

به نام خدا



گزارش پروژه درس هوش مصنوعی پیشرفته با عنوان

پیش بینی شاخص بورس با استفاده از تلفیق تکنیک های یادگیری ماشین

نویسنده

سید حسین فانی یزدی

استاد

محبوبه هوشمند

تاریخ

1399/11/24

فهرست

5.....	مقدمه:
6.....	معرفی مقاله:
8.....	پیاده سازی مقاله:
8.....	داده‌های ورودی:
9.....	شبکه عصبی:
13.....	طبقه بند SVR:
14.....	پیاده سازی الگوریتم ها:
14.....	نتایج:
22.....	نتیجه گیری:
22.....	منابع:

- تصویر 1 - معماری کلی رویکرد تک مرحله ای برای پیش بینی n امین روز آینده.....7
- تصویر 2 - معماری کلی رویکرد دو مرحله ای برای پیش بینی n امین روز آینده.....7
- تصویر 3 - جزئیات رویکرد دو مرحله ای برای پیش بینی n امین روز آینده.....8
- تصویر 4 - مدل شبکه عصبی (مدل پرسپترون با n نرون در لایه مخفی).....11

فهرست جدول (Machine_Learning_A-Z, بدون تاریخ)

- 14..... 1 Table - نتایج بدست آمده از مدل SVR در برابر ورودی ها متفاوت و مقدار Γ متفاوت
- 2 Table - اطلاعات بدست آمده از اجرای الگوریتم های شبکه عصبی با داده های بدست آمده از 10 اندیکاتور و مقایسه با تنظیمات مختلف و دو نوع شبکه عصبی متفاوت..... 16
- 3 Table - تصاویر نمودارهای همگرایی شبکه های عصبی در جدول 2 (نمودار خطای MSE نسبت به تعداد تکرار) (در شکل های ضریب 3، نمودار قرمز میزان خطای آموزش و نمودار آبی میزان خطای اعتبار سنجی می باشد). 17.....
- 4 Table - اطلاعات بدست آمده از اجرای الگوریتم های شبکه عصبی با داده های بدست آمده از قیمت پایانی 60 روز گذشته و مقایسه با تنظیمات مختلف و دو نوع شبکه عصبی متفاوت..... 18
- 5 Table - تصاویر نمودارهای همگرایی شبکه های عصبی در جدول 4 (نمودار خطای آموزش MSE نسبت به تعداد تکرار) (در شکل های ضریب 3، نمودار قرمز میزان خطای آموزش و نمودار آبی میزان خطای اعتبار سنجی می باشد). 19.....

مقدمه:

امروزه به دلیل گسترش فعالیت های اقتصادی بازارهای مالی و رونق سرمایه گذاری در بازارهای سرمایه به خصوص بورس اوراق بهادار توسط اشخاص حقیقی و حقوقی دسترسی به اطلاعات درست و به موقع و تحلیل دقیق و واقع بینانه آنها مهم ترین ابزار جهت اتخاذ تصمیم های درست و کسب منفعت مورد انتظار و استفاده بهینه و مطلوب از امکانات مالی می باشد. به نظر بعضی از کارشناسان سرمایه گذاری در سهام عرضه شده در بورس اوراق بهادار یکی از گزینه های پرسود در بازار سرمایه است. بورس اوراق بهادار، که به معنی یک بازار متشکل و رسمی سرمایه است که در آن، خرید و فروش سهام شرکتها یا اوراق قرضه دولتی یا موسسات معتبر خصوصی، تحت ضوابط و قوانین و مقررات خاصی انجام میشود؛ بنابراین دارای مشخصه مهم حمایت قانون از صاحبان پس اندازها و سرمایه های راکد و الزامات قانونی برای متقاضیان سرمایه به همین دلیل بوده، و پیش بینی این بازار یکی از مهم ترین علاقه پژوهشگران و محققان مالی محسوب شده و یکی از مهم ترین اطلاعات در بازار بورس اوراق بهادار برای سرمایه گذاران اطلاعات قیمت سهام است. قیمت سهام به طور اساسی دینامیک غیرخطی، ناپارامتریک و آشوب گونه می باشد؛ و نشان می دهد که سرمایه گذاران باید سری های زمانی را به کار ببرند که نایستا و دارای ساختار آشوبگونه است. در حقیقت پراکندگی قیمت سهام تحت تأثیر عوامل کلان اقتصادی مثل وقایع سیاسی، سیاست های شرکت ها، شرایط اقتصادی، نرخ بهره، نرخ تورم، انتظارات سرمایه گذاران، سرمایه گذاران سنتی، انتخاب و فاکتورهای فیزیکی و روانی سرمایه گذاران میباشد. بنابراین امروزه، پیش بینی قیمت سهام نه فقط خیلی چالش برانگیز، بلکه مورد علاقه زیاد سرمایه گذاران بوده و از آنجا که نوسان قیمت سهام یک روند غیرخطی دارد، روش های کلاسیک مناسب این کار نیستند؛ به دلیل اینکه درجه خطایشان زیاد است در حالی که باید این خطا کمینه شود.

پیش بینی قیمت سهام به عنوان یکی از چالش برانگیزترین کاربردهای پیش بینی سری های زمانی در نظر گرفته شده است. پیش بینی قیمت سهام به عنوان یک مساله چالش برانگیز، فرآیند پیش بینی سری های زمانی بوده، به دلیل اینکه بازار سهام در اصل دینامیک، غیرخطی، پیچیده، ناپارامتریک و بی نظم است. علاوه براین، قیمت سهام از تعداد زیادی فاکتورهای اقتصاد کلان مثل رویدادهای سیاسی، نرخ تورم، شرایط اقتصاد عمومی، انتظارات و رفتار سرمایه گذاران تأثیر می پذیرد و همچنین متغیرهای مالی مثل حجم مبادلات، نسبت E/P و ... بر قیمت سهام هر شرکتی تأثیر زیادی دارند و در واقع رفتار سرمایه گذاران را جهت میدهند. شبکه های عصبی برای حل مسائل پیچیده و غیرخطی مناسب تر از سری های زمانی هستند، هرچند مشکلات خود رانیز دارند؛ مثلاً در یادگیری دچار محدودیت اند چون داده های مربوط به قیمت سهام دارای اختلافات فاحش و ابعاد پیچیده ای هستند و لذا پیش بینی قیمت سهام کار بسیار مشکلی است.

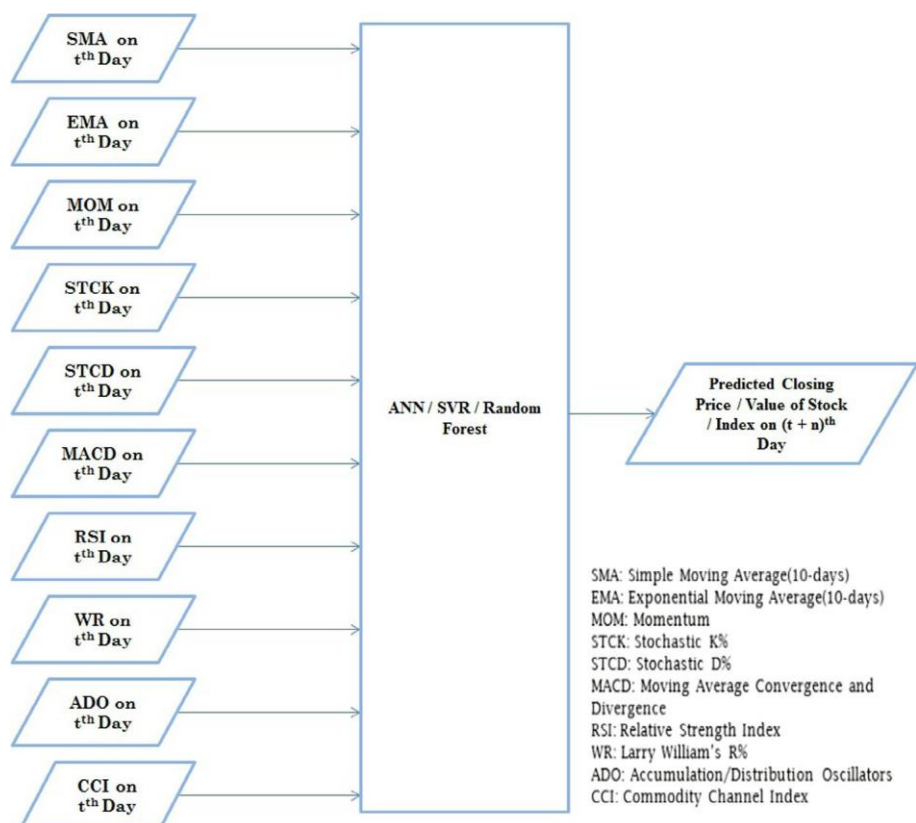
معرفی مقاله:

در این مقاله سعی شده تا با استفاده از تکنیک های مختلف یادگیری ماشین، مدلی را بسازد که بتواند شاخص بازار سرمایه را در روزهای پیشرو پیش بینی کند. در نظر داشته باشید که محقق برای رسیدن به چنین مدلی ابتدا راه حلی را معرفی می کند و در مرحله بعدی سعی دارد تا آنرا بهبود ببخشد. سپس با انجام آزمایشاتی نتایج هر دو مدل رو باهم مقایسه می کند تا بتواند نتیجه گیری مناسبی داشته باشد.

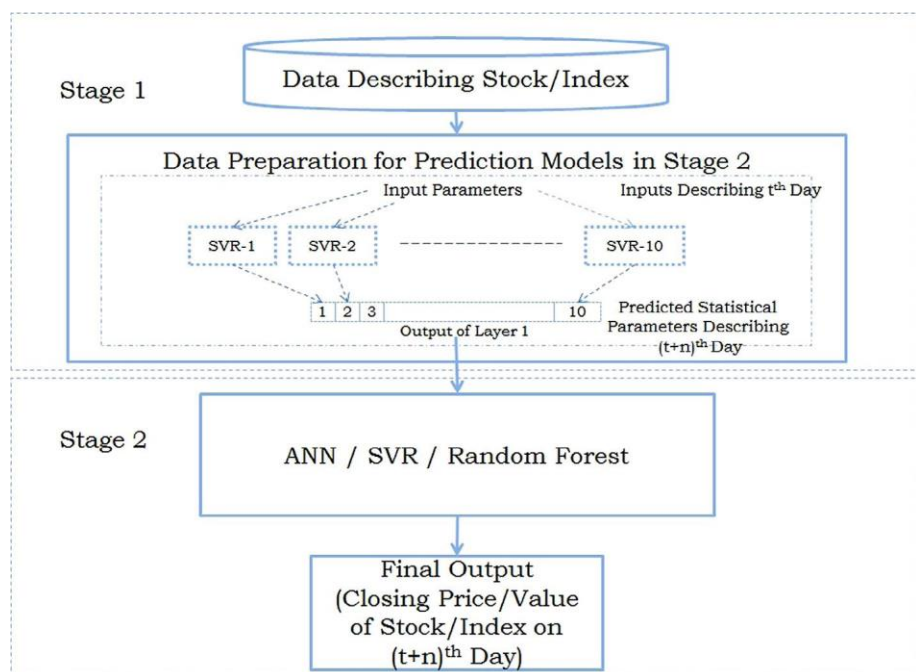
محقق با معرفی طبقه بندی SVR و Random Forest و شبکه عصبی، ابتدا مدلی تک فاز را ارائه می دهد که با توجه به ورودی خود می تواند روزهای پیشرو را پیش بینی کند. در این مدل تک فاز می توان از هر یک از تکنیک ها به صورت جداگانه استفاده کرد. سپس مدلی دو فاز و ترکیبی را پیشنهاد می دهد که در فاز اول ابتدا داده های ورودی توسط تکنیک SVR پرداز می شوند و پس از آن به فاز دوم می روند. در این مدل، فاز دوم همانند مدل تک فاز می باشد با این تفاوت که ورودی ها را از فاز اول دریافت میکند.

در هر یک از این دو مدل پیشنهادی، داده های بازار سرمایه توسط 10 اندیکاتور پردازش میشوند و سپس به مدل ها داده می شوند. باید توجه داشت که در بعضی اندیکاتورها، نیاز است تا تنظیماتی را در نظر بگیریم. به عنوان مثال برای اندیکاتور SMA باید تعیین کنیم که چه تعداد از روزهای گذشته را پردازش کنیم. به همین دلیل با اینکه در مقاله بعضی از تنظیمات بیان شده اند، اما باز هم ممکن است در پیاده سازی برخی تنظیمات و محاسبات متفاوت باشند. قابل توجه است که در یکی از دو مورد از اندیکاتور ها محاسبات و فرمول پیشنهادی در مقاله اشتباه بوده که یقیناً در نتایج تاثیر خواهد داشت. با توجه به همه این احتمالات، میتوان گفت که ممکن است داده های ورودی در پیاده سازی و داده های محقق متفاوت باشند.

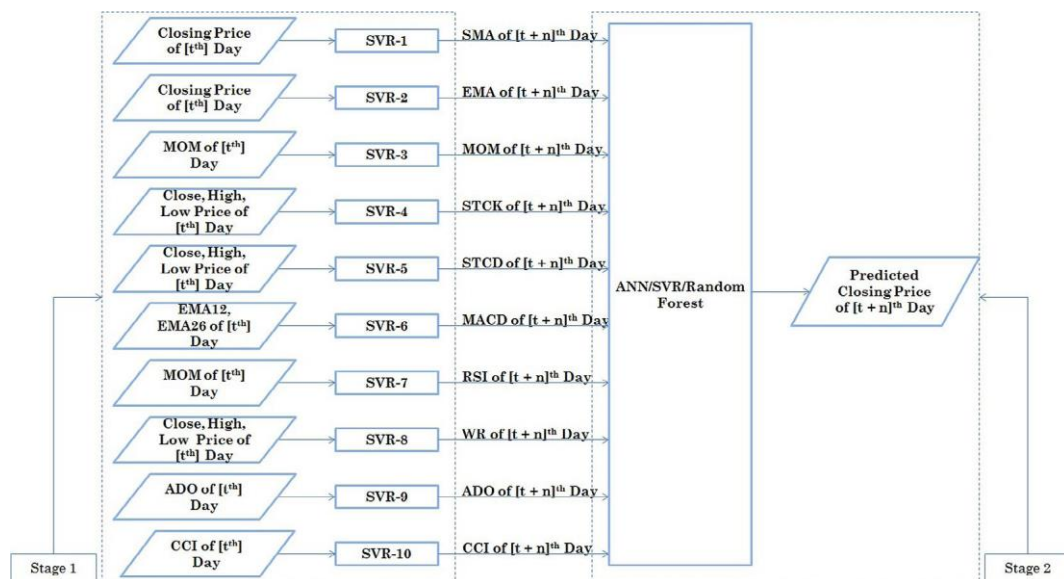
محقق بعد از معرفی مدل ها و معرفی اندیکاتور و داده های آموزشی، مدل های پیشنهادی را آزمایش میکند و نتایج را میسنجد و مقایسه می کند. قابل ذکر است که توضیحات مقاله در مورد پیاده سازی بسیار ناچیز بوده و در بعضی موارد، اطلاعات و راه های مورد استفاده غلط می باشد.



تصویر 1 - معماری کلی رویکرد تک مرحله ای برای پیش بینی n امین روز آینده



تصویر 2 - معماری کلی رویکرد دو مرحله ای برای پیش بینی n امین روز آینده



تصویر 3 - جزئیات رویکرد دو مرحله ای برای پیش بینی h مین روز آینده

پیاده سازی مقاله:

داده‌های ورودی:

برای ارائه داده‌ها به شبکه عصبی و یا تکنیک‌های دیگر، دو راه پیش رو داریم. اول استفاده از 10 اندیکاتور همانند چیزی که در مقاله معرفی شده بود. راهی دیگر اینگونه است که میتوان فقط قیمت پایانی n روز قبل (60 روز) را به عنوان ورودی به شبکه بدهیم تا شبکه بر اساس روند و یادگیری خود بتواند روز بعدی را پیش بینی کند. بدیهی است که در روش دوم تعداد نرون های لایه ورودی برابر n خواهند بود. عدد 60 معرفی شده در این بخش با آزمون و خطا بدست آمده. لازم به ذکر است که روش دوم از اطمینان زیادی برخوردار نیست و ممکن است در نوسانات بزار عمل پیش بینی به درستی عمل نکند.

هر دو روش های فوق را می توان روی داده های بازارهای مختلف منجمله ایران استفاده کرد اما در نظر داشته باشد که برای بعضی اندیکاتورها به داده های بجز قیمت پایانی نیاز است. در ادامه، 10 اندیکاتور را به اختصار معرفی می کنیم. لازم به ذکر است که برای محاسبه اندیکاتور ها از دو کتابخانه معتبر ta و $finta$ استفاده شده است که محاسبات خود را بر اساس استانداردهای جهانی انجام داده اند اما برای کامل بودن کار بعضی اندیکاتورها را نیز خودمان نیز پیاده کرده ایم و آنها را در توابعی به صورت جامع نوشتیم تا کاربر به راحتی از آنها استفاده کند.


```

indicatorName = []
Stock_data_pd = readStockData()
# (use = 1) -> selfwrite
# (use = 2) -> ta library
# (use = 3) -> finta library
SMA(Stock_data_pd, 10, use = 3)
EMA(Stock_data_pd, 50, use = 3)
RSI(Stock_data_pd, 50, use = 3)
CCI(Stock_data_pd, 20, 0.015, use = 3)
MACD(Stock_data_pd, period_fast = 12, period_slow = 26, period_si
gn = 9, use = 3)
WilliamsR(Stock_data_pd, period = 14, use = 3)
Momentum(Stock_data_pd, period = 10, use = 3)
Stochastics(Stock_data_pd, 14, 3, use = 3)
# AccDistIndicator(Stock_data_pd, use = 3)
saveStockData_toCSV(Stock_data_pd, 'With_Indicators')

```

ابتدا داده‌های بازار را توسط تابعی مشخص و با کمک کتابخانه **panda** میخوانیم و سپس با استفاده از توابع ساخته شده برای اندیکاتورها آنها را به داده‌های خود اضافه می‌کنیم. تنظیمات استفاده شده، با توجه به استانداردهای جهانی انتخاب شده‌اند اما باید بدانیم که این اعداد براساس تجربه بدست می‌آیند و باید متناسب با نمودار انتخاب شود.

بعد از آنکه داده‌های خود را بدست آوردیم (چه از روش اول و چه از روش دوم)، باید داده‌ها را نرمالسازی کنیم تا بازه تغییرات داده‌ها باهم اختلاف زیادی نداشته باشد. بدین منظور از روش **MinMaxScaling** استفاده می‌کنیم تا داده‌ها در بازه 0 تا 1 نرمال شوند.

بعد از آن داده‌ها به 3 دسته آموزش، تست و پیش بینی تقسیم می‌شوند. دلیل آن اینست که داده‌ها تست و پیش بینی باید دیده نشده و جدا از داده‌های آموزش باشند. بدین ترتیب 70 درصد کل داده‌ها را برای آموزش، 20 درصد برای تست و 10 درصد برای پیش‌بینی در نظر می‌گیریم. البته می‌توانستیم بجای داده‌های پیش‌بینی از داده‌ها تست هم استفاده کنیم اما بدلیل وجود داده کافی برای بخش آموزش، تصمیم بر این تقسیم بندی گرفته شد.

شبکه عصبی:

از آنجا که محقق در ارائه اطلاعات در مورد نوع، نحوه و چگونگی پیاده سازی کوتاهی کرده است و فقط به ارائه نتایج و توضیحات کوتاه بسنده کرده است، بناچار مجبور بوده‌ایم تا با استفاده از دانش خود اقدام به پیاده سازی کنیم. در پیاده سازی شبکه عصبی، از دو نوع شبکه با ساختار متفاوت استفاده کرده‌ایم. ساختار اول، شبکه عصبی پرسپترون که ساختاری با اتصال کامل و متراکم را دارد و ساختار دوم، شبکه عصبی بازگشتی و دارای حافظه می‌باشد که با توجه به پیچیدگی مناسب و حافظه خود در پیش بینی سری‌های زمانی، در مواردی کمی بهتر از شبکه عصبی پرسپترون می‌باشد.

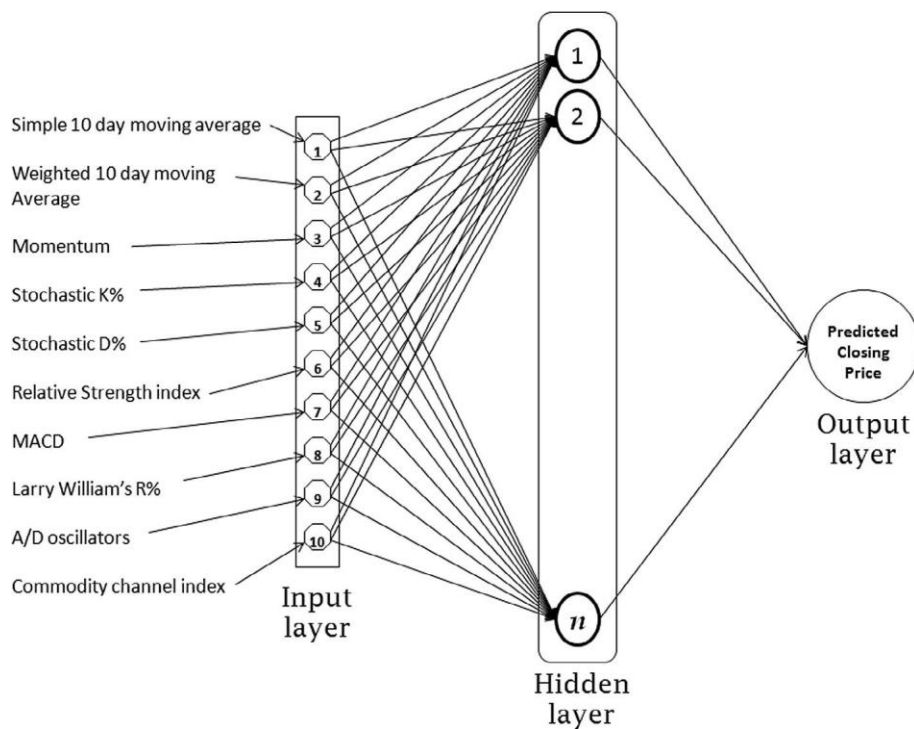
در این پیاده سازی که تماماً با زبان Python می باشد با توجه به نیاز برای توسعه شبکه عصبی از کتابخانه های Keras و Tensorflow استفاده شده است. در ادامه ساختار این دو شبکه را بررسی می کنیم. ابتدا شبکه پرسپترون:

```
layerName = 'Dense'
# Designing the Neural
model = Sequential()
model.add(Dense(50, input_dim = 10, activation = 'sigmoid', name='layer_1'))
model.add(Dense(1, activation='relu', name='layer_2'))
# Compiling and Get Summary
model.compile(optimizer = 'adam' , loss = 'mse')
model.summary()
```

در پیاده سازی این شبکه با توجه به نوع داده های ورودی تعداد 10 نرون در لایه ورودی می باشد که می توان براساس نوع داده ورودی آنرا تغییر داد و 50 نرون در لایه میانی که با فعال ساز سیگموئید عمل می کند و 1 نرون در لایه خروجی که قیمت پایانی روز بعد را برای ما پیش بینی می کند و دارای فعال ساز خطی می باشد. برای محاسبه خطا از معیار میانگین مربعات خطا (MSE) و برای بهینه سازی از گرادیان نزولی تطبیقی (adam) استفاده شده است. در مراحل قبلی با انواع داده های ورودی بکار گرفته شده آشنا شده اید. در قسمت بعدی، پیاده سازی شبکه عصبی بازگشتی و دارای حافظه را بررسی می کنیم.

```
layerName = 'LSTM_Dense'
# Designing the Neural
model = Sequential()
model.add(LSTM(50, return_sequences = True , input_shape = (X_Train.shape[1] , 1) , name='layer_1'))
model.add(LSTM(100, return_sequences = False , name='layer_2'))
model.add(Dense(50 , name='layer_3', activation='relu'))
model.add(Dense(1 , name='layer_4', activation='relu'))
# Compiling and Get Summary
model.compile(optimizer = 'adam' , loss = 'mse')
model.summary()
```

در این پیاده سازی از دو لایه LSTM و دو لایه Dense استفاده شده است که در لایه اول LSTM با 50 نرون عمل بازگشتی انجام می شود و در لایه بعدی با 100 نرون عمل بازگشتی فقط بین نرون ها صورت می گیرد. در دو لایه بعدی Dense، یکی با 50 نرون و دیگری با 1 نرون به عنوان لایه خروجی، از تابع فعالساز خطی و همچنین برای محاسبه خطا از معیار میانگین مربعات خطا (MSE) و برای بهینه سازی از گرادیان نزولی تطبیقی (adam) استفاده شده است.



تصویر 4 - مدل شبکه عصبی (مدل پرسپترون با n نرون در لایه مخفی)

تعیین تعداد نرون‌ها و تعداد لایه‌های میانی، بر اساس تجربه و آزمون و خطاست. در هر دو شبکه عصبی این ساختار نتیجه بهتری را رقم می‌زد. شاید با تغییرات در ساختار بتوان به راه حل‌های بهتری نیز رسید. علاوه بر این، تعیین تعداد تکرار در شبکه عصبی امری مهم است که این مهم نیز با بررسی و آزمایشات بسیار بدست می‌آید. البته راه‌های مثل نمودار اعتبار سنجی نیز می‌تواند به پیدا کردن تکرار مناسب کمک کند.

```
# Trian Setting
epochs          = 10
verbose         = 1
shuffle         = True
validation_split = 0.0

# Training action
model.fit(x      = X_Train,
          y      = Y_Train,
          batch_size = 1,
          verbose = verbose,
          epochs   = epochs,
          shuffle  = shuffle,
          validation_split = validation_split,
```

```
callbacks = [logger]
)
```

در هر دو شبکه عصبی عمل آموزش با دستورات یکسان اجرا می‌شود. تعداد تکرارها، نوع لاگ کردن، بهم ریختن ترتیب داده‌ها و تقسیم داده‌های آموزش برای داشتن داده‌های اعتبار سنجی در این بخش مشخص می‌شود. دستور بالا برای آموزش شبکه استفاده می‌شود.

```
test_Error_MSE = model.evaluate(x = X_Test,
                                y = Y_Test,
                                verbose = 2)
```

در دستور بالا، بعد از عمل آموزش، شبکه با داده‌ها تست، لرزایی می‌شود و خطا با معیار MSE محاسبه شده باز می‌گردد.

```
Pure_Predictions = model.predict(X_Predict)
```

در نهایت نیز، بعد از تست می‌توانیم عمل پیش‌بینی را با کمک دستور بالا انجام دهیم.

بعد از بدست آوردن نتایج از بخش تست و پیش‌بینی، می‌توانیم میزان خطا را برحسب معیارهای متفاوت بیان کنیم. در اینجا از سه معیار MSE، RMSE و MAE استفاده می‌کنیم. خطا در بخش تست و ارزیابی بدینگونه است.

```
print(f"-----TEST-----")
test_Error_RMSE = np.sqrt(test_Error_MSE)
print(f"The MSE for Test data set is: {test_Error_MSE} , RMSE Value : {test_Error_RMSE}")
print(f"The MSE for Test data set is: {test_Error_MSE:.4e} , RMSE Value: {test_Error_RMSE:.4e}")
print(f"-----")
```

خطا در بخش پیش‌بینی نیز اینگونه محاسبه می‌شود.

```
print(f"-----PREDICTION-----")
predict_Error_mse = mean_squared_error(Y_Predict, Predictions)
predict_Error_mae = mean_absolute_error(Y_Predict, Predictions)
print(f"Prediction MSE value: {predict_Error_mse} and RMSE value: {np.sqrt(predict_Error_mse)} and MAE value: {predict_Error_mae}")
print(f"Prediction MSE value: {predict_Error_mse:.4e} and RMSE value: {np.sqrt(predict_Error_mse):.4e} and MAE value: {predict_Error_mae:.4e}")
print(f"-----")
```

طبقه بند SVR:

این طبقه بند از دسته طبقه بند SVM می‌باشد که در تلاش است تا الگوی مناسب را از داده‌ها پیدا کند. برای پیاده سازی این بخش از کتابخانه sklearn استفاده شده است. برای محاسبه خطا در این بخش از همین کتابخانه کمک گرفته شده است.

در قطعه کدهای زیر، بعد از فراخوانی کتابخانه‌ها، ابتدا مدل ساخته می‌شود. سپس آموزش داده شده و پس از آن با داده‌ها تست و پیش‌بینی جداگانه ارزیابی می‌شود. بعد از بدست آوردن نتایج، میزان خطا را برحسب معیارهای متفاوت بیان می‌کنیم. در اینجا از سه معیار MSE، RMSE و MAE استفاده می‌کنیم.

```
# Libraries
from sklearn.svm import SVR
from sklearn.metrics import mean_absolute_error, mean_squared_error

svr = SVR(kernel = 'rbf', gamma = 2)

svr.fit(X_Train, Y_Train)

Predict_Y_Test      = svr.predict(X_Test)
Predict_Y_Predict   = svr.predict(X_Predict)

print(f"-----TESTING-----")
test_Error_mse = mean_squared_error(Y_Test, Predict_Y_Test)
test_Error_mae = mean_absolute_error(Y_Test, Predict_Y_Test)
print(f"Prediction MSE value: {test_Error_mse} and RMSE value: {np.
sqrt(test_Error_mse)} and MAE value: {test_Error_mae}")
print(f"Prediction MSE value: {test_Error_mse:.4e} and RMSE value:
{np.sqrt(test_Error_mse):.4e} and MAE value: {test_Error_mae:.4e}")
print(f"-----PREDICTION-----")
predict_Error_mse = mean_squared_error(Y_Predict, Predict_Y_Predict
)
predict_Error_mae = mean_absolute_error(Y_Predict, Predict_Y_Predict)
print(f"Prediction MSE value: {predict_Error_mse} and RMSE value: {
np.sqrt(predict_Error_mse)} and MAE value: {predict_Error_mae}")
print(f"Prediction MSE value: {predict_Error_mse:.4e} and RMSE valu
e: {np.sqrt(predict_Error_mse):.4e} and MAE value: {predict_Error_m
ae:.4e}")
print(f"-----")
```

پیاده سازی الگوریتم ها:

در پیاده سازی مقاله با توجه به گسترده بوده انتخاب ها برای پیاده سازی، تصمیم بر این شده است که 6 الگوریتم متفاوت در 6 فایل پایتون توسعه یابد که هر کدام از نظر ویژگی با دیگری متفاوت است. در اینجا سه نوع مدل متفاوت (دو نوع شبکه عصبی متفاوت و یک مدل SVR) و دو روش برای تولد داده ها ورودی توسعه یافته است که جمعاً 6 روش را پیشنهاد می دهد. فایل ها در کنار گزارش ارائه خواهند شد.

با توجه با گنگ بودن روش دو مرحله ای پیشنهادی در مقاله، در این گزارش سعی شده تا با تمرکز بر مدل تک مرحله ای و بهینه سازی آن نتایج را بهبود ببخشیم.

نتایج:

این نتایج بر اساس پیش بینی 1 روز آینده جمع آوری شده اند یعنی بعد از آموزش کفایست داده های ورودی لازم برای امروز را جمع آوری کرده تا شبکه قیمت پایانی فردا را پیش بینی کند. در ابتدا نتایج مربوط به مدل SVR را بررسی می کنیم.

Table 1 - نتایج بدست آمده از مدل SVR در برابر ورودی ها متفاوت و مقدار Gamma متفاوت

MAE Prediction	RMSE Prediction	MSE Prediction	MAE TEST	RMSE TEST	MSE TEST	Gamma	نوع داده ها
2.7068e-02	3.4047e-02	1.1592e-03	4.2817e-02	5.3703e-02	2.8841e-03	0.5	با اندیکاتور
2.1437e-02	2.8289e-02	8.0025e-04	4.3878e-02	5.5309e-02	3.0591e-03	1	با اندیکاتور
2.4871e-02	3.1058e-02	9.6457e-04	4.9078e-02	6.1379e-02	3.7673e-03	2	با اندیکاتور
3.0057e-02	3.7657e-02	1.4181e-03	5.6752e-02	7.0906e-02	5.0277e-03	3	با اندیکاتور
3.2596e-02	4.1660e-02	1.7355e-03	6.3070e-02	7.8690e-02	6.1920e-03	4	با اندیکاتور
3.5881e-02	4.6206e-02	2.1350e-03	7.0611e-02	8.6948e-02	7.5600e-03	5	با اندیکاتور
6.9134e-02	7.8667e-02	6.1885e-03	1.1395e-01	1.2958e-01	1.6791e-02	10	با اندیکاتور
3.0113e-02	3.5931e-02	1.2910e-03	4.0352e-02	5.0666e-02	2.5670e-03	0.5	پایانی 60 روز گذشته
2.2110e-02	2.5736e-02	6.6236e-04	4.2616e-02	5.3880e-02	2.9031e-03	1	پایانی 60 روز گذشته

2.0832e-02	2.5312e-02	6.4072e-04	4.8878e-02	6.2643e-02	3.9242e-03	2	پایانی 60 روز گذشته
4.0580e-02	4.6246e-02	2.1387e-03	6.6304e-02	7.9984e-02	6.3975e-03	3	پایانی 60 روز گذشته
6.4689e-02	7.1495e-02	5.1116e-03	8.8095e-02	9.9791e-02	9.9583e-03	4	پایانی 60 روز گذشته
7.6596e-02	8.4527e-02	7.1448e-03	1.0190e-01	1.1358e-01	1.2899e-02	5	پایانی 60 روز گذشته
1.1706e-01	1.2734e-01	1.6216e-02	1.5012e-01	1.6407e-01	7.6596e-02	10	پایانی 60 روز گذشته

با توجه به نتایج بدست آمده از جدول می‌توان عنوان کرد که مقدار gamma کوچکتر نتیجه بهتری را به ارمقان می‌آورد. با این حال در نوع داده‌های ورودی دوم (قیمت پایانی 60 روز گذشته) کمترین مقدار خطا در gamma برابر 2 می‌باشد. اما کمترین خطا در نوع دیگر داده متغییر می‌باشد. علاوه بر این می‌توان بیان کرد که نتایج در نوع داده دوم (قیمت پایانی 60 روز گذشته) بسیار رضایت بخش‌تر نسبت به نوع داده اول (اندیکاتورها) می‌باشد.

در مرحله بعدی نتایج بدست آمده از شبکه عصبی را بررسی می‌کنیم. ابتدا در جدول شماره 2، داده‌های نوع اول یعنی اندیکاتورها را به شبکه‌ها وارد می‌کنیم. نمودار همگرایی و خطای آموزش (MSE) نسبت به تعداد تکرار نتایج این شبکه در جدول شماره 3 نمایش داده می‌شود. در ادامه، در جدول شماره 4، داده‌های نوع دوم (قیمت پایانی 60 روز گذشته) را استفاده می‌کنیم و شبکه را با آن آموزش می‌دهیم. و در نهایت در جدول شماره 5، نمودار همگرایی و خطای آموزش (MSE) نسبت به تعداد تکرار نتایج این آزمایش را بررسی و مشاهده خواهیم کرد. شایان ذکر است که در بعضی از نمودارها با توجه به حضور داده اعتبارسنجی نمودار دارای دو خط روند خواهد بود که خط روند قرمز نمایش دهنده خطای آموزش (MSE) و خط روند آبی نمایش دهنده خطای اعتبارسنجی (MSE) می‌باشد.

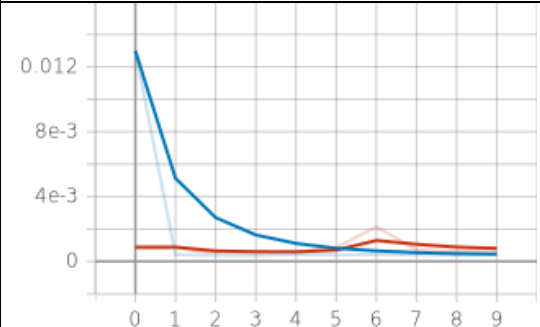
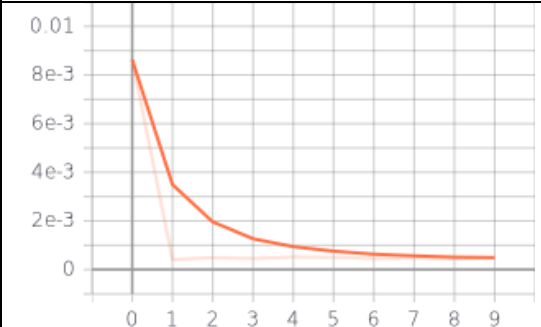

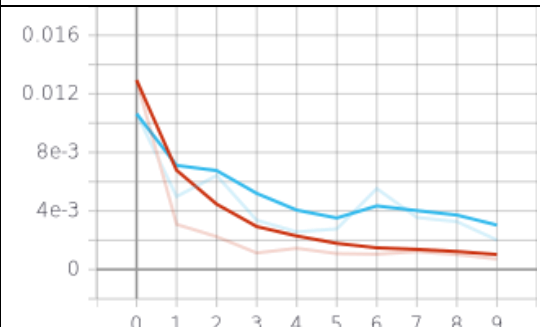
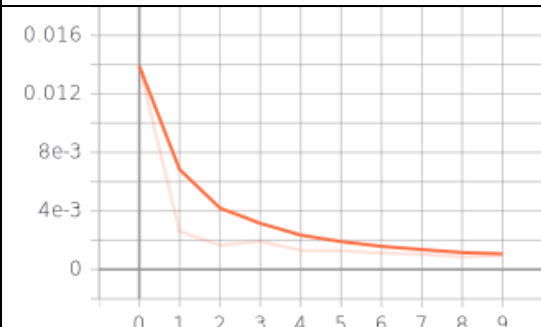

Table 2 - اطلاعات بدست آمده از اجرای الگوریتم های شبکه عصبی با داده های بدست آمده از 10 اندیکاتور و مقایسه با تنظیمات مختلف و دو نوع شبکه عصبی متفاوت

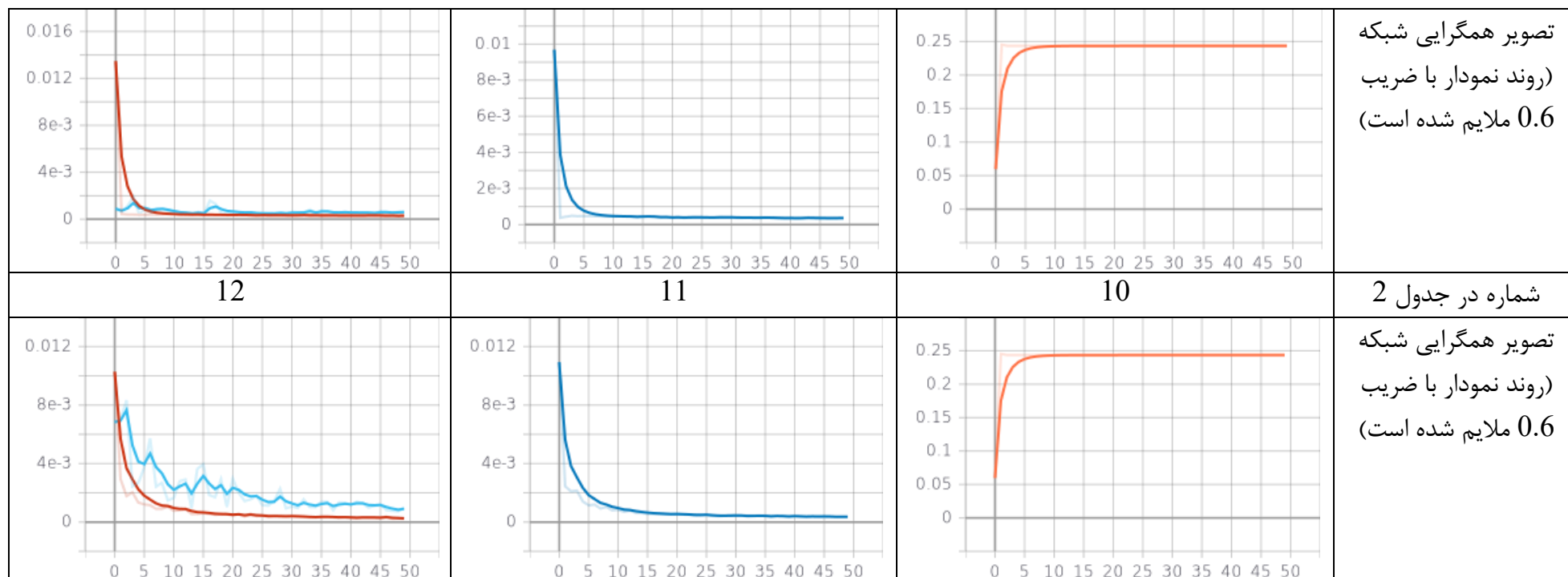
MAE Prediction	RMSE Prediction	MSE Prediction	RMSE TEST	MSE TEST	MSE Train	Validation split (0-1)	Shuffle	Epoch	Input Dim	Input Data	Net	Index
0.8155	0.8171	0.6676	0.836	0.6989	0.2436	0.0	False	10	10	1712	Only Dense	1
8.9696e-03	1.1056e-02	1.2224e-04	1.4401e-02	2.0740e-04	4.7219e-04	0.0	True	10	10	1712	Only Dense	2
1.7603e-02	1.9786e-02	3.9148e-04	2.0895e-02	4.3658e-04	4.1641e-04 (6.8945e-04)	0.2	True	10	10	1369 (Val 343)	Only Dense	3
0.8155	0.8171	0.6676	0.836	0.6989	0.2436	0.0	False	10	10	1712	Dense & LSTM	4
2.3169e-02	2.5926e-02	6.7216e-04	3.1543e-02	9.9499e-04	9.4328e-04	0.0	True	10	10	1712	Dense & LSTM	5
1.7788e-02	2.2132e-02	4.8981e-04	2.2071e-02	4.8713e-04	7.1682e-04 (0.0020)	0.2	True	10	10	1369 (Val 343)	Dense & LSTM	6
0.8155	0.8171	0.6676	0.836	0.6989	0.2436	0.0	False	50	10	1712	Only Dense	7
1.1671e-02	1.4208e-02	2.0187e-04	1.6846e-02	2.8378e-04	3.6827e-04	0.0	True	50	10	1712	Only Dense	8
1.6011e-02	1.8636e-02	3.4731e-04	1.9828e-02	3.9314e-04	3.0650e-04 (6.3284e-04)	0.2	True	50	10	1369 (Val 343)	Only Dense	9
0.8155	0.8171	0.6676	0.836	0.6989	0.2436	0.0	False	50	10	1712	Dense & LSTM	10
1.1479e-02	1.3956e-02	1.9477e-04	2.0234e-02	4.0943e-04	3.5539e-04	0.0	True	50	10	1712	Dense & LSTM	11
1.6784e-02	1.9601e-02	3.8422e-04	6.2253e-04	6.2253e-04	2.2359e-04 (2.2359e-04)	0.2	True	50	10	1369 (Val 343)	Dense & LSTM	12

با توجه به نتایج جدول شماره 2 می توان اظهار کرد، نتایج در تعداد تکرار پایین در شبکه پرسپترون (Only Dense) نتیجه بهتری دارد اما در صورتی که تعداد تکرار بیشتر شود، شبکه عصبی بازگشتی و دارای حافظه (Dense & LSTM) نتایج بهتری می دهد. البته باید ذکر کرد که اگر داده ها به ترتیب به شبکه وارد شود خطای شبکه بسیار زیاد خواهد بود. با توجه به اینکه ویژگی های مورد استفاده شبکه برای هر قیمت پایانی قبلا به صورت مجزا توسط اندیکاتورها استخراج شده است، می توان با بهم ریختن داده ها قدرت یادگیری

شبکه را افزایش و خطای آنرا کاهش دهیم. در نظر داشته باشید، در صورتی که داده‌ها به تریب و مرتب به شبکه وارد شوند، ممکن است شبکه آنها را حفظ کند و یا نتواند آنها را به درستی یاد بگیرد. برای دوری از این مشکل، ترتیب داده‌ها را بهم خواهند ریخت.

Table 3 - تصاویر نمودارهای همگرایی شبکه‌ها عصبی در جدول 2 (نمودار خطای MSE نسبت به تعداد تکرار) (در شکل‌های ضریب 3، نمودار قرمز میزان خطای آموزش و نمودار آبی میزان خطای اعتبار سنجی می‌باشد).

3	2	1	شماره در جدول 2
			تصویر همگرایی شبکه (روند نمودار با ضریب 0.6 ملایم شده است)
6	5	4	شماره در جدول 2
			تصویر همگرایی شبکه (روند نمودار با ضریب 0.6 ملایم شده است)
9	8	7	شماره در جدول 2



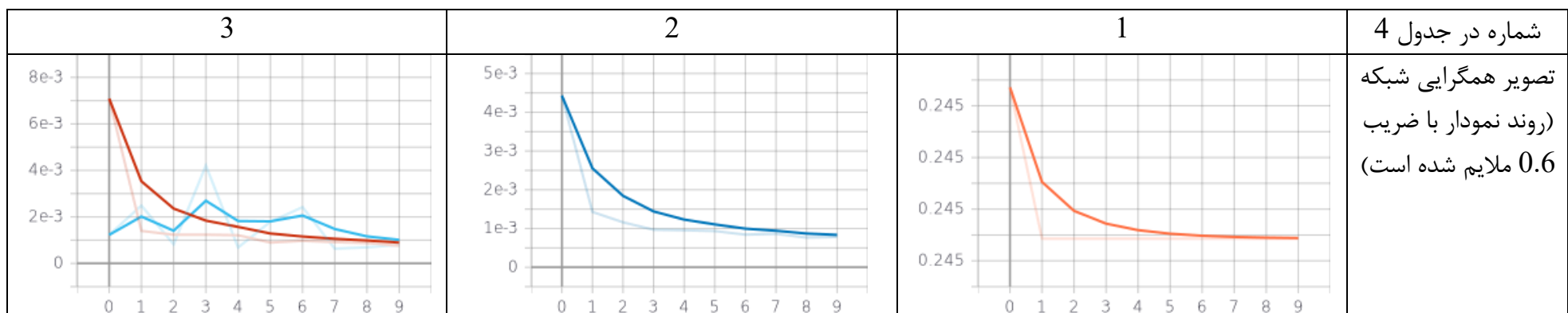
در ادامه نتایج داده‌های نوع دوم (قیمت پایانی 60 روز گذشته) را بررسی می‌کنیم.

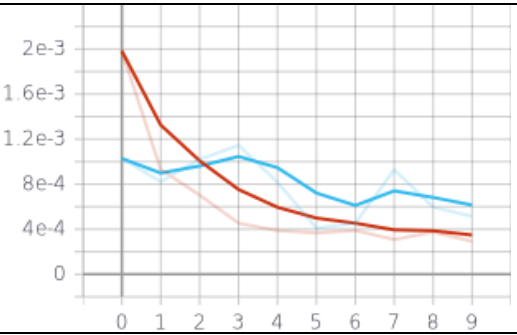
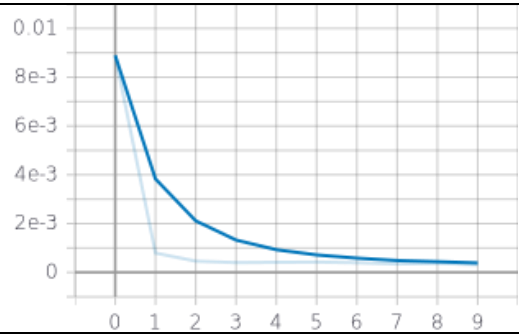
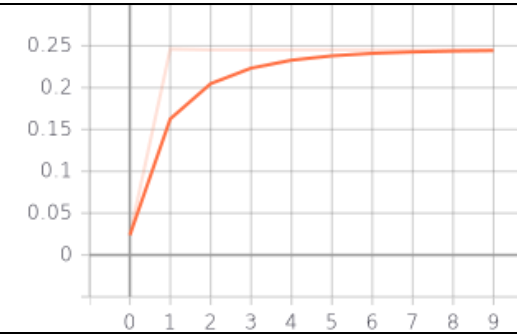
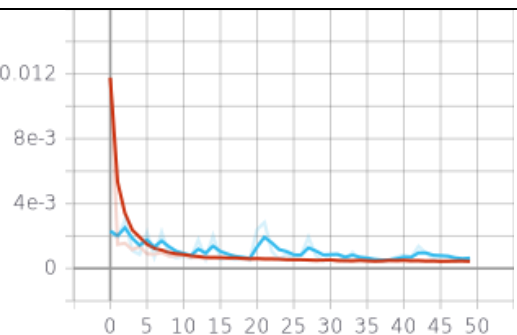
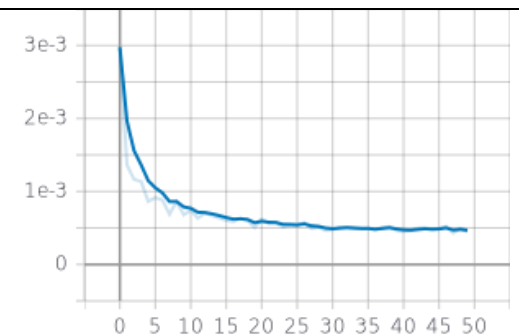
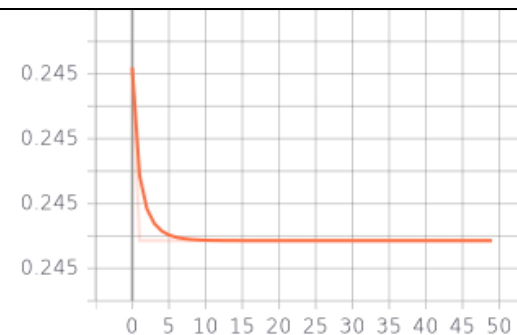
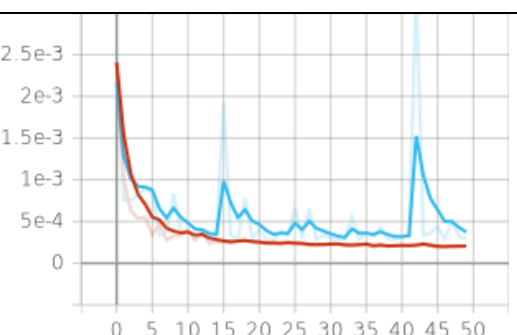
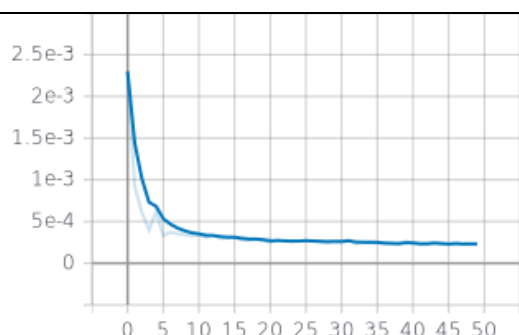
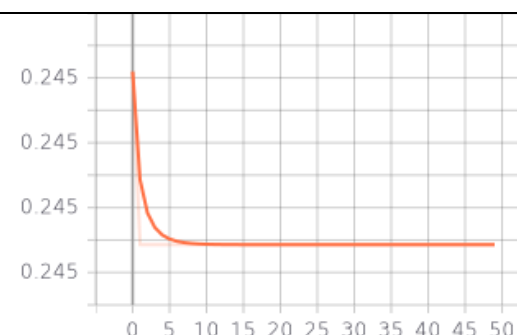
Table 4 - اطلاعات بدست آمده از اجرای الگوریتم‌های شبکه عصبی با داده‌های بدست آمده از قیمت پایانی 60 روز گذشته و مقایسه با تنظیمات مختلف و دو نوع شبکه عصبی متفاوت

MAE Prediction	RMSE Prediction	MSE Prediction	RMSE TEST	MSE TEST	MSE Train	Validation split (0-1)	Shuffle	Epoch	Input Dim	Input Data	Net	Index
0.8155	0.8171	0.6676	0.8364	0.6995	0.2452	0.0	False	10	60	1705	Only Dense	1
1.3633e-02	1.6498e-02	2.7219e-04	2.2564e-02	5.0915e-04	7.8285e-04	0.0	True	10	60	1705	Only Dense	2
1.2002e-02	1.5032e-02	2.2595e-04	2.1091e-02	4.4485e-04	7.6015e-04 (7.9220e-04)	0.2	True	10	60	1364 (Val 341)	Only Dense	3

0.8155	0.8171	0.6676	0.8364	0.6995	0.2452	0.0	False	10	60	1705	Dense & LSTM	4
1.4474e-02	1.6529e-02	2.7321e-04	1.8971e-02	3.5990e-04	3.1054e-04	0.0	True	10	60	1705	Dense & LSTM	5
1.1860e-02	1.3990e-02	1.9573e-04	1.5148e-02	2.2947e-04	2.9200e-04	0.2	True	10	60	1364 (Val 341)	Dense & LSTM	6
0.8155	0.8171	0.6676	0.8364	0.6995	0.2452	0.0	False	50	60	1705	Only Dense	7
9.8938e-03	1.2164e-02	1.4797e-04	3.4953e-04	3.4953e-04	4.4717e-04	0.0	True	50	60	1705	Only Dense	8
2.7193e-02	2.9565e-02	8.7406e-04	3.5179e-02	1.2375e-03	4.2287e-04 (6.7614e-04)	0.2	True	50	60	1364 (Val 341)	Only Dense	9
0.8155	0.8171	0.6676	0.8364	0.6995	0.2452	0.0	False	50	60	1705	Dense & LSTM	10
1.6005e-02	1.7962e-02	3.2265e-04	1.9853e-02	3.9412e-04	2.3175e-04	0.0	True	50	60	1705	Dense & LSTM	11
8.9713e-03	1.1199e-02	1.2543e-04	1.3384e-02	1.7912e-04	2.0905e-04 (2.9087e-04)	0.2	True	50	60	1364 (Val 341)	Dense & LSTM	12

Table 5 - تصاویر نمودارهای همگرایی شبکه ها عصبی در جدول 4 (نمودار خطای آموزش MSE نسبت به تعداد تکرار) (در شکل های ضریب 3، نمودار قرمز قریب میزان خطای اعتبار سنجی می باشد).



6	5	4	شماره در جدول 4
			تصویر همگرایی شبکه (روند نمودار با ضریب 0.6 ملایم شده است)
9	8	7	شماره در جدول 4
			تصویر همگرایی شبکه (روند نمودار با ضریب 0.6 ملایم شده است)
12	11	10	شماره در جدول 4
			تصویر همگرایی شبکه (روند نمودار با ضریب 0.6 ملایم شده است)

با مشاهده نتایج جدول شماره 4، در میابیم که شبکه عصبی بازگشتی و دارای حافظه (Dense & LSTM) نتایج بهتری را در هر دو نوع تعداد تکرار رقم زده و نسبت به شبکه پرسپترون (Only Dense) بهتر عمل کرده است. البته باید عنوان کرد که نتایج شبکه عصبی بازگشتی و دارای حافظه (Dense & LSTM) در این نوع داده نسبت به جدول شماره 2، که با داده اندیکاتورها آموزش دیده است بهتر می باشد.

نتیجه گیری:

از آنجا عمل پیش‌بینی بازار سرمایه با توجه به نوسانات و متغیرات این بازار بسیار سخت می‌باشد و در اکثر مواقع با خطا مواجه است، استفاده از این نوع مدل‌ها بصورت جداگانه و تمام اختیار پیشنهاد نمی‌شود. در مقالات و پیشنهادات بسیاری، مدل‌های مختلفی ارائه و پیشنهاد شده‌اند اما همه آنها در بزرگترین و حساس‌ترین نقطه بازار که رفتار جامعه و بازخورد های سیاسی و تاثیرات اقتصادی است ضعف دارند و این بخش مهم ترین قسمت برای پیش‌بینی روند بازار است.

با اینحال می‌توان بیان کرد که نتایج هر دو شبکه عصبی از مدل SVR بهتر بوده و پیش‌بینی دقیق‌تری را به ما می‌دهد. در بین شبکه‌های عصبی، به طور کلی شبکه عصبی بازگشتی و دارای حافظه (Dense & LSTM) که توسط ما پیشنهاد شد نسبت به شبکه پرسپترون (Only Dense) که در مقاله پیشنهاد شده بود نتایج بهتری را رقم زد. از بین مدل‌های ورودی و داده‌ها نیز مدل دوم (قیمت پایانی 60 روز گذشته) که توسط ما معرفی شده بود، عملکرد و تاثیر بهتری بر مدل های استفاده شده داشت و نتایج بهتری را در خروجی ارائه داد.

در نهایت پیشنهاد می‌شود، برای اینکه بتوانید در بازار سرمایه پیش‌بینی درستی را بدست آورید، بهتر است به سراغ زبان mlq و مشاور خبره (expert advisor) بروید و نتایج را در محیط استاندارد بررسی و مشاهده کنید.

منابع:

- A.R.J. Fredo, G. K. (2015). Automated segmentation and analysis of corpus callosum in autistic MR brain. *Journal of Medical and Biological Engineering*.
- F. Fumero, S. A.-H. (2011). RIM-ONE: An open retinal image database for optic. in *24th International Symposium on Computer-Based Medical Systems (CBMS)*.
- <http://medimrg.webs.ull.es/research/retinal-imaging/rim-one/>. (n.d.). Retrieved from medimrg.
- K. Manickavasagam, S. S. (2014). Development of systems for classification of different plasmodium species in. *Journal of Advanced Microscopy Research*.
- Machine_Learning_A-Z*. (n.d.). Retrieved from https://github.com/srafay/Machine_Learning_A-Z.
- N.S.M. Raja, S. S. (2015). Improved PSO based multi-level thresholding for cancer infected breast thermal images. *Procedia Computer Science*.
- N.S.M. Raja, V. R. (2014). Otsu based optimal multilevel image thresholding using firefly algorithm. *Modelling and Simulation in Engineering*.

- S. Ramakrishnan, N. R. (2014). Analysis of vasculature detection in human retinal images using bacterial foraging optimization based multi thresholding. *International Journal of Swarm Intelligence and Evolutionary Computation*,.
- S.Hore, S. C.-N. (2016). An Integrated Interactive Technique for Image Segmentation using Stack based Seeded Region Growing and Thresholding. *International Journal of Electrical and Computer Engineering*.
- V.S.Lakshmi, S. T. (2016). Chaotic cuckoo search and Kapur/Tsallis approach in segmentation of t.cruzi from blood smear images. *International Journal of Computer Science and Information Security (IJCSIS)*.