Torch Trade

딥러닝을 이용한 코인 예측

https://github.com/DogRing/TorchTrade

데이터 수집

upbit API를 이용 2018-09-19 23:50 ~ 2022-04-23 02:01 기간의 분당 데이터를 수집했다.

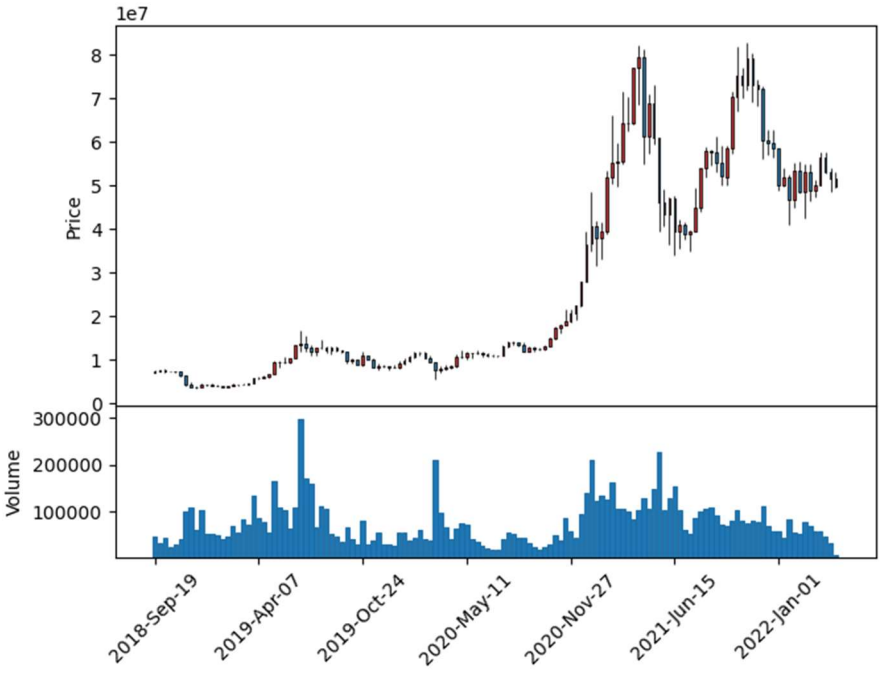


그림 1 10일 간격 데이터

결측값은 총데이터의 수는 결측값을 제외하고 1867800이며, 결측값은 총 20172개이다.

데이터 전처리

데이터

데이터의 결측값은 전체 데이터의 1.068%로, 크게 영향을 주지 않는다고 판단하여 선형 보간법을 사용하여 채워 넣었다.

전체 데이터는 ohlcv와 총 가치인 value로 나타나 있다. value의 경우 volume에 의존성이 높다고 판단하여 제외하였으며, 나머지 특징들은 전부 사용한다.

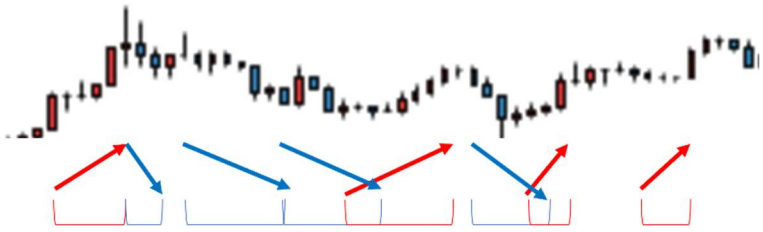
예측 결괏값

주가 예측에서 지도학습을 위한 결괏값은 대부분 다음 가격을 구하는 데에 초점을 두고있다. 랜덤성이 높은 지표는 딥러닝을 이용하여 예측하기가 매우 어렵다. 여러 번 시도하였을 경우에도 잘 되지 않았으며, 이미 많은 논문들이 나와 있다. 특히 실시간으로 바뀌는 시장에서 긴 단위의 예측은 더욱 어렵다.

해당 프로젝트는 기계거래의 장점을 살리기 위해 긴 단위의 예측보다는 짧은 단위의 예측을 필요로 하며, 거래를 위해서는 해당 지점에서 올라갈 지 내려갈 지만 확인하면 된다고 생각했다.

때문에 목표 변화량을 정하고, 해당 변화량까지 도달하는 기간을 예측값으로 정하였다.

예를 들어 목표 변화량이 5% 라면 현 시점에서 5%의 증가나 감소가 나타난 지점까지의 거리를 구하였다.



해당하는 기간이 클수록 완만하게 변동하며, 해당하는 기간이 작을수록 크게 변동한다.

정규화

데이터의 경우 원본 데이터와 15분, 60분, 1일로 총 4개의 데이터로 샘플링한다.

각 데이터들을 해당 값으로 정규화하지 않고, 전 데이터의 종가에 따른 변화량으로 정규화를 시행한다. 전날에 비해서 50% 올랐으면 0.5로 정규화한다.

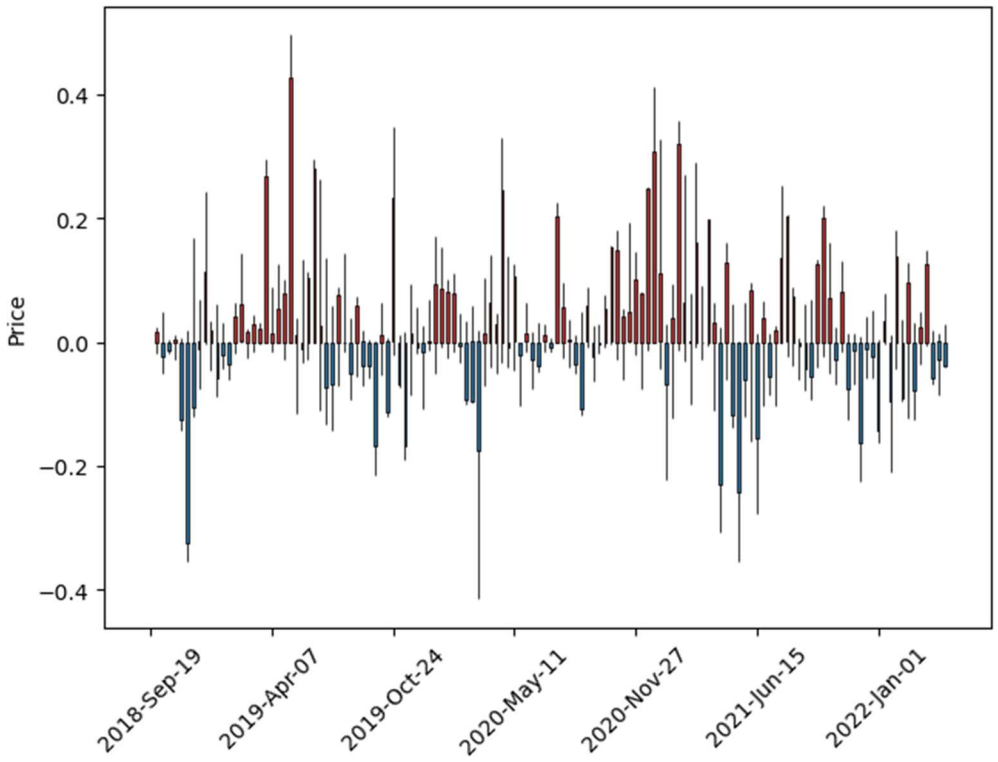


그림 2 10일 간격 데이터의 정규화

해당 데이터들을 모두 사용하며, 각 데이터의 길이는 50개로 하여, 모델의 input은 [4,50,5]의 형태를 가진다.

Ohlc의 경우 음수값이 가치가 있다고 판단 max abs scaling을 시행하였다.

결괏값의 경우 0을 기점으로 0에 가까울수록 변화량이 크지만 정반대의 결과이기에 역수를 취하였다.

결과 데이터는 −0.5~0.5까지 나타나며 이또한 max abd scaling을 시행하였다.

Model 1

단변량 시계열 데이터(etth)에서 SOTA를 기록했던 SCINet을 참고하여 제작하였다.

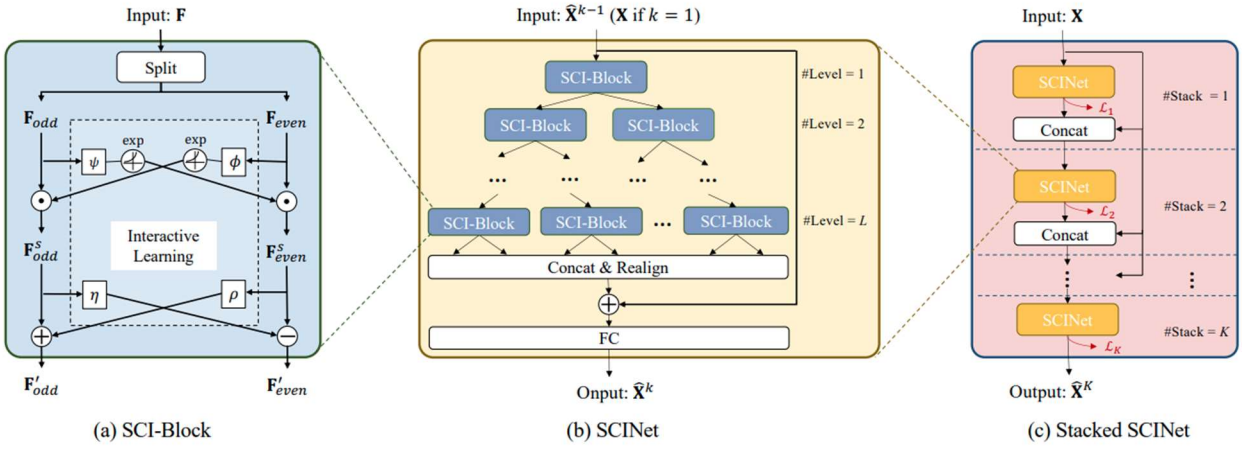


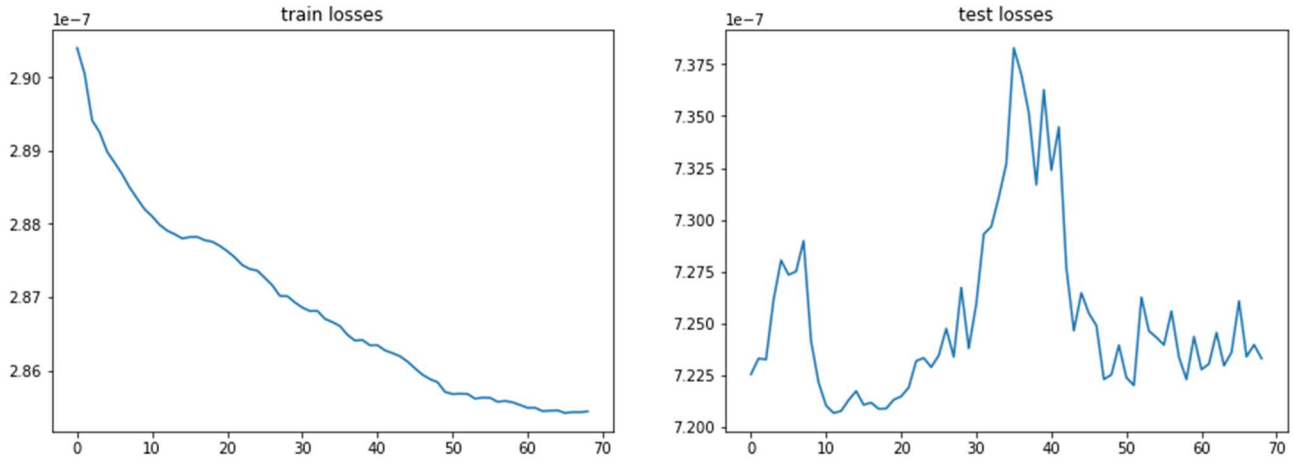
Figure 2: The overall architecture of Sample Convolution and Interaction Network (SCINet).

총 데이터 1887902개 중에 262144개를 테스트 데이터로 하였으며, batch size는 train 512, test 128개로 하였다.

입력 데이터는 64개의 종가 데이터를 입력으로 하였고, 다음 5개의 종가 데이터를 예측하게 **결과 데이터**를 설정하였다.

학습환경의 한계로 SCINet의 Level은 4, Stack은 1로 설정하여 총 70번의 학습을 수행하였으며, Loss 와 Optimizer함수는 현재 가장 일반적인 MSELoss와 AdamW를 사용하였다.

학습 (epoch 당 약 7.5분 소요)



Train 데이터의 loss가 계속해서 줄어들고 있지만 test loss가 진동하는 것으로 보아 데이터의 랜덤성을 모델이 학습하지 못하는 것으로 보인다.

데이터의 크기가 많아 train 데이터만 학습되어도 유의미한 결과가 나타날 것으로 생각하여 loss가 수렴할 때까지 더 해본다.

Model 2

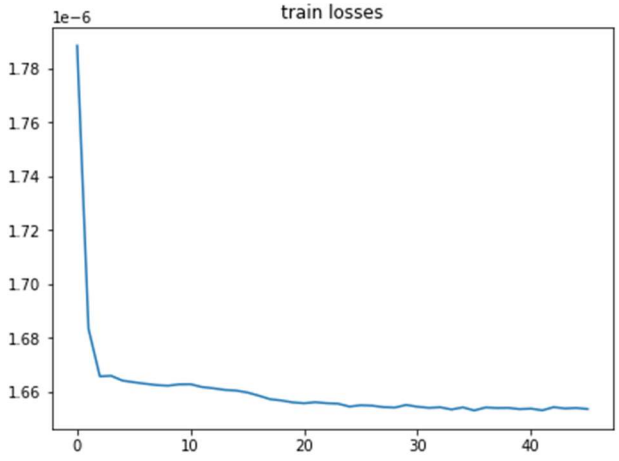
위의 모델과 같은 SCINet을 사용하지만 다음 데이터를 예측하는 것이 아닌 현재 시점에서 특정 변동률이 나타난 기간을 예측한다.

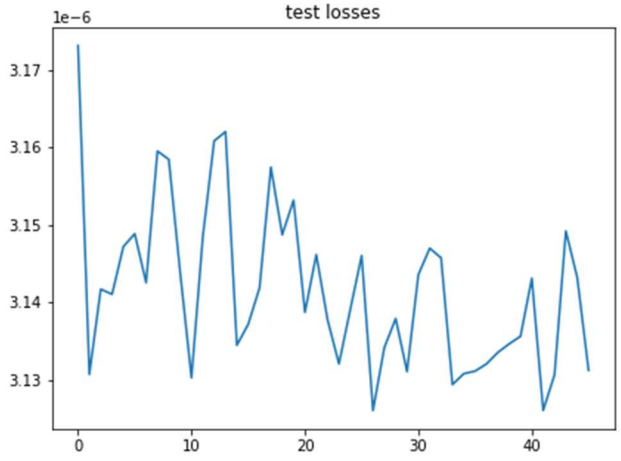
하락하면 음수, 상승하면 양수로 나타낸다. 0에 가까울수록 기울기가 급격하며 0에서 멀어질수록 기울기가 완만하다.

이를 역수를 취하여 0에서 멀어질수록 급격한 변동이 나타나게 표현하고, 0~1로 정규화 하였다.

마찬가지로 SCINet의 Level은 4, Stack은 1로 설정하여 총 50번의 학습을 수행하였으며, Loss와 Optimizer함수는 MSELoss와 AdamW를 사용하였다.

학습 (epoch당 약 7.61분 소요)





앞선 모델과 마찬가지로 train의 경우 지속적으로 줄어들고 있지만 test의 경우 진동하면서 줄어드는 경향이 나타난다.

하지만 계속해서 줄어드는 것으로 보이므로 계속해서 최적화 하였을 경우 기대해볼 여지가 있다.

모델의 효용성 확인

다른 코인(“ETC”)의 데이터셋을 이용하여 검증하였다. 총 2407653개의 분 데이터로 이루어져 있다.

간단하게 모델 결과값이 0보다 크고 구매한 것이 없다면 1개 구매하고,
모델 결과값이 0보다 작고 구매한 것이 있으면 1개를 판매한다.

총 2523번의 판매와 구매가 이루어 졌으며, 약 1028254.485 원의 이득이 나타났다.
전체 평균 코인의 가격은 22426.599 원으로 1개씩 거래하였다고 하였을 때 약 45.8배의 수익률을 나타낸다.

전체 데이터 셋에서 코인의 시작가는 14300 원이였고, 마지막은 42660 원으로 3배 오른 것에 비하면 많은 수익률을 낸 것을 알 수 있다.

사실 확인

전체 데이터와 구매, 판매의 기록을 같이 확인하여 제대로 예측을 하였는지 확인해 보자.

해당 그림을 확인해 보면 눈에 띄게 예측을 잘 하고 있지는 않다. 하강에서도 계속 하강하는 것이 보이며, 그저 랜덤보다 더 나은 수익을 내고 있는 것으로 보인다.

개선요소 : 더 짧은 예측

코드 : https://github.com/DogRing/TorchTrade