SOURCE: https://storage.googleapis.com/kaggle-datasets-images/1912819/3141205/d400432fd098ed79ec12db4d5d34414a/dataset-cover.jpeg?t=2022-02-05-05-11-16

Netflix Stock Price Prediction

MOTIVATION

- The Dataset contains data for 5 years ie. from 5th Feb 2018 to 5th Feb 2022
- The art of forecasting stock prices has been a difficult task for many of the researchers and analysts. In fact, investors are highly interested in the research area of stock price prediction. For a good and successful investment, many investors are keen on knowing the future situation of the stock market. Good and effective prediction systems for the stock market help traders, investors, and analyst by providing supportive information like the future direction of the stock market.

Source: https://www.kaggle.com/datasets/jainilcoder/netflix-stock-price-prediction

	Date	Open	High	Low	Close	Adj Close	Volume
0	2018-02-05	262.000000	267.899994	250.029999	254.259995	254.259995	11896100
1	2018-02-06	247.699997	266.700012	245.000000	265.720001	265.720001	12595800
2	2018-02-07	266.579987	272.450012	264.329987	264.559998	264.559998	8981500
3	2018-02-08	267.079987	267.619995	250.000000	250.100006	250.100006	9306700
4	2018-02-09	253.850006	255.800003	236.110001	249.470001	249.470001	16906900
5	2018-02-12	252.139999	259.149994	249.000000	257.950012	257.950012	8534900
6	2018-02-13	257.290009	261.410004	254.699997	258.269989	258.269989	6855200
7	2018-02-14	260.470001	269.880005	260.329987	266.000000	266.000000	10972000
8	2018-02-15	270.029999	280.500000	267.630005	280.269989	280.269989	10759700
9	2018-02-16	278.730011	281.959991	275.690002	278.519989	278.519989	8312400

TARGET VARIABLE IS 'CLOSE' .

The feature "close" in Netflix stock price prediction refers to the closing price of Netflix stock on a given day.

METHODOLOGY

- PREPOCESSING STAGE:
- As was proposed in Pearson Correlation Coefficient-Based Performance Enhancement of Broad Learning System for Stock Price Prediction by Guanzhi Li, Aining Zhang, Qizhi Zhang, Di Wu , and Choujun Zhan , Member, IEEE we choose for prepocessing min-max scaler:

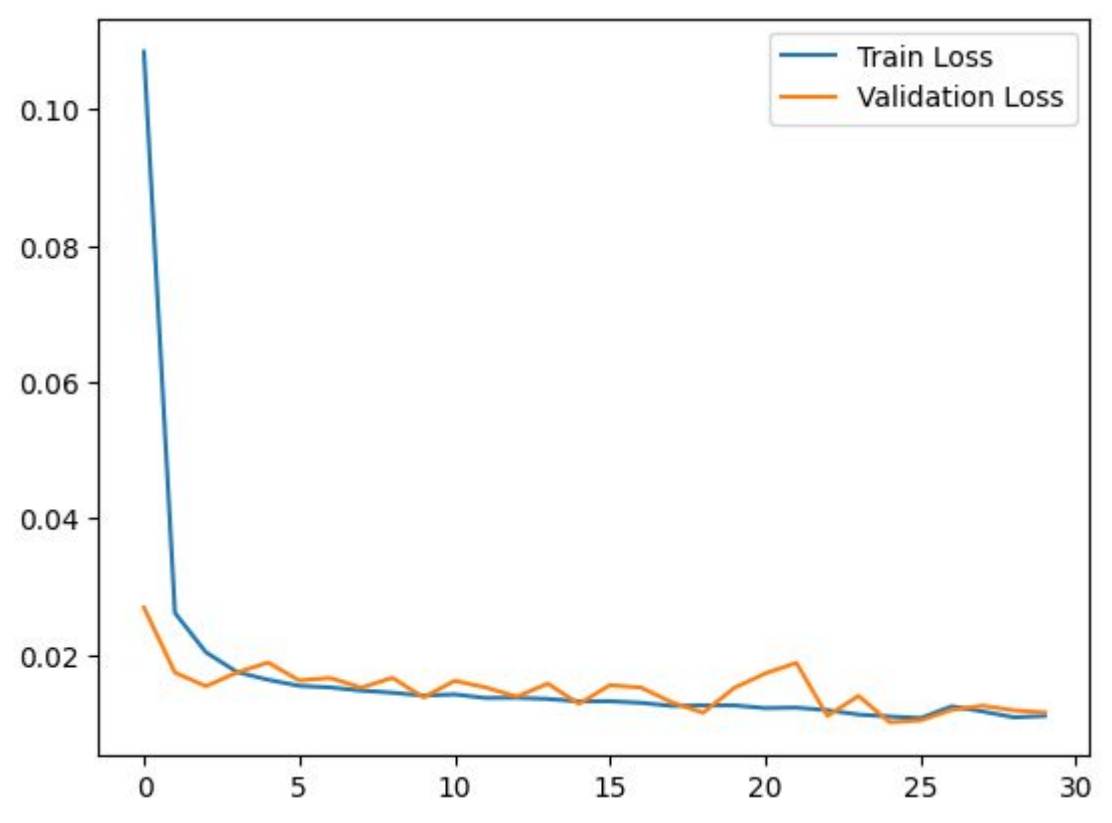
$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Source:https://miro.medium.com/v2/resize:fit:720/format:webp/1*_XNKwOmeAwYOCffKFVhrDg.png

METHODOLOGY

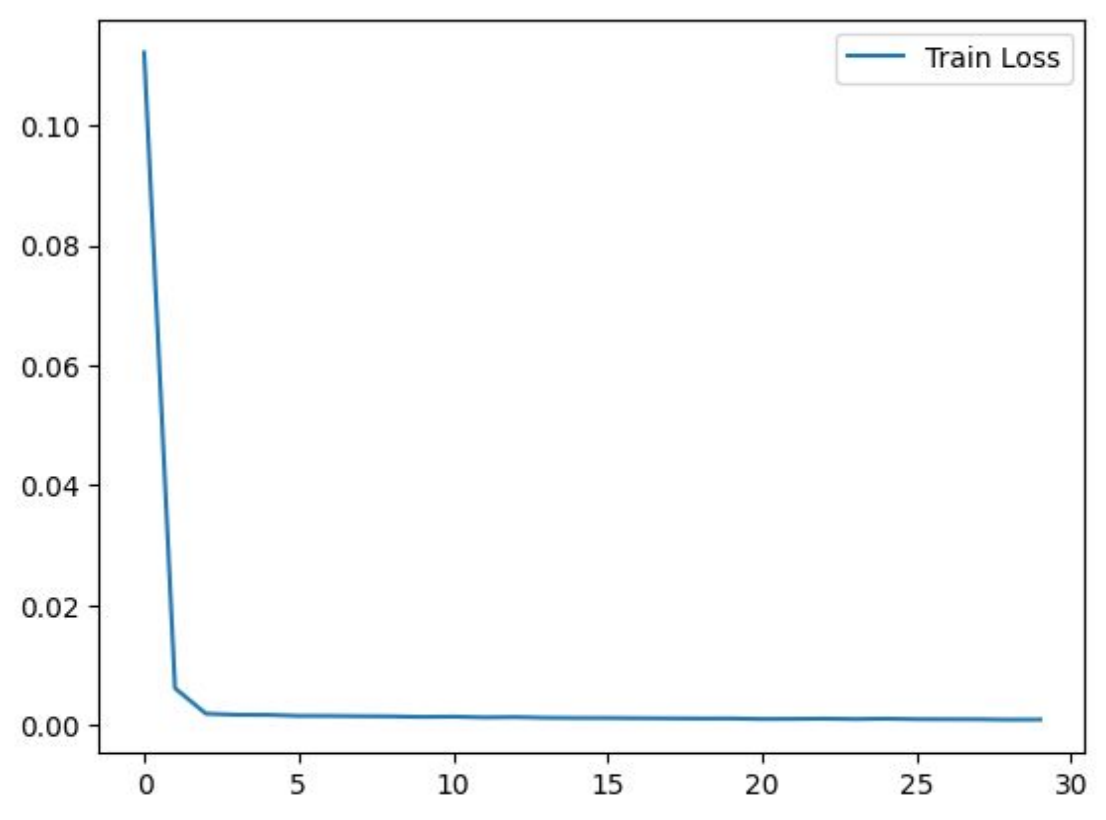
- LEARNING STAGE:
- We use previous values of the target variable to predict the next and using this data to fit neural networks is an effective approach to time series prediction.
- The motivation for using this approach is that it can capture temporal autocorrelation. Temporal autocorrelation is the idea that the value of a variable at a given time is correlated with its values at previous times. This is often true for time series data, such as stock prices, sales, and weather patterns. By using previous values of the target variable as features, neural networks can learn to capture this temporal autocorrelation and make more accurate predictions.
- We applied it on the next of neural networks: Simple RNN, RNN, LSTM and GRU.

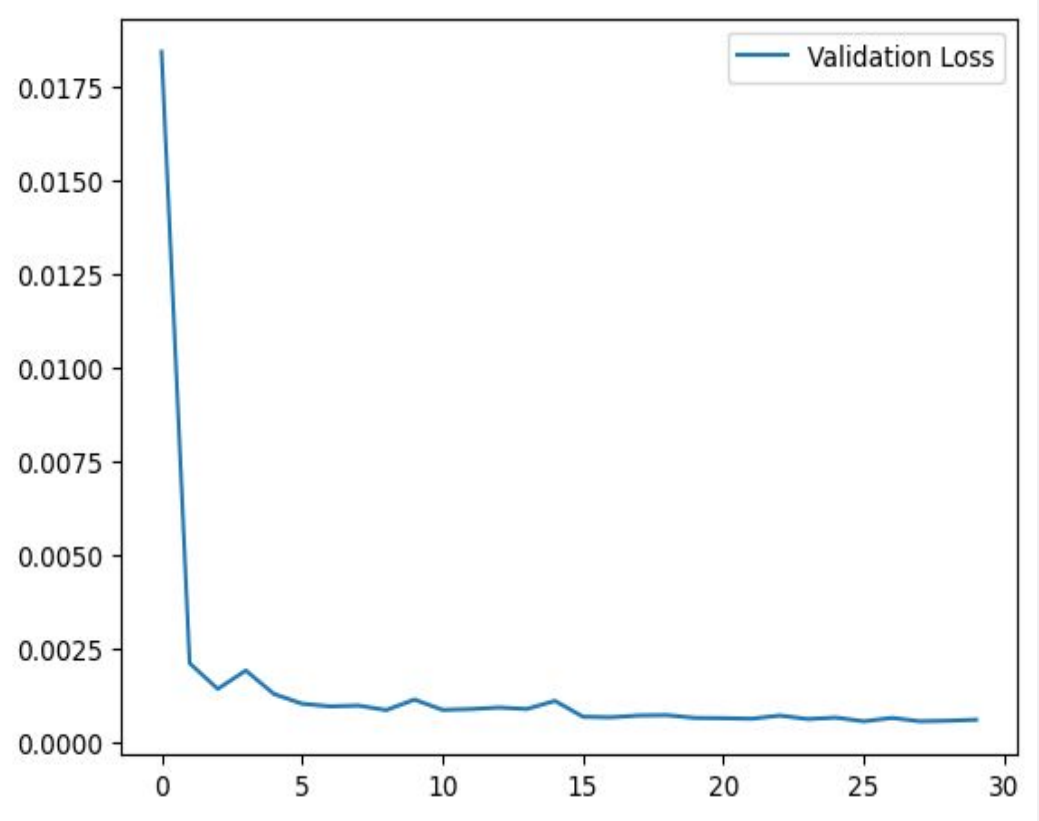
LSTM



R^2 score on test value from LSTM model 0.873 (R2^2 was calculated on inverse transformed data)

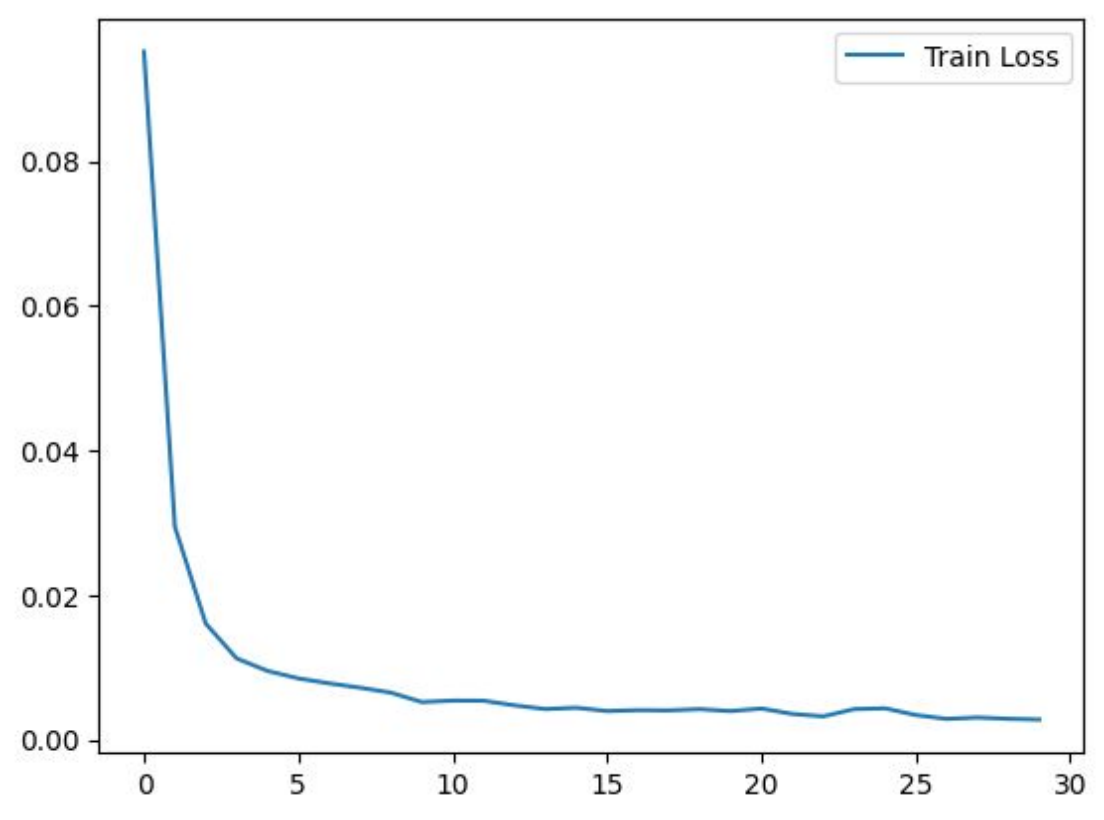
SIMPLE RNN

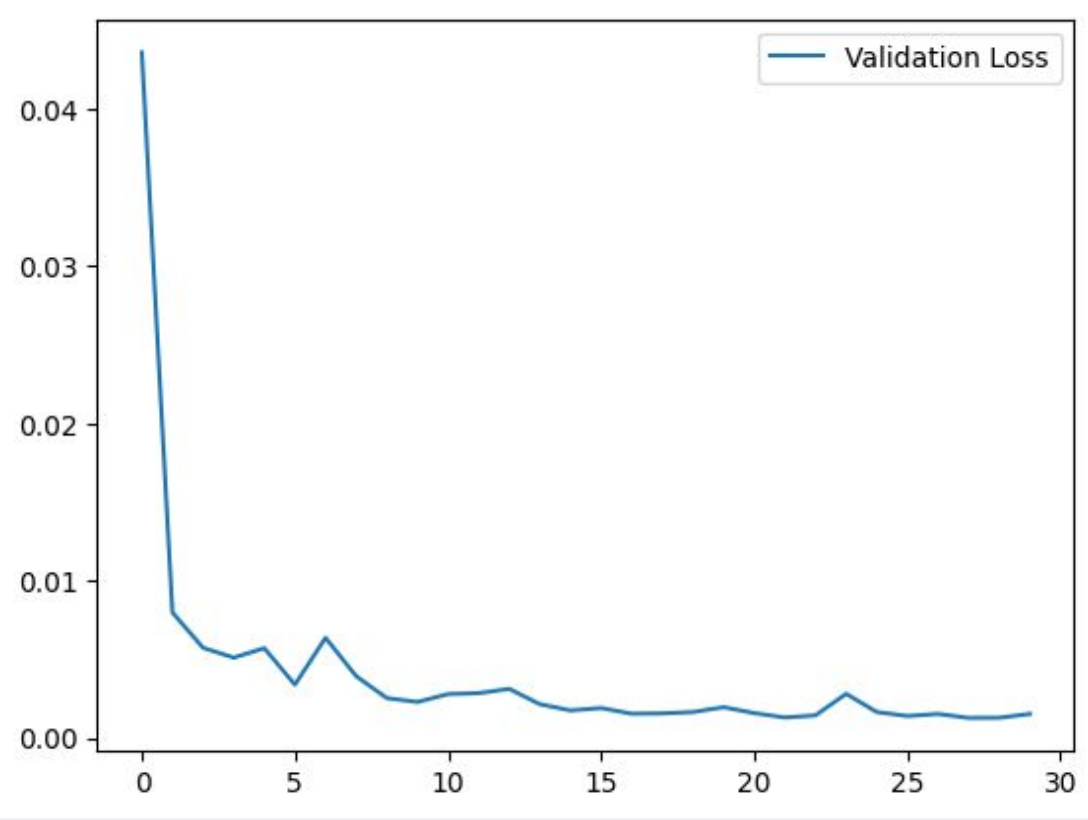




R^2 score on test value from Simple RNN model 0.9799. (R2^2 was calculated on inverse transformed data)

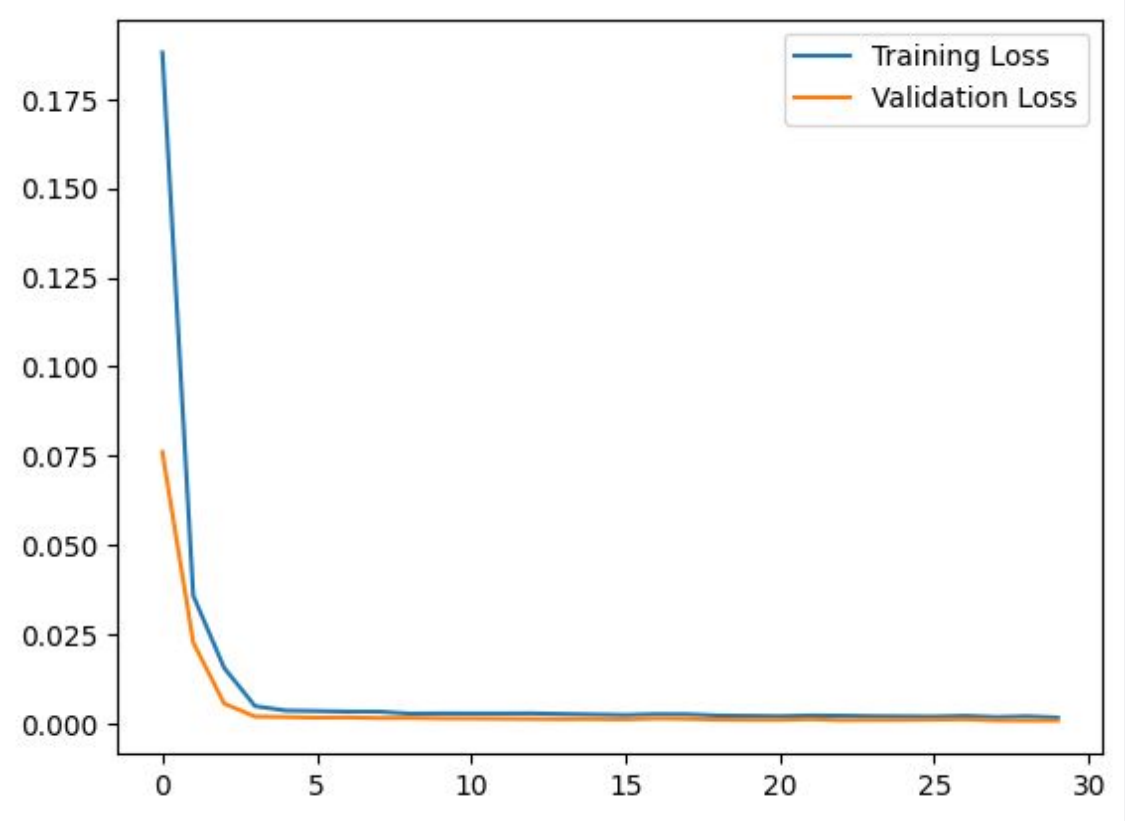
RNN





R^2 score on test value from RNN model 0.958. (R2^2 was calculated on inverse transformed data)

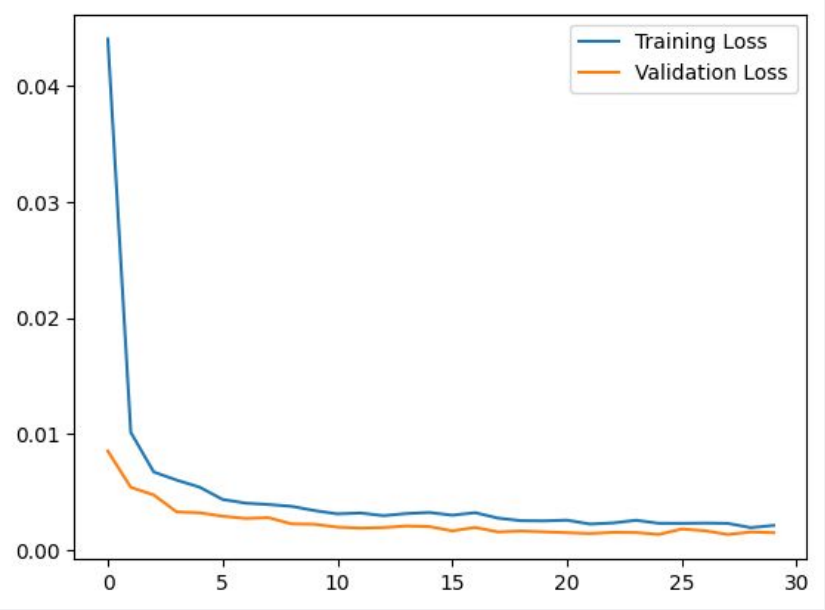
GRU



R^2 score on test value from GRU model 0.967 (R2^2 was calculated on inverse transformed data)

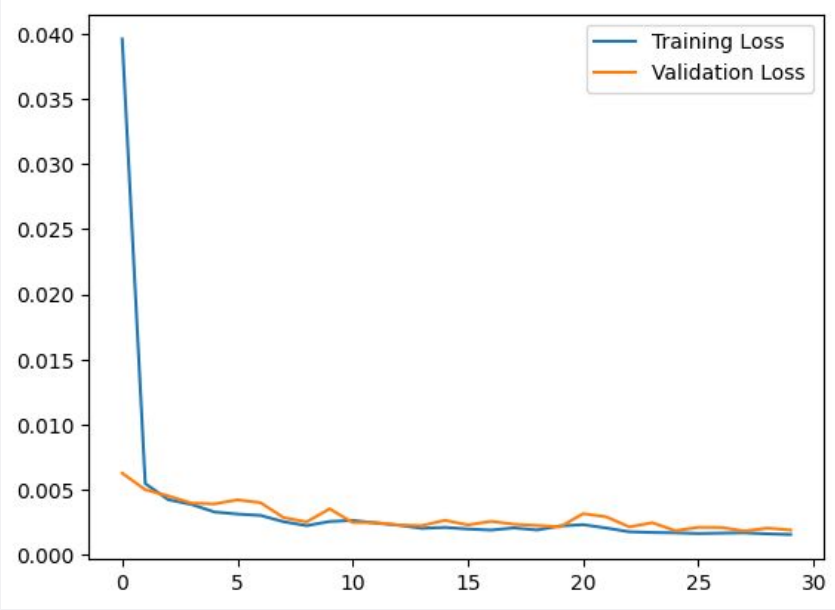
BiDirectional LSTM

R^2 = .97



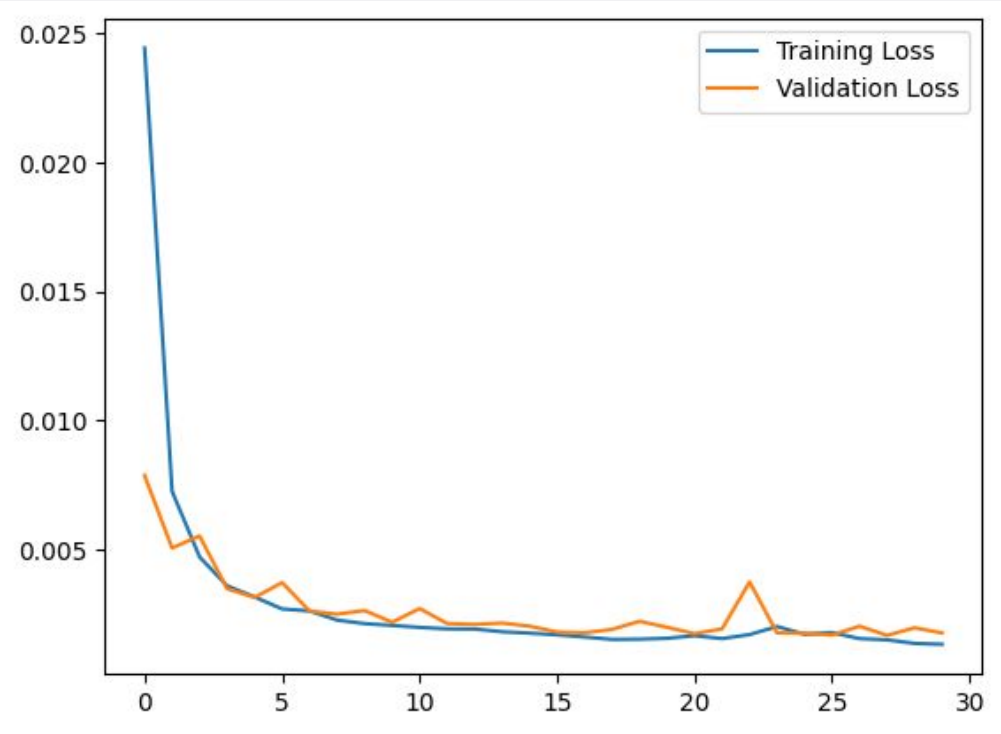
LSTM w/ Attention

R^2 = .962



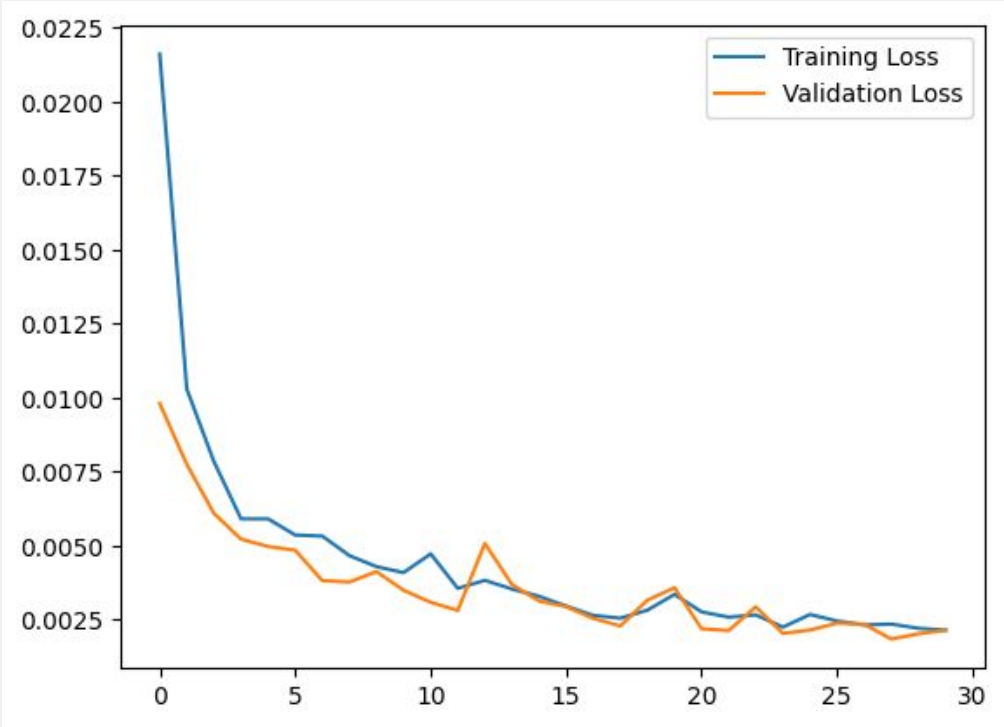
CNN

R^2 = .964



LSTM + CNN Hybrid

R^2 = .96



Existing Models

- We used two existing libraries to run predictions without training. These are:
- Pycaret
  - Supports a wide range of forecasting methods such as ARIMA, and LSTM. Suitable for univariate and multivariate time series forecasting tasks.
- Nixtla
  - boasts fast execution speed, but is still in development and has limited functionality

Pycaret

	Model	MASE	RMSSE	MAE	RMSE	MAPE	SMAPE	R2	TT (Sec)
ets	ETS	2.4104	1.8608	33.0018	47.6400	0.0611	0.0618	0.6965	0.4400
arima	ARIMA	2.4021	1.8573	32.8644	47.4836	0.0607	0.0612	0.6952	0.3000
exp_smooth	Exponential Smoothing	2.4206	1.8769	33.1468	48.0698	0.0614	0.0622	0.6919	0.3733
snaive	Seasonal Naive Forecaster	2.5380	1.8902	34.7659	48.4927	0.0647	0.0662	0.6874	0.1500
stlf	STLF	2.5089	1.9063	34.3154	48.6909	0.0640	0.0643	0.6780	0.2067
huber_cds_dt	Huber w/ Cond. Deseasonalize & Detrending	2.9502	2.0394	40.4113	52.3085	0.0782	0.0798	0.6370	2.8600
knn_cds_dt	K Neighbors w/ Cond. Deseasonalize & Detrending	2.9963	2.0628	40.9921	52.8575	0.0765	0.0797	0.6264	4.2367
omp_cds_dt	Orthogonal Matching Pursuit w/ Cond. Deseasonalize & Detrending	3.4360	2.2459	47.1145	57.7498	0.0934	0.0952	0.5546	2.8867
rf_cds_dt	Random Forest w/ Cond. Deseasonalize & Detrending	3.5294	2.3614	48.4333	60.7299	0.0940	0.0967	0.4881	5.7233
ridge_cds_dt	Ridge w/ Cond. Deseasonalize & Detrending	3.7527	2.4474	51.4294	62.7661	0.1049	0.1055	0.4754	3.0800
lr_cds_dt	Linear w/ Cond. Deseasonalize & Detrending	3.7527	2.4474	51.4294	62.7661	0.1049	0.1055	0.4754	2.8600
en_cds_dt	Elastic Net w/ Cond. Deseasonalize & Detrending	3.7560	2.4488	51.4749	62.8007	0.1050	0.1056	0.4748	3.1133
lasso_cds_dt	Lasso w/ Cond. Deseasonalize & Detrending	3.7573	2.4491	51.4921	62.8104	0.1050	0.1057	0.4746	2.7433
llar_cds_dt	Lasso Least Angular Regressor w/ Cond. Deseasonalize & Detrending	3.7573	2.4491	51.4920	62.8105	0.1050	0.1057	0.4746	2.9900
br_cds_dt	Bayesian Ridge w/ Cond. Deseasonalize & Detrending	3.7643	2.4529	51.5888	62.9075	0.1053	0.1059	0.4730	3.0467
gbr_cds_dt	Gradient Boosting w/ Cond. Deseasonalize & Detrending	3.5513	2.4543	48.6875	63.0107	0.0972	0.0987	0.4673	3.1900
auto_arima	Auto ARIMA	3.5588	2.4409	48.7994	62.6364	0.0944	0.0984	0.4436	353.1267
xgboost_cds_dt	Extreme Gradient Boosting w/ Cond. Deseasonalize & Detrending	4.1006	2.7063	56.2242	69.5382	0.1124	0.1149	0.3530	4.5500
et_cds_dt	Extra Trees w/ Cond. Deseasonalize & Detrending	4.2337	2.7162	58.0339	69.6923	0.1186	0.1193	0.3494	5.6133
lightgbm_cds_dt	Light Gradient Boosting w/ Cond. Deseasonalize & Detrending	4.4420	2.7630	60.8585	70.7564	0.1274	0.1261	0.3233	3.7333
theta	Theta Forecaster	5.5323	3.3034	75.7583	84.4589	0.1594	0.1575	0.0308	0.1800
ada_cds_dt	AdaBoost w/ Cond. Deseasonalize & Detrending	5.8063	3.3208	79.4435	84.8199	0.1574	0.1647	0.0179	3.6000
croston	Croston	6.1236	3.5834	83.8722	91.6810	0.1750	0.1746	-0.1409	0.1167
polytrend	Polynomial Trend Forecaster	6.3452	3.5798	86.8005	91.3770	0.1793	0.1811	-0.1532	0.0700
dt_cds_dt	Decision Tree w/ Cond. Deseasonalize & Detrending	5.9894	4.1113	81.8737	104.3018	0.1704	0.1735	-0.7303	3.0767
naive	Naive Forecaster	7.3475	4.9686	100.5897	127.1524	0.2034	0.2105	-1.1630	0.1167
grand_means	Grand Means Forecaster	8.7640	5.6017	119.9151	142.8801	0.2181	0.2576	-1.8708	0.0900

Using median for filling the missing values for the numerica target imputation and calculation the pycaret models

	Test	Test Name	Data	Property	Setting	Value
0	Summary	Statistics	Transformed	Length		1461.0
1	Summary	Statistics	Transformed	# Missing Values		0.0
2	Summary	Statistics	Transformed	Mean		406.52333
3	Summary	Statistics	Transformed	Median		378.670013
4	Summary	Statistics	Transformed	Standard Deviation		91.891424
5	Summary	Statistics	Transformed	Variance		8444.033742
6	Summary	Statistics	Transformed	Kurtosis		0.232967
7	Summary	Statistics	Transformed	Skewness		0.901258
8	Summary	Statistics	Transformed	# Distinct Values		988.0
9	White Noise	Ljung-Box	Transformed	Test Statictic	{'alpha': 0.05, 'K': 24}	14627.906285
10	White Noise	Ljung-Box	Transformed	Test Statictic	{'alpha': 0.05, 'K': 48}	27265.904496
11	White Noise	Ljung-Box	Transformed	p-value	{'alpha': 0.05, 'K': 24}	0.0
12	White Noise	Ljung-Box	Transformed	p-value	{'alpha': 0.05, 'K': 48}	0.0
13	White Noise	Ljung-Box	Transformed	White Noise	{'alpha': 0.05, 'K': 24}	False
14	White Noise	Ljung-Box	Transformed	White Noise	{'alpha': 0.05, 'K': 48}	False
15	Stationarity	ADF	Transformed	Stationarity	{'alpha': 0.05}	False
16	Stationarity	ADF	Transformed	p-value	{'alpha': 0.05}	0.42757
17	Stationarity	ADF	Transformed	Test Statistic	{'alpha': 0.05}	-1.706883

Nixtla

	ds	cutoff	y	AutoARIMA
0	2021-03-07	2021-02-28	663.840027	664.247559
1	2021-03-14	2021-02-28	641.900024	665.084473
2	2021-03-21	2021-02-28	617.770020	666.858887
3	2021-03-28	2021-02-28	616.469971	666.574036
4	2021-04-04	2021-02-28	602.130005	667.061768

unique_id	AutoARIMA	SeasonalNaive	best_model
NETFLIX	12407.507812	13180.722656	AutoARIMA