Coin Data를 이용한 선형관계분석

2018314020
오승엽

데이터

특징

<소셜 감성 데이터를 이용한 딥러닝 기반 비트코인 가격 예측 모델 (고광종 외., 2021)>연구에서 사용된 데이터 셋
2020-05-15 00:00 ~ 2021-05-31 23:00까지의 각종 데이터가 담겨 있음

close: Bitcoin 시간봉 종가
dollar: 해당 시간대의 달러 지수
oil: 해당 시간대의 서부 텍사스 유가
AMD: 해당 시간대의 그래픽 카드 제조 업체 AMD 사의 주가
NVDA: 해당 시간대의 그래픽 카드 제조 업체 NVIDIA 사의 주가
gal_p: 해당 시간대의 디씨인사이드 비트코인갤러리의 비트코인 관련 게시글 중 긍정 게시글 의 양
gal_n: 해당 시간대의 디씨인사이드 비트코인갤러리의 비트코인 관련 게시글 중 부정 게시글 의 양
gal_t: 해당 시간대의 디씨인사이드 비트코인갤러리의 비트코인 관련 게시글 의 양
twt_p: 해당 시간대의 트위터의 비트코인 관련 게시글 중 긍정 게시글 의 양
twt_n: 해당 시간대의 트위터의 비트코인 관련 게시글 중 부정 게시글 의 양
twt_t: 해당 시간대의 트위터의 비트코인 관련 게시글 의 양
dec_p: 해당 시간대의 디센터의 비트코인 관련 뉴스 중 긍정 뉴스 의 양
dec_n: 해당 시간대의 디센터의 비트코인 관련 뉴스 중 부정 뉴스 의 양
dec_t: 해당 시간대의 디센터의 비트코인 관련 뉴스의 양

Task

1. 긍정적인 커뮤니티 게시글이 시간봉 종가에 미치는 영향
2. 상관계수 상위 2컬럼으로 선형관계 예측

데이터 불러오기

```
data=pd.read_excel('D:/data_set/coindata_set.xlsx',index_col=0)
data_set = data.iloc[:,0:14] # data를 불러 왔을 때 지정하지 않은 컬럼까지 가져와서 기입된 컬럼까지만 가져오기
data_set.head()
```

	close	dollar	oil	AMD	NVDA	gal_p	gal_n	gal_t	twt_p	twt_n	twt_t	dec_p	dec_n	dec_t
2020-05-15 00:00:00	11744000	100.400002	31.33	54.200001	339.630005	0	0	0	0	0	0	0	0	0
2020-05-15 01:00:00	11676000	100.400002	31.33	54.200001	339.630005	0	0	0	0	0	0	0	0	0
2020-05-15 02:00:00	11616000	100.400002	31.33	54.200001	339.630005	0	0	0	1	0	1	0	0	0
2020-05-15 03:00:00	11658000	100.400002	31.33	54.200001	339.630005	0	0	0	0	0	0	0	0	0
2020-05-15 04:00:00	11635000	100.400002	31.33	54.200001	339.630005	0	0	0	0	0	0	0	0	0

1. head()로 데이터 형태 확인
2. iloc를 통해 쓸 데 없는 컬럼 없이 가져오기

데이터 정보 및 기술통계 확인

```
# 데이터 확인
data_set.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 9168 entries, 2020-05-15 00:00:00
Data columns (total 14 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   close   9168 non-null   int64
 1   dollar  9168 non-null   float64
 2   oil     9168 non-null   float64
 3   AMD     9168 non-null   float64
 4   NVDA    9168 non-null   float64
 5   gal_p   9168 non-null   int64
 6   gal_n   9168 non-null   int64
 7   gal_t   9168 non-null   int64
 8   twt_p   9168 non-null   int64
 9   twt_n   9168 non-null   int64
 10  twt_t   9168 non-null   int64
 11  dec_p   9168 non-null   int64
 12  dec_n   9168 non-null   int64
```

```
# 기술통계 확인
data_set.describe(include='all')
```

	close	dollar	oil	AMD	NVDA
count	9.168000e+03	9168.000000	9168.000000	9168.000000	9168.000000
mean	3.117141e+07	92.876885	49.314215	78.245550	506.110079
std	2.236326e+07	2.546071	10.467773	12.728450	75.639388
min	1.053700e+07	89.440002	31.330000	50.099998	339.480011
25%	1.247375e+07	90.779999	40.910000	76.220001	476.519989
50%	1.960100e+07	92.485000	44.110001	81.615002	522.489990
75%	5.113775e+07	93.680000	60.610001	86.150002	550.510010
max	8.128200e+07	100.400002	67.519997	97.250000	645.489990

1. Info()를 통해 데이터들의 요약정보 확인
2. describe()를 통해 데이터의 기술통계적인 부분을 확인

결측치 및 상관계수 확인

```
# 컬럼 별 결측치 확인
data_set.isnull().sum()
```

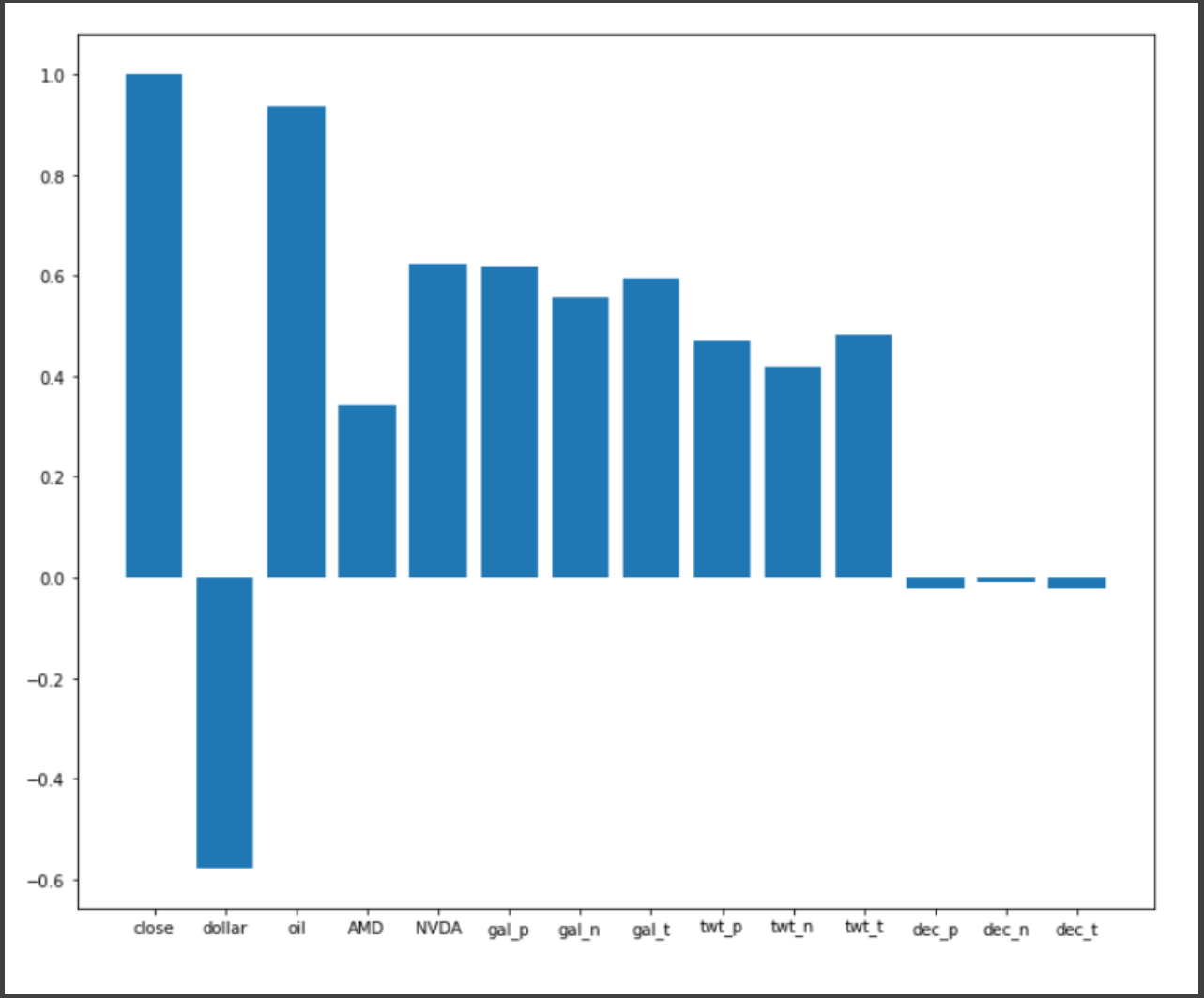
```
close     0
dollar    0
oil       0
AMD       0
NVDA      0
gal_p     0
gal_n     0
gal_t     0
twt_p     0
twt_n     0
twt_t     0
dec_p     0
dec_n     0
dec_t     0
dtype: int64
```

```
# close에 대한 다른 변수들의 상관계수를 내림차 순으로 정렬하여 확인
corr_data_set=data_set.corr()
corr_data_set['close'].sort_values(ascending=False)
```

```
close     1.000000
oil       0.935422
NVDA      0.623908
gal_p     0.617485
gal_t     0.595700
gal_n     0.554558
twt_t     0.482461
twt_p     0.469824
twt_n     0.416955
AMD       0.341236
dec_n    -0.009399
dec_p    -0.021489
dec_t    -0.022230
dollar   -0.578837
Name: close, dtype: float64
```

1. Isnull()과 sum()을 통해 결측치를 확인
2. Close에 대한 다른 변수들의 상관계수를 내림차 순으로 정렬.

상관계수 시각화




```
# 분석에 활용할 데이터를 선택하기 위해 조금 더 보기쉽게 시각화
plt.figure(figsize=(12,10))
plt.bar(corr_data_set.index, corr_data_set["close"])
```

1. 상관계수가 높은 Oil, NVDA로

Multiple Linear Regression

2. 긍정적인 게시글에 해당하는

gal_p, twt_p와 NVDA로

Multiple Linear Regression

코드부분

https://github.com/SEUNGYEOPOH/Python_breakers_RP/blob/main/ML.ipynb

Shaply

Shapley Value

- 특정 변수가 예측력에 얼마나 기여하는지 파악하기 위해 모든 변수의 조합들을 입력시켰을 때 나온 결과값을 비교하면서 변수의 기여도를 계산하는 방식.

Case	x1	x2	x3	결과값
Case 1	X	X	X	28
Case 2	O	X	X	32
Case 3	X	O	X	31
Case 4	X	X	O	30
Case 5	O	O	X	32
Case 6	O	X	O	33
Case 7	X	O	O	32
Case 8	O	O	O	35

X1에 대한 Shaply Value

- Case 2 - Case 1 = 32 - 28 = 4
- Case 5 - Case 3 = 32 - 31 = 1
- Case 6 - Case 4 = 33 - 30 = 3
- Case 8 - Case 7 = 35 - 32 = 3

각각에 대해 경우의 수로 나누어 가중치를 구함

4*1/3 + 1*1/6 + 3*1/6 + 3*1/3 = 3

Shap

Shap 패키지
- Shapley value를 근간으로 하는 XAI(eXplainable)

TreeExplainer - 트리 및 트리 앙상블에 대한 SHAP

DeepExplainer - SHAP과 DeepLIFT 연결을 기반으로 딥러닝에 대한 SHAP

GradientExplainer - 딥러닝 모델의 SHAP 값을 근사화

LinearExplainer - Linear 모델에 대한 SHAP

KernelExplainer - 모든 기계 학습 모델의 SHAP

사용방법
(전제조건 : skimage와 shap 패키지 설치가 완료 되어 있어야함.)

explain=Shap.LinearExplainer(model,test_data)
-> shap value 객체 지정
Shap_values = explain.shap_values(test_data)
-> shap_value 구하기
Shap.summary_plot(shap_values, test_data)
-> 시각화

결론

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
rf	Random Forest Regressor	869057.9556	9150749871973.3652	2951624.4587	0.9817	0.0926	0.0287	0.1530
et	Extra Trees Regressor	1037306.1592	9727671703393.2715	3079139.1894	0.9806	0.0989	0.0345	0.1380
lightgbm	Light Gradient Boosting Machine	1291957.9689	9976337769279.4688	3115312.6250	0.9800	0.0995	0.0485	0.0180
dt	Decision Tree Regressor	841331.8491	11701266783776.3398	3359989.1393	0.9766	0.1041	0.0258	0.0070
knn	K Neighbors Regressor	1637296.8990	18022146276509.4258	4188723.3338	0.9643	0.1068	0.0419	0.0170
gbr	Gradient Boosting Regressor	2658299.8466	23458169115783.0703	4830675.9308	0.9531	0.1562	0.0974	0.0330
ada	AdaBoost Regressor	5188944.3142	52706282903636.9375	7251210.9766	0.8947	0.2518	0.2118	0.0140

어려움이 있고, Non-Linear Regression model을 적용시키는 것이 적합할 것이다.

결론

1. 상관관계가 높은 두 독립변수 (oil, NVDA)로 Multiple Linear Regression을 진행했을 경우에는 설명력이 87%로 꽤나 준수한 결과를 보여주었다.

-> oil, NVDA 데이터로 시간봉 종가를 예측할 수 있다.

2. 긍정적인 게시글에 해당하는 gal_p, twt_p와 NVDA로 Multiple Linear Regression을 진행했을 경우에는 설명력이 55%로 애매한 결과를 보여주었다. 상관계수가 높은 데이터로 학습을 진행했을 경우에는 조금 더 나은 결과를 안겨줬겠지만 이 과정을 통해서 선형모델의 한계를 알 수 있었다. 비연속적인 데이터를 선형으로 표현하기에는 어려움이 있고, Non-Linear Regression model을 적용시키는 것이 적합할 것이다.