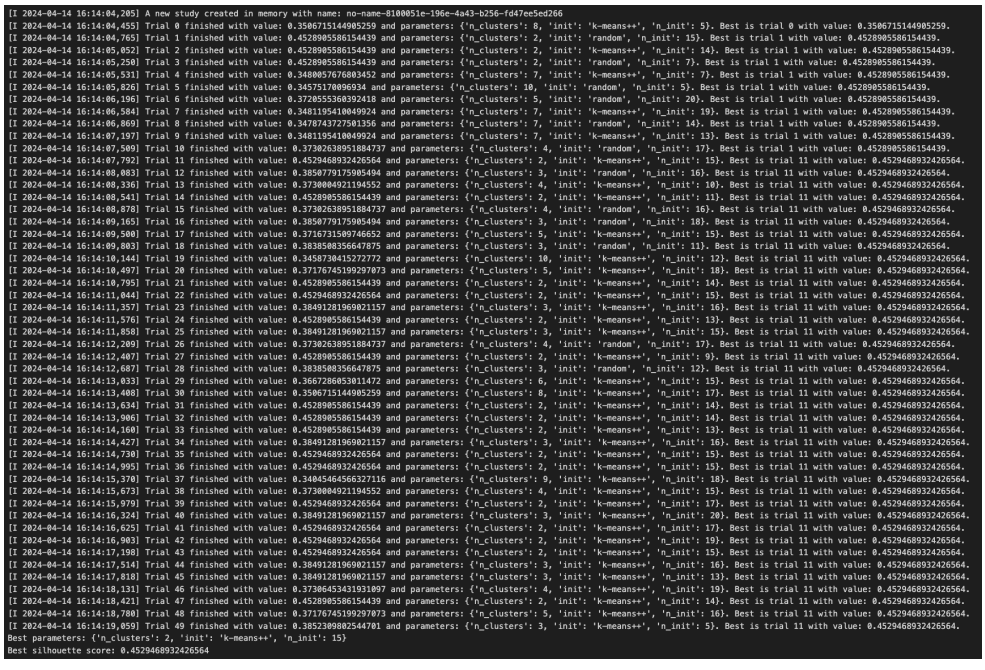
Final Project Part 4 | Christian Ruiz, cr72@rice.edu | Janel Straach, COMP 642-901

Summary

In part 3 of our final project, I was able to analyze three models that could be applied to indicate timing of equity purchases for the TQQQ ETF. We conducted a review of LSTM, XGBClassifier, and KMeans. We found that LSTM and KMeans yielded the most meaningful results for what could be used in the final model. Here in part 4, we will conduct hyperparameter tuning to optimize and fine tune our models to yield the most accurate results. In this report we will cover what has been completed in our notebook. First, we will normalize the data in preparation for kmeans and LSTM. Second, we will explore more details in the hyperparameter tuning of kmeans to increase the silhouette score for the model. Third, we will train the model on training set and conduct predictions on the test set to predict what cluster that the observation falls under. Fourth, we will conduct hyper parameterization on the RNN LSTM model to decrease the MSE results and attempt to better predict future prices with the addition of the kmeans clustering results. Finally, once we have completed hyper parameterization, we will run some test data to predict the outcomes and document our findings. Some data manipulation tasks that are necessary are set up in another data_etl_rnn_prep.py file that will be uploaded with the notebook and this report. This py file is called on in our notebook to complete these tasks and keep our notebook clean.

Testing

KMeans: We first established a baseline by re-iterating the kmeans model that was originally created in our part 3 which yielded a silhouette score of 0.45. Our goal was to initiate parameter optimization methods to find any possible combinations in parameter tuning which might yield better results. To do this we used a package called optuna which was used to automate the hyperparameter tuning process. Below are the results of running this optimization:



```
[I 2024-04-14 16:14:04,205] A new study created in memory with name: no-name-8100051e-196e-4a43-b256-fd47ee5ed266
[I 2024-04-14 16:14:04,455] Trial 0 finished with value: 0.3506715144905259 and parameters: {'n_clusters': 8, 'init': 'k-means++', 'n_init': 5}. Best is trial 0 with value: 0.3506715144905259.
[I 2024-04-14 16:14:04,765] Trial 1 finished with value: 0.4528905586154439 and parameters: {'n_clusters': 2, 'init': 'random', 'n_init': 15}. Best is trial 1 with value: 0.4528905586154439.
[I 2024-04-14 16:14:05,052] Trial 2 finished with value: 0.4528905586154439 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 14}. Best is trial 1 with value: 0.4528905586154439.
[I 2024-04-14 16:14:05,250] Trial 3 finished with value: 0.4528905586154439 and parameters: {'n_clusters': 2, 'init': 'random', 'n_init': 7}. Best is trial 1 with value: 0.4528905586154439.
[I 2024-04-14 16:14:05,531] Trial 4 finished with value: 0.3480057676003452 and parameters: {'n_clusters': 7, 'init': 'k-means++', 'n_init': 7}. Best is trial 1 with value: 0.4528905586154439.
[I 2024-04-14 16:14:05,826] Trial 5 finished with value: 0.345751708969034 and parameters: {'n_clusters': 10, 'init': 'random', 'n_init': 5}. Best is trial 1 with value: 0.4528905586154439.
[I 2024-04-14 16:14:06,196] Trial 6 finished with value: 0.37205553603092418 and parameters: {'n_clusters': 5, 'init': 'random', 'n_init': 20}. Best is trial 1 with value: 0.4528905586154439.
[I 2024-04-14 16:14:06,584] Trial 7 finished with value: 0.3481195410049924 and parameters: {'n_clusters': 7, 'init': 'k-means++', 'n_init': 19}. Best is trial 1 with value: 0.4528905586154439.
[I 2024-04-14 16:14:06,869] Trial 8 finished with value: 0.3478743727501356 and parameters: {'n_clusters': 7, 'init': 'random', 'n_init': 14}. Best is trial 1 with value: 0.4528905586154439.
[I 2024-04-14 16:14:07,197] Trial 9 finished with value: 0.3481195410049924 and parameters: {'n_clusters': 7, 'init': 'k-means++', 'n_init': 13}. Best is trial 1 with value: 0.4528905586154439.
[I 2024-04-14 16:14:07,509] Trial 10 finished with value: 0.37302638951884737 and parameters: {'n_clusters': 4, 'init': 'random', 'n_init': 17}. Best is trial 1 with value: 0.4528905586154439.
[I 2024-04-14 16:14:07,792] Trial 11 finished with value: 0.4529468932426564 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 15}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:08,083] Trial 12 finished with value: 0.3850779175905494 and parameters: {'n_clusters': 3, 'init': 'random', 'n_init': 16}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:08,336] Trial 13 finished with value: 0.37300492119455 2 and parameters: {'n_clusters': 4, 'init': 'k-means++', 'n_init': 10}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:08,541] Trial 14 finished with value: 0.4528905586154439 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 11}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:08,878] Trial 15 finished with value: 0.37302638951884737 and parameters: {'n_clusters': 4, 'init': 'random', 'n_init': 16}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:09,165] Trial 16 finished with value: 0.3850779175905494 and parameters: {'n_clusters': 3, 'init': 'random', 'n_init': 18}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:09,500] Trial 17 finished with value: 0.3716731509746652 and parameters: {'n_clusters': 5, 'init': 'k-means++', 'n_init': 15}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:09,803] Trial 18 finished with value: 0.3838503566647875 and parameters: {'n_clusters': 3, 'init': 'random', 'n_init': 11}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:10,144] Trial 19 finished with value: 0.3458730415272772 and parameters: {'n_clusters': 10, 'init': 'k-means++', 'n_init': 12}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:10,497] Trial 20 finished with value: 0.3717674519929073 and parameters: {'n_clusters': 5, 'init': 'k-means++', 'n_init': 18}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:10,795] Trial 21 finished with value: 0.4528905586154439 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 14}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:11,044] Trial 22 finished with value: 0.4529468932426564 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 15}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:11,357] Trial 23 finished with value: 0.3849128196902115 7 and parameters: {'n_clusters': 3, 'init': 'k-means++', 'n_init': 16}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:11,576] Trial 24 finished with value: 0.4528905586154439 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 13}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:11,858] Trial 25 finished with value: 0.3849128196902115 7 and parameters: {'n_clusters': 3, 'init': 'k-means++', 'n_init': 15}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:12,209] Trial 26 finished with value: 0.37302638951884737 and parameters: {'n_clusters': 4, 'init': 'random', 'n_init': 17}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:12,407] Trial 27 finished with value: 0.4528905586154439 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 9}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:12,607] Trial 28 finished with value: 0.3838503566647875 and parameters: {'n_clusters': 3, 'init': 'random', 'n_init': 12}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:13,033] Trial 29 finished with value: 0.3667286053011472 and parameters: {'n_clusters': 6, 'init': 'k-means++', 'n_init': 15}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:13,408] Trial 30 finished with value: 0.3506715144905259 and parameters: {'n_clusters': 8, 'init': 'k-means++', 'n_init': 17}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:13,634] Trial 31 finished with value: 0.4528905586154439 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 14}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:13,906] Trial 32 finished with value: 0.4528905586154439 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 14}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:14,160] Trial 33 finished with value: 0.4528905586154439 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 13}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:14,427] Trial 34 finished with value: 0.3849128196902115 7 and parameters: {'n_clusters': 3, 'init': 'k-means++', 'n_init': 16}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:14,730] Trial 35 finished with value: 0.4529468932426564 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 15}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:14,995] Trial 36 finished with value: 0.4529468932426564 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 15}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:15,370] Trial 37 finished with value: 0.3404546456632711 6 and parameters: {'n_clusters': 9, 'init': 'k-means++', 'n_init': 18}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:15,673] Trial 38 finished with value: 0.3730004921194552 and parameters: {'n_clusters': 4, 'init': 'k-means++', 'n_init': 15}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:15,979] Trial 39 finished with value: 0.4529468932426564 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 17}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:16,324] Trial 40 finished with value: 0.3849128196902115 7 and parameters: {'n_clusters': 3, 'init': 'k-means++', 'n_init': 20}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:16,625] Trial 41 finished with value: 0.4529468932426564 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 17}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:16,903] Trial 42 finished with value: 0.4529468932426564 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 19}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:17,198] Trial 43 finished with value: 0.4529468932426564 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 15}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:17,514] Trial 44 finished with value: 0.3849128196902115 7 and parameters: {'n_clusters': 3, 'init': 'k-means++', 'n_init': 16}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:17,810] Trial 45 finished with value: 0.3849128196902115 7 and parameters: {'n_clusters': 3, 'init': 'k-means++', 'n_init': 13}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:18,131] Trial 46 finished with value: 0.3730645343193109 7 and parameters: {'n_clusters': 4, 'init': 'k-means++', 'n_init': 19}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:18,421] Trial 47 finished with value: 0.4528905586154439 and parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 14}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:18,780] Trial 48 finished with value: 0.3717674519929073 and parameters: {'n_clusters': 5, 'init': 'k-means++', 'n_init': 16}. Best is trial 11 with value: 0.4529468932426564.
[I 2024-04-14 16:14:19,059] Trial 49 finished with value: 0.3852309802544701 and parameters: {'n_clusters': 3, 'init': 'k-means++', 'n_init': 5}. Best is trial 11 with value: 0.4529468932426564.
Best parameters: {'n_clusters': 2, 'init': 'k-means++', 'n_init': 15}
Best silhouette score: 0.4529468932426564
```

It is difficult to see, but after running this optimization algorithm, we still end up with a silhouette score near our original score, 0.45. We attempted to make some changes to the parameters by setting a large range of clusters to determine if there were any unforeseen better optimized clusters in the larger range. Set the options parameter for k-means++ and random, which essentially affects the rate of convergence. However, the default option for convergence, k-means++, is already the most optimized option for convergence compared with the random option, which sets observations at random from the data for the initial centroids. We also tested a range of parameters for n_init, which identify the number of times that the algorithm runs with different centroid seeds. A larger value of n_init can yield greater stability and quality of clusters but may take more time. Considering the many iterations, we ran here not making a large impact on the final score, we've decided keep the original parameters for the kmeans model and use the results of this model as an additional independent variable of our data for running our LSTM model.

LSTM RNN: We established a baseline with our LSTM model, just like we did for kmeans, by iterating the same model which we used in part 3 of our project. However, we did include the results of our kmeans clusters in our data. We did not transform or normalize these clusters considering it is only reporting 1 and 0. The results are insignificant with a model performance of 0.20 when running on our testing data. Rather than change many parameters, I decided to increase the number of epochs from 50 to 150 and see if that changed anything. It did, but in the wrong direction, resulting in model performance of 0.21. Rather than make many changes by hand for hyper parameter tuning, I decided to use a keras_tuner package which allowed me to iteratively run through many parameter specifications.

We initiated this optimization with range of parameters for the LSTM layer units with a min value of 20, max of 200, and increasing/decreasing in steps of 10. Changing the units changes the number of neurons in each LSTM layer which can have very significant effects on the model's performance and ability to generalize. We set parameters for the dropout layer with a min value of 0.1, max of 0.5 and step increase of 0.1. The dropout rate helps model in reducing overfitting. If this parameter is set too high then we could possibly underfit the model. The optimizer was set for adam and rmsprop which essentially affects the learning rate and ultimately impacts the training dynamics and final performance. After about 20 minutes of this optimization running, we received better results for our LSTM model:

```
The best number of units in the LSTM layers: 170
The best dropout rate: 0.5
The best optimizer: rmsprop
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - loss: 0.1337
Best Model Test Loss: 0.12416800111532211
20/20 ━━━━━━━━━━━━━━━━━━━━ 1s 21ms/step
mse: 0.12794881990493165
```

```
MSE for 20 day window, 1 day close: 10.387939869342365
MSE for 20 day window, 3 day close: 13.633824648833965
MSE for 20 day window, 5 day close: 20.822151599782917
MSE for 20 day window, 10 day close: 23.15540902811913
MSE for 20 day window, 20 day close: 29.402208722379402
```

Model: "LSTM_20_Day_Window"

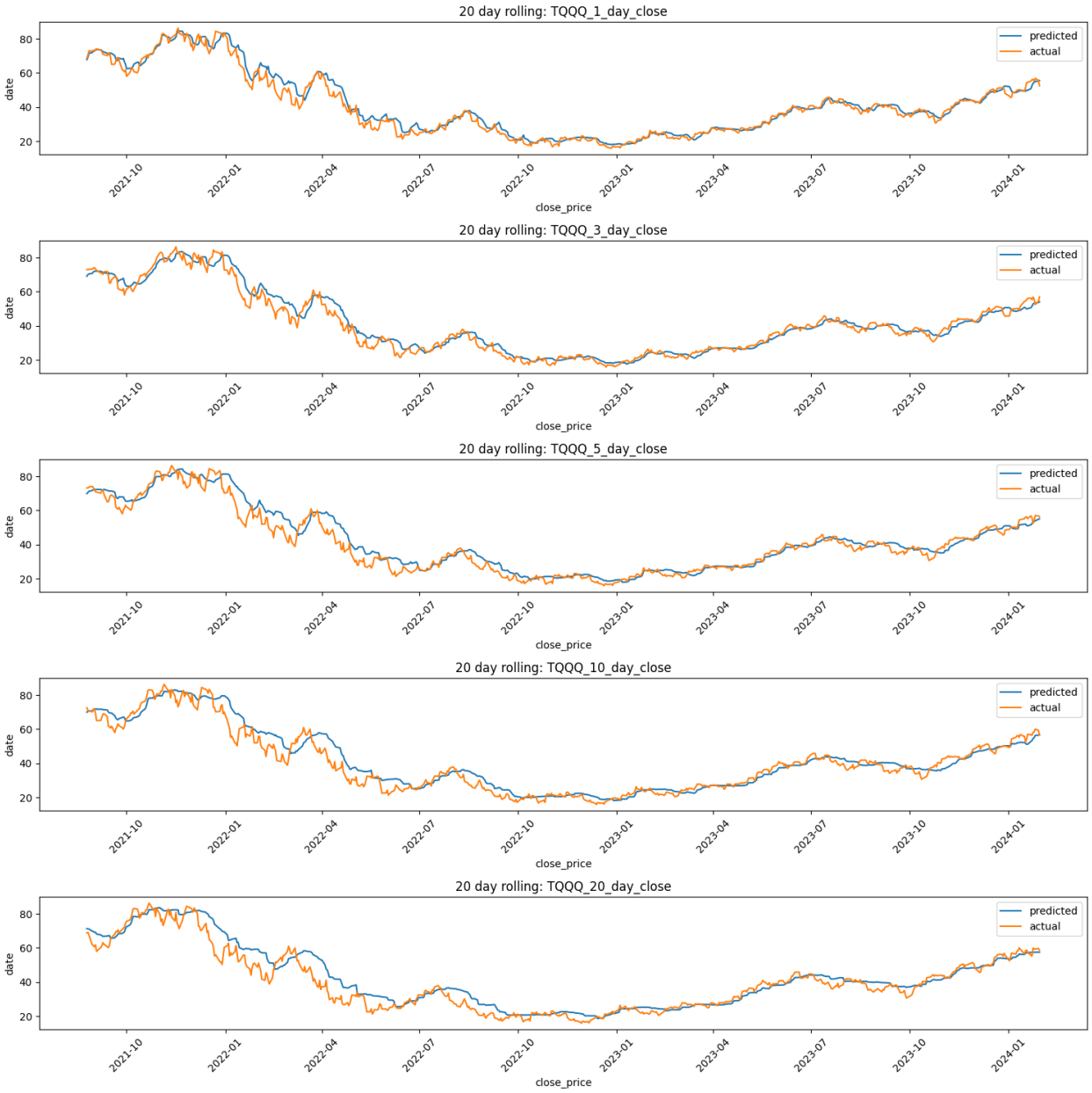
Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 20, 170)	134,640
dropout (Dropout)	(None, 20, 170)	0
lstm_1 (LSTM)	(None, 20, 170)	231,880
dropout_1 (Dropout)	(None, 20, 170)	0
lstm_2 (LSTM)	(None, 170)	231,880
dropout_2 (Dropout)	(None, 170)	0
dense (Dense)	(None, 5)	855

Total params: 599,255 (2.29 MB)

Trainable params: 599,255 (2.29 MB)

Non-trainable params: 0 (0.00 B)

When we un-scale the results, and compare to the predicted to the actual we can see very little difference in our plots than what we had initially seen in part 3 of our project.



Conclusion

I am satisfied with the results of these test and parameter optimizations. Of course we cannot predict the price movements perfectly, but the above chart which plots the predicted vs the actual is considerably impressive and near to the actual movements. I believe that more feature engineering could be conducted which could possibly lead to a product which can be utilized and optimized through back testing. The framework is there for these models to be appropriately set up for an automated or production use case where, at the end of each day the price data can be fed into the model and the predictions can be made which can guide an individual to act. I believe an immediate next step would be to apply an investment simulation model for back testing based on these predictions and see how well it performs against the general market.