Stock price prediction

Dataset

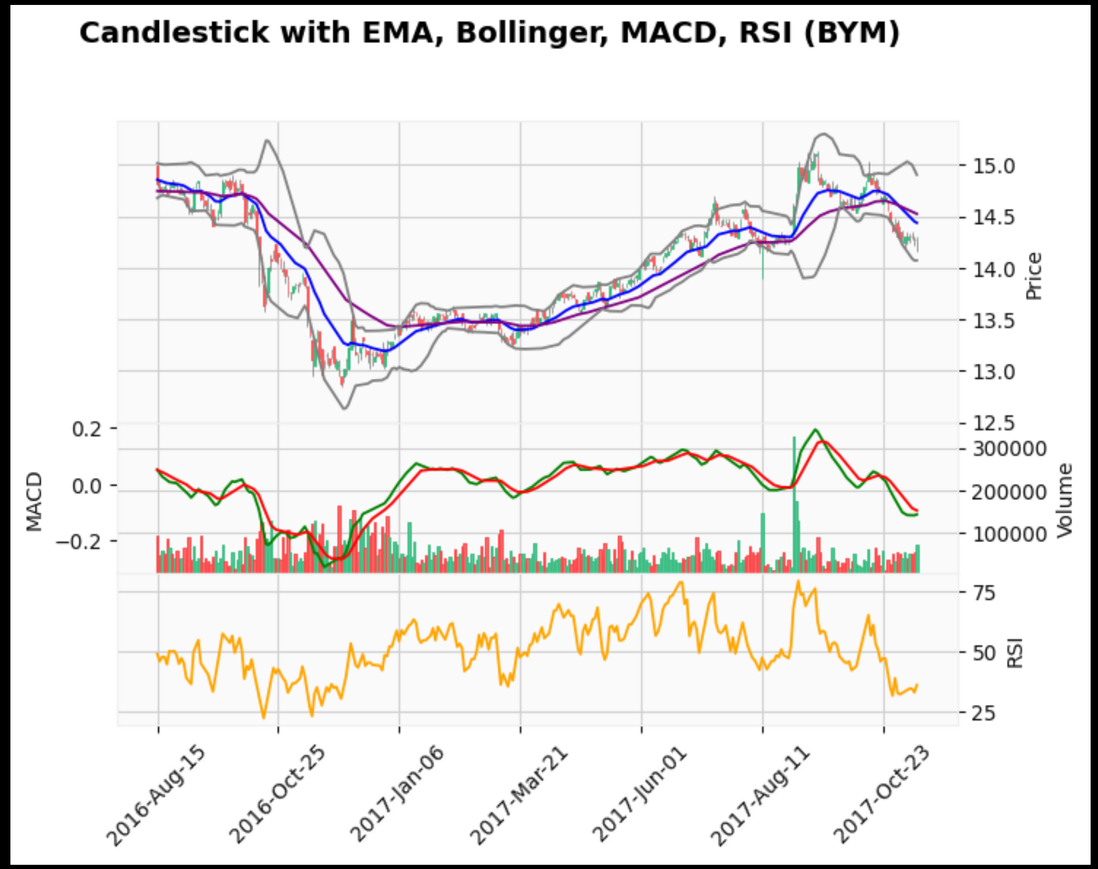
- Relatively large stock market dataset from Kaggle
- Full historical daily price and volume data for all US-based stocks and ETFs trading on the NYSE, NASDAQ, and NYSE MKT
- After preprocessing: 4571 stocks, 1155 ETFs
- Only a subset of the dataset is used due to computational limitations
- Date, Open, High, Low, Close, Volume (daily)

EDA

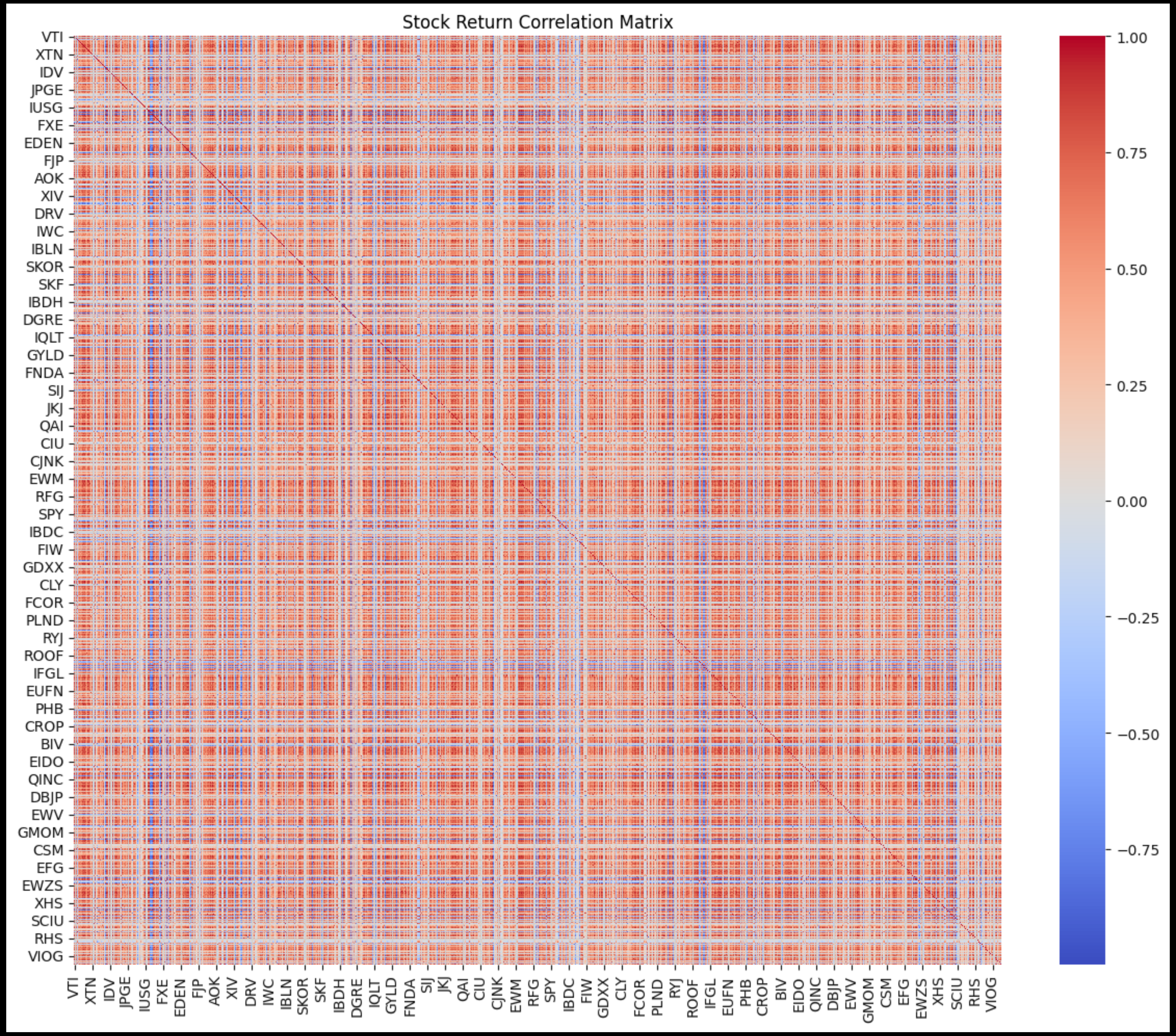
- Part of the data was empty or corrupted
- We only kept files with at least 300 entries (≈ 10 months)
- Used a correlation matrix and PCA + K-means
- ETFs were much more correlated (as expected)

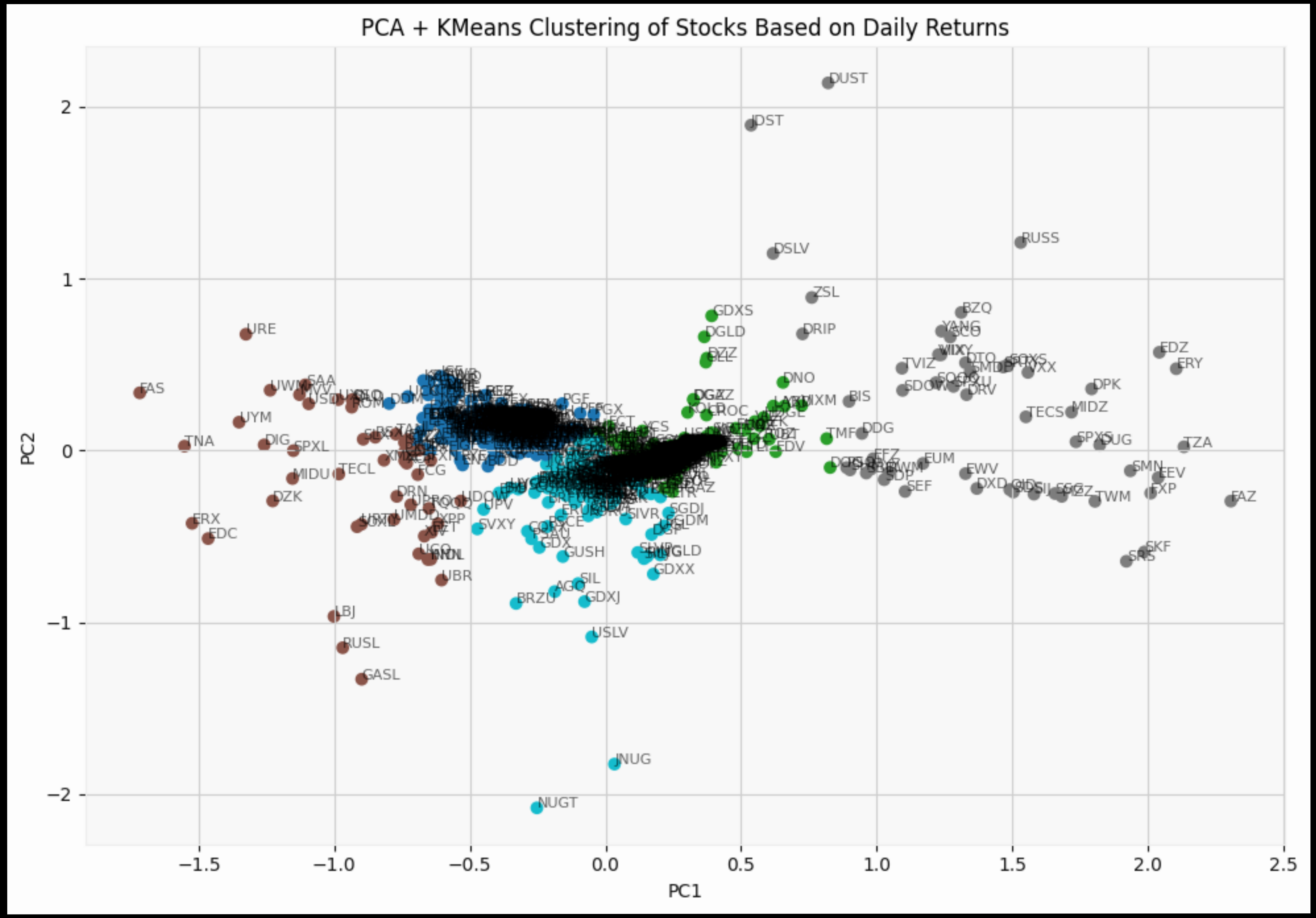
Example from EDA

- Candlestick plot of BYM with additional technical indicators



EDA of ETFs

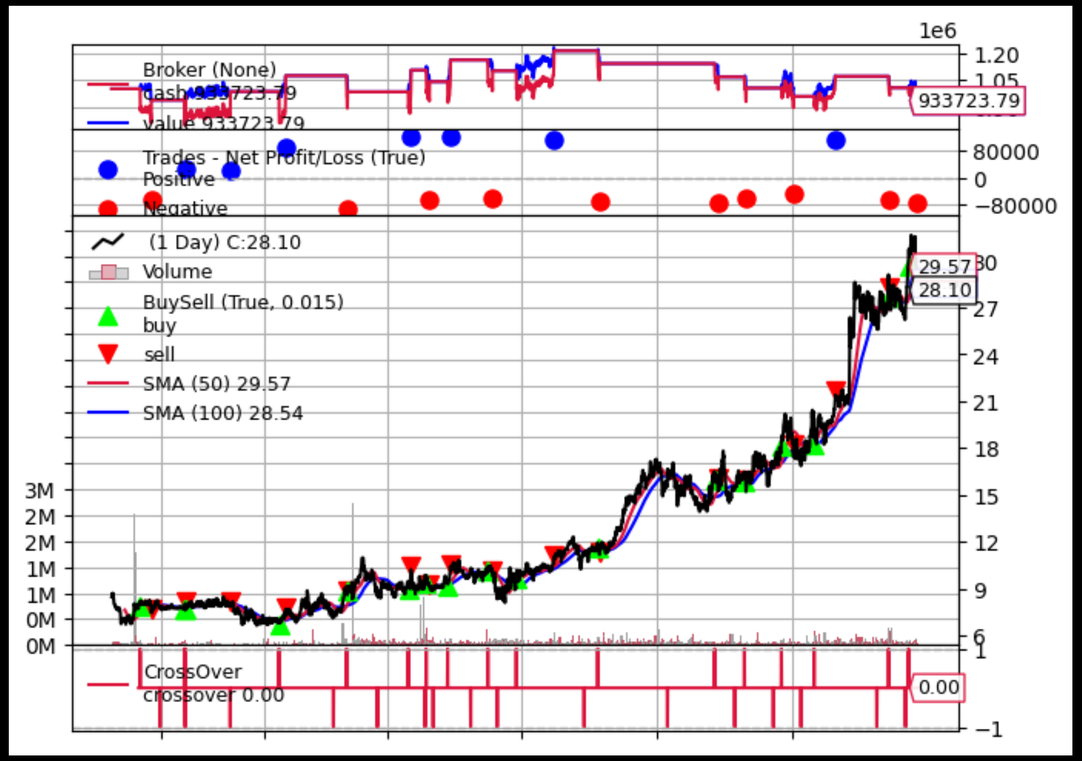

Stock Return Correlation Matrix
VTI
XTN
IDV
JPGE
IUSG
FXE
EDEN
FJP
AOK
XIV
DRV
IWC
IBLN
SKOR
SKF
IBDH
DGRE
IQLT
GYLD
FNDA
SIJ
JKJ
QAI
CIU
CJNK
EWM
RFG
SPY
IBDC
FIW
GDXX
CLY
FCOR
PLND
RYJ
ROOF
IFGL
EUFN
PHB
CROP
BIV
EIDO
QINC
DBJP
EWV
GMOM
CSM
EFG
EWZS
XHS
SCIU
RHS
VIOG
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

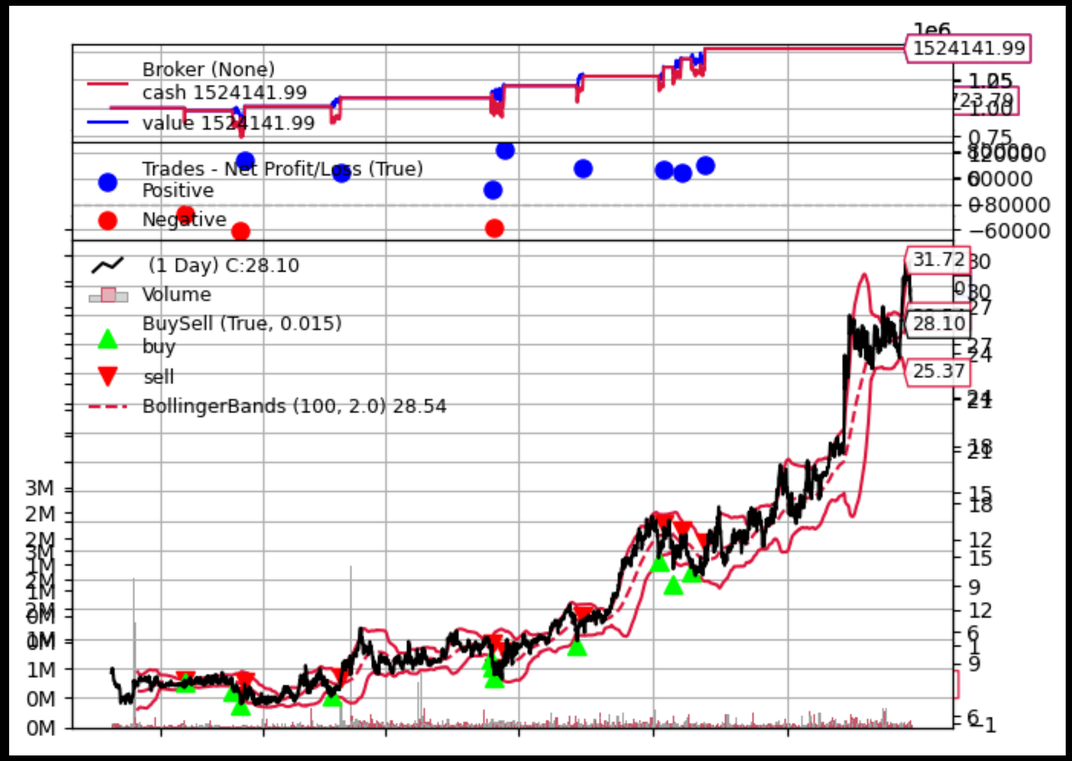

PCA + KMeans Clustering of Stocks Based on Daily Returns
PC1
PC2
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
2
1
0
−1
−2
DUST
JDST
RUSS
DSLV
ZSL
GDXS
DGLD
DRIP
BZQ
URE
FAS
EDZ
ERY
DPK
TZA
FAZ
MIDZ
TECS
UYM
TNA
SPXL
MIDU
TECL
DZK
ERX
EDC
SVXY
GUSH
BRZU
SIL
GDXJ
USLV
LBJ
RUSL
GASL
JNUG
NUGT
GDXX
SKF
SMN
TWM
EWV
EUM
SEF
DDG
TMF
BIS
DNO
UBR

Backtesting

- We backtested common trading strategies to serve as a benchmark for comparison
- EMA, MACD, Bollinger Bands, RSI and the simple buy & hold strategies were tested (with several parameters)
- Markets are hard to beat: buy & hold performed fairly well on most stocks and ETFs

Examples from backtesting





MDP for trading

- MDP $= (\mathcal{S}, \mathcal{A}, P, R, \gamma)$
- $s_t = \left[p_t, p_{t-1}, \ldots, p_{t-k}\right]$ is a vector of closing prices k units back in time
- $\mathcal{A} = \{\text{Buy, Hold, Sell}\}$
- $r_t = w_{t+1} - w_t$ means the reward at time t is the portfolio value at time (t + 1) minus the value at time t
- $$Q(s, a) = \mathbb{E}\left[r + \gamma \max_{a'} Q(s', a') \mid s, a\right]$$
- The former equation defines the expected return from taking action a in state s and then following the optimal policy

DQN

- When the number of states is large we approximate the Q-value with a neural network
- It’s a basically a neural network that takes a state as input and outputs a score (Q-value) for each action (Buy, Sell, Hold)
- Experience Replay: Instead of learning only from the latest trade, the agent remembers past trades and learns from random ones
- This prevents it from overreacting to recent events

PPO

- Policy gradient algorithm, meaning it directly learns the policy — a function that maps states to actions — by optimizing expected reward
- It learns a stochastic policy
- Policy: $\pi_\theta(a \mid s)$ is the probability that we take action a in state s (parameterized by a neural network)
- Advantage function: $A_t = Q(s_t, a_t) - V(s_t)$ tells the agent how much better (or worse) an action was compared to the average
- The goal is to maximize expected advantage while clipping changes that are too large

DDPG

- Designed for problems where actions are continuous, not discrete
- In trading, instead of just buy / hold / sell, we can decide how much to buy or sell — like allocating 23% of your capital to an asset
- Actor-Critic Framework: Actor is the decision maker, Critic is the evaluation function

DDPG: continued

- Observe market state s_t
- Actor outputs action $a_t = \mu(s_t)$
- Execute action: get reward r_t and next state s_{t+1}
- Critic learns how good that action was
- Actor updates to make better decisions next time

RL vs Backtest