوک، اپل، امازون، نتفلیکس و گوگل را در یک سال گذشته مورد بررسی قرار دهیم. این برنامه از یک شبکه عصبی مکرر مصنوعی به نام حافظه کوتاه مدت (LSTM) برای پیش بینی قیمت نهایی سهام استفاده می‌کند. در اینجا برای تجسم تاریخچه داده‌ها از نمودار استفاده می‌کنیم و به بررسی چند سوال می‌پردازیم.

۱) تغییر قیمت سهام در طول زمان چگونه بود؟
۲) بازده روزانه سهام به طور متوسط چقدر بود؟
۳) قیمت لحظه‌ای سهام چقدر است؟
۴) چگونه می توانیم رفتار آینده سهام را پیش‌بینی کنیم؟

۱-۱-۳. افزودن کتابخانه‌های مورد نیاز به برنامه:

کتابخانه‌هایی که در این برنامه استفاده می‌شوند و مورد نیاز است را به برنامه اضافه می‌کنیم.

```
# Import the Libraries
import pandas as pd
import numpy as np

# To draw a diagram
import matplotlib.pyplot as plt
plt.style.use("fivethirtyeight")
%matplotlib inline

# To draw a hit map
import seaborn as sns
sns.set_style('whitegrid')

# To get information from Yahoo
import yfinance as yf
from yahoo_fin import stock_info as si #To get an instant price


# For time stamps
from datetime import datetime

# For the LSTM
import math
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, LSTM
```

شکل ۱-۳. اضافه کردن کتابخانه‌ها

۲-۱-۳. دریافت دیتاست‌ها از yahoo finance:

ابتدا باید مدت زمان انتخابی دریافت داده‌ها را مشخص کنیم. با استفاده از کتابخانه datetime زمان اغاز و پایان را مشخص می‌کنیم. سپس توسط کتابخانه yfinance دیتاست‌ها را از yahoo! دریافت و ان‌ها را به فایل csv تبدیل می‌کنیم.

```
# Set up End and Start times for data grab
end = datetime.now()
start = datetime(end.year - 1, end.month, end.day)

# Download stock data then export as CSV
df_fb = yf.download("FB", start, end)
df_fb.to_csv('facebook.csv')

df_aapl = yf.download("AAPL", start, end)
df_aapl.to_csv('apple.csv')

df_amzn = yf.download("AMZN", start, end)
df_amzn.to_csv('amazon.csv')

df_nflx = yf.download("NFLX", start, end)
df_nflx.to_csv('netflix.csv')

df_goog = yf.download("GOOG", start, end)
df_goog.to_csv('google.csv')
```

```
[**********100%**********]  1 of 1 compl
eted
[**********100%**********]  1 of 1 compl
eted
[**********100%**********]  1 of 1 compl
eted
[**********100%**********]  1 of 1 compl
eted
[**********100%**********]  1 of 1 compl
eted
```

شکل ۲-۳. دانلود دیتاست‌ها

پس از دریافت دیتاست‌ها می‌توانیم ان‌ها را با استفاده از کتابخانه pandas بخوانیم.

```
# read google dataset
df = pd.read_csv('google.csv')
df.head()
```

Out[3]:

	Date	Open	High	Low	Close	Adj Close	Vol
0	2020-08-14	1515.660034	1521.900024	1502.880005	1507.729980	1507.729980	135
1	2020-08-17	1514.670044	1525.609985	1507.969971	1517.979980	1517.979980	137
2	2020-08-18	1526.180054	1562.469971	1523.709961	1558.599976	1558.599976	202
3	2020-08-19	1553.310059	1573.680054	1543.949951	1547.530029	1547.530029	166
4	2020-08-20	1543.449951	1585.869995	1538.199951	1581.750000	1581.750000	170

شکل ۳-۳. خواندن دیتاست

دیتاست‌های سهام شامل ستون‌های متفاوتی است که هرکدام بیانگر یک مقداری هستند.

- ستون Date، تاریخ را نشان می‌دهد.
- ستون‌های High و Low قیمت بالا و پائین سهام در یک تاریخ مشخص را نشان می‌دهند.
- ستون‌های Open و Close قیمت‌های شروع سهام و بسته شدن سهام در یک تاریخ مشخص را نشان می‌دهند.
- ستون Volume حجم معملات در یک تاریخ مشخص را نشان می‌دهد.

۲-۳. رسم نمودار برای تجسم تاریخچه داده‌ها:

در تابع زیر یک نمودار بر اساس تاریخچه بسته شدن سهام (Close) را ایجاد می‌کنیم و سپس با استفاده از کتابخانه matplotlib نمودار را رسم می‌کنیم.

```
# Plot stock company['Close']
def figure_close(stockname , name):
    df = pd.read_csv(stockname)
    df["Date"] = pd.to_datetime(df.Date,format="%Y-%m-%d")
    df.index = df['Date']

    fig = plt.figure(figsize=(16,8))
    ax = fig.add_subplot()
    ax.set_title(name)
    ax.set_xlabel('Date', fontsize=16)
    ax.set_ylabel('Close Price USD ($)', fontsize=16)
    plt.plot(df["Close"],label='Close Price history')
```

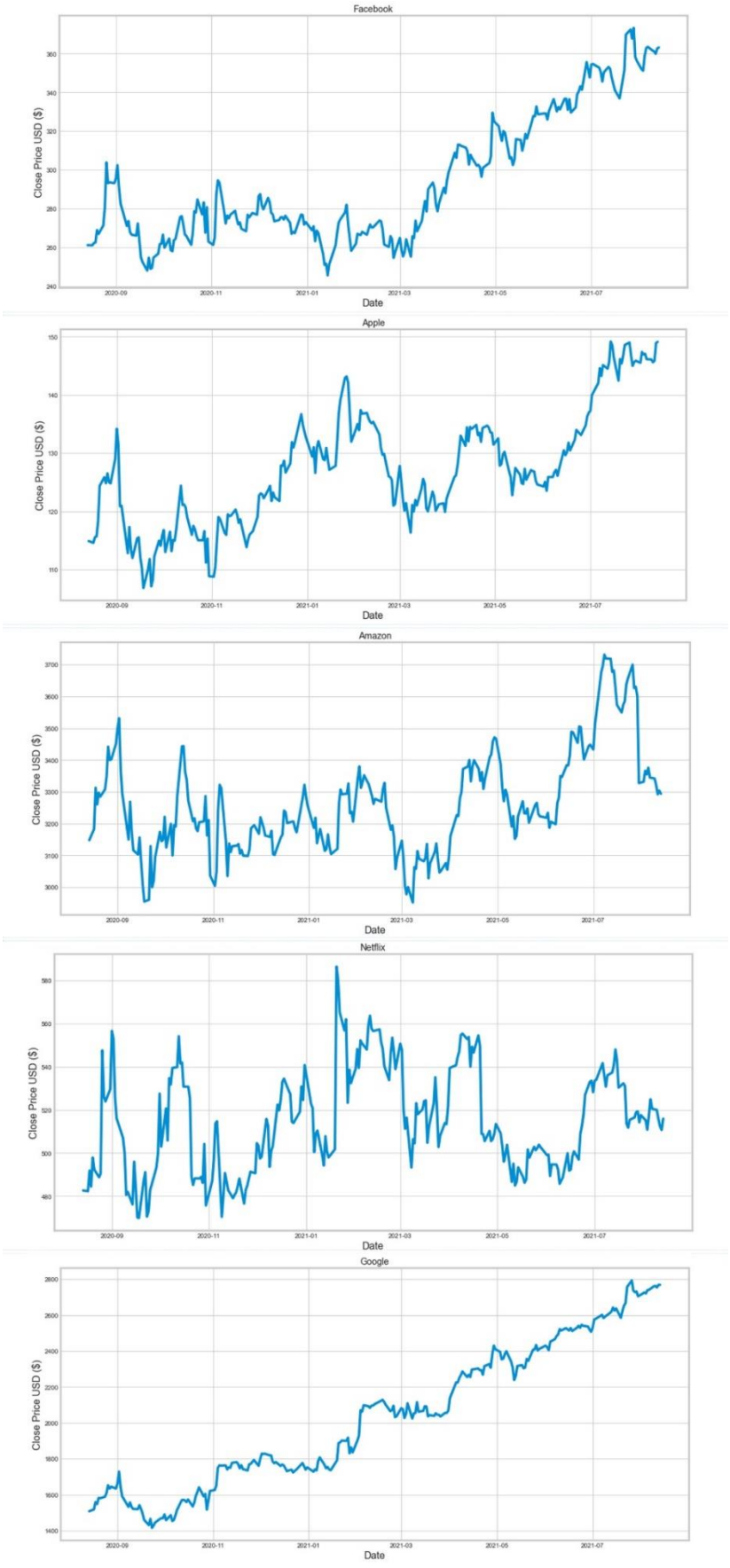
شکل ۴-۳. تابع رسم نمودار close

سپس با فراخوانی تابع figure_close می‌توان نمودارهای سهام هر شرکت را رسم کرد، به عنوان مثال:

```
figure_close('facebook.csv' , 'Facebook')
```

شکل ۵-۳. فراخوانی تابع figure_close

۱-۲-۳. نمودارهای تاریخچه قیمت پایانی سهام‌ها:



شکل ۶-۳. نمودارهای سهام بر اساس close

۲-۲-۳. نمودارهای حجم کل سهام‌های مورد معامله:

در تابع زیر یک نمودار بر اساس تاریخچه حجم سهام‌های معامله شده (Volume) را ایجاد می‌کنیم و سپس با استفاده از کتابخانه matplotlib نمودار را رسم می‌کنیم.

```
# Plot stock compony['Volume']
def figure_Sales(stockname , name):
    df = pd.read_csv(stockname)
    df["Date"] = pd.to_datetime(df.Date,format="%Y-%m-%d")
    df.index = df['Date']

    fig = plt.figure(figsize=(16,8))
    ax = fig.add_subplot()
    ax.set_title(name)
    ax.set_xlabel('Date', fontsize=16)
    ax.set_ylabel('Close Price USD ($)', fontsize=16)
    plt.plot(df["Volume"],label='Close Price history')
```

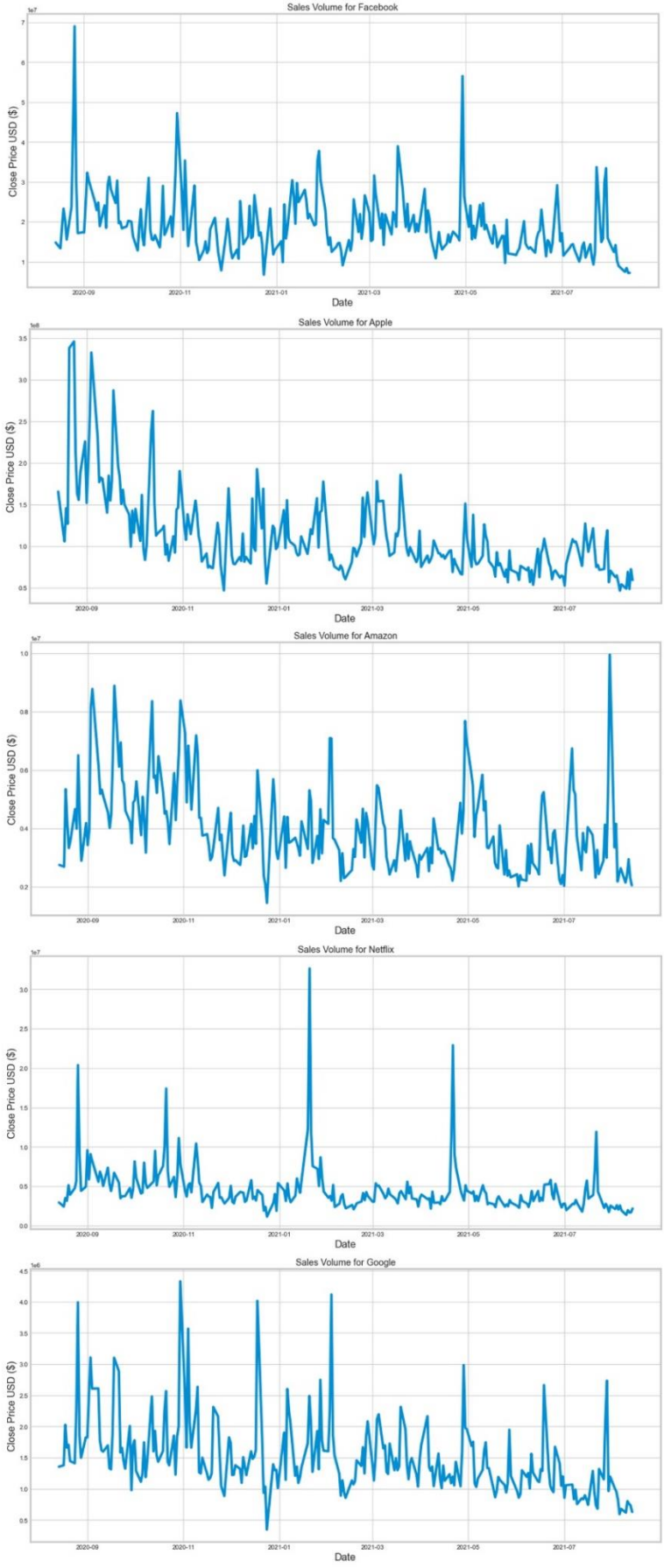
شکل ۷-۳. تابع رسم نمودار Sales

سپس با فراخوانی تابع figure_sales می‌توان نمودارهای سهام هر شرکت را رسم کرد، به عنوان مثال:

```
figure_Sales('facebook.csv' , 'Sales Volume for Facebook')
```

شکل ۸-۳. فراخوانی تابع figure_sales

نمودارهای تاریخچه حجم کل سهام‌های معامله شده:



شکل ۹–۳. نمودارهای sales

۳-۲-۳. محاسبه بازده روزانه سهام‌ها:

اکنون که تجزیه و تحلیل پایه را انجام داده‌ایم، کمی عمیق‌تر می‌شویم و تغییرات روزانه سهام‌ها را بررسی کنیم.

حالا می‌خواهیم بازدهی همه سهام موجود در لیست خود را تجزیه و تحلیل کنیم. یک دیتاست که شامل فقط مقدار نهایی سهام‌های شرکت‌ها باشد یعنی فقط شامل ستون close باشد را از yahoo! دریافت می‌کنیم و ان را به فایل csv تبدیل می‌کنیم.

```
# The tech stocks we'll use for this analysis
tech_list = ['FB', 'AAPL', 'AMZN', 'NFLX', 'GOOG']

# Download stock data then export as CSV
df_close = yf.download(tech_list, start, end)['Adj Close']
df_close.to_csv('closing.csv')
```

```
[**********100%**********]  5 of 5 compl
eted
```

شکل ۱۰-۳. دریافت دیتاست با ستون‌های close

دیتاست دریافت شده را مشاهده می‌کنیم که فقط شامل قیمت بسته شدن سهام‌ها است.

```
print(df_close.head())
```

```
                 AAPL         AMZN          FB         GOOG
NFLX
Date
2020-08-14  114.173164  3148.020020  261.239990  1507.729980  48
2.679993
2020-08-17  113.875069  3182.409912  261.160004  1517.979980  48
2.350006
2020-08-18  114.823975  3312.489990  262.339996  1558.599976  49
1.869995
2020-08-19  114.968048  3260.479980  262.589996  1547.530029  48
4.529999
2020-08-20  117.519142  3297.370117  269.010010  1581.750000  49
7.899994
```

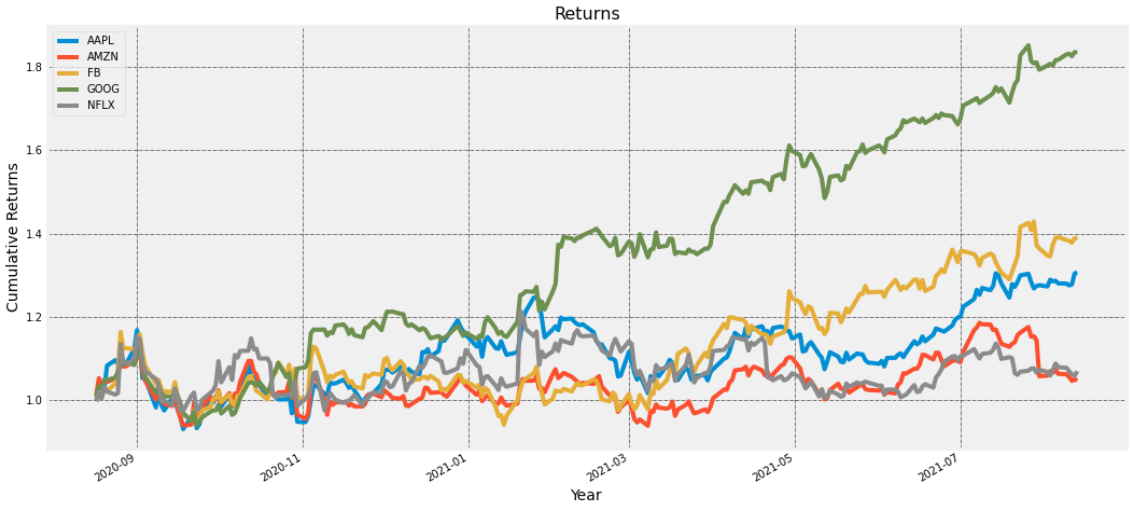
شکل ۱۱-۳. خواندن دیتاست close

از pct_change برای یافتن درصد تغییر در هر روز استفاده می‌کنیم. سپس، نمودار بازدهی روزانه سهام‌ها را با استفاده از matplotlib رسم می‌کنیم.

```
# Plot all the close prices
((df_close.pct_change()+1).cumprod()).plot(figsize=(16,8))

# Show the legend
plt.legend()

# Define the label for the title of the figure
plt.title("Returns", fontsize=16)
plt.ylabel('Cumulative Returns', fontsize=14)
plt.xlabel('Year', fontsize=14)

# Plot the grid lines
plt.grid(which="major", color='k', linestyle='-.', linewidth=0.5)
plt.show()
```

شکل ۱۲-۳. برای به دست اوردن daily return

نمودار بازدهی روزانه سهام‌ها:



شکل ۱۳-۳. نمودار Daily returns

همچنین می‌توانیم یک نمودار همبستگی ایجاد کنیم تا مقادیر عددی واقعی را برای همبستگی بین مقادیر بازده روزانه سهام مشاهده کنیم. از sebron برای رسم نودار heatmap استفاده می‌کنیم.

```
sns.heatmap(df_close.corr(), annot=True, cmap='summer')
```

شکل ۱۴-۳. رسم نمودار heatmap

نمودار Heatmap:

شکل ۱۵-۳. نمودار heatmap

۳-۳. قیمت لحظه‌ایی سهام‌ها:

می‌توان با استفاده از کتابخانه‌ی Yahoo_fin از قیمت‌های لحظه‌ایی سهام ها مطلع شویم.

```
# import stock_info module from yahoo_fin
def get_Price(stock , name):
    price = si.get_live_price(stock)
    print(name,"live stock price: " , price)


print(datetime.now())
print()
get_Price('fb' , 'Facebook Inc.')
get_Price('aapl' , 'Apple Inc.')
get_Price('amzn' , 'Amazon Inc.')
get_Price('nflx' , 'Netflix Inc.')
get_Price('goog' , 'Google Inc.')
```

```
2021-08-15 20:17:01.967604

Facebook Inc. live stock price:  363.17999267578125
Apple Inc. live stock price:  149.10000610351562
Amazon Inc. live stock price:  3293.969970703125
Netflix Inc. live stock price:  515.9199829101562
Google Inc. live stock price:  2768.1201171875
```

شکل ۱۶-۳. قیمت لحظه‌ای سهام‌ها

۳-۴. محاسبه پیش بینی قیمت سهام با LSTM:

تا به اینجا ما داده‌ها را بر اساس تاریخچه بسته شدن سهام، حجم کل سهام‌های معامله شده و بازدهی روزانه سهام‌ها مورد بررسی قرار دادیم. اینک به سراغ پیش بینی قیمت سهام‌ها با استفاده از تکنیک LSTM می‌رویم.

ابتدا یک dataframe جدید فقط با قیمت پایانی ایجاد می‌کنیم و آن را به یک آرایه تبدیل می‌کنیم. سپس یک متغیر برای ذخیره طول مجموعه داده‌های آموزشی ایجاد می‌کنیم. در این‌جا حدود ۸۰ درصد از داده‌ها را به عنوان مجموعه داده‌های آموزشی در نظر گرفته‌ایم.

```
#Create a new dataframe with only the 'Close' column
data = df.filter(['Close'])

#Converting the dataframe to a numpy array
dataset = data.values

#Get /Compute the number of rows to train the model on
training_data_len = math.ceil( len(dataset) *.8)
```

شکل ۱۷-۳. مشخص کردن محدوده داده‌های اموزشی

اکنون مجموعه داده‌ها را بین ۰ تا ۱ توسط کتابخانه sklearn.preprocessing مقیاس بندی می‌کنیم، این کار را به این دلیل انجام می‌دهیم زیرا خوب است که داده‌ها را قبل از ارائه به شبکه عصبی مقیاس بندی کرد.

```
#Scale the all of the data to be values between 0 and 1
scaler = MinMaxScaler(feature_range=(0, 1))
scaled_data = scaler.fit_transform(dataset)
scaled_data
```

```
Out[6]:

array([[0.0671564 ],
       [0.07459644],
       [0.10408079],
       [0.09604558],
       [0.12088442],
       [0.11991906],
       [0.12556617],
       [0.14009786],
       [0.17215177],
       [0.15905   ],
       [0.16636671],
       [0.15894119],
       [0.17819814],
       [0.22724441],
       [0.1645012 ],
       [0.12762767],
       [0.08505608],
       [0.10289037],
```

شکل ۱۸-۳. مقیاس بندی داده‌ها بین ۰ و ۱

یک مجموعه داده آموزشی ایجاد می‌کنیم که شامل قیمت‌های پایانی ۶۰ روز گذشته است که می‌خواهیم از آن برای پیش بینی قیمت سهام روز ۶۱ ام استفاده کنیم.

مجموعه داده‌های آموزشی مقیاس بندی شده را ایجاد می‌کنیم و سپس، داده‌ها را به مجموعه داده‌های x_train و y_train تقسیم می‌کنیم.

```
#Create the scaled training data set
train_data = scaled_data[0:training_data_len  , : ]

#Split the data into x_train and y_train data sets
x_train=[]
y_train = []
for i in range(60,len(train_data)):
    x_train.append(train_data[i-60:i,0])
    y_train.append(train_data[i,0])
```

شکل ۱۹-۳. تقسیم بندی داده‌ها به x_train و y_train

اکنون مجموعه داده‌های x_train و y_train را به آرایه های numpy تبدیل می‌کنیم.

```
#Convert x_train and y_train to numpy arrays
x_train, y_train = np.array(x_train), np.array(y_train)
```

شکل ۲۰-۳. تبدیل داده‌ها به ارایه numpy

مدل LSTM روی مجموعه داده‌های سه بعدی کار می‌کند. در نتیجه داده‌ها را به صورت سه بعدی بر اساس [تعداد نمونه ، تعداد مراحل زمانی و تعداد ویژگی‌ها] تغییر می‌دهیم.

```
#Reshape the data into the shape accepted by the LSTM
x_train = np.reshape(x_train, (x_train.shape[0],x_train.shape[1],1))
```

شکل ۲۱-۳. تغییر ابعاد داده‌ها

حال شروع به ساخت مدل LSTM می‌کنیم که دارای دو لایه LSTM با ۵۰ نورون و دو لایه متراکم(Dense)، یکی با ۲۵ نورون و دیگری با ۱ نورون باشد.

```
#Build the LSTM network model
model = Sequential()
model.add(LSTM(units=50, return_sequences=True,input_shape=(x_train.shape[1],1)))
model.add(LSTM(units=50, return_sequences=False))
model.add(Dense(units=25))
model.add(Dense(units=1))
```

شکل ۲۲-۳. ساخت مدل LSTM

مدل را با استفاده از تابع از دست دادن خطای میانگین مربع (MSE) و بهینه ساز adam کامپایل می‌کنیم.

```
#Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')
```

شکل ۲۳-۳. کامپایل مدل

مدل را با استفاده از مجموعه داده‌های آموزشی آموزش می‌دهیم. باید خاطر نشان کرد که fit نام دیگری برای اموزش(train) است. Batch size عبارت است از تعداد کل نمونه‌های آموزشی موجود در یک دسته و epoch تعداد تکرارها هنگامی است که یک مجموعه داده کامل از طریق شبکه عصبی به جلو و عقب منتقل می‌شود.

```
#Train the model
model.fit(x_train, y_train, batch_size=1, epochs=1)
```

```
142/142 [==============================] - 19s 19ms/step - loss:
0.0178
```

شکل ۲۴-۳. اموزش مدل

حال یک مجموعه داده آزمایشی ایجاد می‌کنیم.

```
#Test data set
test_data = scaled_data[training_data_len - 60: , : ]
#Create the x_test and y_test data sets
x_test = []

y_test =  dataset[training_data_len : , : ]
for i in range(60,len(test_data)):
    x_test.append(test_data[i-60:i,0])
```

شکل ۲۵-۳. مجموعه داده اموزشی

سپس، مجموعه داده‌های آزمایشی "x_test" را به یک آرایه numpy تبدیل می‌کنیم تا بتوانیم از آن برای آزمایش مدل LSTM استفاده کنیم.

```
#Convert x_test to a numpy array
x_test = np.array(x_test)
```

شکل ۲۶-۳. تبدیل مجموعه داده‌های اموزشی x_test به ارایه numpy

داده‌های ازمایشی را نیز به صورت سه بعدی بر اساس [تعداد نمونه ، تعداد مراحل زمانی و تعداد ویژگی ها] تغییر می‌دهیم زیرا مدل LSTM روی مجموعه داده‌های سه بعدی کار می‌کند.

```
#Reshape the data into the shape accepted by the LSTM
x_test = np.reshape(x_test, (x_test.shape[0],x_test.shape[1],1))
```

شکل ۲۷-۳. سه بعدی کردن داده های اموزشی

اکنون مقادیر پیش‌بینی شده را با استفاده از داده‌های آزمایش از مدل بدست می‌اوریم.

```
#Getting the models predicted price values
predictions = model.predict(x_test)
predictions = scaler.inverse_transform(predictions)#Undo scaling
```

شکل ۲۸-۳. به دست اوردن مقادیر پیش بینی شده

خطای میانگین مربع مربع (RMSE) را بدست می‌اوریم، که معیار خوبی برای اندازه گیری دقیق مدل است. مقدار ۰ نشان می‌دهد که مدل‌های پیش بینی شده با مقادیر واقعی مجموعه داده‌های آزمون کاملاً مطابقت دارند. هرچه مقدار کمتر باشد، مدل بهتر عمل می‌کند. اما معمولاً بهتر است از معیارهای دیگر نیز استفاده کرد تا واقعاً از عملکرد خوب مدل مطلع شویم.

```
#Calculate/Get the value of RMSE
rmse=np.sqrt(np.mean(((predictions- y_test)**2)))
rmse
```

شکل ۲۹-۳. محاسبه RMSE

داده‌ها برای رسم نمودار ایجاد می‌کنیم.

```
#Plot/Create the data for the graph
train = data[:training_data_len]
valid = data[training_data_len:]
valid['Predictions'] = predictions
```

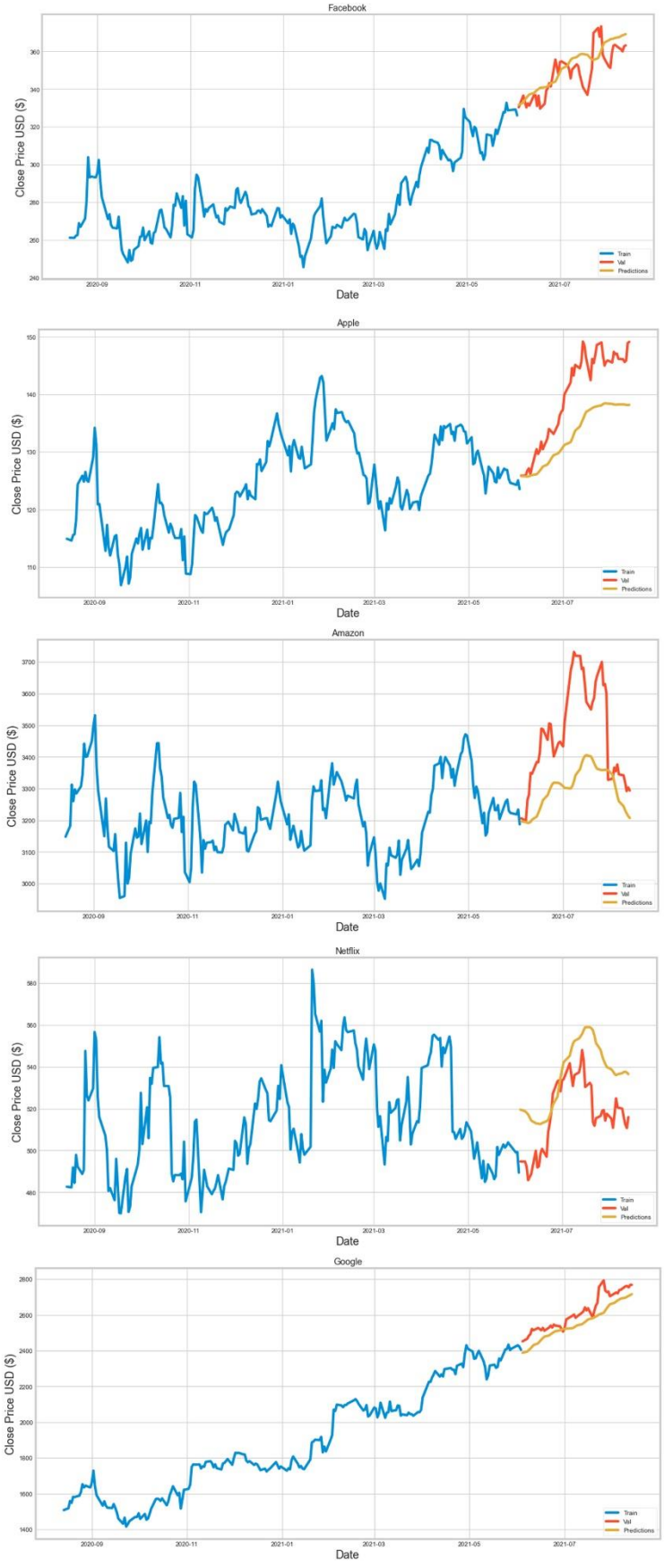
شکل ۳۰-۳. ایجاد داده برای رسم نمودار پیش بینی

نمودار پیش بینی را رسم می‌کنیم.

```
#Visualize the data
plt.figure(figsize=(16,8))
plt.title('Model')
plt.xlabel('Date', fontsize=18)
plt.ylabel('Close Price USD ($)', fontsize=18)
plt.plot(train['Close'])
plt.plot(valid[['Close', 'Predictions']])
plt.legend(['Train', 'Val', 'Predictions'], loc='lower right')
plt.show()
```

شکل ۳۱-۳. رسم نمودار پیش بینی قیمت سهام

نمودارهای پیش بینی قیمت سهام‌ها:



شکل ۳۲-۳. نمودارهای پیش بینی قیمت سهام

۳-۵. ایجاد داشبورد:

برای نشان دادن نتایج به دست امده از این پژوهش شروع به ساخت یک داشبورد برای نمایش نتایج به صورت یک وب‌سایت می‌نماییم. برای ساخت داشبورد از کتابخانه Dash استفاده می‌کنیم. فریم ورک Python Dash به عنوان یک کتابخانه منبع باز توسط Plotly توسعه یافته است و روی Flask، Plotly.js و React.js ساخته شده است. Dash اجازه می‌دهد تا برنامه‌های کاربردی وب تعاملی در پایتون ایجاد شود و مخصوصاً برای به اشتراک گذاری نتایج به دست آمده از داده‌ها مناسب است. برای استفاده بهتر از dash یک دانش نسبی از html و css مفید است.

۳-۵-۱. نحوه راه اندازی برنامه Dash:

پس از نصب dash بر روی سیستم خود dash بسته‌های ضروری dash_html_component و dash_core_component را به طور خودکار نصب می‌کند. در اینجا برای راحتی از افزونه‌های bootstarp استفاده کردیم که برای استفاده از ان‌ها نیاز به نصب ان‌ها روی سیستم است.

ابتدا کتابخانه‌های مورد نیاز برای کار با dash را به برنامه اضافه می‌کنیم.

```
# For Dashboard
import dash
from dash_bootstrap_components._components.CardImg import CardImg
from dash_bootstrap_components._components.Row import Row
import dash_core_components as dcc
import dash_html_components as html
from dash.dependencies import Output, Input
from dash_html_components.Br import Br
from dash_html_components.Img import Img
from numpy.core.numeric import outer

# for Plot
import plotly.express as px
import plotly.graph_objs as go

# use boostrap
import dash_bootstrap_components as dbc
```

شکل ۳۳-۳. افزودن کتابخانه‌های مورد استفاده برای dash

با استفاده از دستور زیر برنامه را ایجاد می‌کنیم و یک عنوان برای ان در نظر می‌گیریم.

```
app = dash.Dash(external_stylesheets=[dbc.themes.BOOTSTRAP])
app.title = "Stock Price Prediction"
```

شکل ۳۴-۳. راه اندازی اولیه برنامه

با استفاده از دستور app.layout شروع به ساخت داشبورد می‌نماییم. یک نمای کلی از ساختار داشبورد به صورت زیر است.

```
app.layout = dbc.Container([
    dbc.Row([
        dbc.Col([
            # show title dashboard
        ])
    ]),

    # creat two tabs
    dcc.Tabs(id="tabs", children=[
        # tab 1
        dcc.Tab(label='FAANG Stocks',children=[
            # Show stock price closing chart
            # show stock histogram
        ]),

        # tab 2
        dcc.Tab(label='Google stocks',children=[
            # Display Google stock closing chart
            # Display Google stock price forecast chart
        ])

    ])
], style = {"background-color": "#F7F7F7"}) # style bpdy
```

شکل ۳۵-۳. قسمت app.layout برنامه

برای راه اندازی سرور و نمایش وب‌سایت دستور زیر را اجرا می‌کنیم.

```
if __name__ == "__main__":
    app.run_server()
```

شکل ۳۶-۳. راه اندازی سرور

داشبورد را می‌توان برای راحتی کار به سطر و ستون‌های متفاوت توسط bootsrap تبدیل کرد. در اینجا در سطر اول برنامه عنوان سایت قرار دارد که برای نمایش ان در داشبورد از دستور زیر استفاده می‌کنیم.

```
# Row-01 ( Header )
dbc.Row([
    dbc.Col(
        html.H1("Stock Market Dashboard" ,className='text-center text-dark, my-4'),
                    width=12)
]),
```

شکل ۳۷-۳. عنوان سایت

سپس دو تب تعریف می‌کنیم که یکی برای بررسی سهام‌های FAANG است و دیگری به بررسی سهام گوگل و پیش بینی قیمت سهام ان می‌پردازد.

تب اول به چهار سطر تقسیم می‌شود که شامل :

- سطر اول شامل dropdown است.
- سطر دوم شامل نمودارهای بسته شدن قیمت سهام‌ها.
- سطر سوم عنوان بخش چهارم.
- سطر چهارم شامل نمودار هیستوگرام که مجموع قیمت پایانی سهام ها را نشان می‌دهد.

تب دوم به دو سطر تبدیل می‌شود که شامل:

- سطر اول نمودار بسته شدن قیمت سهام گوگل را نشان می‌دهد.
- سطر دوم نمودار پیش بینی قیمت سهام گوگل است.

در ابتدا داده‌های مورد نیاز را از سایت stooq توسط دستورات زیر خوانده و به فایل csv تبدیل می‌کنیم تا در ادامه پردازش‌های لازم روی ان انجام شود.

```
# Select time range
end = datetime.now()
start = datetime(end.year - 1, end.month, end.day)

# read stocks dataset from 'stooq'
df_stock = web.DataReader(['FB','AAPL','AMZN','NFLX','GOOGL'],
                          'stooq', start = start, end = end)
df_stock = df_stock.stack().reset_index()
df_stock.to_csv("mystocks.csv", index = False)
dfs = pd.read_csv("mystocks.csv")    # read dataset
```

شکل ۳۸-۳. دریافت داده‌ها از stooq

در تب اول داریم:

- سطر اول، dropdown:

ابتدا عنوان بخش را با استفاده از html مشخص می‌کنیم و سپس درون یک card ، dropdown را تعریف می‌کنیم و یک id برای ان در نظر می‌گیریم. مقادیر dropdown توسط یک حلقه for از دیتابیس خوانده می‌شود که شامل نام شرکت‌ها است و سپس توسط css می‌توان style مورد نظر خود را اعمال کنیم.

```
# Row-0 ( Drodown list )
dbc.Row([
    # Col-0
    dbc.Col([
        # title
        html.H3("Close Price USD ($) " ,className='text-center text-dark, my-4'),

        # dropdown
        dbc.Card([
            dcc.Dropdown(id = 'my-dpdn', multi = True , value = '',
            options = [{'label':x, 'value':x}
                       for x in sorted(dfs['Symbols'].unique())
            ], style = {"width": "95%" , "margin": "auto" })

        # style css code
        ], style = {"width": "85%", "height":"6rem",
                    "box-shadow": "0 4px 6px 0 rgba(0, 0, 0, 0.18)",
                    "margin": "auto" , "padding": "10px"})
    ])
]),
```

شکل ۳۹-۳. ایجاد Dropdown در داشبورد

- سطر دوم، نمودارهای تاریخچه بسته شدن قیمت سهام‌ها:

در این بخش گراف را ایجاد می‌کنیم و یک id برای ان تعریف می‌کنیم و در ادامه توسط css، style مورد نظر خود را اعمال می‌کنیم.

```
# Row-01 ( Close figuers )
dbc.Row([
    dbc.Col([
        # figure
        dcc.Graph(id = 'fig',
        figure = {
                "layout": {"title": "Close Prise"}
        # style
         }, className = "my-4",
            style = {"box-shadow": "0 4px 6px 0 rgba(0, 0, 0, 0.15)",
                     "width": "85%", "margin": "auto"}) # style code
    ])
]),
```

شکل ۴۰-۳. نمودارهای close سهام‌ها در داشبورد

در قسمت Callback در داشبورد می‌توان نتایج مورد نظر را اعمال کرد تا در داشبورد نمایش داده شوند.

Callback شامل دو بخش input و output است که:

- در input مقادیر اولیه که دریافت می‌شوند را مشخص می‌کنیم.
- در output خروجی را به همراه نوعی که قرار است نمایش داده شود، مشخص می‌کنیم.

در این بخش ورودی ما، مقداری است که از dropdown دریافت می‌شود و خروجی نمودار تاریخچه قیمت بسته شدن سهام‌ها است که توسط تابع close_graph مشخص شده است.

```
# Close Price figure
@app.callback(
    Output('fig', 'figure'),
    Input('my-dpdn', 'value')
)
def close_graph(stock_slctd):
    dff = dfs[dfs['Symbols'].isin(stock_slctd)]
    figln = px.line(dff, x = 'Date', y = 'Close', color = 'Symbols')
    return figln
```

شکل ۴۱-۳. قسمت callback برای نمودارهای close

- سطر سوم، عنوان بخش چهارم:

```
# Row-02
dbc.Row([
    dbc.Col([
        # titel ( histogram )
        html.H1("Histogram Chart",className='text-center text-dark, my-4')
    ])
]),
```

شکل ۴۲-۳. عنوان بخش چهارم در تب اول

- سطر چهارم، نمودار هیستوگرام سهام‌ها:

در این بخش ابتدا یک card را تعریف می‌کنیم و درون ان یک checklist با id مشخص ایجاد می‌کنیم که عناصر چک‌لیست اسم‌های شرکت‌ها هستند و با انتخاب هر کدام از ان‌ها ،نمودار هیستوگرام مربوطه نمایش داده می‌شود.

```
# Row-03
dbc.Row([
    # Show histogeram figure
    dbc.Col([
        dbc.Card([
            dbc.CardBody([
                # title
                html.H6("Select the Compony:", className = "font-weight-bolder"),
                dcc.Checklist(id = 'my-chek', value = '',
                # select compony name from database
                options=[{'label':x, 'value':x}
                        for x in sorted(dfs['Symbols'].unique())]
                        , labelClassName="mr-3"),
            # figure
            dcc.Graph(id='my-hist', figure={}),
            ])
        ], style = {"width": "85%", "margin": "auto" ,   # style css
                    "box-shadow": "0 4px 6px 0 rgba(0, 0, 0, 0.18)"}),
        html.Br()
    ])
])
```

شکل ۴۳-۳. نمودار هیستوگرام

سپس، در قسمت callback داشبورد، خروجی را مشخص می‌کنیم. به این صورت که ورودی این بخش از عنصرهای انتخاب شده در چک لیست هست و خروجی به صورت یک نمودار هیستوگرام است که در تابع مشخص شده است.

```
# Histogram figure
@app.callback(
    Output('my-hist', 'figure'),
    Input('my-chek', 'value')
)
def update_graph(stock_slctd):
    dff = dfs[dfs['Symbols'].isin(stock_slctd)]
    dff = dff[dff['Date']=='2021-08-06']  # Date Today
    fighist = px.histogram(dff, x = 'Symbols', y = 'Close', color = 'Symbols')
    return fighist
```

شکل ۴۴-۳. بخش callback نمودار هیستوگرام

در تب دوم داریم:

- سطر اول، نمودار تاریخچه گوگل بر اساس close:

ابتدای این بخش یک عنوان با استفاده از html نمایش داده می‌شود. بخش نمودار شامل دو بخش data و layout می‌باشد که:

- در بخش data از نتایج به دست امده در بخش LSTM استفاده می‌شود. در این نمودار مقادیر اموزش دیده close در نمودار نمایش داده می‌شود.
- در بخش layout ، عنوان نمودار و عناوین سطر و ستون نمودار مشخص می‌شود.

```
# Row-00 ( Close figure )
dbc.Row([
    dbc.Col([
        # title
        html.H2("Close figure", className = 'text-center blockquote, my-4'),

        # figure
        dcc.Graph(id = "Close",
        figure= {
            "data":[
                go.Scatter(
                    x=train.index,
                    y=valid["Close"],
                    mode = 'lines',
                    line = dict(shape = 'linear', color = 'rgb(10, 120, 24)', dash = 'dot'),
                    connectgaps = True
            )],
            "layout":go.Layout(
                title='scatter plot',
                xaxis={'title':'Date'},
                yaxis={'title':'Closing Rate'}
            )}

            # style
            , className = "my-4",
            style = {"box-shadow": "0 4px 6px 0 rgba(0, 0, 0, 0.15)",
                     "width": "85%", "margin": "auto"}
          )
    ])
]),
```

شکل ۴۵-۳. نمودار گوگل، بخش close

- سطر دوم، نمودار پیش بینی سهام گوگل:

در این بخش نیز یک عنوان با استفاده از html نمایش داده می‌شود. بخش نمودار شامل دو بخش data و layout می‌باشد که:

- در بخش data از نتایج به دست امده در بخش LSTM استفاده می‌شود. در این نمودار مقادیر پیش بینی شده نمایش داده می‌شود.
- در بخش layout ، عنوان نمودار و عناوین سطر و ستون نمودار مشخص می‌شود.

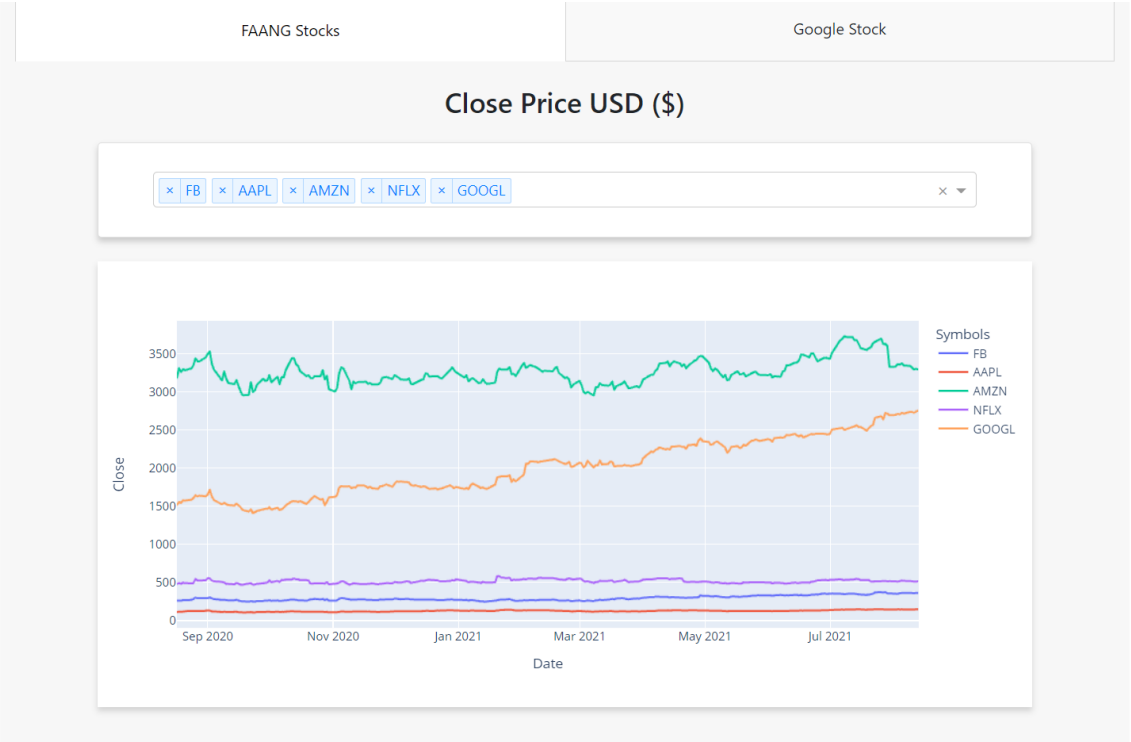
```
# Row-01 ( predict figure )
dbc.Row([
    dbc.Col([
        #  Header and Description
        html.H2("Predicted figure", className = 'text-center blockquote, my-4'),
        html.P("This program uses an artificial recurrent neural network called"
               " Long Short Term Memory (LSTM)",
               className="text-decoration-none text-center"),

        # figure
        dcc.Graph(id = "Predicted",
        figure={
            "data":[
                go.Scatter(
                    x=valid.index,
                    y=valid["Predictions"],
                    mode = 'lines',
                    line = dict(shape = 'linear', color = 'rgb(100, 10, 100)', dash = 'dot'),
                    connectgaps = True
                )],

            "layout":go.Layout(
                title='scatter plot',
                xaxis={'title':'Date'},
                yaxis={'title':'Closing Rate'}
            )}
            , className = "my-4",

            # style code css
            style = {"box-shadow": "0 4px 6px 0 rgba(0, 0, 0, 0.15)",
                     "width": "85%", "margin": "auto"}),
        # next line
        html.Br(),
        html.Br()
    ])
])
```

شکل ۴۶-۳. نمودار گوگل، بخش predictions

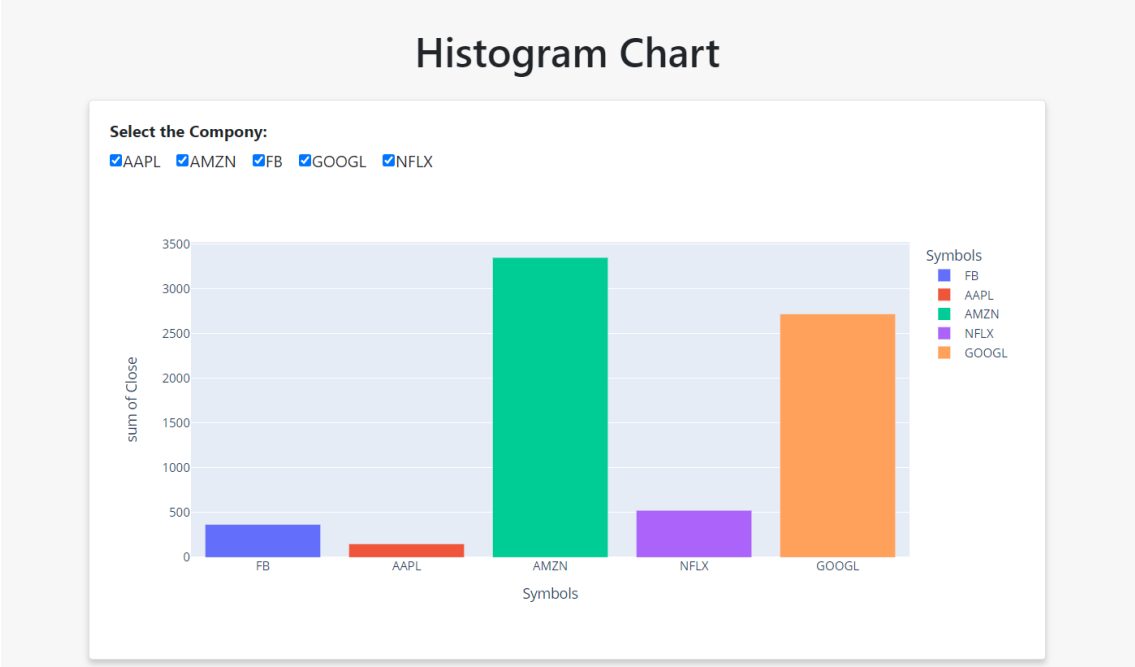
اکنون داشبورد اماده است و با اجرای سرور و برنامه می‌توان ان را مشاهده کرد.

۲-۵-۳. داشبورد:

تب اول، FAANG Stocks

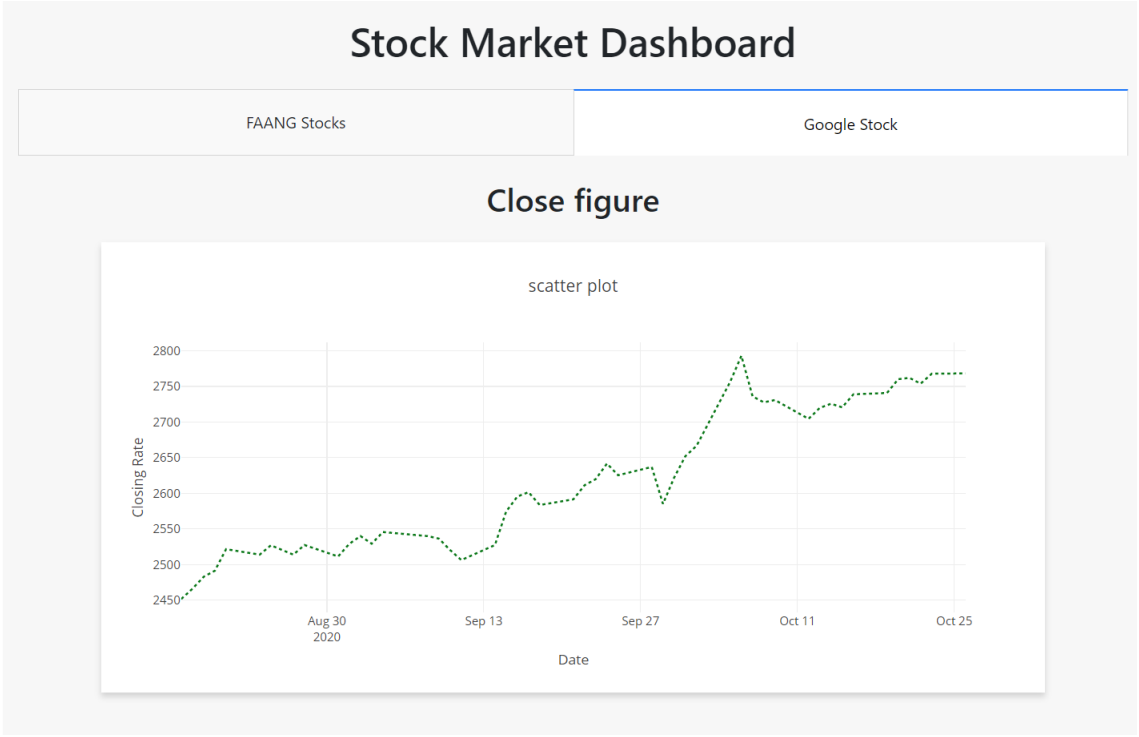


شکل ۴۷-۳. داشبورد بخش اول

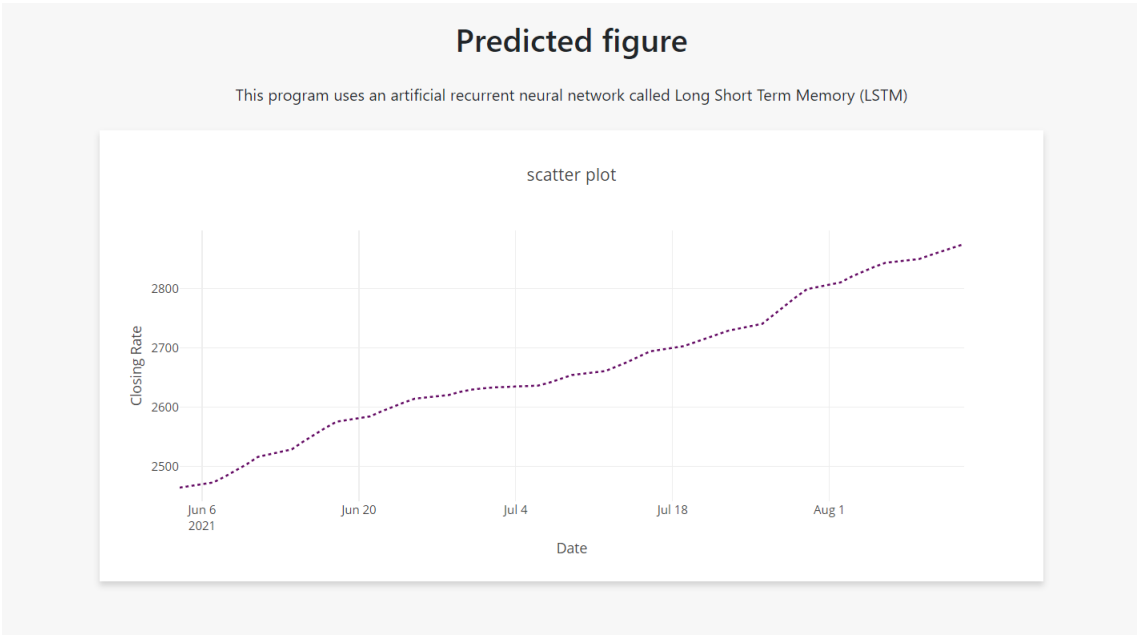


شکل ۴۸-۳. داشبورد بخش دوم

تب دوم، Google Stocks



شکل ۴۹-۳. داشبورد بخش سوم



شکل ۵۰-۳. داشبورد بخش چهارم

جمع بندی و نتیجه گیری:

هدف از انجام این پژوهش پیش بینی قیمت سهام FAANG با استفاده از تکنیک شبکه عصبی مصنوعی LSTM بود و همانطور که در متن پژوهش اشاره شده RMSE یک شاخص برای اندازه گیری دقت مدل است و مقادیری RMSE که در این پژوهش برای سهام‌ها به دست امده به صورت زیر است:

جدول ۱-۳. مقادیر RMSE

Compony	**RMSE**
Facebook	19.5679753344781
Apple	2.5280764366648945
Amazon	168.94183926463404
Netflix	12.915784445442592
Google	45.45845439251702

مقدار RMSE هر چه به صفر نزدیک‌تر باشد، دقت مدل بالاتر است. اما همانطور که مشاهده کردیم دقت هیچ‌کدام نزدیک به صفر نیست و این بدان معناست که هیچ الگوریتم معاملاتی نمی‌تواند ۱۰۰ درصد موثر باشد. گستره دقت‌هایی که به دست امده متفاوت است؛ مثلا برای امازون حدود ۱۶۹ است اما دقت مدل برای اپل حدود ۳ است و برخلاف دقت مدل برای امازون این یک دقت بسیار خوب است.

برای کارآمدتر و بهتر شدن دقت، می‌توان از مجموعه داده‌ای حجیم که میلیون‌ها ورودی دارند و می توانند دستگاه را با قدرت بیشتری آموزش دهند، استفاده کرد.

فعالیت‌های مختلف سهام می‌تواند منجر به افزایش یا پایین آمدن قیمت پیش‌بینی شده شود، از این حرکات می‌توان برای تعیین اینکه آیا یک شرکت برای انتخاب خرید سهام مناسب هست یا نه، استفاده کرد. هیچ داده آموزشی هرگز نمی‌تواند پایدار باشد، از این رو همیشه ناهمواری‌هایی وجود دارد که در گستره داده‌های موجود در پژوهش مشاهده می‌شود ، اما هم‌چنان اگر پیش بینی مدل نزدیک به نتیجه باشد یعنی اگر دارای دقت کمتر از ۴۰ باشد، منجر به یک رویکرد خوب می‌شود.

فهرست علائم اختصاری

ANN	Artificial neural network	شبکه های عصبی مصنوعی
CNN	convolutional neural networ	شبکه عصبی متحرک
FAANG Facebook,	Apple, Amazon, Netflix, Google	فیسبوک، اپل، امازون، نتفلیکس، گوگل
LSTM	Long short-term memory	حافظه کوتاه مدت طولانی
MACD	Moving average convergence divergence	واگرایی همگرایی میانگین متحرک
RMSE	root mean squared error	ریشه میانگین مربع خطا
RNN	Recurrent neural network	شبکه عصبی مکرر
SVM	Support vector machine	پشتیبانی از دستگاه بردار

فهرست مراجع

استناد به مراجع الکترونیکی

[1] https://randerson112358.medium.com/stock-price-prediction-using-python-machine-learning-e82a039ac2bb

[2] https://data-flair.training/blogs/stock-price-prediction-machine-learning-project-in-python/

[3] https://www.kaggle.com/faressayah/stock-market-analysis-prediction-using-lstm

[4] https://medium.com/swlh/dashboards-in-python-for-beginners-and-everyone-else-using-dash-f0a045a86644