Artificial Intelligence in Finance Project

Project topic & Seed programs

The following project is studied: Explain and extend a case model in: Machine Learning and Data Science Blueprints for finance by Tatsat for the final project. The case model we will explain and extend is case study 1: Stock Price Prediction from Chapter 5 Supervised Learning: Regression.

The seed programs we used for our project is under the repository fin-ml/Chapter 5 - Sup. Learning - Regression and Time Series models/Case Study 1 - Stock Price Prediction/ from Github of Hariom Tatsat.

The link for this seed program: https://github.com/tatsath/fin-ml/blob/master/Chapter%205%20-%20Sup.%20Learning%20-%20Regression%20and%20Time%20Series%20models/Case%20Study%201%20-%20Stock%20Price%20Prediction/StockPricePrediction.ipynb

Sources of Data

The database we used to download our data is WRDS, the Wharton Research Data Services.

The link for the database:

https://wrds-www.wharton.upenn.edu/pages/get-data/nyse-trade-and-quote/millisecond-trade-and-quote-daily-product-2003-present-updated-daily/consolidated-trades/

We downloaded data for IBM, GOOGL, MSFT, ETF data for Dow Jones, S&P 500, VIX and currencies data for USD/JPY and GDP/USD. The period we used is from 2018/01/01 to 2018/12/31. A one year period provides enough data for our program to process.

We processed our data using WRDSHourlyDataProcessing.py, WRDSHourlyDataProcessing_OPEN.py,WRDSHourlyDataProcessing_HIGH.py and WRDSHourlyDataProcessing_LOW.py from lecture to obtain hourly data results.

We named our data csv IBM_result.csv, GOOGLE_result.csv, MSFT_result.csv, dia_result.csv, SPY_result.csv, vixy_result.csv, fxy_result.csv, fxb_result.csv, MSFT_OPEN.csv, MSFT_LOW.csv and MSFT_HIGH.csv.

List of improvements

1. Apply model to new data (frequency)
- We applied model to WRDS Hourly data including: GOOGL, IBM, MSFT open, MSFT close, MSFT high, MSFT low, ETF data for Dow Jones, S&P 500, VIX , currencies data for USD/JPY, currencies data GDP/USD.
2. Include new indicator inputs
- Talib features, smoothing features: wavelets, lags
3. Include more model parameter grids during cross validation
- Added SVR, SVC, PCA, wavelets parameter grids
4. Used scikit-learn and ta-lib (=parametrized features) move the feature parameter choice to a function inside the pipeline
5. Answered questions at the end of chapters in Tatsat's book
6. Replace a model with a more advanced version of the same model

Inputs and Target

Inputs:

- IBM
- GOOGL
- MSFT open
- MSFT close
- MSFT high
- MSFT low
- ETF data for Dow Jones
- S&P 500
- VIX
- Currencies data for USD/JPY
- Currencies data GDP/USD

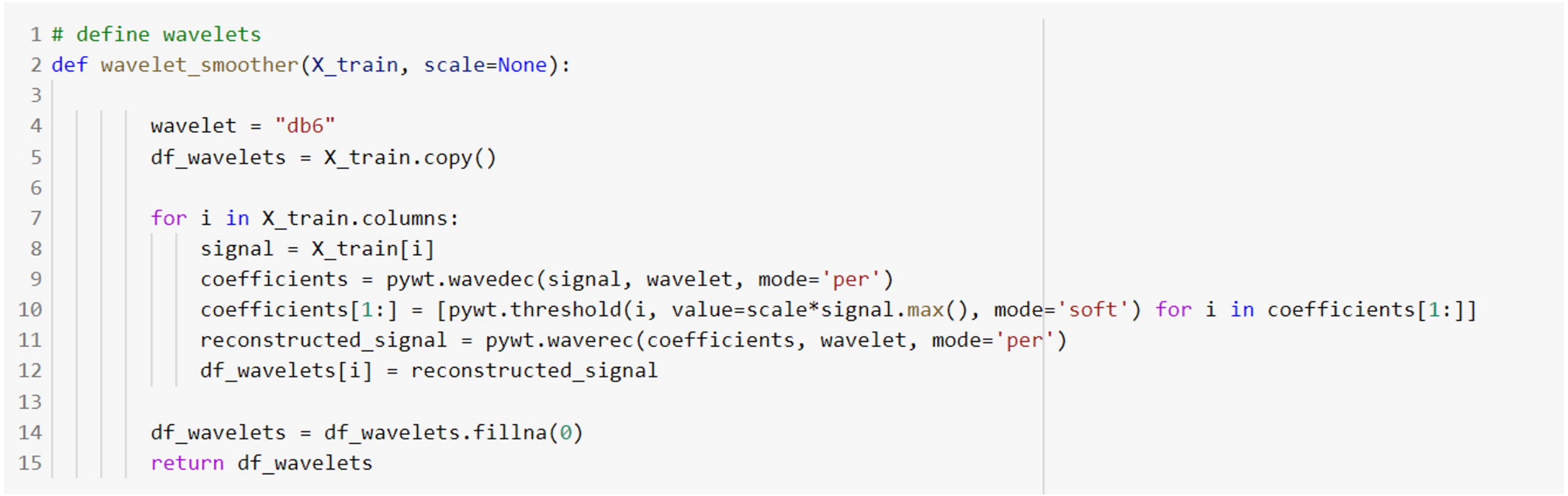
Target:

- MSFT future returns

We have downloaded the data from WRDS website: https://wrds-www.wharton.upenn.edu/pages/get-data/nyse-trade-and-quote/millisecond-trade-and-quote-daily-product-2003-present-updated-daily/consolidated-quotes/

The timeframe for the data is from 2018/01/01 to 2018/12/31, and the frequency is hourly.

Input Processing



```python
# define wavelets
def wavelet_smoother(X_train, scale=None):

    wavelet = "db6"
    df_wavelets = X_train.copy()

    for i in X_train.columns:
        signal = X_train[i]
        coefficients = pywt.wavedec(signal, wavelet, mode='per')
        coefficients[1:] = [pywt.threshold(i, value=scale*signal.max(), mode='soft') for i in coefficients[1:]]
        reconstructed_signal = pywt.waverec(coefficients, wavelet, mode='per')
        df_wavelets[i] = reconstructed_signal

    df_wavelets = df_wavelets.fillna(0)
    return df_wavelets
```

We have covered wavelets in the LinearSVC model in our program.

Input Selection or Extraction

- `pca=PCA()`
- `pipe = Pipeline([('scaler',scaler), ('pca', pca),('svc', svc)])`
- `ncomponents_rs = list(range(10,X_test.shape[1]))`
- `param_grid = [{'pca__n_components':ncomponents_rs, 'svc__C': c_rs}]`

We have used PCA for input selection and features extraction.

Models Used

The models used for this project include:

- Linear Regression
- Lasso
- Elastic Net
- KNeighborsRegressor
- DecisionTreeRegressor
- MLPregressor
- AdaBoostRegressor
- GradientBoostingRegressor
- RandomForestRegressor
- ExtraTreesRegressor
- LinearSVR
- LinearSVC

Model Parameters

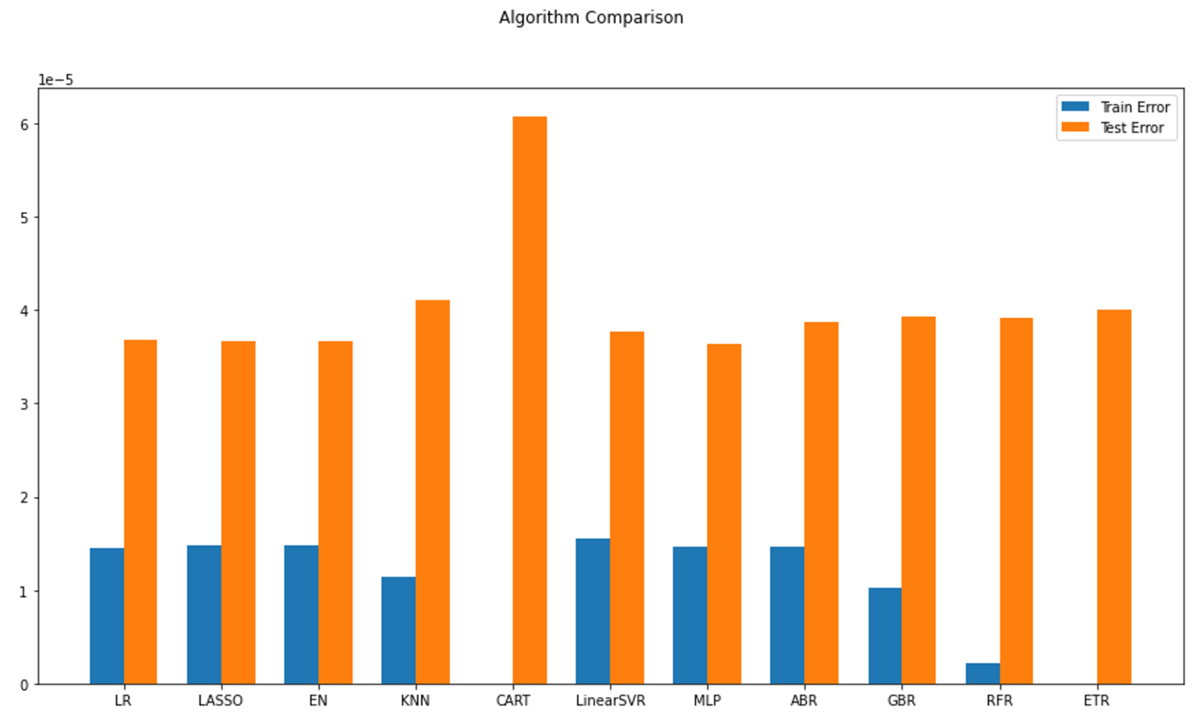
The model parameters for LinearSVR are:

```
param_grid_LinearSVR = [{'pca__n_components':ncomponents_rs_LinearSVR, 'svr__C': c_rs_LinearSVR,
'svr__epsilon': epsilon_rs_LinearSVR}]
```

The model parameters for LinearSVC are:

```
param_grid = [{'pca__n_components':ncomponents_rs, 'svc__C': c_rs}]
```

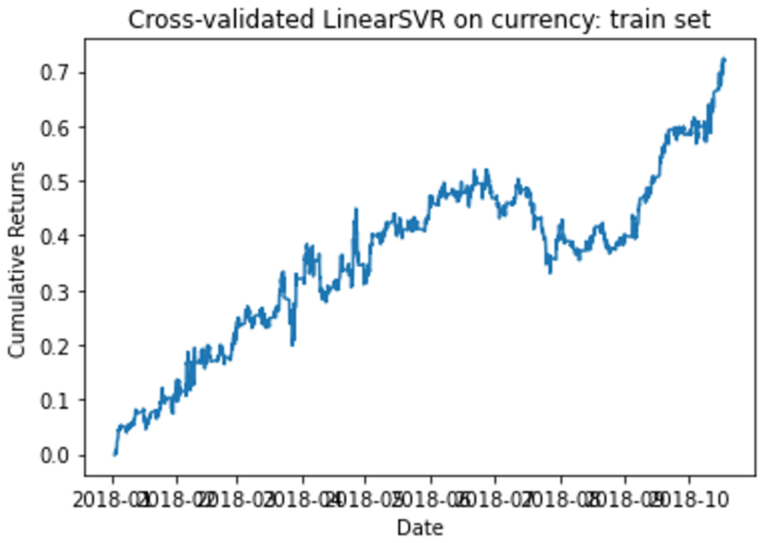
Model Performance

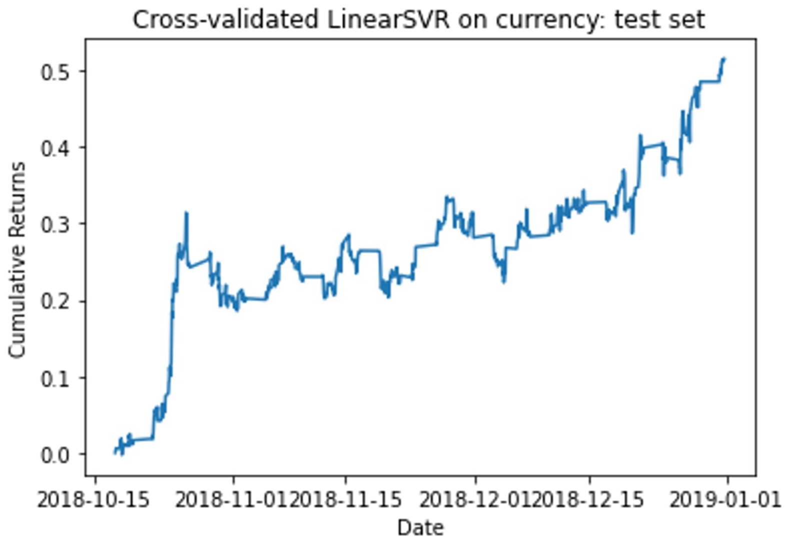


In the plot of Algorithm Comparison, the train error and test error of all the models are compared.

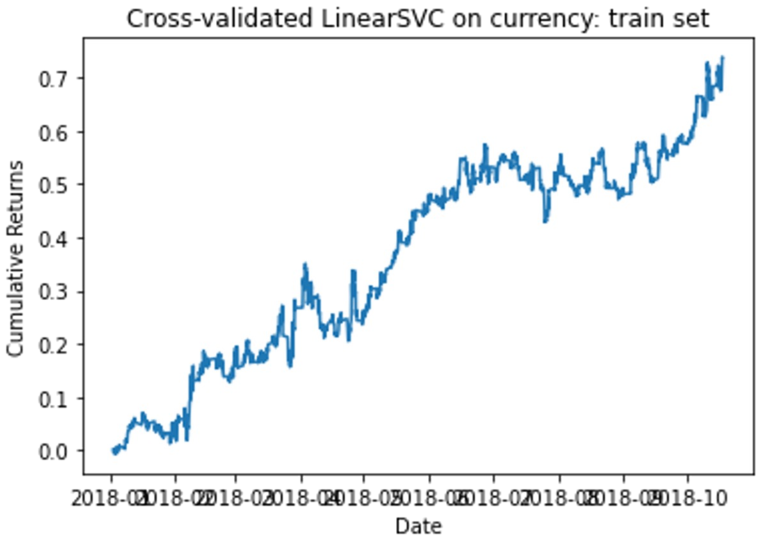
Examining the training and test error, we could see a stronger performance from the linear models. Among the linear models, linear SVM regression seems to perform best. Linear SVMs are also more robust (not as vulnerable to outliers) and faster trainers, so we select LinearSVR for further model tuning and compare it with LinearSVC, in order to make a comparison between regression and classification.

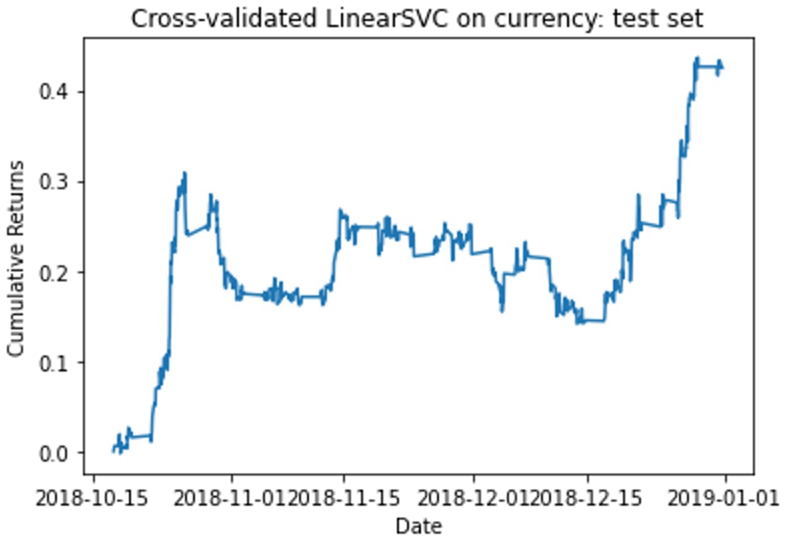
Equity Curves for LinearSVR



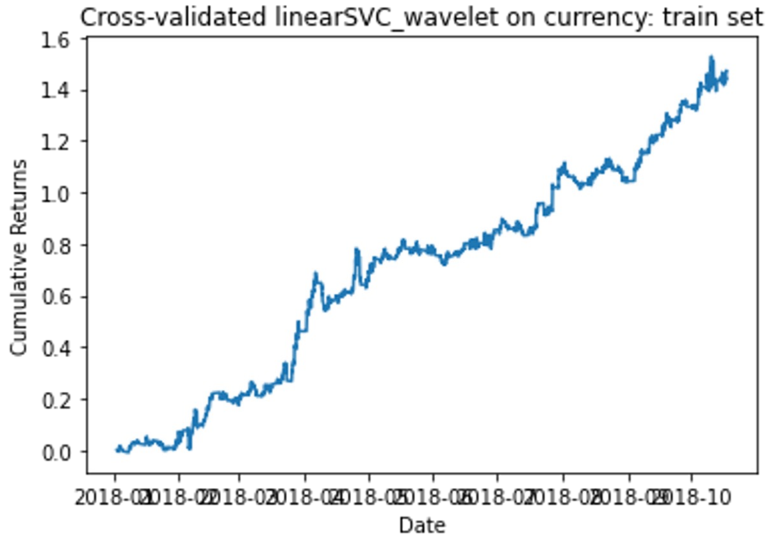


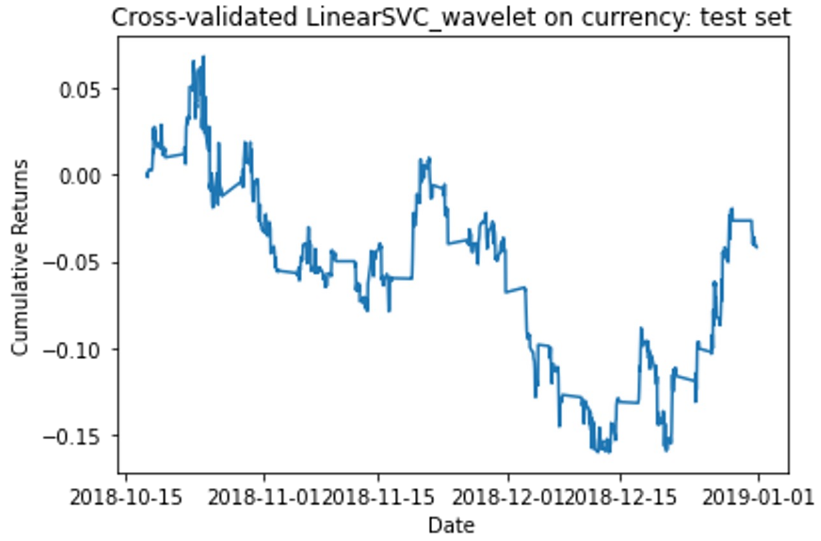
Equity Curves for LinearSVC





Equity Curves for LinearSVC with wavelets





Statistical & Financial Metrics

The financial metrics for LinearSVR are:

In-sample: CAGR=0.0410983 Sharpe ratio=0.691297 maxDD=-0.124226 maxDDD=938 Calmar ratio=0.330836

Out-of-sample: CAGR=0.131052 Sharpe ratio=1.33326 maxDD=-0.097679 maxDDD=355 Calmar ratio=1.34166 Rho=0.107173 PVal=0.00176489 MSE=3.64218e-05

The financial metrics for LinearSVC are:

In-sample: CAGR=0.041849 Sharpe ratio=0.703134 maxDD=-0.107667 maxDDD=845 Calmar ratio=0.388689

Out-of-sample: CAGR=0.110771 Sharpe ratio=1.14333 maxDD=-0.127992 maxDDD=678 Calmar ratio=0.865459 accuracy_score=0.542992

The financial metrics for LinearSVC with wavelets are:

In-sample: CAGR=0.0694158 Sharpe ratio=1.1333 maxDD=-0.0872723 maxDDD=352 Calmar ratio=0.795393

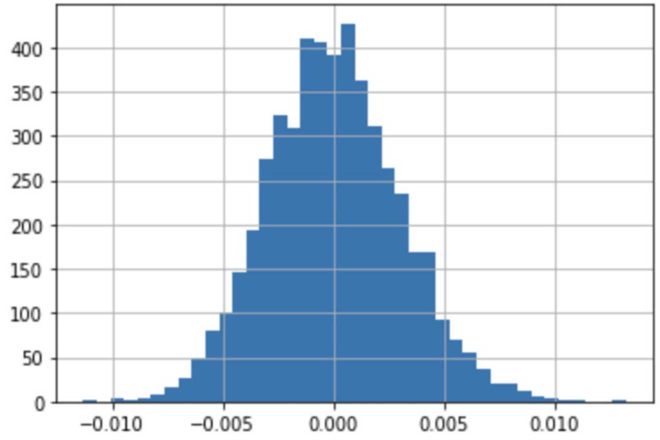
Out-of-sample: CAGR=-0.0126269 Sharpe ratio=-0.0840197 maxDD=-0.213068 maxDDD=764 Calmar ratio=-0.0592622 accuracy_score=0.504122

Statistical & Financial Metrics Comparisons

	LinearSVR	LinearSVC	LinearSVC w/ wavelets
CAGR	0.131052	0.110771	-0.0126269
Sharpe Ratio	1.33326	1.14333	-0.0840197
P-value	0.0114	0.0282	0.4310
If reject Ho	Reject Ho	Reject Ho	Do not reject Ho
Statistical metrics (accuracy/mse)	MSE=3.64218e-05	accuracy_score=0.542992	accuracy_score=0.504122

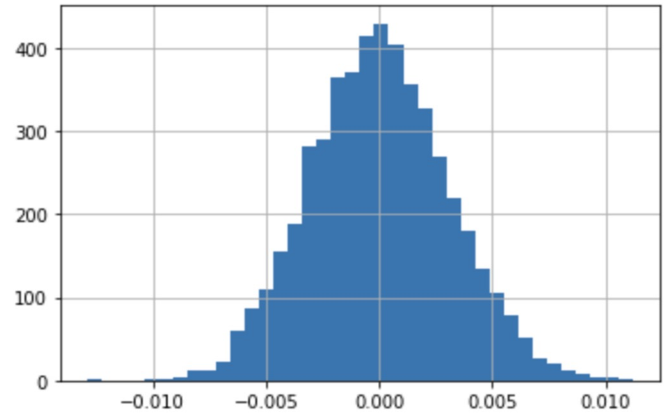
Final White Reality Check for LinearSVR

```
average return 0.007389
[-0.00567126  0.00629418]
Reject Ho = The population distribution of rule returns has an expected value of zero or less (because p_value is sma
ll enough)
p_value:
0.01139999999999966
```



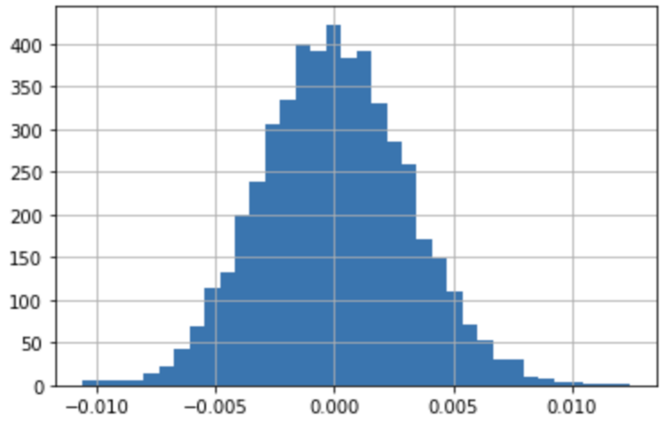
Final White Reality Check for LinearSVC

```
average return 0.005980
[-0.00580374  0.00615183]
Reject Ho = The population distribution of rule returns has an expected value of zero or less (because p_value is sma
ll enough)
p_value:
0.028200000000000003
```



Final White Reality Check for LinearSVC with wavelets

```
average return 0.000508
[-0.00579911  0.00615807]
Do not reject Ho = The population distribution of rule returns has an expected value of zero or less (because p_value
is not small enough)
p_value:
0.43100000000000005
```



White Reality Check Results Comparison

From the results returned for the White Reality Check, the LinearSVR model has an average return of 0.007389 while the LinearSVC model has an average return of 0.005980 and the LinearSVC with wavelets model has an average return of 0.000508.

The LinearSVR model has a p-value of 0.0114 while the LinearSVC model has a p-value of 0.0282 and the LinearSVC with wavelets has a p-value of 0.4310. The LinearSVR model's p-value is significantly smaller than the p-value of the LinearSVC model and the LinearSVC with wavelets. Therefore, we prefer to use the LinearSVR model.

Conclusion

From the results above, we can conclude that linear SVMs are promising modeling approaches for stock price prediction problems, and LinearSVR outperforms LinearSVC in this case. Also, we found that wavelets did not improve the performance of our linearSVC model. Therefore, we should not apply waveltes to our LinearSVMs model and finalize our best model as LinearSVR(C=0.0001, epsilon=0) with StandardScaler and PCA(n_components=12) applied.

Thanks for watching!