Algorithmic Trading, Risk Management, Trading Bot and Event-Based Backtesting

1111AIFQA10
MBA, IM, NTPU (M6132) (Fall 2022)
Tue 2, 3, 4 (9:10-12:00) (B8F40)

https://meet.google.com/paj-zhhj-mya

Min-Yuh Day, Ph.D, Associate Professor

Institute of Information Management, National Taipei University

https://web.ntpu.edu.tw/~myday

2022-12-13

Syllabus

国立臺北大學
National Taipei University

Week	Date	Subject/Topics
1	2022/09/13	Introduction to Artificial Intelligence in Finance and Quantitative Analysis
2	2022/09/20	AI in FinTech: Metaverse, Web3, DeFi, NFT, Financial Services Innovation and Applications
3	2022/09/27	Investing Psychology and Behavioral Finance
4	2022/10/04	Event Studies in Finance
5	2022/10/11	Case Study on AI in Finance and Quantitative Analysis I
6	2022/10/18	Finance Theory

Syllabus

Week	Date	Subject/Topics
7	2022/10/25	Data-Driven Finance
8	2022/11/01	Midterm Project Report
9	2022/11/08	Financial Econometrics and Machine Learning
10	2022/11/15	AI-First Finance
11	2022/11/22	Deep Learning in Finance; Reinforcement Learning in Finance
12	2022/11/29	Case Study on AI in Finance and Quantitative Analysis II

Syllabus

Week	Date	Subject/Topics
13	2022/12/06	Industry Practices of AI in Finance and Quantitative Analysis
14	2022/12/13	Algorithmic Trading; Risk Management; Trading Bot and Event-Based Backtesting
15	2022/12/20	Final Project Report I
16	2022/12/27	Final Project Report II
17	2023/01/03	Self-learning
18	2023/01/10	Self-learning

Algorithmic Trading
Risk Management
Trading Bot
Event-Based Backtesting

Outline

- Algorithmic Trading
- Risk Management
- Trading Bot
- Event-Based Backtesting

Deep learning for financial applications: A survey

Applied Soft Computing (2020)

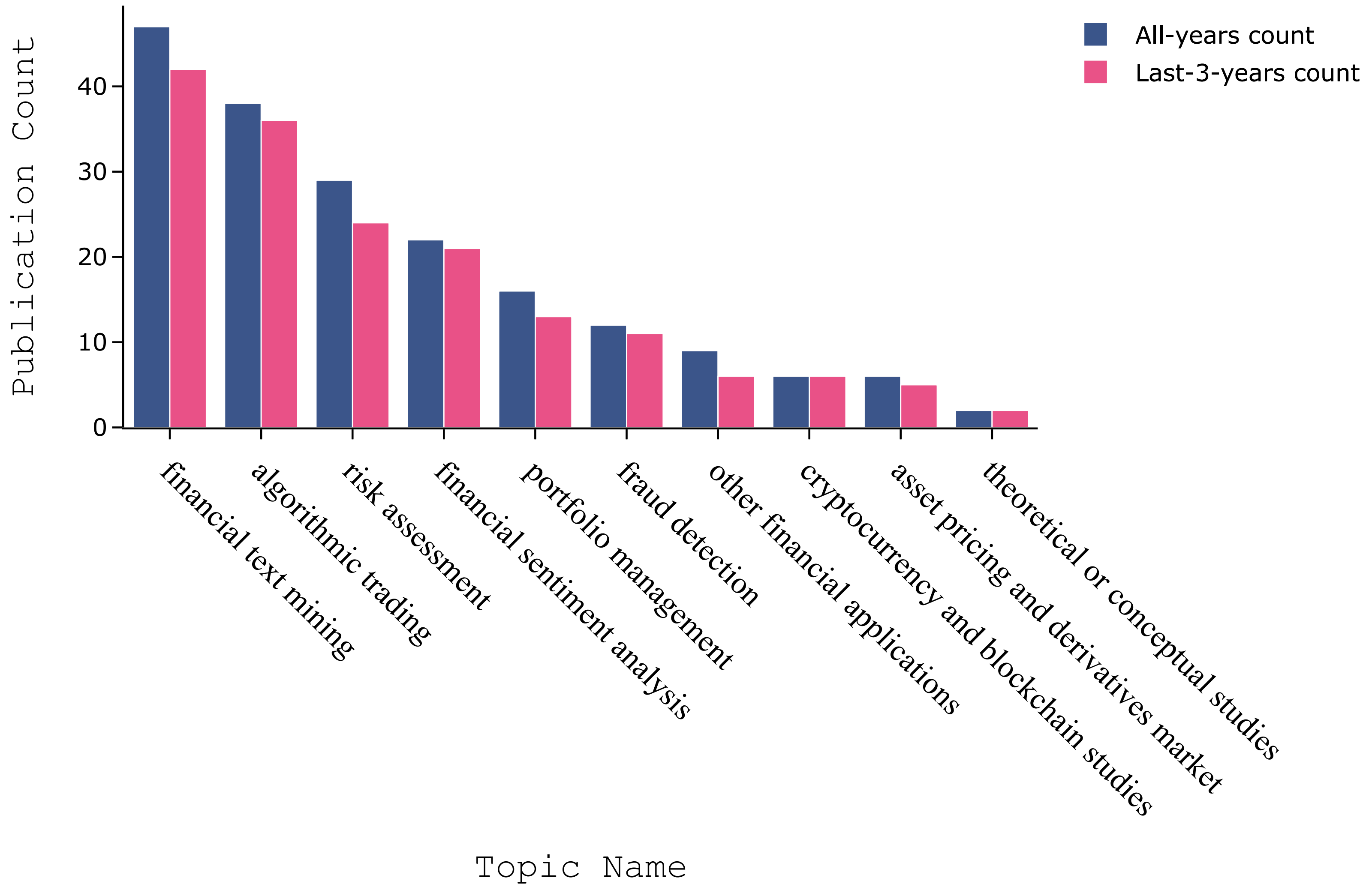
Source:
Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey."
Applied Soft Computing (2020): 106384.

Financial time series forecasting with deep learning: A systematic literature review: 2005–2019

Applied Soft Computing (2020)

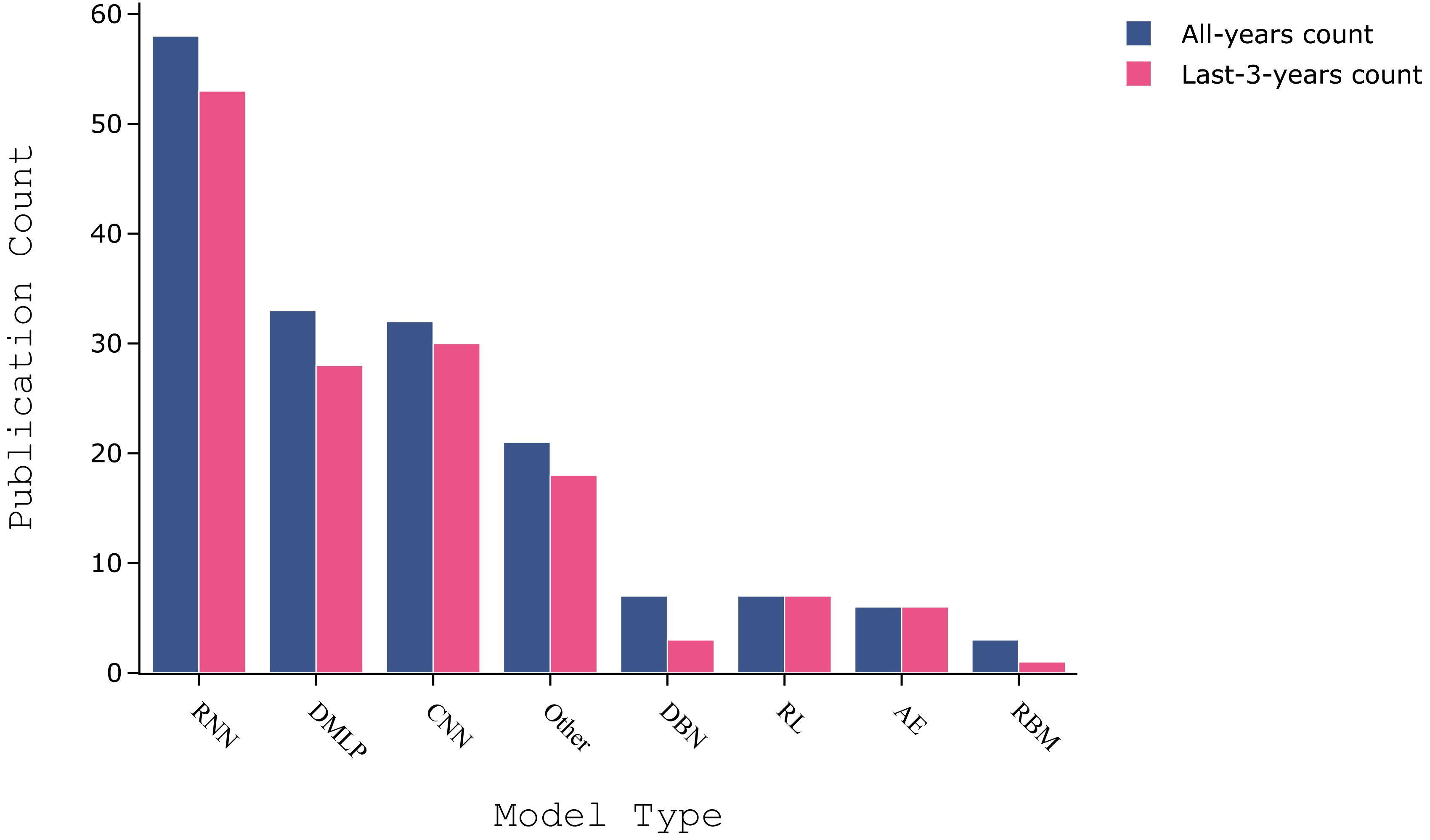
Source:
Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu (2020), "Financial time series forecasting with deep learning: A systematic literature review: 2005–2019." Applied Soft Computing 90 (2020): 106181.

Deep learning for financial applications: Topics



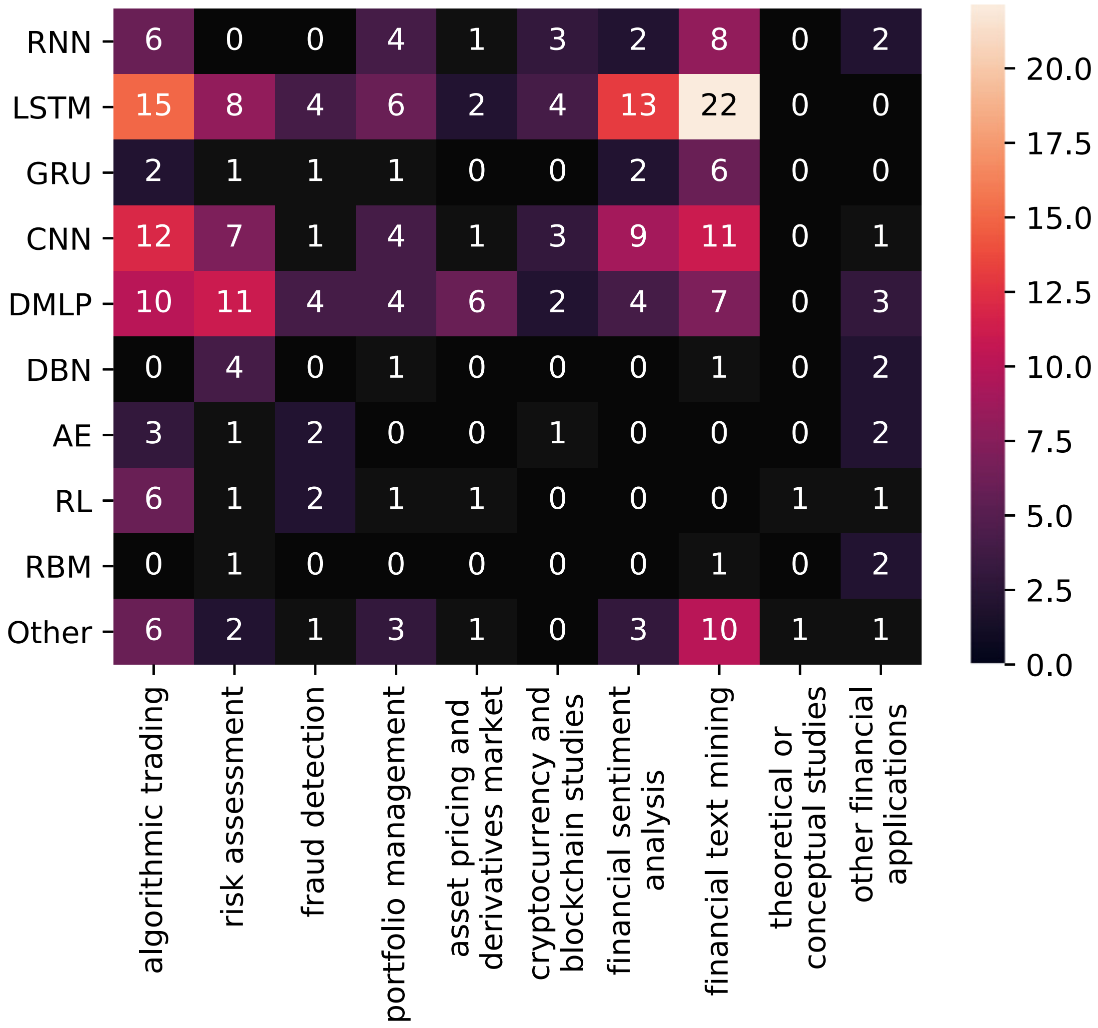
Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications: Deep Learning Models



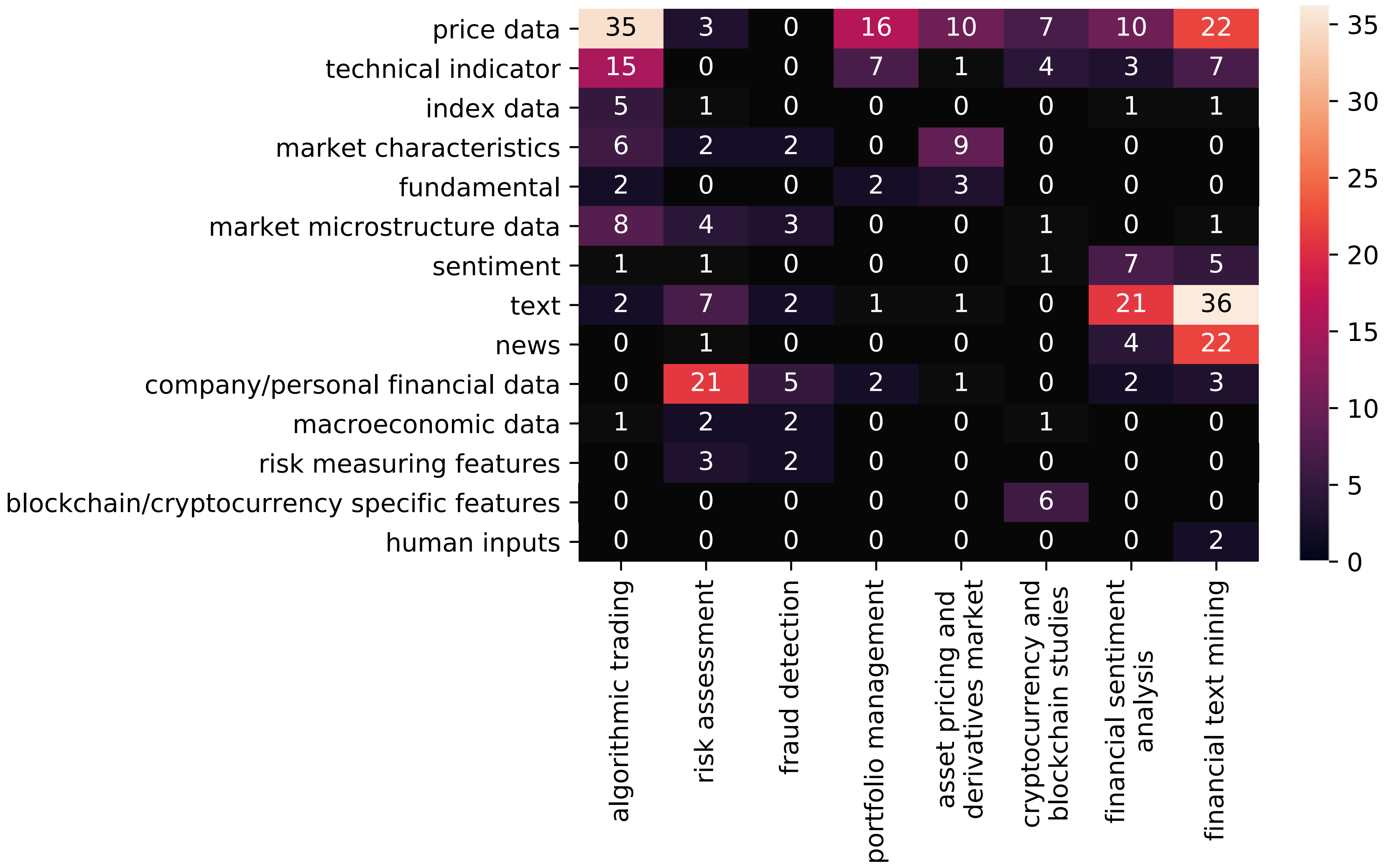
Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications: Topic-Model Heatmap



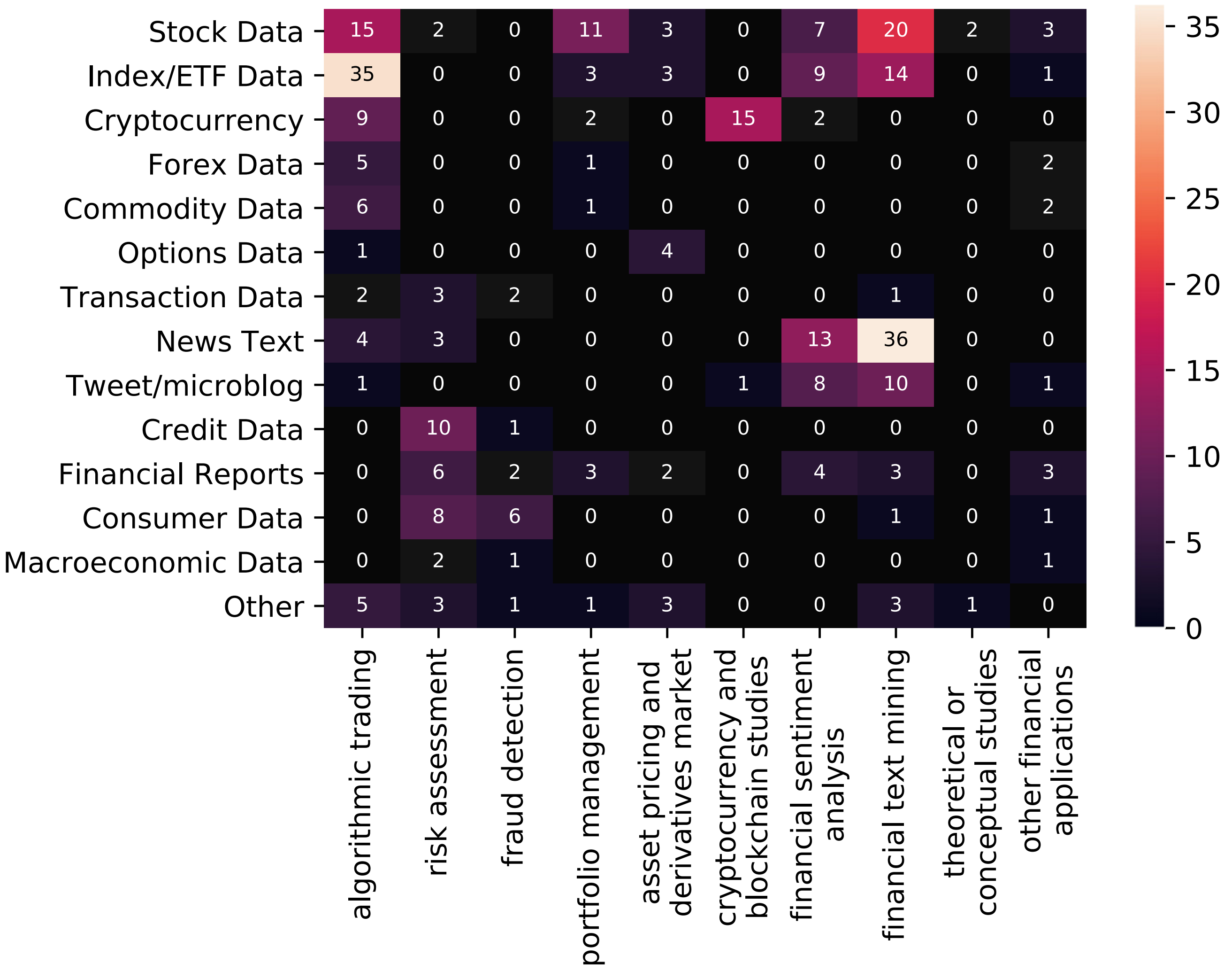
Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications: Topic-Feature Heatmap



Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications: Topic-Dataset Heatmap



Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:

Algo-trading applications embedded with time series forecasting models

Art.	Data set	Period	Feature set	Method	Performance criteria	Environment
[33]	GarantiBank in BIST, Turkey	2016	OCHLV, Spread, Volatility, Turnover, etc.	PLR, Graves LSTM	MSE, RMSE, MAE, RSE, Correlation R-square	Spark
[34]	CSI300, Nifty50, HSI, Nikkei 225, S&P500, DJIA	2010–2016	OCHLV, Technical Indicators	WT, Stacked autoencoders, LSTM	MAPE, Correlation coefficient, THEIL-U	–
[35]	Chinese Stocks	2007–2017	OCHLV	CNN + LSTM	Annualized Return, Mxm Retracement	Python
[36]	50 stocks from NYSE	2007–2016	Price data	SFM	MSE	–
[37]	The LOB of 5 stocks of Finnish Stock Market	2010	FI-2010 dataset: bid/ask and volume	WMTR, MDA	Accuracy, Precision, Recall, F1-Score	–
[38]	300 stocks from SZSE, Commodity	2014–2015	Price data	FDDR, DMLP+RL	Profit, return, SR, profit-loss curves	Keras
[39]	S&P500 Index	1989–2005	Price data, Volume	LSTM	Return, STD, SR, Accuracy	Python, TensorFlow, Keras, R, H2O
[40]	Stock of National Bank of Greece (ETE).	2009–2014	FTSE100, DJIA, GDAX, NIKKEI225, EUR/USD, Gold	GASVR, LSTM	Return, volatility, SR, Accuracy	Tensorflow
[41]	Chinese stock-IF-IH-IC contract	2016–2017	Decisions for price change	MODRL+LSTM	Profit and loss, SR	–
[42]	Singapore Stock Market Index	2010–2017	OCHL of last 10 days of Index	DMLP	RMSE, MAPE, Profit, SR	–
[43]	GBP/USD	2017	Price data	Reinforcement Learning + LSTM + NES	SR, downside deviation ratio, total profit	Python, Keras, Tensorflow
[44]	Commodity, FX future, ETF	1991–2014	Price Data	DMLP	SR, capability ratio, return	C++, Python
[45]	USD/GBP, S&P500, FTSE100, oil, gold	2016	Price data	AE + CNN	SR, % volatility, avg return/trans, rate of return	H2O

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:

Algo-trading applications embedded with time series forecasting models

Art.	Data set	Period	Feature set	Method	Performance criteria	Environment
[46]	Bitcoin, Dash, Ripple, Monero, Litecoin, Dogecoin, Nxt, Namecoin	2014–2017	MA, BOLL, the CRIX returns, Euribor interest rates, OCHLV	LSTM, RNN, DMLP	Accuracy, F1-measure	Python, Tensorflow
[47]	S&P500, KOSPI, HSI, and EuroStoxx50	1987–2017	200-days stock price	Deep Q-Learning, DMLP	Total profit, Correlation	–
[48]	Stocks in the S&P500	1990–2015	Price data	DMLP, GBT, RF	Mean return, MDD, Calmar ratio	H2O
[49]	Fundamental and Technical Data, Economic Data	–	Fundamental , technical and market information	CNN	–	–

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:

Classification (buy–sell signal, or trend detection) based algo-trading models

Art.	Data set	Period	Feature set	Method	Performance criteria	Environment
[51]	Stocks in Dow30	1997–2017	RSI	DMLP with genetic algorithm	Annualized return	Spark MLlib, Java
[52]	SPY ETF, 10 stocks from S&P500	2014–2016	Price data	FFNN	Cumulative gain	MatConvNet, Matlab
[53]	Dow30 stocks	2012–2016	Close data and several technical indicators	LSTM	Accuracy	Python, Keras, Tensorflow, TALIB
[54]	High-frequency record of all orders	2014–2017	Price data, record of all orders, transactions	LSTM	Accuracy	–
[55]	Nasdaq Nordic (Kesko Oyj, Outokumpu Oyj, Sampo, Rautaruukki, Wartsila Oyj)	2010	Price and volume data in LOB	LSTM	Precision, Recall, F1-score, Cohen's k	–
[56]	17 ETFs	2000–2016	Price data, technical indicators	CNN	Accuracy, MSE, Profit, AUROC	Keras, Tensorflow
[57]	Stocks in Dow30 and 9 Top Volume ETFs	1997–2017	Price data, technical indicators	CNN with feature imaging	Recall, precision, F1-score, annualized return	Python, Keras, Tensorflow, Java
[58]	FTSE100	2000–2017	Price data	CAE	TR, SR, MDD, mean return	–
[59]	Nasdaq Nordic (Kesko Oyj, Outokumpu Oyj, Sampo, Rautaruukki, Wartsila Oyj)	2010	Price, Volume data, 10 orders of the LOB	CNN	Precision, Recall, F1-score, Cohen's k	Theano, Scikit learn, Python
[60]	Borsa Istanbul 100 Stocks	2011–2015	75 technical indicators and OCHLV	CNN	Accuracy	Keras
[61]	ETFs and Dow30	1997–2007	Price data	CNN with feature imaging	Annualized return	Keras, Tensorflow
[62]	8 experimental assets from bond/derivative market	–	Asset prices data	RL, DMLP, Genetic Algorithm	Learning and genetic algorithm error	–
[63]	10 stocks from S&P500	–	Stock Prices	TDNN, RNN, PNN	Missed opportunities, false alarms ratio	–
[64]	London Stock Exchange	2007–2008	Limit order book state, trades, buy/sell orders, order deletions	CNN	Accuracy, kappa	Caffe
[65]	Cryptocurrencies, Bitcoin	2014–2017	Price data	CNN, RNN, LSTM	Accumulative portfolio value, MDD, SR	–

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:

Stand-alone and/or other algorithmic models

Art.	Data set	Period	Feature set	Method	Performance criteria	Environment
[66]	DAX, FTSE100, call/put options	1991–1998	Price data	Markov model, RNN	Ewa-measure, iv, daily profits' mean and std	–
[67]	Taiwan Stock Index Futures, Mini Index Futures	2012–2014	Price data to image	Visualization method + CNN	Accumulated profits,accuracy	–
[68]	Energy-Sector/ Company-Centric Tweets in S&P500	2015–2016	Text and Price data	LSTM, RNN, GRU	Return, SR, precision, recall, accuracy	Python, Tweepy API
[69]	CME FIX message	2016	Limit order book, time-stamp, price data	RNN	Precision, recall, F1-measure	Python, TensorFlow, R
[70]	Taiwan stock index futures (TAIFEX)	2017	Price data	Agent based RL with CNN pre-trained	Accuracy	–
[71]	Stocks from S&P500	2010–2016	OCHLV	DCNL	PCC, DTW, VWL	Pytorch
[72]	News from NowNews, AppleDaily, LTN, MoneyDJ for 18 stocks	2013–2014	Text, Sentiment	DMLP	Return	Python, Tensorflow
[73]	489 stocks from S&P500 and NASDAQ-100	2014–2015	Limit Order Book	Spatial neural network	Cross entropy error	NVIDIA's cuDNN
[74]	Experimental dataset	–	Price data	DRL with CNN, LSTM, GRU, DMLP	Mean profit	Python

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:
Credit scoring or classification studies

Art.	Data set	Period	Feature set	Method	Performance criteria	Env.
[77]	The XR 14 CDS contracts	2016	Recovery rate, spreads, sector and region	DBN+RBM	AUROC, FN, FP, Accuracy	WEKA
[78]	German, Japanese credit datasets	–	Personal financial variables	SVM + DBN	Weighted-accuracy, TP, TN	–
[79]	Credit data from Kaggle	–	Personal financial variables	DMLP	Accuracy, TP, TN, G-mean	–
[80]	Australian, German credit data	–	Personal financial variables	GP + AE as Boosted DMLP	FP	Python, Scikit-learn
[81]	German, Australian credit dataset	–	Personal financial variables	DCNN, DMLP	Accuracy, False/Missed alarm	–
[82]	Consumer credit data from Chinese finance company	–	Relief algorithm chose the 50 most important features	CNN + Relief	AUROC, K-s statistic, Accuracy	Keras
[83]	Credit approval dataset by UCI Machine Learning repo	–	UCI credit approval dataset	Rectifier, Tanh, Maxout DL	–	AWS EC2, H2O, R

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:

Financial distress, bankruptcy, bank risk, mortgage risk, crisis forecasting studies.

Art.	Data set	Period	Feature set	Method	Performance criteria	Env.
[84]	966 french firms	–	Financial ratios	RBM+SVM	Precision, Recall	–
[85]	883 BHC from EDGAR	2006–2017	Tokens, weighted sentiment polarity, leverage and ROA	CNN, LSTM, SVM, RF	Accuracy, Precision, Recall, F1-score	Keras, Python, Scikit-learn
[86]	The event data set for large European banks, news articles from Reuters	2007–2014	Word, sentence	DMLP +NLP preprocess	Relative usefulness, F1-score	–
[87]	Event dataset on European banks, news from Reuters	2007–2014	Text, sentence	Sentence vector + DFFN	Usefulness, F1-score, AUROC	–
[88]	News from Reuters, fundamental data	2007–2014	Financial ratios and news text	doc2vec + NN	Relative usefulness	Doc2vec
[89]	Macro/Micro economic variables, Bank charac-teristics/performance variables from BHC	1976–2017	Macro economic variables and bank performances	CGAN, MVN, MV-t, LSTM, VAR, FE-QAR	RMSE, Log likelihood, Loan loss rate	–
[90]	Financial statements of French companies	2002–2006	Financial ratios	DBN	Recall, Precision, F1-score, FP, FN	–
[91]	Stock returns of American publicly-traded companies from CRSP	2001–2011	Price data	DBN	Accuracy	Python, Theano
[92]	Financial statements of several companies from Japanese stock market	2002–2016	Financial ratios	CNN	F1-score, AUROC	–
[93]	Mortgage dataset with local and national economic factors	1995–2014	Mortgage related features	DMLP	Negative average log-likelihood	AWS
[94]	Mortgage data from Norwegian financial service group, DNB	2012–2016	Personal financial variables	CNN	Accuracy, Sensitivity, Specificity, AUROC	–
[95]	Private brokerage company's real data of risky transactions	–	250 features: order details, etc.	CNN, LSTM	F1-Score	Keras, Tensorflow
[96]	Several datasets combined to create a new one	1996–2017	Index data, 10-year Bond yield, exchange rates,	Logit, CART, RF, SVM, NN, XGBoost, DMLP	AUROC, KS, G-mean, likelihood ratio, DP, BA, WBA	R

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:
Fraud detection studies

Art.	Data set	Period	Feature set	Method	Performance criteria	Env.
[114]	Debit card transactions by a local Indonesia bank	2016–2017	Financial transaction amount on several time periods	CNN, Stacked-LSTM, CNN-LSTM	AUROC	–
[115]	Credit card transactions from retail banking	2017	Transaction variables and several derived features	LSTM, GRU	Accuracy	Keras
[116]	Card purchases' transactions	2014–2015	Probability of fraud per currency/origin country, other fraud related features	DMLP	AUROC	–
[117]	Transactions made with credit cards by European cardholders	2013	Personal financial variables to PCA	DMLP, RF	Recall, Precision, Accuracy	–
[118]	Credit-card transactions	2015	Transaction and bank features	LSTM	AUROC	Keras, Scikit-learn
[119]	Databases of foreign trade of the Secretariat of Federal Revenue of Brazil	2014	8 Features: Foreign Trade, Tax, Transactions, Employees, Invoices, etc	AE	MSE	H2O, R
[120]	Chamber of Deputies open data, Companies data from Secretariat of Federal Revenue of Brazil	2009–2017	21 features: Brazilian State expense, party name, Type of expense, etc.	Deep Autoencoders	MSE, RMSE	H2O, R
[121]	Real-world data for automobile insurance company labeled as fradulent	–	Car, insurance and accident related features	DMLP + LDA	TP, FP, Accuracy, Precision, F1-score	–
[122]	Transactions from a giant online payment platform	2006	Personal financial variables	GBDT+DMLP	AUROC	–
[123]	Financial transactions	–	Transaction data	LSTM	t-SNE	–
[124]	Empirical data from Greek firms	–	–	DQL	Revenue	Torch

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:
Portfolio management studies

Art.	Data set	Period	Feature set	Method	Performance criteria	Env.
[65]	Cryptocurrencies, Bitcoin	2014–2017	Price data	CNN, RNN, LSTM	Accumulative portfolio value, MDD, SR	–
[127]	Stocks from NYSE, AMEX, NASDAQ	1965–2009	Price data	Autoencoder + RBM	Accuracy, confusion matrix	–
[128]	20 stocks from S&P500	2012–2015	Technical indicators	DMLP	Accuracy	Python, Scikit Learn, Keras, Theano
[129]	Chinese stock data	2012–2013	Technical, fundamental data	Logistic Regression, RF, DMLP	AUC, accuracy, precision, recall, f1, tpr, fpr	Keras, Tensorflow, Python, Scikit learn
[130]	Top 5 companies in S&P500	–	Price data and Financial ratios	LSTM, Auto-encoding, Smart indexing	CAGR	–
[131]	IBB biotechnology index, stocks	2012–2016	Price data	Auto-encoding, Calibrating, Validating, Verifying	Returns	–
[132]	Taiwans stock market	–	Price data	Elman RNN	MSE, return	–
[133]	FOREX (EUR/USD, etc.), Gold	2013	Price data	Evolino RNN	Return	Python
[134]	Stocks in NYSE, AMEX, NASDAQ, TAQ intraday trade	1993–2017	Price, 15 firm characteristics	LSTM+DMLP	Monthly return, SR	Python,Keras, Tensorflow in AWS
[135]	S&P500	1985–2006	monthly and daily log-returns	DBN+MLP	Validation, Test Error	Theano, Python, Matlab
[136]	10 stocks in S&P500	1997–2016	OCHLV, Price data	RNN, LSTM, GRU	Accuracy, Monthly return	Keras, Tensorflow
[137]	Analyst reports on the TSE and Osaka Exchange	2016–2018	Text	LSTM, CNN, Bi-LSTM	Accuracy, R^2	R, Python, MeCab
[138]	Stocks from Chinese/American stock market	2015–2018	OCHLV, Fundamental data	DDPG, PPO	SR, MDD	–
[139]	Hedge fund monthly return data	1996–2015	Return, SR, STD, Skewness, Kurtosis, Omega ratio, Fund alpha	DMLP	Sharpe ratio, Annual return, Cum. return	–
[140]	12 most-volumed cryptocurrency	2015–2016	Price data	CNN + RL	SR, portfolio value, MDD	–

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:

Asset pricing and derivatives market studies

Art.	Der. type	Data set	Period	Feature set	Method	Performance criteria	Env.
[137]	Asset pricing	Analyst reports on the TSE and Osaka Exchange	2016–2018	Text	LSTM, CNN, Bi-LSTM	Accuracy, R^2	R, Python, MeCab
[142]	Options	Simulated a range of call option prices	–	Price data, option strike/maturity, dividend/risk free rates, volatility	DMLP	RMSE, the average percentage pricing error	Tensorflow
[143]	Futures, Options	TAIEX Options	2017	OCHLV, fundamental analysis, option price	DMLP, DMLP with Black scholes	RMSE, MAE, MAPE	–
[144]	Equity returns	Returns in NYSE, AMEX, NASDAQ	1975–2017	57 firm characteristics	Fama–French n-factor model DL	R^2,RMSE	Tensorflow

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:
Cryptocurrency and blockchain studies

Art.	Data set	Period	Feature set	Method	Performance criteria	Env.
[46]	Bitcoin, Dash, Ripple, Monero, Litecoin, Dogecoin, Nxt, Namecoin	2014–2017	MA, BOLL, the CRIX daily returns, Euribor interest rates, OCHLV of EURO/UK, EURO/USD, US/JPY	LSTM, RNN, DMLP	Accuracy, F1-measure	Python, Tensorflow
[65]	Cryptocurrencies, Bitcoin	2014–2017	Price data	CNN	Accumulative portfolio value, MDD, SR	–
[140]	12 most-volumed cryptocurrency	2015–2016	Price data	CNN + RL	SR, portfolio value, MDD	
[145]	Bitcoin data	2010–2017	Hash value, bitcoin address, public/private key, digital signature, etc.	Takagi–Sugeno Fuzzy cognitive maps	Analytical hierarchy process	–
[146]	Bitcoin data	2012, 2013, 2016	TransactionId, input/output Addresses, timestamp	Graph embedding using heuristic, laplacian eigen-map, deep AE	F1-score	–
[147]	Bitcoin, Litecoin, StockTwits	2015–2018	OCHLV, technical indicators, sentiment analysis	CNN, LSTM, State Frequency Model	MSE	Keras, Tensorflow
[148]	Bitcoin	2013–2016	Price data	Bayesian optimized RNN, LSTM	Sensitivity, specificity, precision, accuracy, RMSE	Keras, Python, Hyperas

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:

Financial sentiment studies coupled with text mining for forecasting

Art.	Data set	Period	Feature set	Method	Performance criteria	Env.
[137]	Analyst reports on the TSE and Osaka Exchange	2016–2018	Text	LSTM, CNN, Bi-LSTM	Accuracy, R^2	R, Python, MeCab
[150]	Sina Weibo, Stock market records	2012–2015	Technical indicators, sentences	DRSE	F1-score, precision, recall, accuracy, AUROC	Python
[151]	News from Reuters and Bloomberg for S&P500 stocks	2006–2015	Financial news, price data	DeepClue	Accuracy	Dynet software
[152]	News from Reuters and Bloomberg, Historical stock security data	2006–2013	News, price data	DMLP	Accuracy	–
[153]	SCI prices	2008–2015	OCHL of change rate, price	Emotional Analysis + LSTM	MSE	–
[154]	SCI prices	2013–2016	Text data and Price data	LSTM	Accuracy, F1-Measure	Python, Keras
[155]	Stocks of Google, Microsoft and Apple	2016–2017	Twitter sentiment and stock prices	RNN	–	Spark, Flume,Twitter API,
[156]	30 DJIA stocks, S&P500, DJI, news from Reuters	2002–2016	Price data and features from news articles	LSTM, NN, CNN and word2vec	Accuracy	VADER
[157]	Stocks of CSI300 index, OCHLV of CSI300 index	2009–2014	Sentiment Posts, Price data	Naive Bayes + LSTM	Precision, Recall, F1-score, Accuracy	Python, Keras
[158]	S&P500, NYSE Composite, DJIA, NASDAQ Composite	2009–2011	Twitter moods, index data	DNN, CNN	Error rate	Keras, Theano

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:

Text mining studies without sentiment analysis for forecasting

Art.	Data set	Period	Feature set	Method	Performance criteria	Env.
[68]	Energy-Sector/ Company-Centric Tweets in S&P500	2015–2016	Text and Price data	RNN, KNN, SVR, LinR	Return, SR, precision, recall, accuracy	Python, Tweepy API
[165]	News from Reuters, Bloomberg	2006–2013	Financial news, price data	Bi-GRU	Accuracy	Python, Keras
[166]	News from Sina.com, ACE2005 Chinese corpus	2012–2016	A set of news text	Their unique algorithm	Precision, Recall, F1-score	–
[167]	CDAX stock market data	2010–2013	Financial news, stock market data	LSTM	MSE, RMSE, MAE, Accuracy, AUC	TensorFlow, Theano, Python, Scikit-Learn
[168]	Apple, Airbus, Amazon news from Reuters, Bloomberg, S&P500 stock prices	2006–2013	Price data, news, technical indicators	TGRU, stock2vec	Accuracy, precision, AUROC	Keras, Python
[169]	S&P500 Index, 15 stocks in S&P500	2006–2013	News from Reuters and Bloomberg	CNN	Accuracy, MCC	–
[170]	S&P500 index news from Reuters	2006–2013	Financial news titles, Technical indicators	SI-RCNN (LSTM + CNN)	Accuracy	–
[171]	10 stocks in Nikkei 225 and news	2001–2008	Textual information and Stock prices	Paragraph Vector + LSTM	Profit	–
[172]	NIFTY50 Index, NIFTY Bank/Auto/IT/Energy Index, News	2013–2017	Index data, news	LSTM	MCC, Accuracy	–
[173]	Price data, index data, news, social media data	2015	Price data, news from articles and social media	Coupled matrix and tensor	Accuracy, MCC	Jieba
[174]	HS300	2015–2017	Social media news, price data	RNN-Boost with LDA	Accuracy, MAE, MAPE, RMSE	Python, Scikit-learn

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:

Text mining studies without sentiment analysis for forecasting

Art.	Data set	Period	Feature set	Method	Performance criteria	Env.
[175]	News and Chinese stock data	2014–2017	Selected words in a news	HAN	Accuracy, Annual return	–
[176]	News, stock prices from Hong Kong Stock Exchange	2001	Price data and TF-IDF from news	ELM, DLR, PCA, BELM, KELM, NN	Accuracy	Matlab
[177]	TWSE index, 4 stocks in TWSE	2001–2017	Technical indicators, Price data, News	CNN + LSTM	RMSE, Profit	Keras, Python, TALIB
[178]	Stock of Tsugami Corporation	2013	Price data	LSTM	RMSE	Keras, Tensorflow
[179]	News, Nikkei Stock Average and 10-Nikkei companies	1999–2008	news, MACD	RNN, RBM+DBN	Accuracy, *P*-value	–
[180]	ISMIS 2017 Data Mining Competition dataset	–	Expert identifier, classes	LSTM + GRU + FFNN	Accuracy	–
[181]	Reuters, Bloomberg News, S&P500 price	2006–2013	News and sentences	LSTM	Accuracy	–
[182]	APPL from S&P500 and news from Reuters	2011–2017	Input news, OCHLV, Technical indicators	CNN + LSTM, CNN+SVM	Accuracy, F1-score	Tensorflow
[183]	Nikkei225, S&P500, news from Reuters and Bloomberg	2001–2013	Stock price data and news	DGM	Accuracy, MCC, %profit	–
[184]	Stocks from S&P500	2006–2013	Text (news) and Price data	LAR+News, RF+News	MAPE, RMSE	–

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:

Financial sentiment studies coupled with text mining without forecasting

Art.	Data set	Period	Feature set	Method	Performance criteria	Env.
[85]	883 BHC from EDGAR	2006–2017	Tokens, weighted sentiment polarity, leverage and ROA	CNN, LSTM, SVM, Random Forest	Accuracy, Precision, Recall, F1-score	Keras, Python, Scikit-learn
[185]	SemEval-2017 dataset, financial text, news, stock market data	2017	Sentiments in Tweets, News headlines	Ensemble SVR, CNN, LSTM, GRU	Cosine similarity score, agreement score, class score	Python, Keras, Scikit Learn
[186]	Financial news from Reuters	2006–2015	Word vector, Lexical and Contextual input	Targeted dependency tree LSTM	Cumulative abnormal return	–
[187]	Stock sentiment analysis from StockTwits	2015	StockTwits messages	LSTM, Doc2Vec, CNN	Accuracy, precision, recall, f-measure, AUC	–
[188]	Sina Weibo, Stock market records	2012–2015	Technical indicators, sentences	DRSE	F1-score, precision, recall, accuracy, AUROC	Python
[189]	News from NowNews, AppleDaily, LTN, MoneyDJ for 18 stocks	2013–2014	Text, Sentiment	LSTM, CNN	Return	Python, Tensorflow
[190]	StockTwits	2008–2016	Sentences, StockTwits messages	CNN, LSTM, GRU	MCC, WSURT	Keras, Tensorflow
[191]	Financial statements of Japan companies	–	Sentences, text	DMLP	Precision, recall, f-score	–
[192]	Twitter posts, news headlines	–	Sentences, text	Deep-FASP	Accuracy, MSE, R^2	–
[193]	Forums data	2004–2013	Sentences and keywords	Recursive neural tensor networks	Precision, recall, f-measure	–
[194]	News from Financial Times related US stocks	–	Sentiment of news headlines	SVR, Bidirectional LSTM	Cosine similarity	Python, Scikit Learn, Keras, Tensorflow

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:
Other text mining studies

Art.	Data set	Period	Feature set	Method	Performance criteria	**Env.**
[72]	News from NowNews, AppleDaily, LTN, MoneyDJ for 18 stocks	2013–2014	Text, Sentiment	DMLP	Return	Python, Tensorflow
[86]	The event data set for large European banks, news articles from Reuters	2007–2014	Word, sentence	DMLP +NLP preprocess	Relative usefulness, F1-score	–
[87]	Event dataset on European banks, news from Reuters	2007–2014	Text, sentence	Sentence vector + DFFN	Usefulness, F1-score, AUROC	–
[88]	News from Reuters, fundamental data	2007–2014	Financial ratios and news text	doc2vec + NN	Relative usefulness	Doc2vec
[121]	Real-world data for automobile insurance company labeled as fradulent	–	Car, insurance and accident related features	DMLP + LDA	TP, FP, Accuracy, Precision, F1-score	–
[123]	Financial transactions	–	Transaction data	LSTM	t-SNE	–
[195]	Taiwan's National Pension Insurance	2008–2014	Insured's id, area-code, gender, etc.	RNN	Accuracy, total error	Python
[196]	StockTwits	2015–2016	Sentences, StockTwits messages	Doc2vec, CNN	Accuracy, precision, recall, f-measure, AUC	Python, Tensorflow

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications: Other theoretical or conceptual studies

Art.	SubTopic	IsTimeSeries?	Data set	Period	Feature set	Method
[197]	Analysis of AE, SVD	Yes	Selected stocks from the IBB index and stock of Amgen Inc.	2012–2014	Price data	AE, SVD
[198]	Fraud Detection in Banking	No	Risk Management / Fraud Detection	–	–	DRL

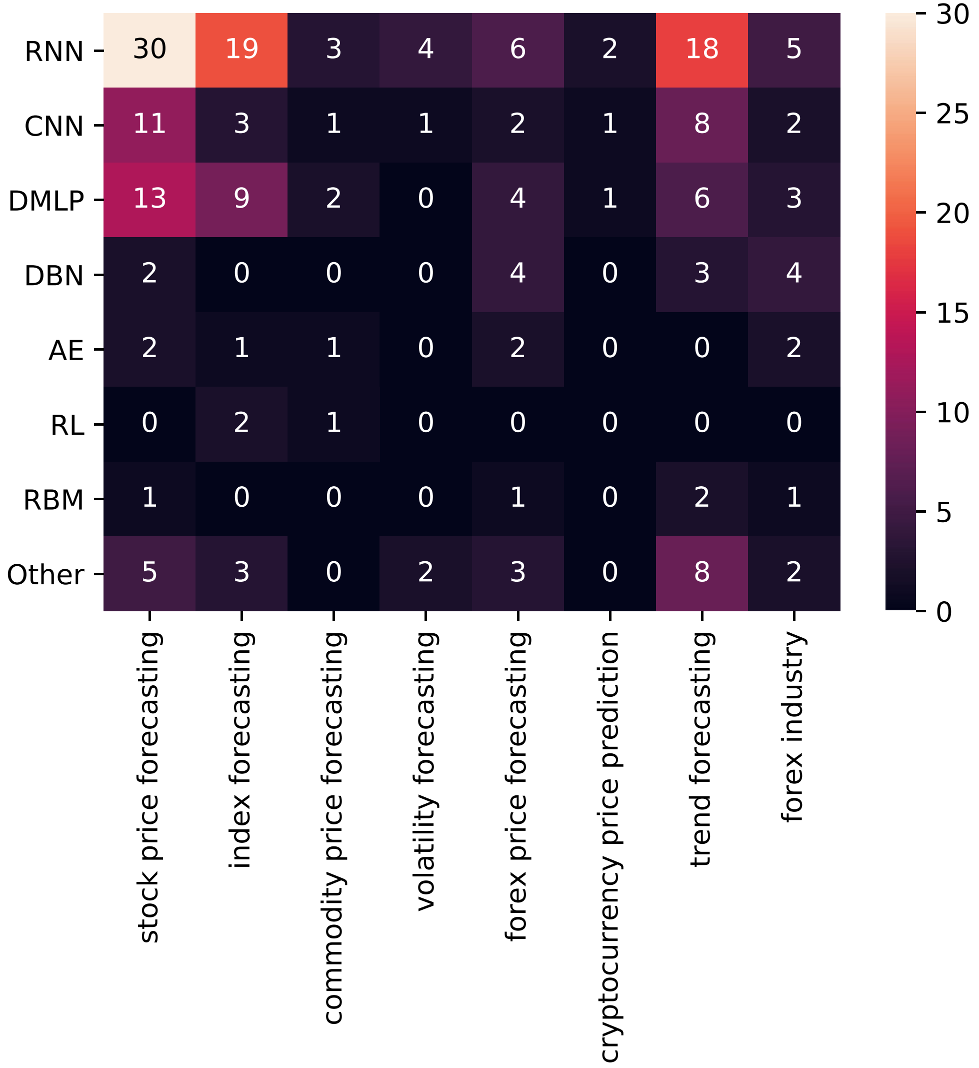
Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications:
Other financial applications

Art.	Subtopic	Data set	Period	Feature set	Method	Performance criteria	Env.
[47]	Improving trading decisions	S&P500, KOSPI, HSI, and EuroStoxx50	1987–2017	200-days stock price	Deep Q-Learning and DMLP	Total profit, Correlation	–
[193]	Identifying Top Sellers In Underground Economy	Forums data	2004–2013	Sentences and keywords	Recursive neural tensor networks	Precision, recall, f-measure	–
[195]	Predicting Social Ins. Payment Behavior	Taiwan's National Pension Insurance	2008–2014	Insured's id, area-code, gender, etc.	RNN	Accuracy, total error	Python
[199]	Speedup	45 CME listed commodity and FX futures	1991–2014	Price data	DNN	–	–
[200]	Forecasting Fundamentals	Stocks in NYSE, NASDAQ or AMEX exchanges	1970–2017	16 fundamental features from balance sheet	DMLP, LFM	MSE, Compound annual return, SR	–
[201]	Predicting Bank Telemarketing	Phone calls of bank marketing data	2008–2010	16 finance-related attributes	CNN	Accuracy	–
[202]	Corporate Performance Prediction	22 pharmaceutical companies data in US stock market	2000–2015	11 financial and 4 patent indicator	RBM, DBN	RMSE, profit	–

Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Financial time series forecasting with deep learning: Topic-model heatmap



Source: Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu (2020), "Financial time series forecasting with deep learning: A systematic literature review: 2005–2019." Applied Soft Computing 90 (2020): 106181.

Stock price forecasting using only raw time series data

Art.	Data set	Period	Feature set	Lag	Horizon	Method	Performance criteria	Env.
[80]	38 stocks in KOSPI	2010–2014	Lagged stock returns	50 min	5 min	DNN	NMSE, RMSE, MAE, MI	–
[81]	China stock market, 3049 Stocks	1990–2015	OCHLV	30 d	3 d	LSTM	Accuracy	Theano, Keras
[82]	Daily returns of 'BRD' stock in Romanian Market	2001–2016	OCHLV	–	1 d	LSTM	RMSE, MAE	Python, Theano
[83]	297 listed companies of CSE	2012–2013	OCHLV	2 d	1 d	LSTM, SRNN, GRU	MAD, MAPE	Keras
[84]	5 stock in NSE	1997–2016	OCHLV, Price data, turnover and number of trades.	200 d	1..10 d	LSTM, RNN, CNN, MLP	MAPE	–
[85]	Stocks of Infosys, TCS and CIPLA from NSE	2014	Price data	–	–	RNN, LSTM and CNN	Accuracy	–
[86]	10 stocks in S&P500	1997–2016	OCHLV, Price data	36 m	1 m	RNN, LSTM, GRU	Accuracy, Monthly return	Keras, Tensorflow
[87]	Stocks data from S&P500	2011–2016	OCHLV	1 d	1 d	DBN	MSE, norm-RMSE, MAE	–
[88]	High-frequency transaction data of the CSI300 futures	2017	Price data	–	1 min	DNN, ELM, RBF	RMSE, MAPE, Accuracy	Matlab
[89]	Stocks in the S&P500	1990–2015	Price data	240 d	1 d	DNN, GBT, RF	Mean return, MDD, Calmar ratio	H2O
[90]	ACI Worldwide, Staples, and Seagate in NASDAQ	2006–2010	Daily closing prices	17 d	1 d	RNN, ANN	RMSE	–
[91]	Chinese Stocks	2007–2017	OCHLV	30 d	1..5 d	CNN + LSTM	Annualized Return, Mxm Retracement	Python
[92]	20 stocks in S&P500	2010–2015	Price data	–	–	AE + LSTM	Weekly Returns	–
[93]	S&P500	1985–2006	Monthly and daily log-returns	*	1 d	DBN+MLP	Validation, Test Error	Theano, Python, Matlab
[94]	12 stocks from SSE Composite Index	2000–2017	OCHLV	60 d	1..7 d	DWNN	MSE	Tensorflow
[95]	50 stocks from NYSE	2007–2016	Price data	–	1d, 3 d, 5 d	SFM	MSE	–

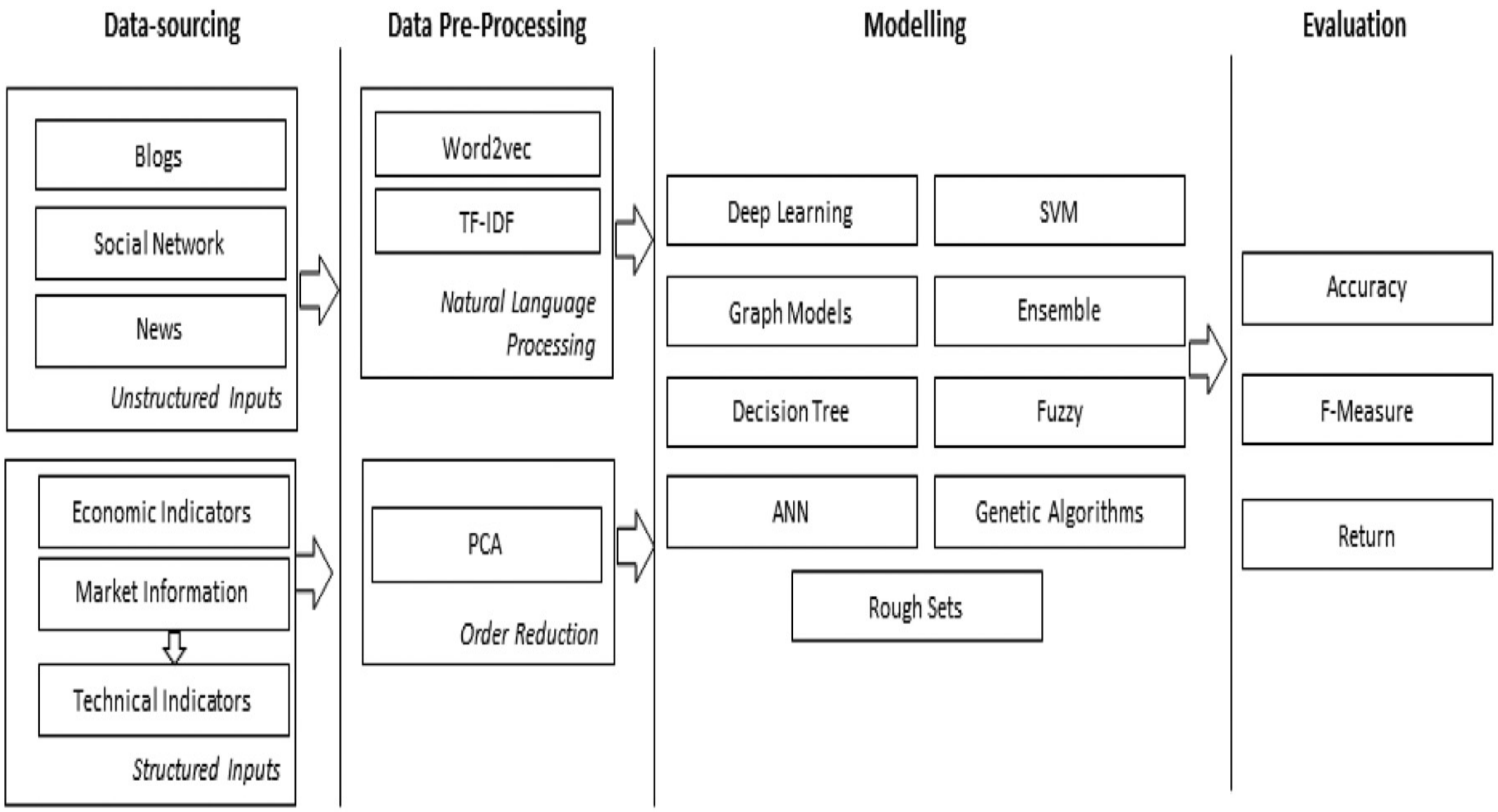
Source: Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu (2020), "Financial time series forecasting with deep learning: A systematic literature review: 2005–2019." Applied Soft Computing 90 (2020): 106181.

Stock price forecasting using various data

Art.	Data set	Period	Feature set	Lag	Horizon	Method	Performance criteria	Env.
[96]	Japan Index constituents from WorldScope	1990–2016	25 Fundamental Features	10 d	1 d	DNN	Correlation, Accuracy, MSE	Tensorflow
[97]	Return of S&P500	1926–2016	Fundamental Features:	–	1 s	DNN	MSPE	Tensorflow
[98]	U.S. low-level disaggregated macroeconomic time series	1959–2008	GDP, Unemployment rate, Inventories, etc.	–	–	DNN	R^2	–
[99]	CDAX stock market data	2010–2013	Financial news, stock market data	20 d	1 d	LSTM	MSE, RMSE, MAE, Accuracy, AUC	TensorFlow, Theano, Python, Scikit-Learn
[100]	Stock of Tsugami Corporation	2013	Price data	–	–	LSTM	RMSE	Keras, Tensorflow
[101]	Stocks in China's A-share	2006–2007	11 technical indicators	–	1 d	LSTM	AR, IR, IC	–
[102]	SCI prices	2008–2015	OCHL of change rate, price	7 d	–	EmotionalAnalysis + LSTM	MSE	–
[103]	10 stocks in Nikkei 225 and news	2001–2008	Textual information and Stock prices	10 d	–	Paragraph Vector + LSTM	Profit	–
[104]	TKC stock in NYSE and QQQQ ETF	1999–2006	Technical indicators, Price	50 d	1 d	RNN (Jordan–Elman)	Profit, MSE	Java
[105]	10 Stocks in NYSE	–	Price data, Technical indicators	20 min	1 min	LSTM, MLP	RMSE	–
[106]	42 stocks in China's SSE	2016	OCHLV, Technical Indicators	242 min	1 min	GAN (LSTM, CNN)	RMSRE, DPA, GAN-F, GAN-D	–
[107]	Google's daily stock data	2004–2015	OCHLV, Technical indicators	20 d	1 d	$(2D)^2$ PCA + DNN	SMAPE, PCD, MAPE, RMSE, HR, TR, R^2	R, Matlab
[108]	GarantiBank in BIST, Turkey	2016	OCHLV, Volatility, etc.	–	–	PLR, Graves LSTM	MSE, RMSE, MAE, RSE, R^2	Spark
[109]	Stocks in NYSE, AMEX, NASDAQ, TAQ intraday trade	1993–2017	Price, 15 firm characteristics	80 d	1 d	LSTM+MLP	Monthly return, SR	Python,Keras, Tensorflow in AWS
[110]	Private brokerage company's real data of risky transactions	–	250 features: order details, etc.	–	–	CNN, LSTM	F1-Score	Keras, Tensorflow
[111]	Fundamental and Technical Data, Economic Data	–	Fundamental , technical and market information	–	–	CNN	–	–
[112]	The LOB of 5 stocks of Finnish Stock Market	2010	FI-2010 dataset: bid/ask and volume	–	*	WMTR, MDA	Accuracy, Precision, Recall, F1-Score	–
[113]	Returns in NYSE, AMEX, NASDAQ	1975–2017	57 firm characteristics	*	–	Fama–French n-factor model DL	R^2, RMSE	Tensorflow

Source: Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu (2020), "Financial time series forecasting with deep learning: A systematic literature review: 2005–2019." Applied Soft Computing 90 (2020): 106181.

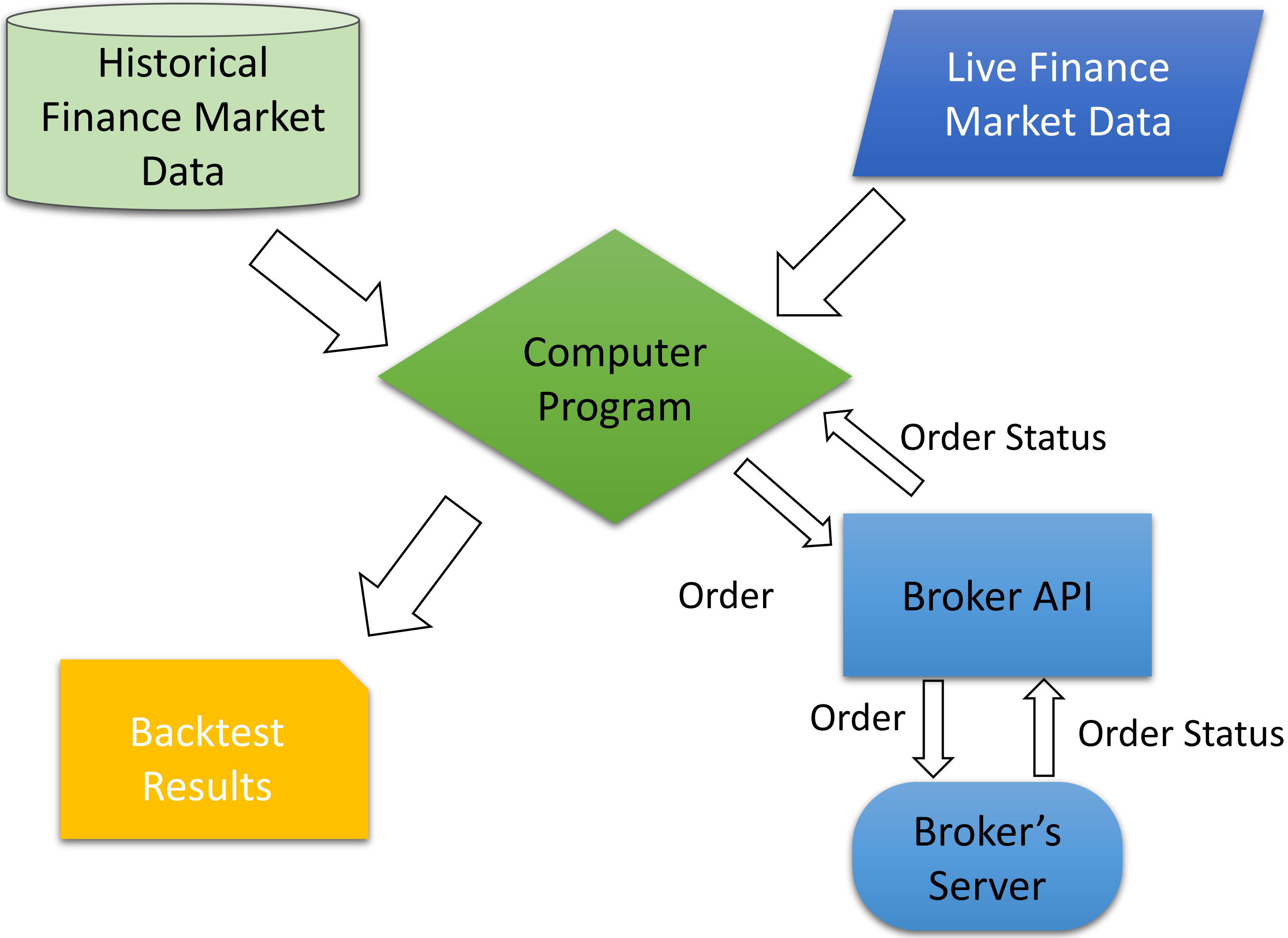
Stock Market Movement Forecast:
Phases of the stock market modeling



Source: O. Bustos and A. Pomares-Quimbaya (2020), "Stock Market Movement Forecast: A Systematic Review." Expert Systems with Applications (2020): 113464.

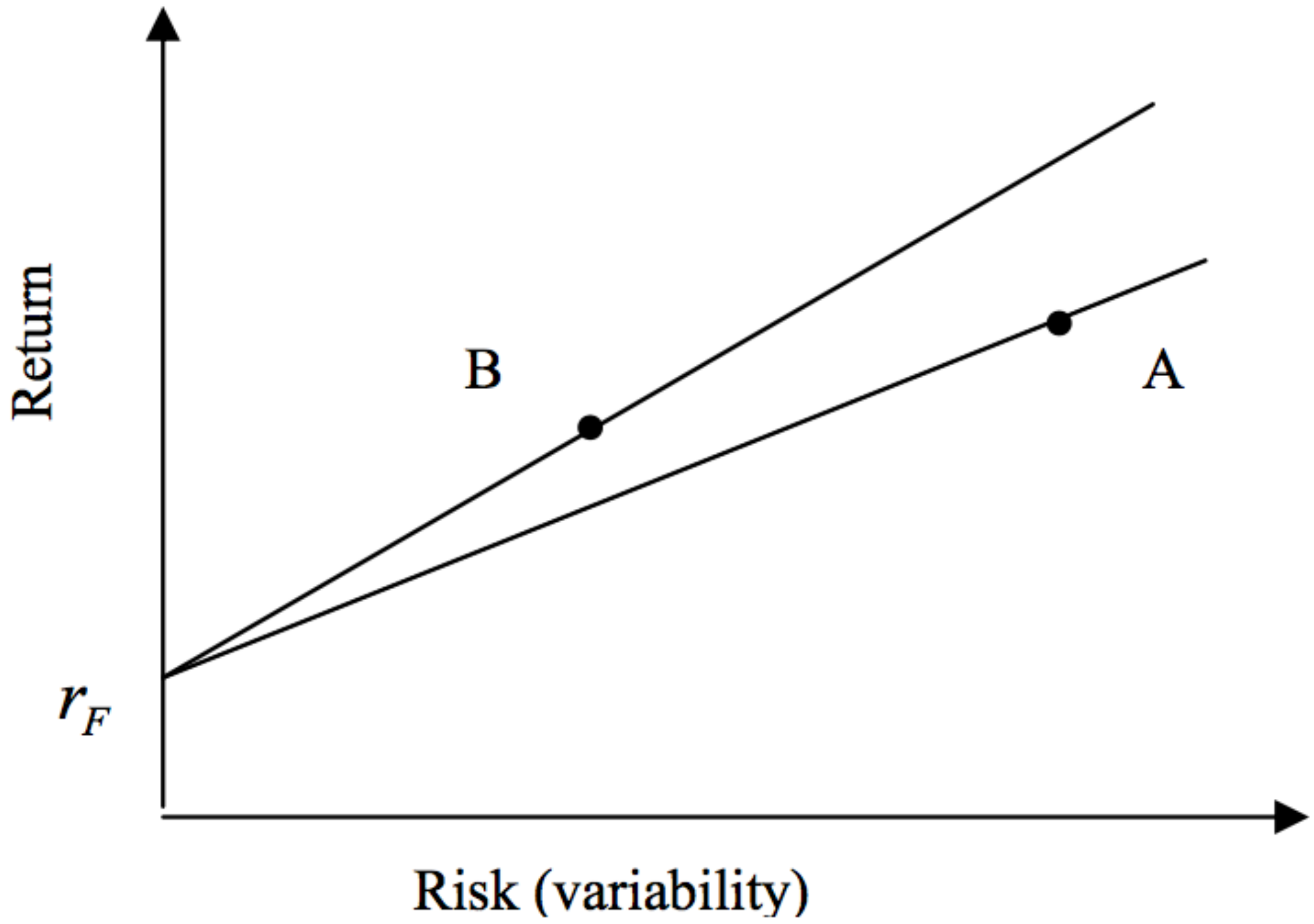
Algorithmic Trading

Algorithmic Trading



Source: Ernest P. Chan (2017), Machine Trading: Deploying Computer Algorithms to Conquer the Markets, Wiley

Risk and Return



Source: Bacon, Carl. "How sharp is the Sharpe-ratio?-Risk-adjusted Performance Measures." *Statpro White Paper* (2000).

Sharpe Ratio

$$\textbf{Sharpe Ratio} = \frac{Portofolio\ Return - Risk\ Free\ Return}{Portofolio\ Risk}$$

Source: Bacon, Carl. "How sharp is the Sharpe-ratio?-Risk-adjusted Performance Measures." *Statpro White Paper* (2000).

Sharpe Ratio

$$\textbf{Sharpe Ratio } SR = \frac{r_P - r_F}{\sigma_P}$$

Where

r_P = portfolio return

r_F = risk free rate

σ_P = portfolio risk (variability, standard deviation of return)

Source: Bacon, Carl. "How sharp is the Sharpe-ratio?-Risk-adjusted Performance Measures." *Statpro White Paper* (2000).

Sortino Ratio

$$\textbf{Sortino Ratio} = \frac{r_P - r_T}{\sigma_D}$$

Where

r_P = portfolio return

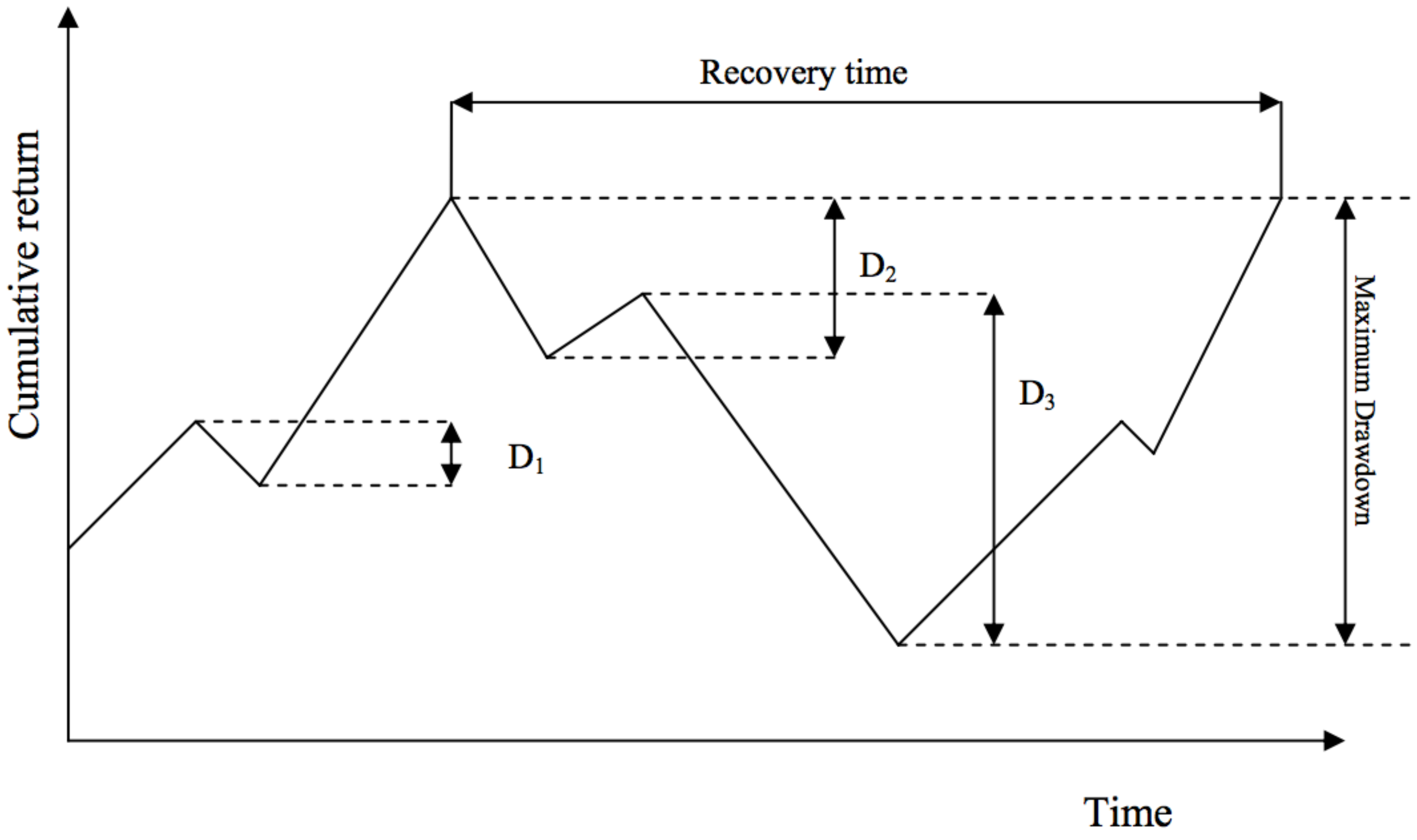
r_T = Minimum Target Return

σ_D = Downside Risk

$$\textbf{Downside Risk } \sigma_D = \sqrt{\sum_{i=1}^{n} \frac{\min[(r_i - rT), 0]^2}{n}}$$

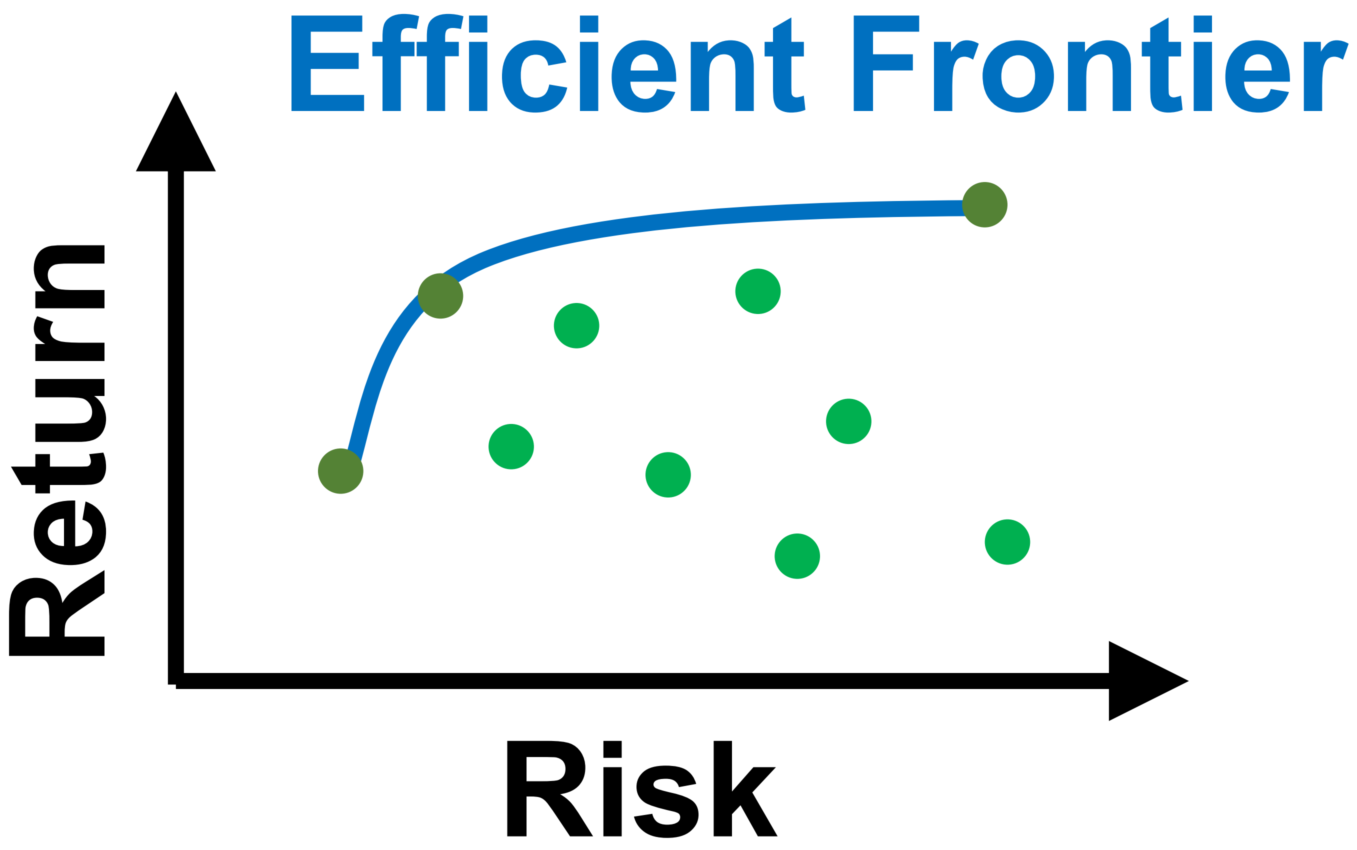
Source: Bacon, Carl. "How sharp is the Sharpe-ratio?-Risk-adjusted Performance Measures." *Statpro White Paper* (2000).

Max Drawdown



Source: Bacon, Carl. "How sharp is the Sharpe-ratio?-Risk-adjusted Performance Measures." *Statpro White Paper* (2000).

Portfolio Optimization
Efficient Frontier



Source: Tucker Balch (2012), Investment Science: Portfolio Optimization, https://www.youtube.com/watch?v=5qbMhXXq0vI

Backtesting

- Financial Functions (ffn)
  - https://pmorissette.github.io/ffn/
- backtesting.py
  - https://kernc.github.io/backtesting.py/
- Visualization
  - Plotly Express (px)
    - https://plotly.com/python/plotly-express/
  - Bokeh
    - https://bokeh.org/

Financial Functions (ffn)
plotly.express (px)

```
!pip install ffn
import ffn
import plotly.express as px
%pylab inline
#BTC-USD Bitcoin USD
df = ffn.get('btc-usd', start='2016-01-01', end='2021-12-31')
print('df')
print(df.head())
print(df.tail())
print(df.describe())
df.plot(figsize=(14,10))

returns = df.to_returns().dropna()
print('returns')
print(returns.head())
print(returns.tail())
print(returns.describe())
#ax = df.plot(figsize=(12,9))

perf = df.calc_stats()
perf.plot(figsize=(14, 10))
print(perf.display())

fig = px.line(df, x=df.index, y="btcusd", title='btcusd')
fig.update_layout(title='btcusd price', xaxis_title='Date', yaxis_title='Price')
#fig.update_traces(mode='markers+lines')
fig.show()

fig = px.line(returns, x=returns.index, y="btcusd", title='btcusd')
fig.update_layout(title='btcusd returns', xaxis_title='Date', yaxis_title='Returns')
fig.show()

fig = px.histogram(returns, x='btcusd', nbins=40, histnorm='probability', width=800, height=400)
fig.update_layout(title='btcusd returns histogram')
fig.show()

fig = px.box(returns, y='btcusd', points = 'all')
fig.update_layout(title='btcusd returns box')
fig.update_traces(boxmean='sd')
fig.show()
```

Financial Functions (ffn)
plotly.express (px)

```
# Upgrade pandas-datareader
!pip install --upgrade pandas
!pip install --upgrade pandas-datareader
```

```
!pip install ffn
import ffn
import plotly.express as px
%pylab inline
#BTC-USD Bitcoin USD
df = ffn.get('btc-usd', start='2016-01-01', end='2021-12-31')
print('df')
print(df.head())
print(df.tail())
print(df.describe())
df.plot(figsize=(14,10))
```

Financial Functions (ffn) plotly.express (px)

```
returns = df.to_returns().dropna()
print('returns')
print(returns.head())
print(returns.tail())
print(returns.describe())
#ax = df.plot(figsize=(12,9))
```

Financial Functions (ffn)
plotly.express (px)

```
perf = df.calc_stats()
perf.plot(figsize=(14, 10))
print(perf.display())

fig = px.line(df, x=df.index, y="btcusd", title='btcusd')
fig.update_layout(title='btcusd price', xaxis_title='Date',
yaxis_title='Price')
#fig.update_traces(mode='markers+lines')
fig.show()

fig = px.line(returns, x=returns.index, y="btcusd", title='btcusd')
fig.update_layout(title='btcusd returns', xaxis_title='Date',
yaxis_title='Returns')
fig.show()
```

Financial Functions (ffn)
plotly.express (px)

```
fig = px.histogram(returns, x='btcusd', nbins=40,
histnorm='probability', width=800, height=400)

fig.update_layout(title='btcusd returns histogram')
fig.show()

fig = px.box(returns, y='btcusd', points = 'all')
fig.update_layout(title='btcusd returns box')
fig.update_traces(boxmean='sd')
fig.show()
```

Financial Functions (ffn)

```
 btcusd
Date
2016-01-01  434.334015
2016-01-02  433.437988
2016-01-03  430.010986
2016-01-04  433.091003
2016-01-05  431.959991
                  btcusd
Date
2021-12-28  47588.855469
2021-12-29  46444.710938
2021-12-30  47178.125000
2021-12-31  46306.445312
2022-01-01  47686.812500
             btcusd
count   2193.000000
mean   13025.164562
std    16489.530523
min      364.330994
25%     2589.409912
50%     7397.796875
75%    11358.662109
max    67566.828125
```

Financial Functions (ffn)

calc_stats() display()

```
Stat                   btcusd
-------------------    ----------
Start                  2016-01-01
End                    2022-01-01
Risk-free rate         0.00%

Total Return           10879.29%
Daily Sharpe           1.18
Daily Sortino          1.95
CAGR                   118.79%
Max Drawdown           -83.40%
Calmar Ratio           1.42
```

Financial Functions (ffn)

calc_stats() display()

```
MTD                   2.98%
3m                    -0.89%
6m                    42.04%
YTD                   2.98%
1Y                    62.34%
3Y (ann.)             131.46%
5Y (ann.)             116.71%
10Y (ann.)            -
Since Incep. (ann.)   118.79%
```

Financial Functions (ffn)

`calc_stats() display()`

```
Daily Sharpe          1.18
Daily Sortino         1.95
Daily Mean (ann.)     74.04%
Daily Vol (ann.)      62.94%
Daily Skew            -0.10
Daily Kurt            7.30
Best Day              25.25%
Worst Day             -37.17%
```

Financial Functions (ffn)

`calc_stats() display()`

```
Monthly Sharpe        1.38
Monthly Sortino       3.75
Monthly Mean (ann.)   114.20%
Monthly Vol (ann.)    82.59%
Monthly Skew          0.43
Monthly Kurt          -0.16
Best Month            69.63%
Worst Month           -36.41%
```

Financial Functions (ffn)

calc_stats() display()

```
Yearly Sharpe           0.54
Yearly Sortino          9.73
Yearly Mean             292.22%
Yearly Vol              542.38%
Yearly Skew             2.17
Yearly Kurt             4.86
Best Year               1368.90%
Worst Year              -73.56%
```

Financial Functions (ffn)

calc_stats() display()

```
Avg. Drawdown           -10.25%
Avg. Drawdown Days      36.55
Avg. Up Month           25.13%
Avg. Down Month         -12.35%
Win Year %              83.33%
Win 12m %               85.48%
```

Visualization
plotly.express (px)

Backtesting Output

```
backtesing output
Start                     2016-01-01 00:00:00
End                       2022-01-01 00:00:00
Duration                   2192 days 00:00:00
Exposure Time [%]                   97.993616
Equity Final [$]               4237449.058157
Equity Peak [$]                6165339.439633
Return [%]                        4137.449058
Buy & Hold Return [%]            10879.294935
Return (Ann.) [%]                   86.557668
Volatility (Ann.) [%]              144.748975
Sharpe Ratio                         0.597985
Sortino Ratio                        1.946086
Calmar Ratio                         1.362652
Max. Drawdown [%]                  -63.521467
Avg. Drawdown [%]                  -12.142095
Max. Drawdown Duration      557 days 00:00:00
Avg. Drawdown Duration       44 days 00:00:00
# Trades                                  116
Win Rate [%]                        35.344828
Best Trade [%]                     119.026467
Worst Trade [%]                    -23.393531
Avg. Trade [%]                       3.291328
Max. Trade Duration          74 days 00:00:00
Avg. Trade Duration          19 days 00:00:00
Profit Factor                        2.293983
Expectancy [%]                       5.036865
SQN                                  1.236071
_strategy                            SmaCross
_equity_curve                             ...
_trades                          Size  Entry..
```

describe()

	High	Low	Open	Close	Volume	Adj Close
count	2193.00	2193.00	2193.00	2193.00	2.193000e+03	2193.00
mean	13363.00	12616.08	13005.79	13025.16	1.757591e+10	13025.16
std	16935.24	15960.65	16480.00	16489.53	2.085247e+10	16489.53
min	374.95	354.91	365.07	364.33	2.851400e+07	364.33
25%	2682.26	2510.48	2577.77	2589.41	1.182870e+09	2589.41
50%	7535.72	7233.40	7397.13	7397.80	9.175292e+09	7397.80
75%	11570.79	11018.13	11354.30	11358.66	2.886756e+10	11358.66
max	68789.62	66382.06	67549.73	67566.83	3.509679e+11	67566.83

Backtesting

```
# Upgrade pandas-datareader
!pip install --upgrade pandas
!pip install --upgrade pandas-datareader
```

```
!pip install backtesting
from backtesting import Backtest, Strategy
from backtesting.lib import crossover
from backtesting.test import SMA

import pandas as pd
import pandas_datareader.data as web
df = web.DataReader("BTC-USD", 'yahoo', '2016-01-01', '2021-12-31')
df.to_csv('BTC-USD.csv')
print(df.head().round(2))
print(df.tail().round(2))
print(df.describe().round(2))

class SmaCross(Strategy):
      n1 = 5
      n2 = 20

      def init(self):
            close = self.data.Close
            self.sma1 = self.I(SMA, close, self.n1)
            self.sma2 = self.I(SMA, close, self.n2)

      def next(self):
            if crossover(self.sma1, self.sma2):
                self.buy()
            elif crossover(self.sma2, self.sma1):
                self.sell()

bt = Backtest(df, SmaCross, cash=100000, commission=.002, exclusive_orders=True)

output = bt.run()
print('backtesing output')
print(output)

bt.plot()
```

Backtesting

```
#!pip install backtesting
from backtesting import Backtest, Strategy
from backtesting.lib import crossover
from backtesting.lib import plot_heatmaps
from backtesting.test import SMA

import pandas as pd
import pandas_datareader.data as web

from google.colab import files
import time
#BTC-USD ETH-USD
v_symbol = 'BTC-USD'
v_time_start = '2016-01-01'
v_time_end = '2021-12-31'
v_to_csv_filename = v_symbol + '_' + v_time_start + '_' + v_time_end + '.csv'
df = web.DataReader(v_symbol, 'yahoo', v_time_start, v_time_end)
df.to_csv(v_to_csv_filename)

print(df.head().round(2))
print(df.tail().round(2))
print(df.describe().round(2))
v_n1 = 5 #5 #20 #60 #120
v_n2 = 200 #20 #60 #120 #240
```

Backtesting

```python
class SmaCross(Strategy):
    n1 = v_n1 #5
    n2 = v_n2 #60

    def init(self):
        close = self.data.Close
        self.sma1 = self.I(SMA, close, self.n1)
        self.sma2 = self.I(SMA, close, self.n2)

    def next(self):
        if crossover(self.sma1, self.sma2):
                self.buy()
        elif crossover(self.sma2, self.sma1):
                self.sell()

bt = Backtest(df, SmaCross, cash=100000, commission=.002, exclusive_orders=True)

stats = bt.run()
```

Backtesting

```
filename = v_symbol + '_' + v_time_start + '_' + v_time_end + '_' + 'MA_' +
str(v_n1) + '_' + str(v_n2) + '.csv'
print('filename:', filename)
stats.to_csv(filename)

print('backtesing stats')
print(stats)
bt.plot()

print('filename:\t', filename)
print("stats._strategy:\t", stats._strategy)
print("# Trades:\t", stats['# Trades'])
print("stats['Equity Final [$]']:\t", round(stats['Equity Final [$]'], 4))
print("stats['Avg. Trade [%]']:\t", round(stats['Avg. Trade [%]'], 4))
print("Sharpe Ratio:\t", round(stats['Sharpe Ratio'], 4))

#download file
time.sleep(1) # time sleep 1 second
files.download(filename)
print('file downloaded:', filename)
```

Backtesting

```
print('*****bt.optimize*****')
stats, heatmap = bt.optimize(
    n1 = range(5, 65, 5),
    n2 = range(10, 205, 5),
    constraint = lambda param: param.n1 <param.n2,
    maximize = 'Avg. Trade [%]',
    max_tries = 600,
    random_state = 0,
    return_heatmap = True)

#'Equity Final [$]' 'Avg. Trade [%]'

optimize_strategy = stats._strategy
```

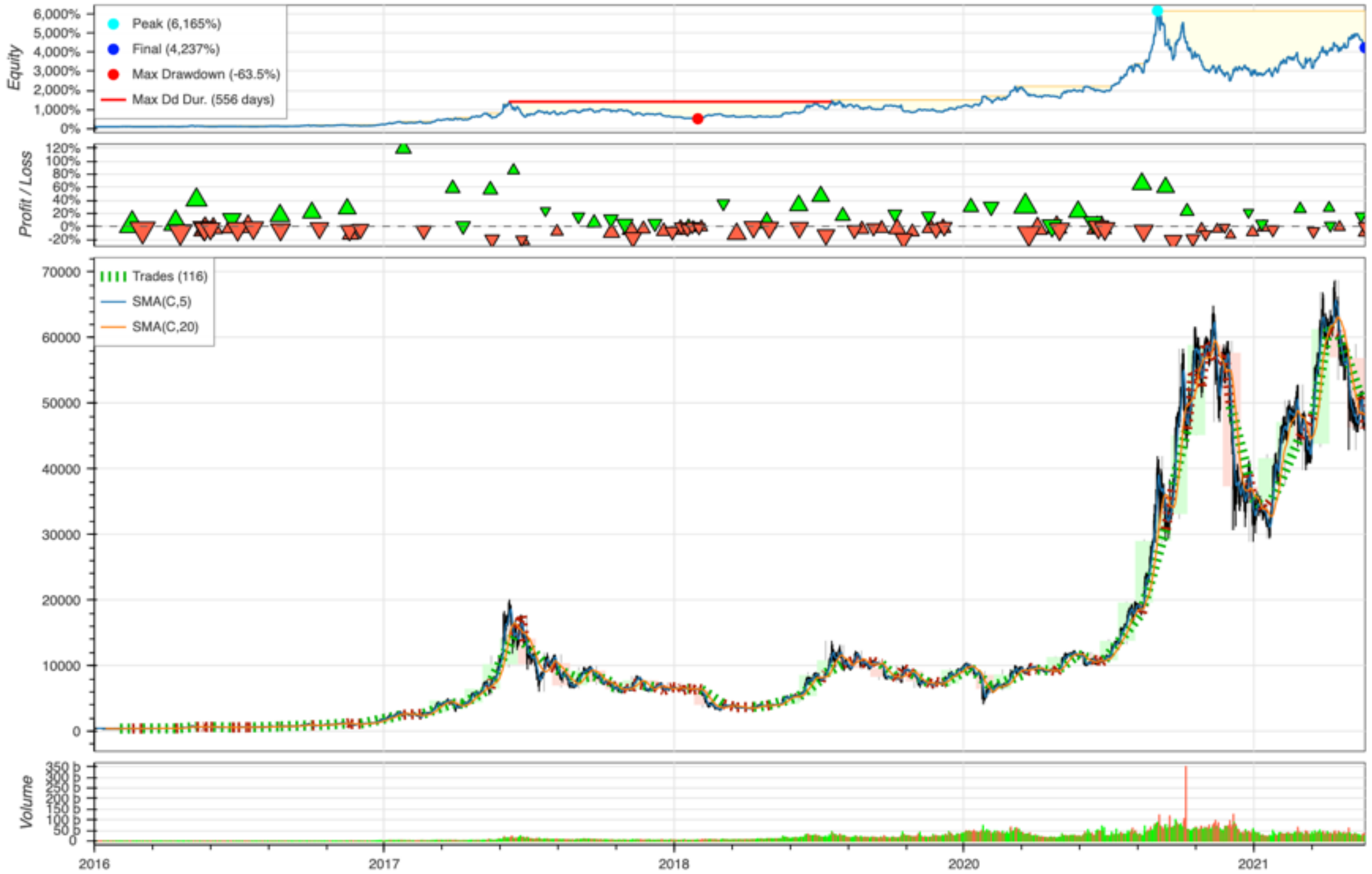
Backtesting

```
optimize_filename = v_symbol + '_' + v_time_start + '_' + v_time_end + '_' +
'bt_optimize_strategy' + str(optimize_strategy) + '.csv'
print('optimize_filename:', optimize_filename)
print('backtesing optimize strategy stats')
print(stats)
stats.to_csv(optimize_filename)
plot_heatmaps(heatmap, agg='mean', plot_width = 1800)

print('backtesting optimize strategy heatmap')
print(heatmap)
print('backtesting optimize strategy heatmap Top 10')
print(heatmap.sort_values().iloc[-10:])
hm = heatmap.groupby(['n1', 'n2']).mean().unstack()
print('backtesting optimize strategy heatmap mean')
print(hm)
hm_filename = v_symbol + '_' + v_time_start + '_' + v_time_end + '_' +
'hm_heatmap.csv'
hm.to_csv(hm_filename)
```

Backtesting

```
print("filename:\t", optimize_filename)
print("stats._strategy:\t", stats._strategy)
print("# Trades:\t", stats['# Trades'])
print("stats['Equity Final [$]']:\t", round(stats['Equity Final [$]'], 4))
print("stats['Avg. Trade [%]']:\t", round(stats['Avg. Trade [%]'], 4))
print("Sharpe Ratio:\t", round(stats['Sharpe Ratio'], 4))

#download file
time.sleep(1) # time sleep 1 second
files.download(hm_filename)
print('file downloaded:', hm_filename)
files.download(optimize_filename)
print('file downloaded:', optimize_filename)
```

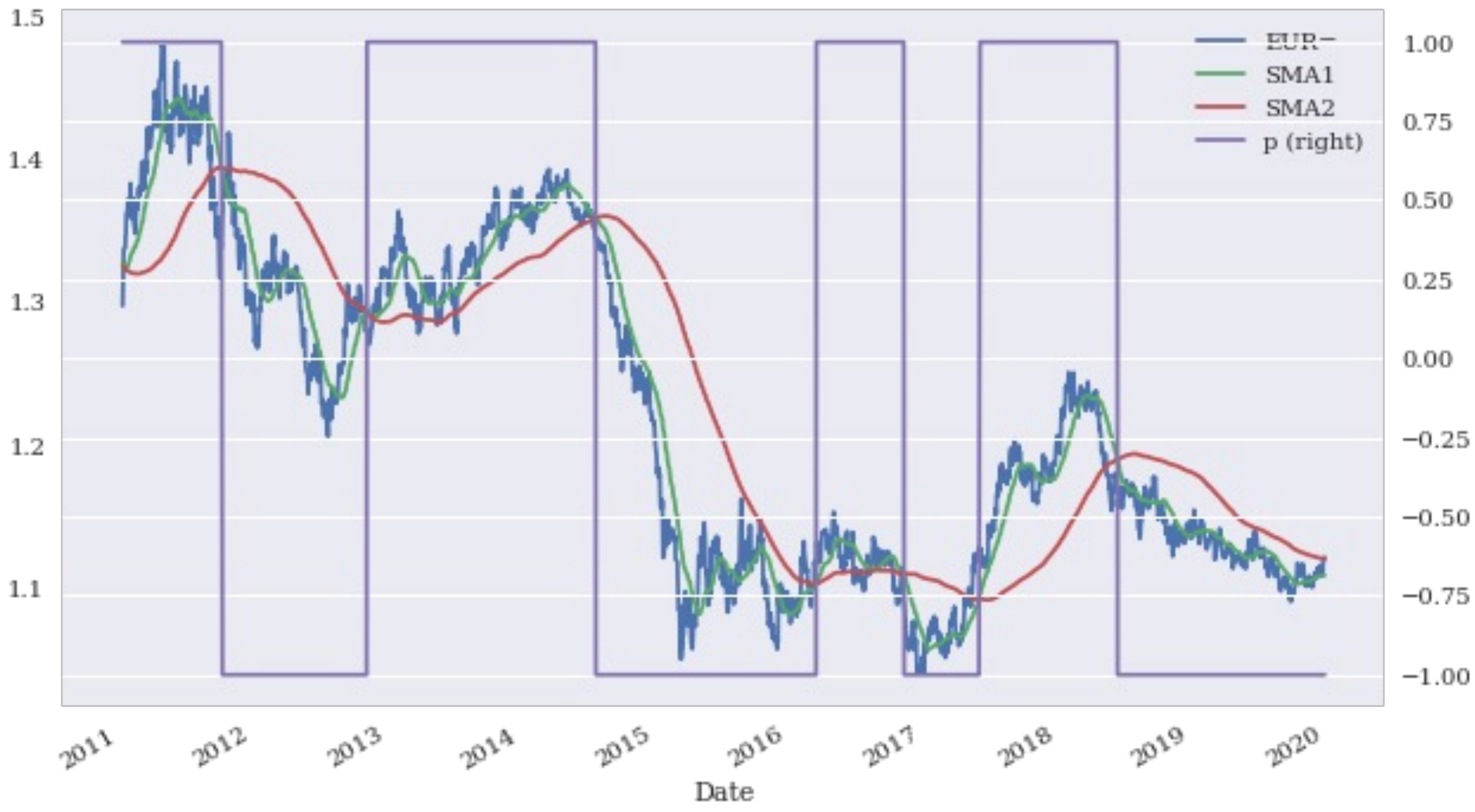
Backtesting



Time series data for EUR/USD and SMAs

Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Time series data for EUR/USD, SMAs, and resulting positions



Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

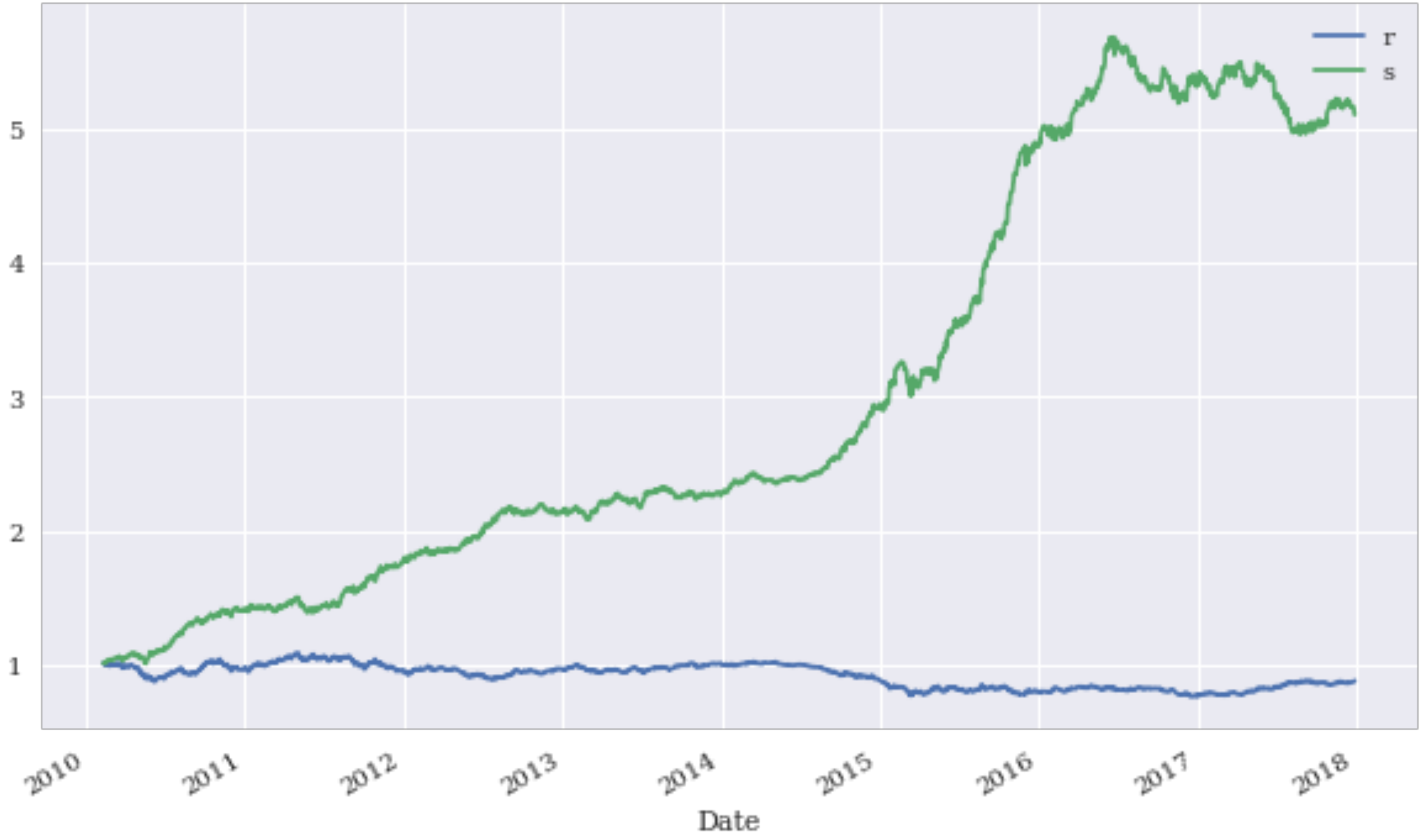
Gross performance of passive benchmark investment and SMA strategy

Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Gross performance of the SMA strategy before and after transaction costs

Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Gross performance of the passive benchmark investment and the daily DNN strategy (in-sample)



Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Gross performance of the passive benchmark investment and the daily DNN strategy (out-of-sample)

Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

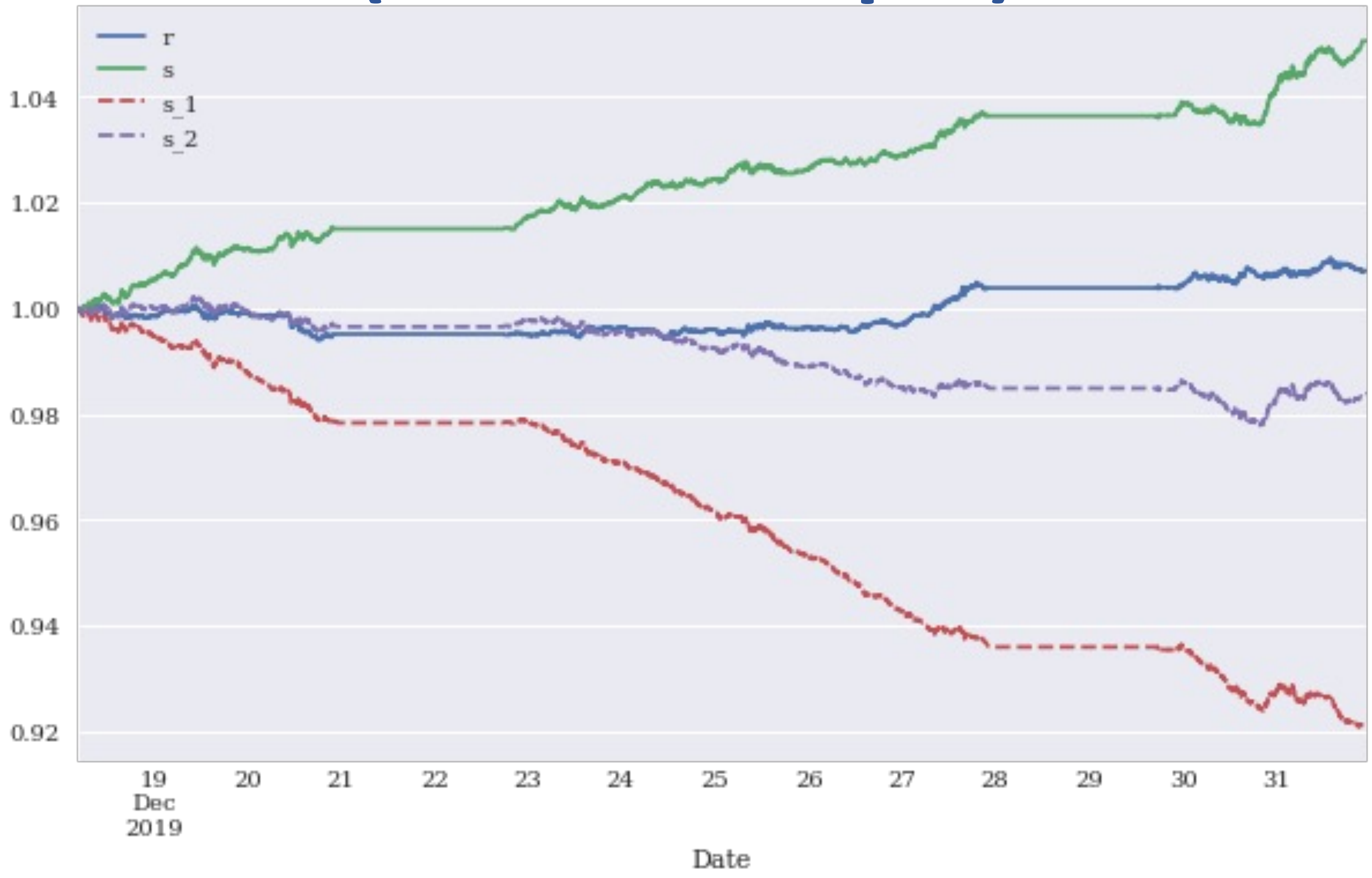
Gross performance of the daily DNN strategy before and after transaction costs (out-of-sample)

Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Gross performance of the passive benchmark investment and the DNN intraday strategy (out-of-sample)

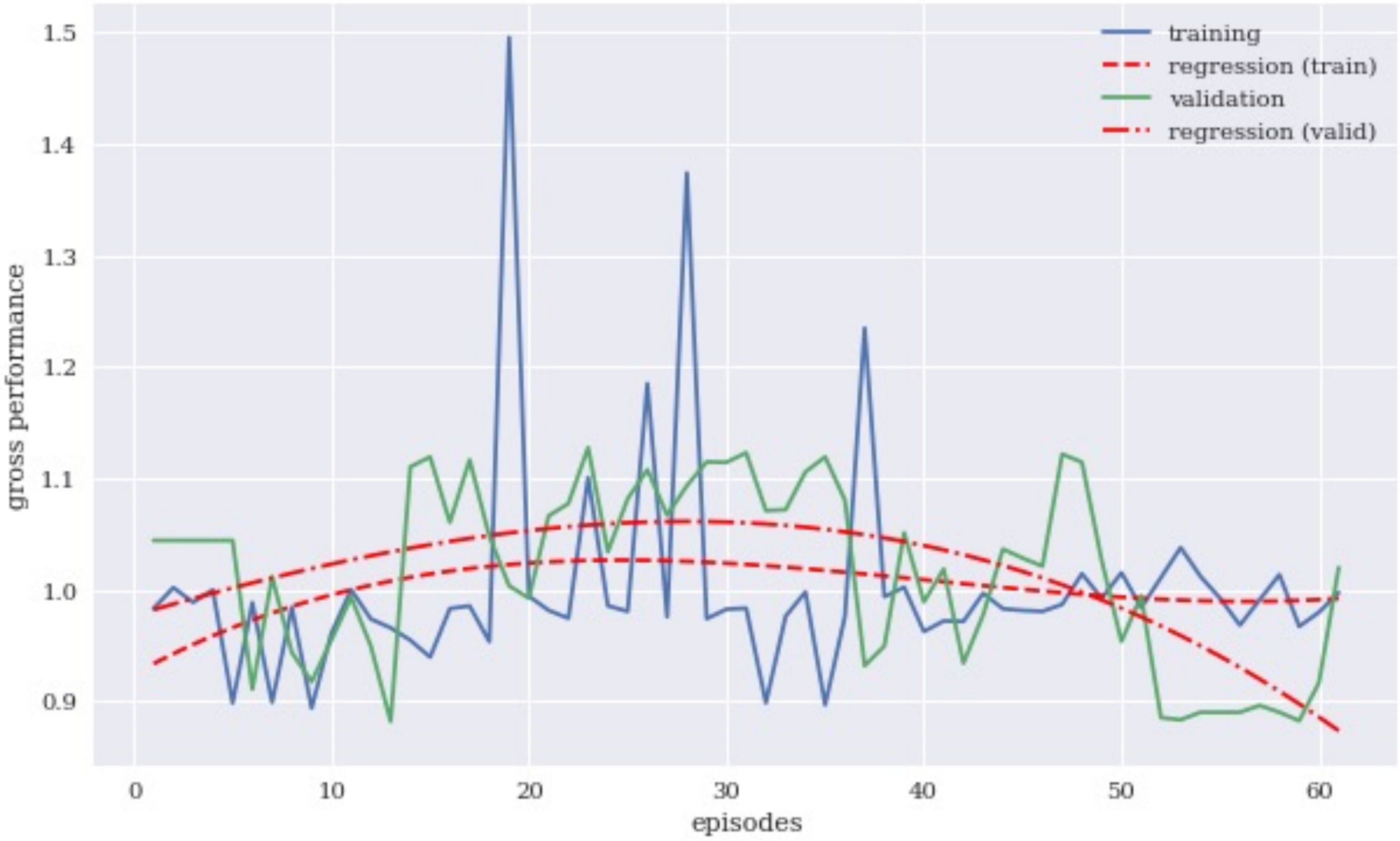
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Gross performance of the DNN intraday strategy before and after higher/ lower transaction costs (out-of-sample)



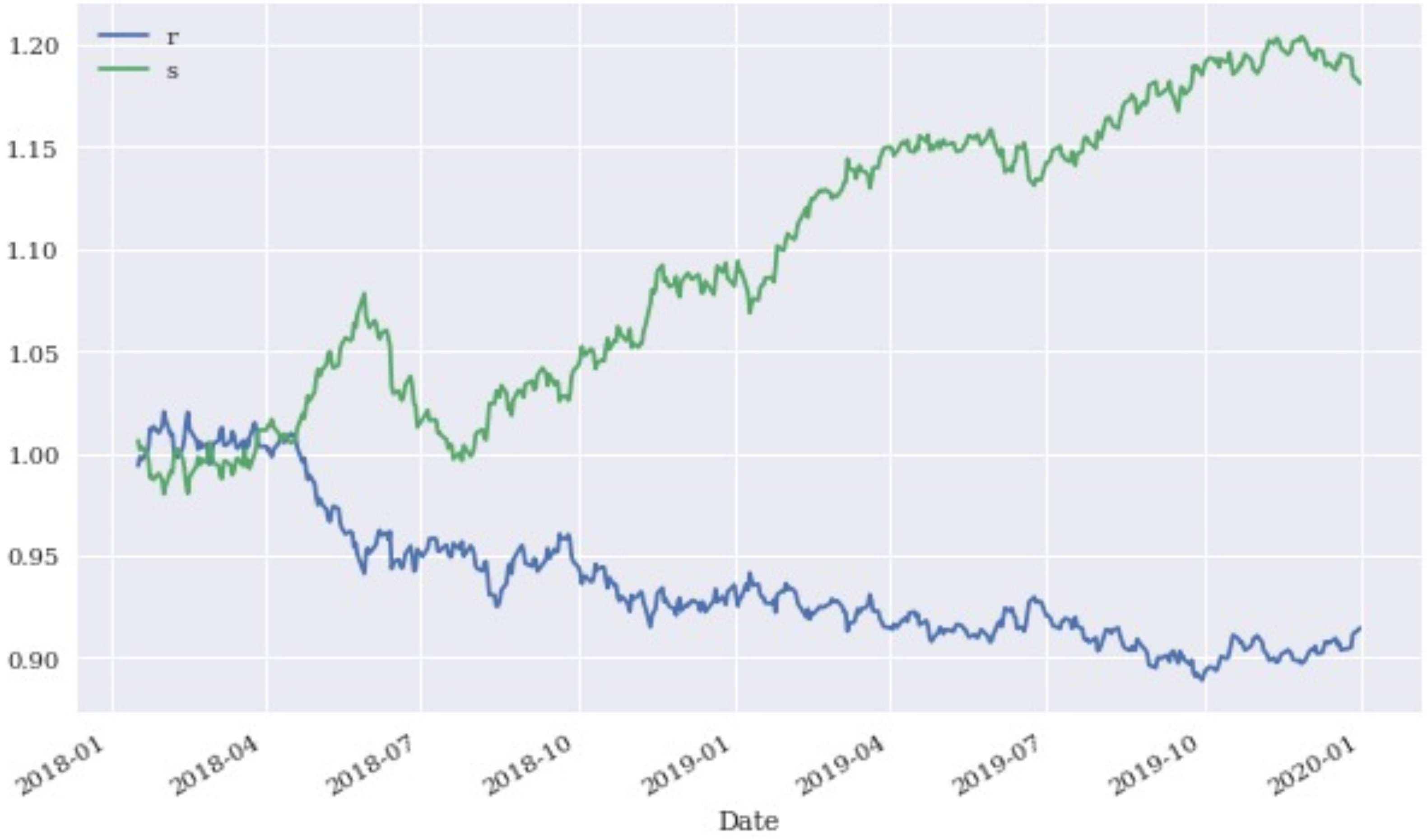
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Gross performance on training and validation data set



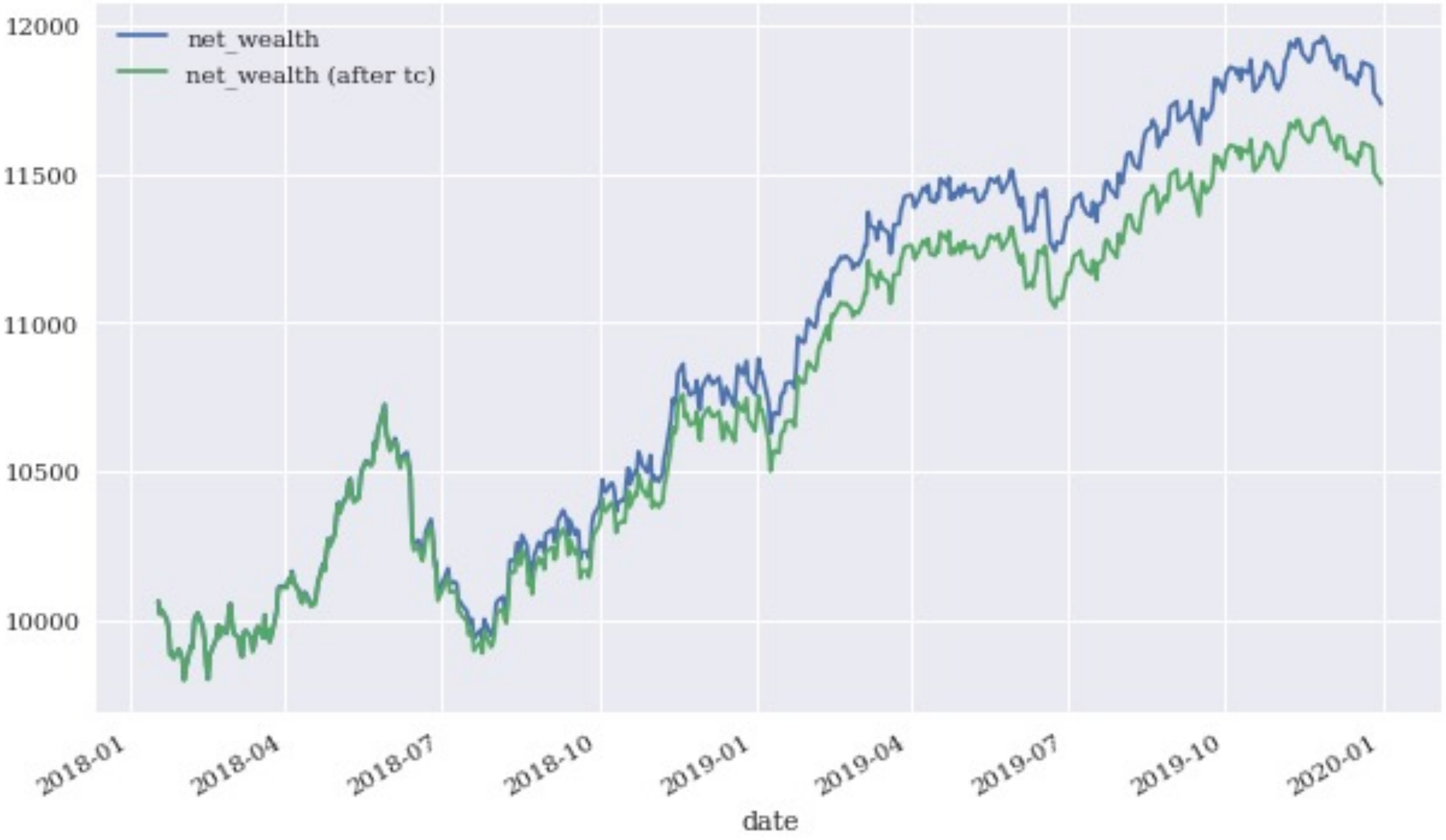
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Gross performance of the passive benchmark investment and the trading bot (out-of-sample)



Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Gross performance of the trading bot before and after transaction costs (in-sample)

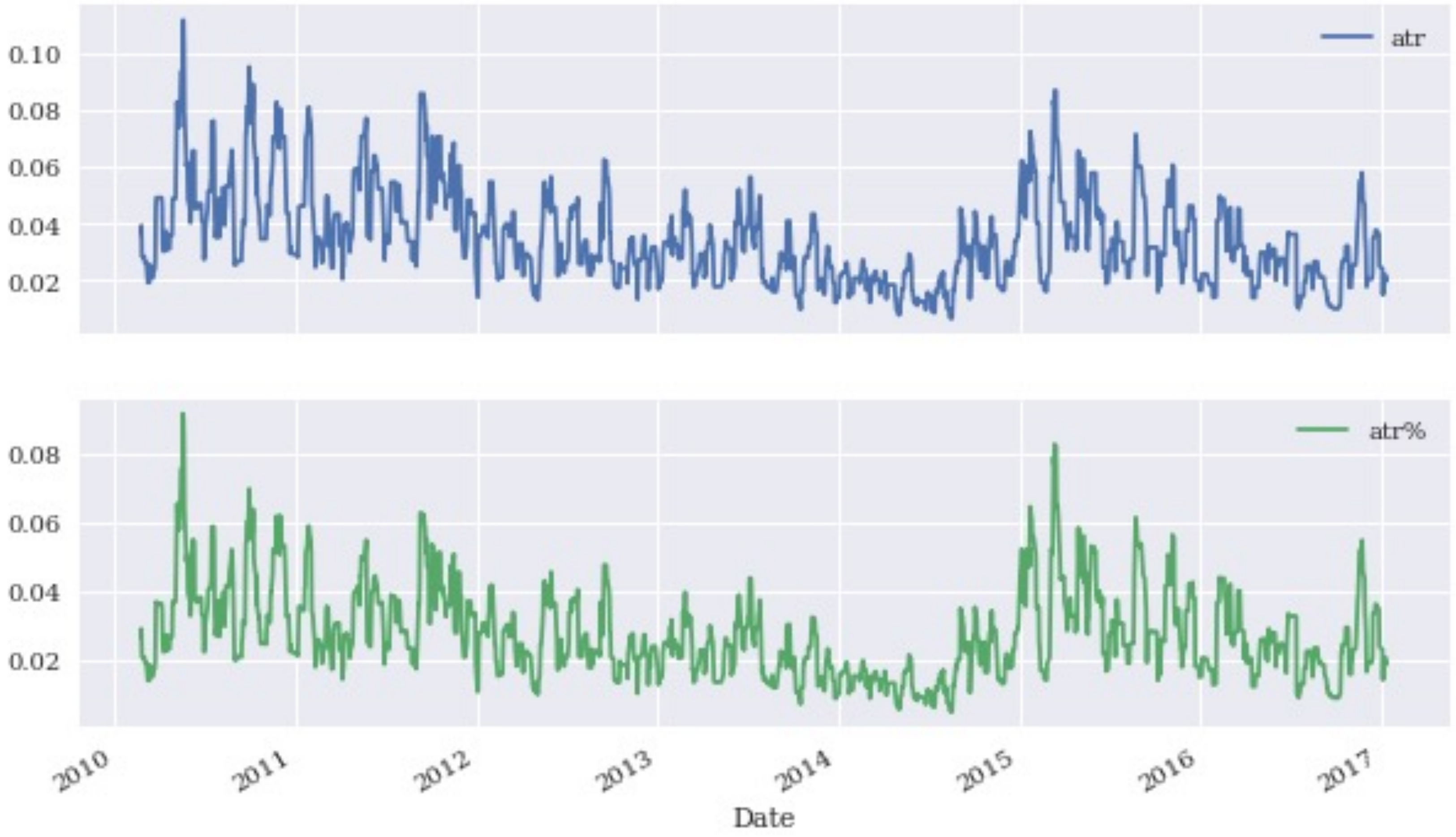


Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Gross performance of the passive benchmark investment and the trading bot (vectorized and event-based backtesting)

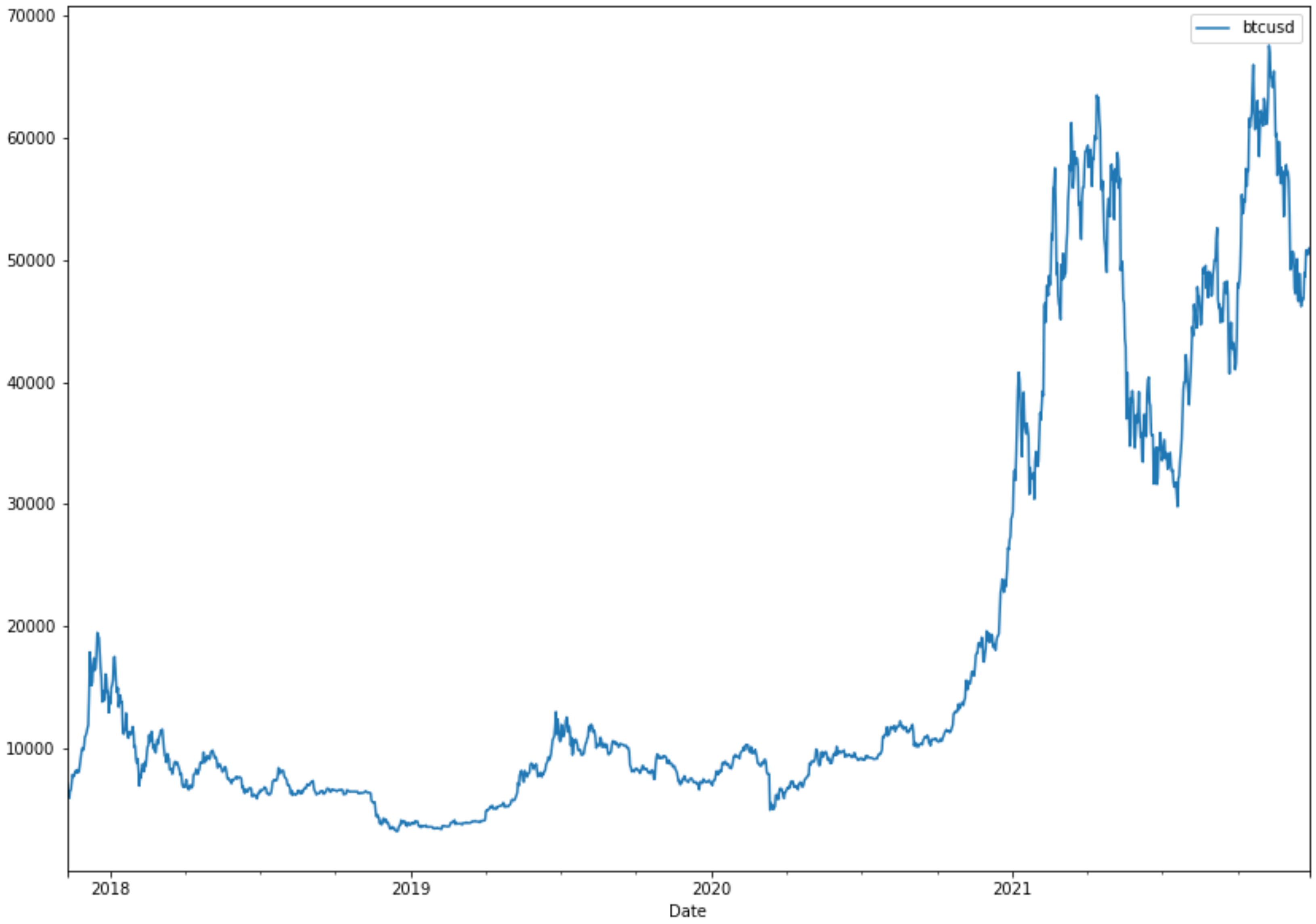
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Average true range (ATR) in absolute (price) and relative (%) terms



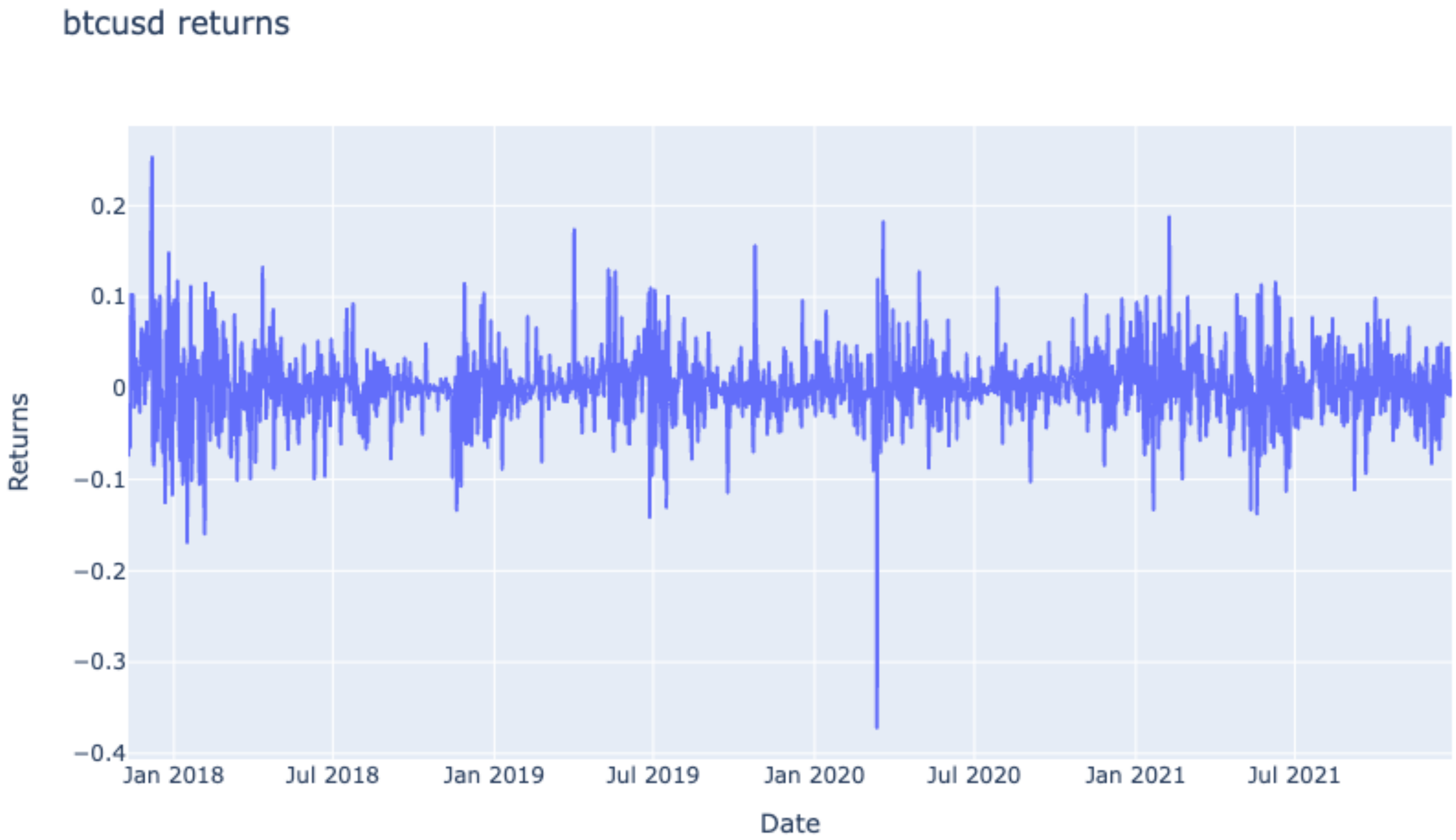
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

BTC-USD



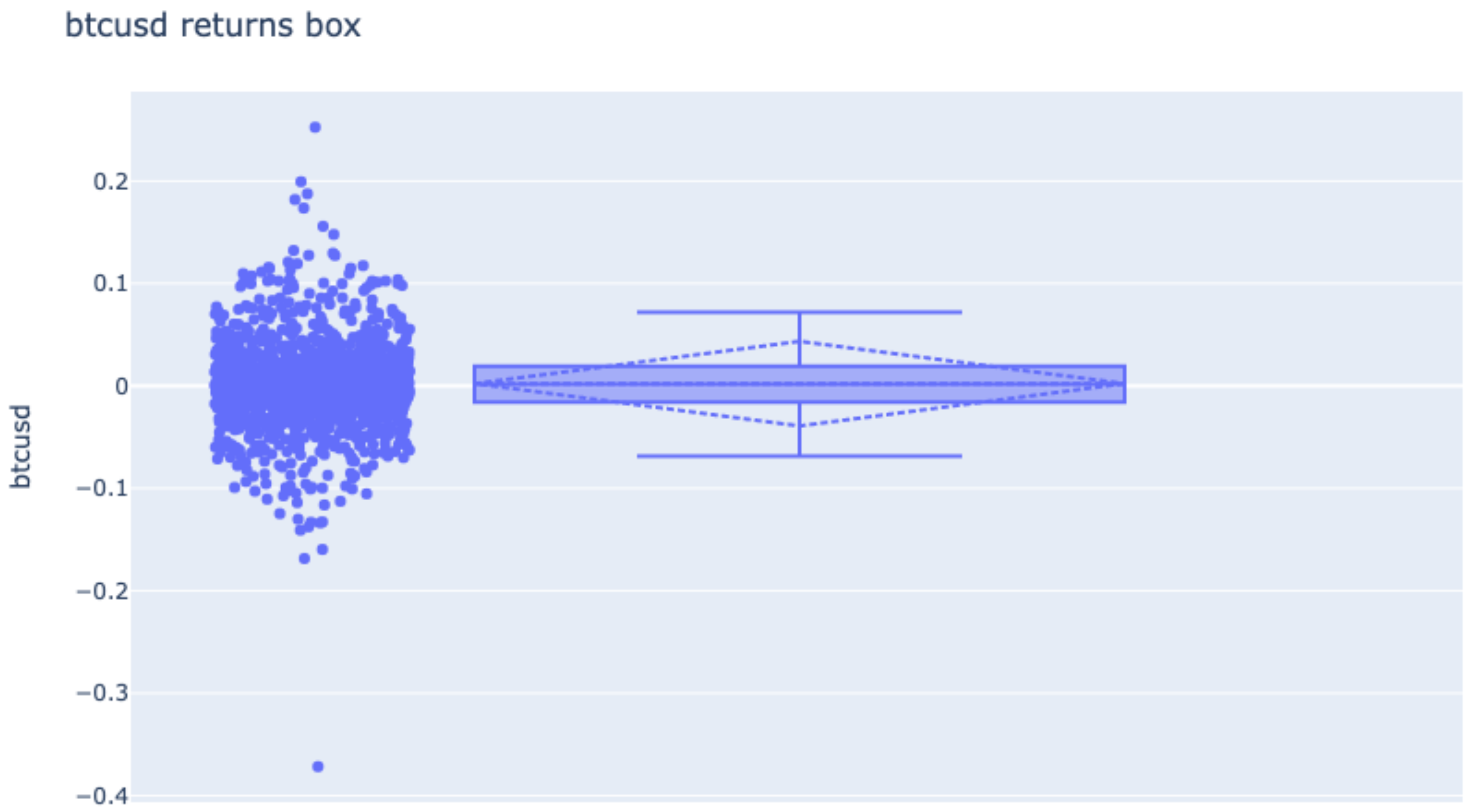
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

BTC-USD Returns



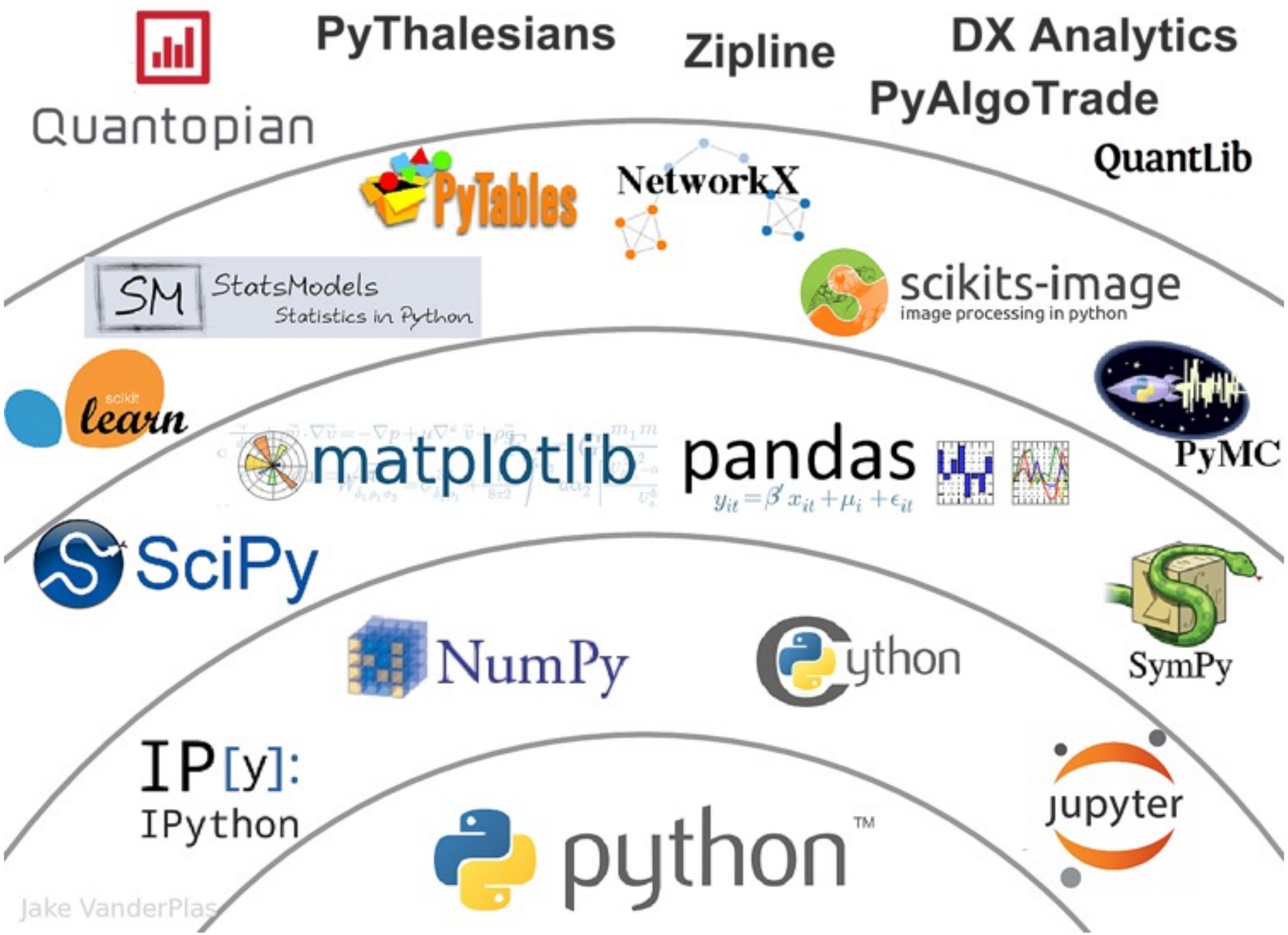
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

BTC-USD Returns Box



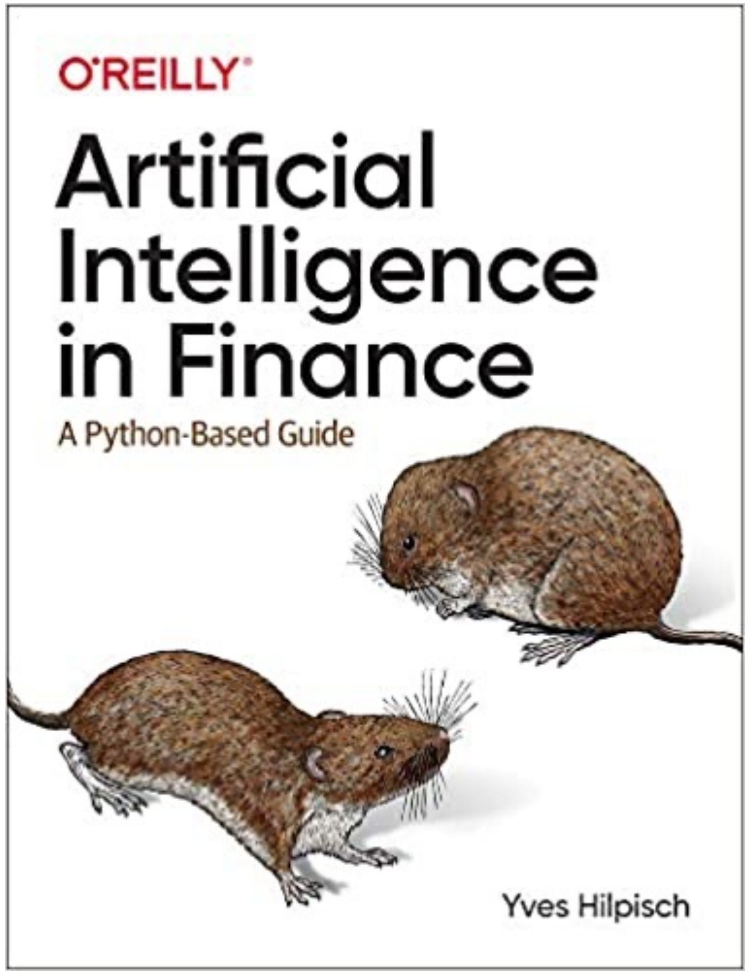
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

The Quant Finance PyData Stack



Source: http://nbviewer.jupyter.org/format/slides/github/quantopian/pyfolio/blob/master/pyfolio/examples/overview_slides.ipynb#/5

Yves Hilpisch (2020),
Artificial Intelligence in Finance: A Python-Based Guide,
O'Reilly



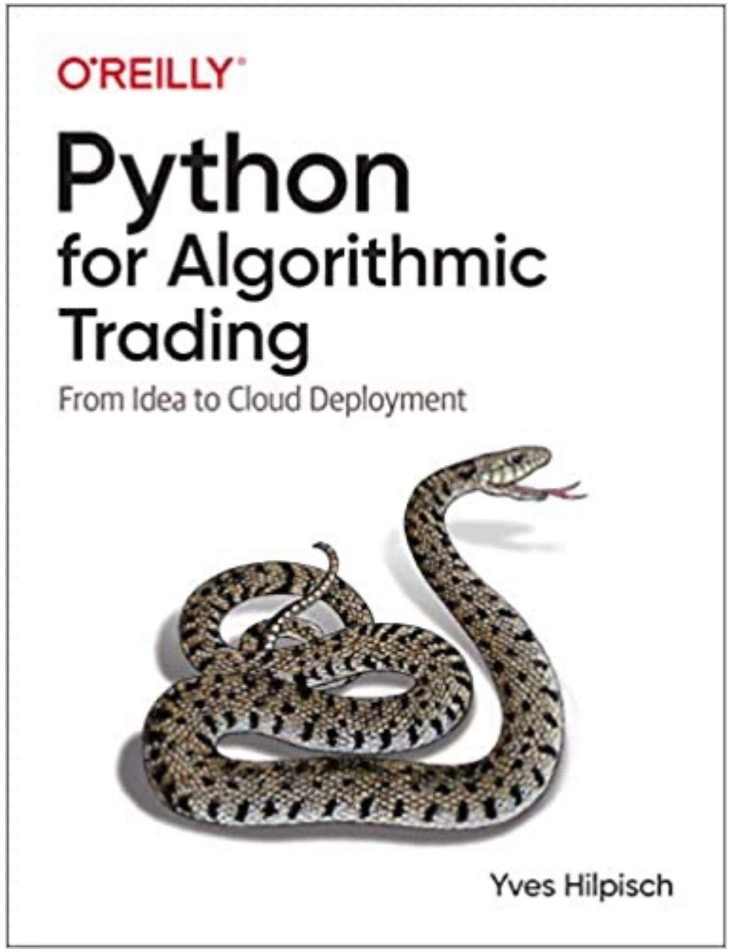
Source: https://www.amazon.com/Artificial-Intelligence-Finance-Python-Based-Guide/dp/1492055433

Yves Hilpisch (2020),

Python for Algorithmic Trading:

From Idea to Cloud Deployment,

O'Reilly



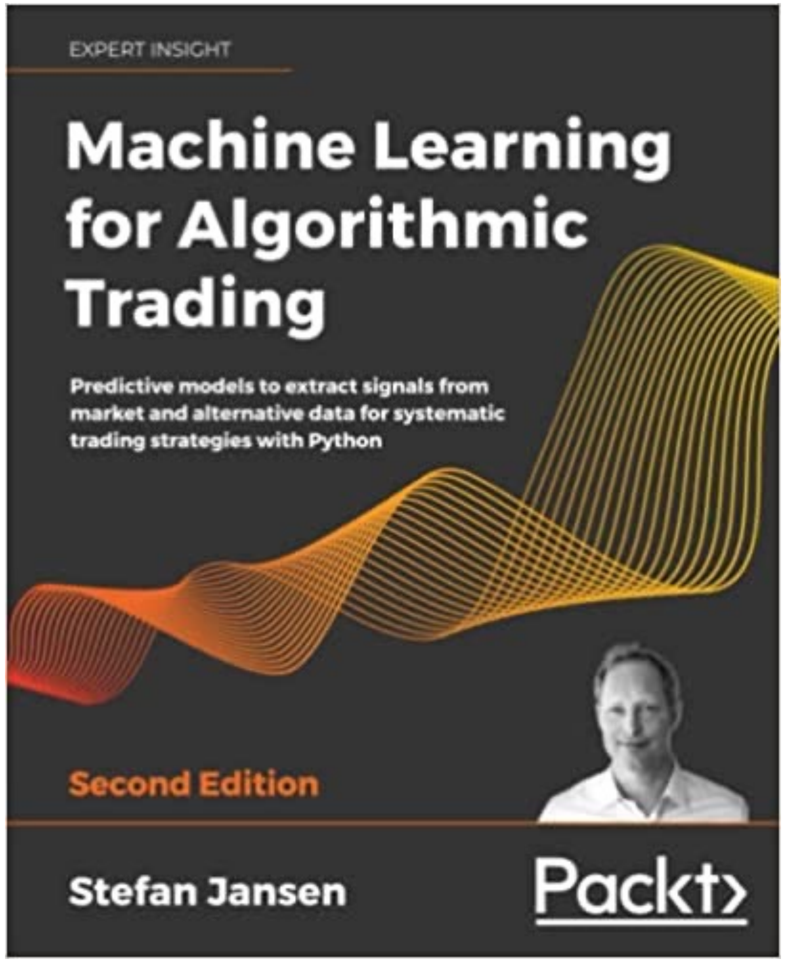
Source: https://www.amazon.com/Python-Algorithmic-Trading-Cloud-Deployment/dp/149205335X

Stefan Jansen (2020),

Machine Learning for Algorithmic Trading:

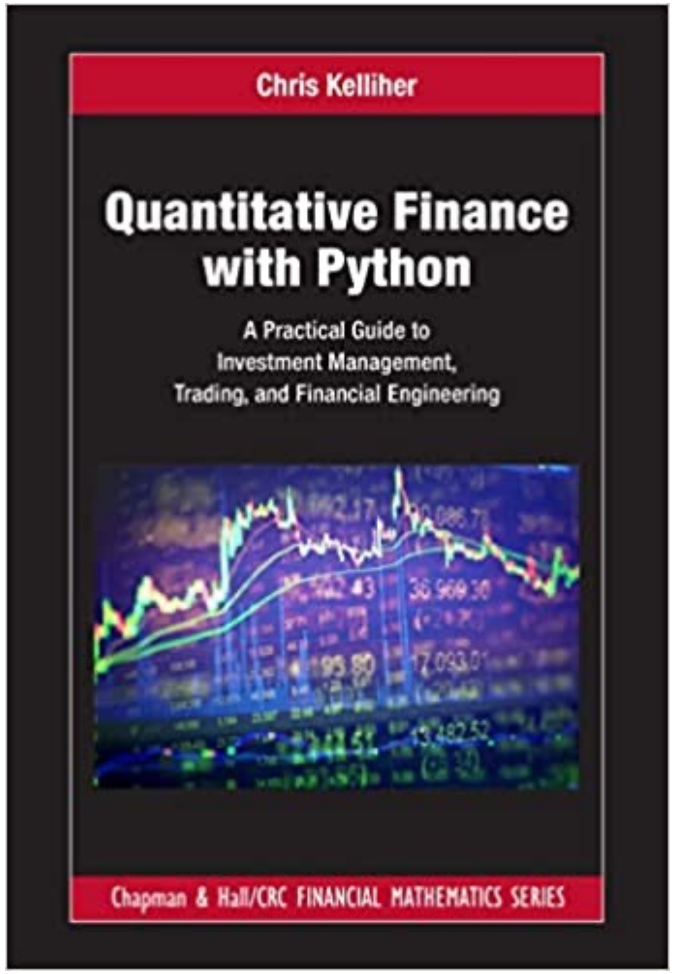
Predictive models to extract signals from market and alternative data for systematic trading strategies with Python, 2nd Edition,

Packt Publishing.



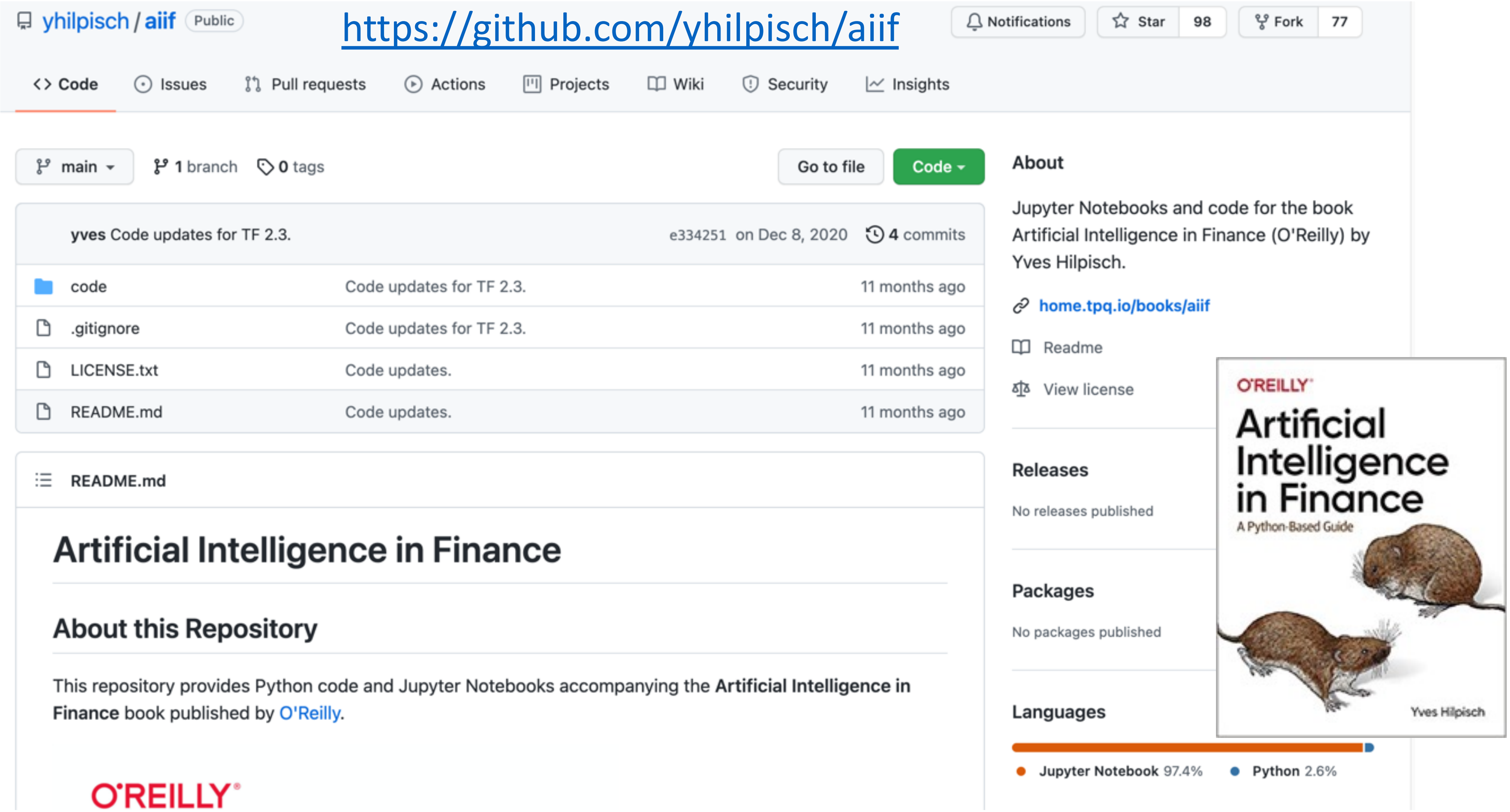
Source: https://www.amazon.com/Machine-Learning-Algorithmic-Trading-alternative/dp/1839217715/

Chris Kelliher (2022),
Quantitative Finance With Python:
A Practical Guide to Investment Management, Trading, and Financial Engineering,
Chapman and Hall/CRC.



Source: https://www.amazon.com/Quantitative-Finance-Python-Engineering-Mathematics/dp/1032014431/

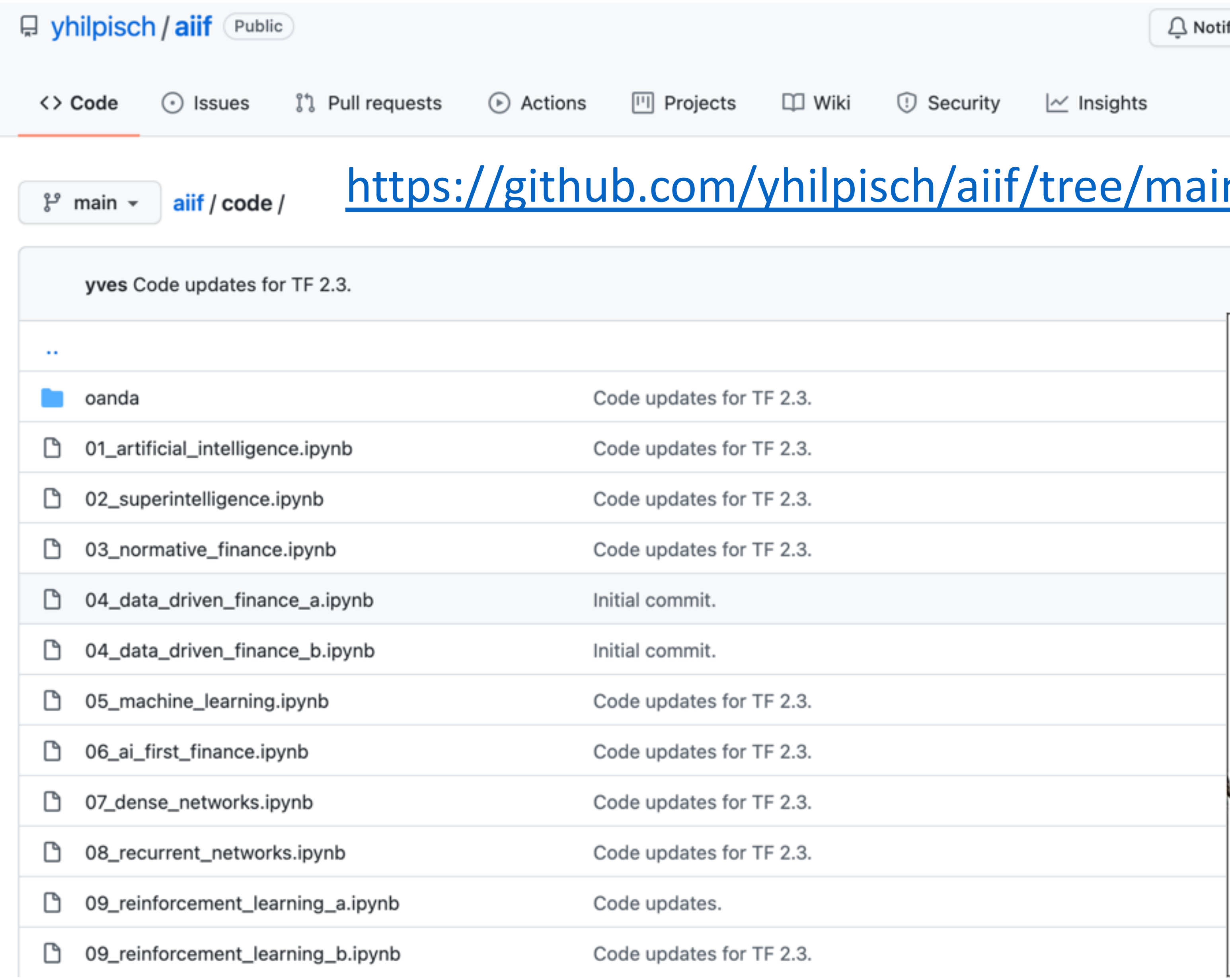
Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly

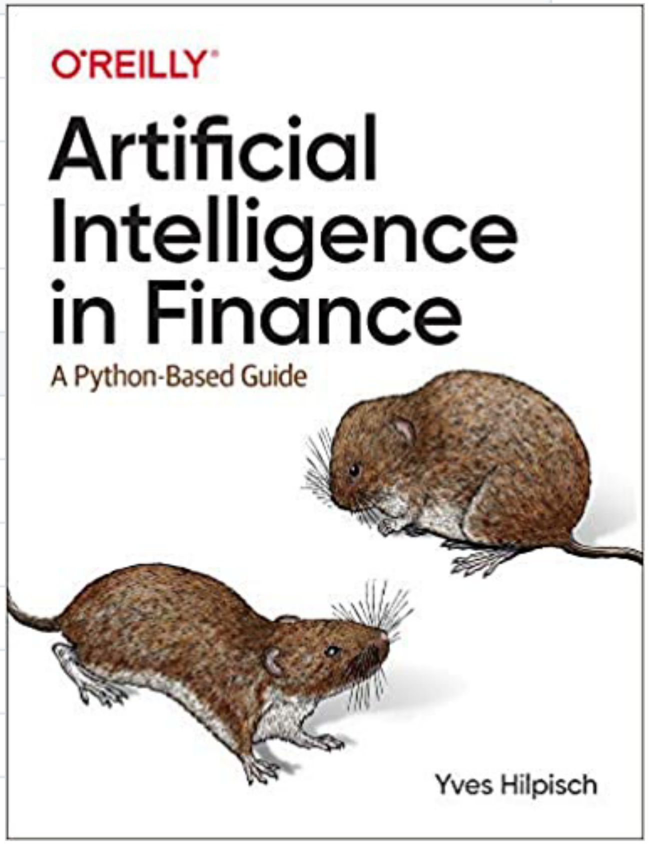


Source: https://github.com/yhilpisch/aiif

Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly

https://github.com/yhilpisch/aiif/tree/main/code





Source: https://github.com/yhilpisch/aiif/tree/main/code

Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

python101.ipynb - Colaboratory

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT?authuser=2#scrollTo=wsh36fLxDKC3

python101.ipynb

File Edit View Insert Runtime Tools Help

COMMENT SHARE

CODE TEXT CELL CELL CONNECTED EDITING

```python
# Future Value
pv = 100
r = 0.1
n = 7
fv = pv * ((1 + (r)) ** n)
print(round(fv, 2))
```

```
194.87
```

[11]

```python
amount = 100
interest = 10 #10% = 0.01 * 10
years = 7

future_value = amount * ((1 + (0.01 * interest)) ** years)
print(round(future_value, 2))
```

```
194.87
```

[12]

```python
# Python Function def
def getfv(pv, r, n):
    fv = pv * ((1 + (r)) ** n)
    return fv
fv = getfv(100, 0.1, 7)
print(round(fv, 2))
```

```
194.87
```

[13]

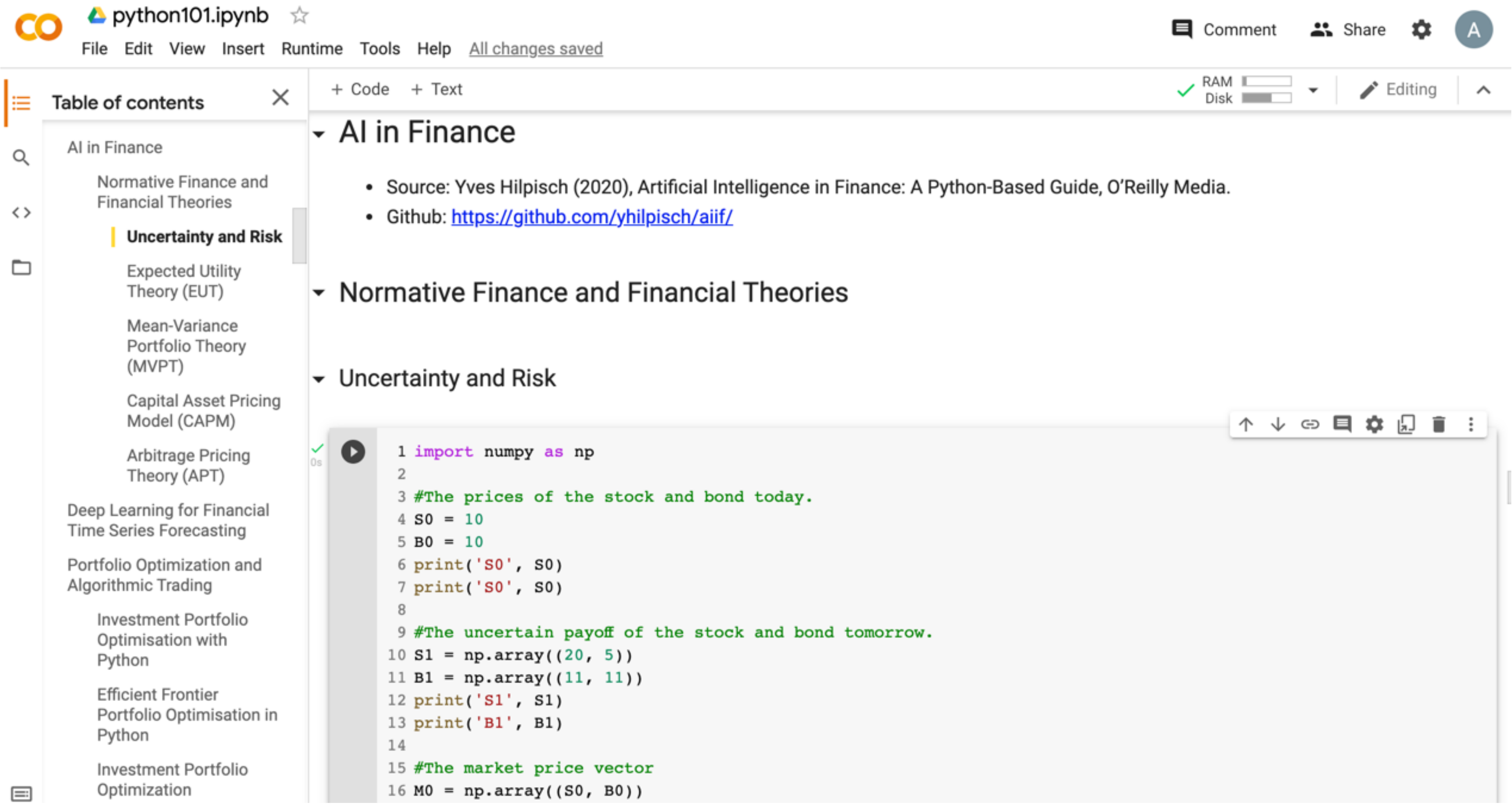
```python
# Python if else
score = 80
if score >=60 :
    print("Pass")
else:
    print("Fail")
```

```
Pass
```

https://tinyurl.com/aintpupython101

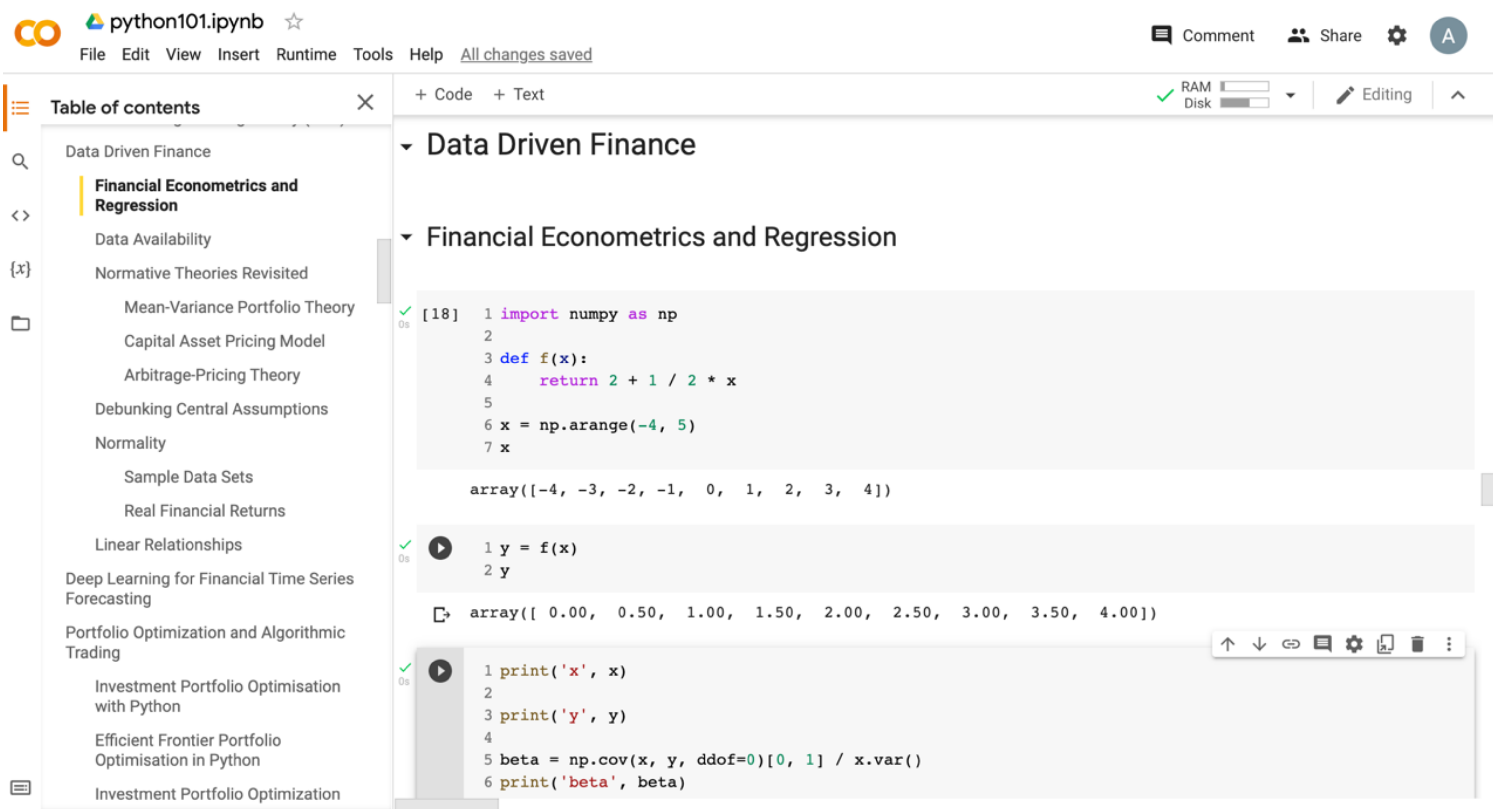
Python in Google Colab (Python101)

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT



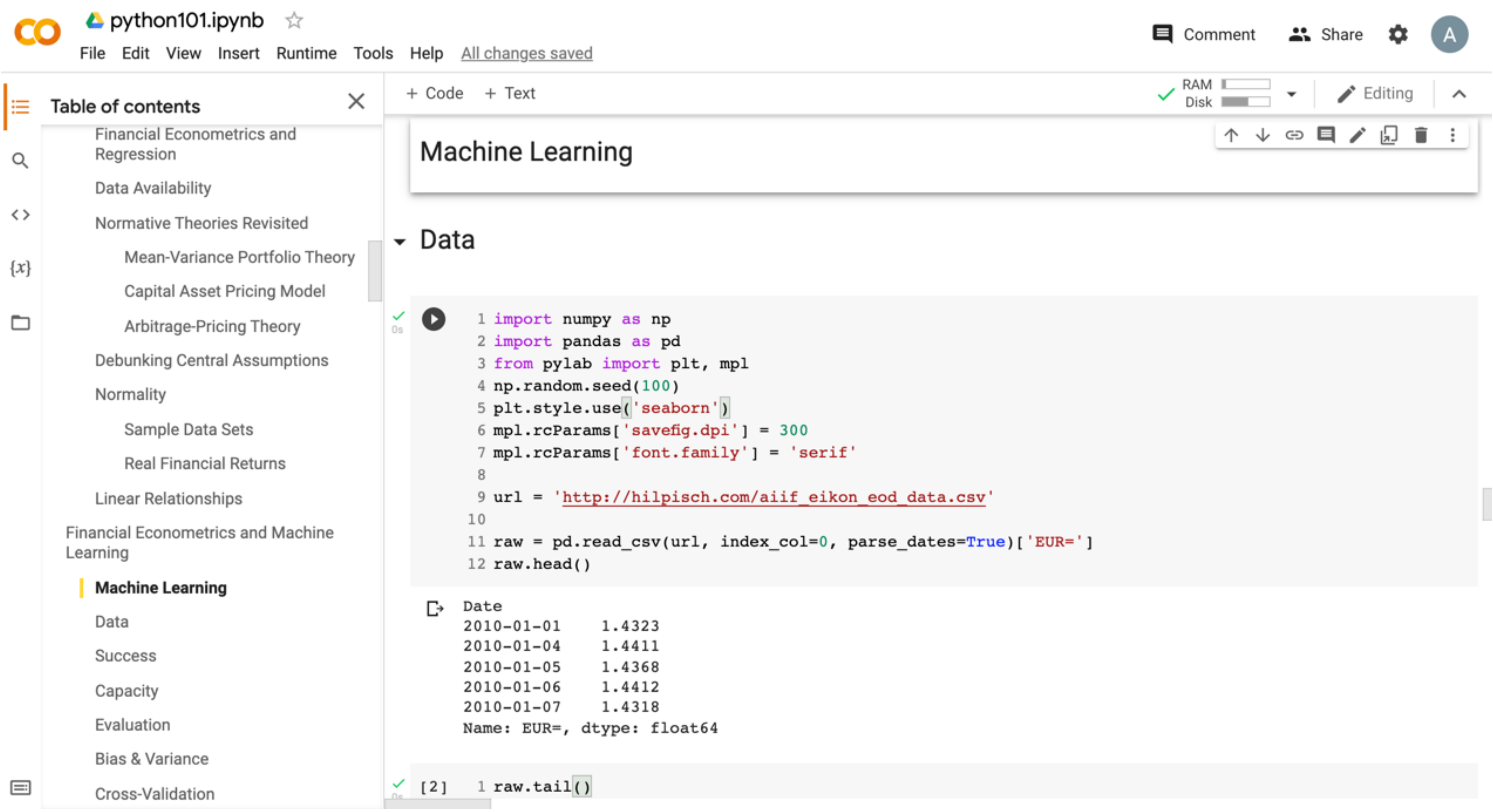
https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



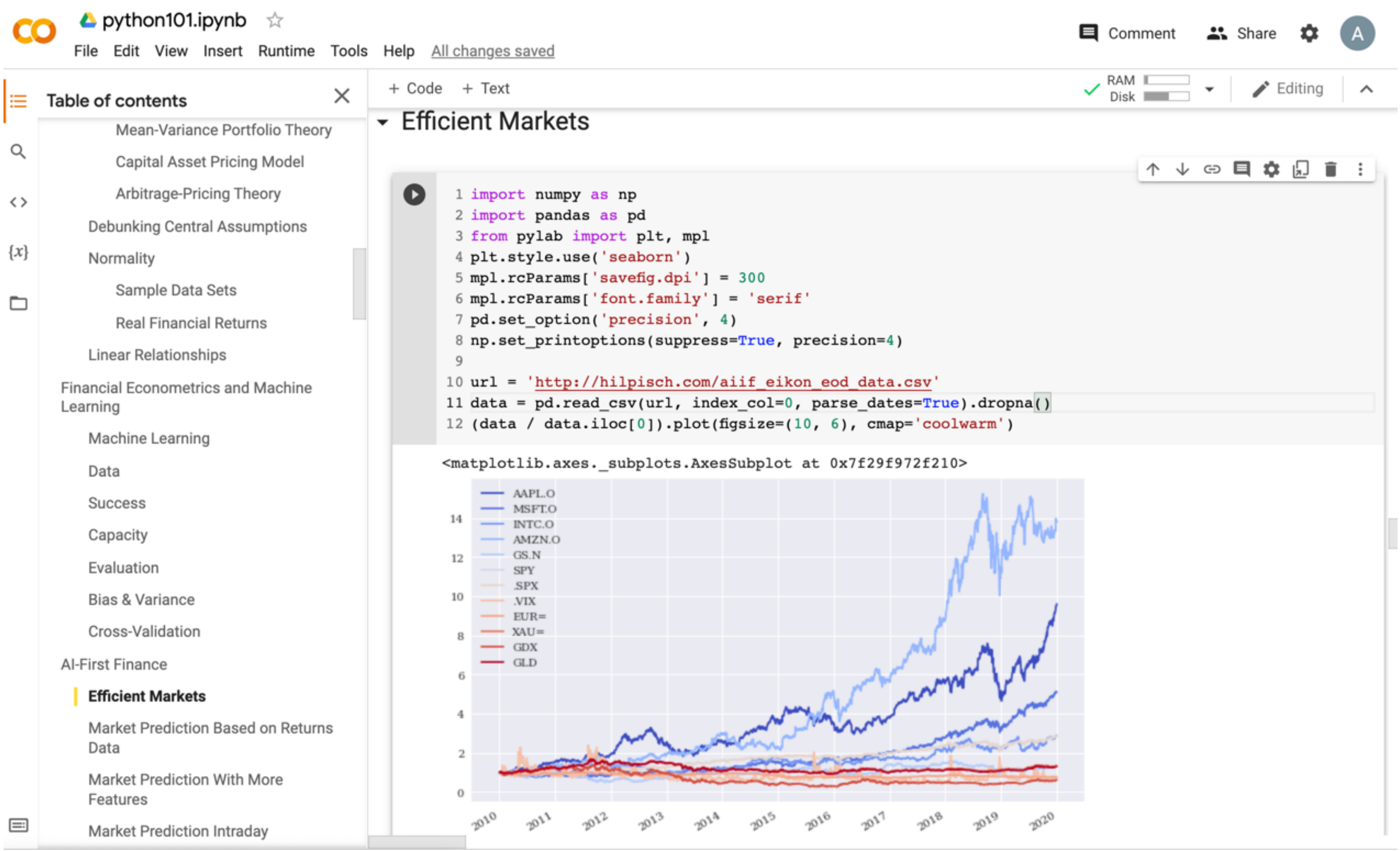
https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

python101.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text Connect Editing

Table of contents

Deep Learning (DL) in Finance

- Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.
- Github: https://github.com/yhilpisch/aiif/

Dense Neural Networks (DNN)

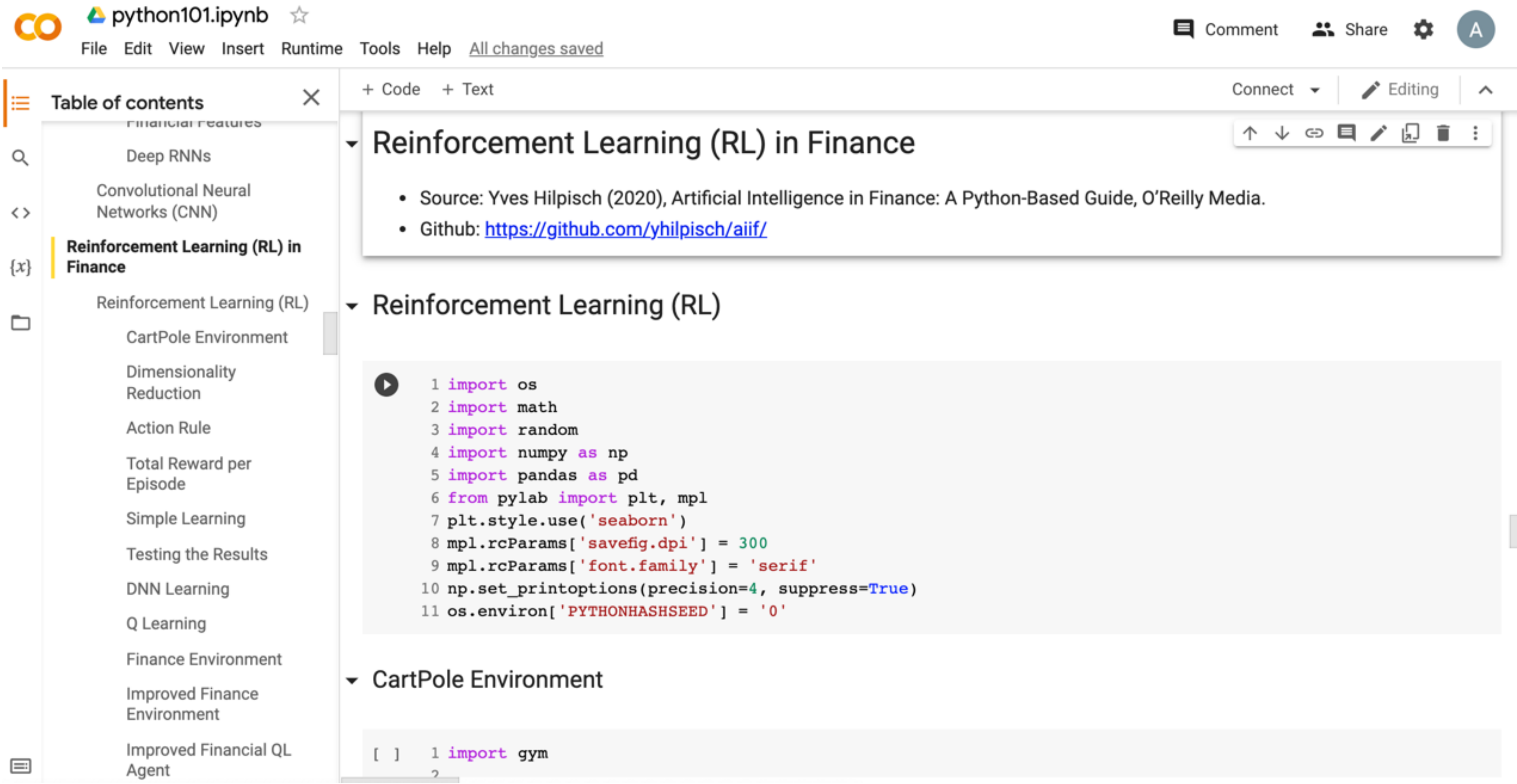
```
import os
import numpy as np
import pandas as pd
from pylab import plt, mpl
plt.style.use('seaborn')
mpl.rcParams['savefig.dpi'] = 300
mpl.rcParams['font.family'] = 'serif'
pd.set_option('precision', 4)
np.set_printoptions(suppress=True, precision=4)
os.environ['PYTHONHASHSEED'] = '0'
```

```
url = 'http://hilpisch.com/aiif_eikon_id_eur_usd.csv'
symbol = 'EUR_USD'
raw = pd.read_csv(url, index_col=0, parse_dates=True)
raw.head()
```

HIGH LOW OPEN CLOSE

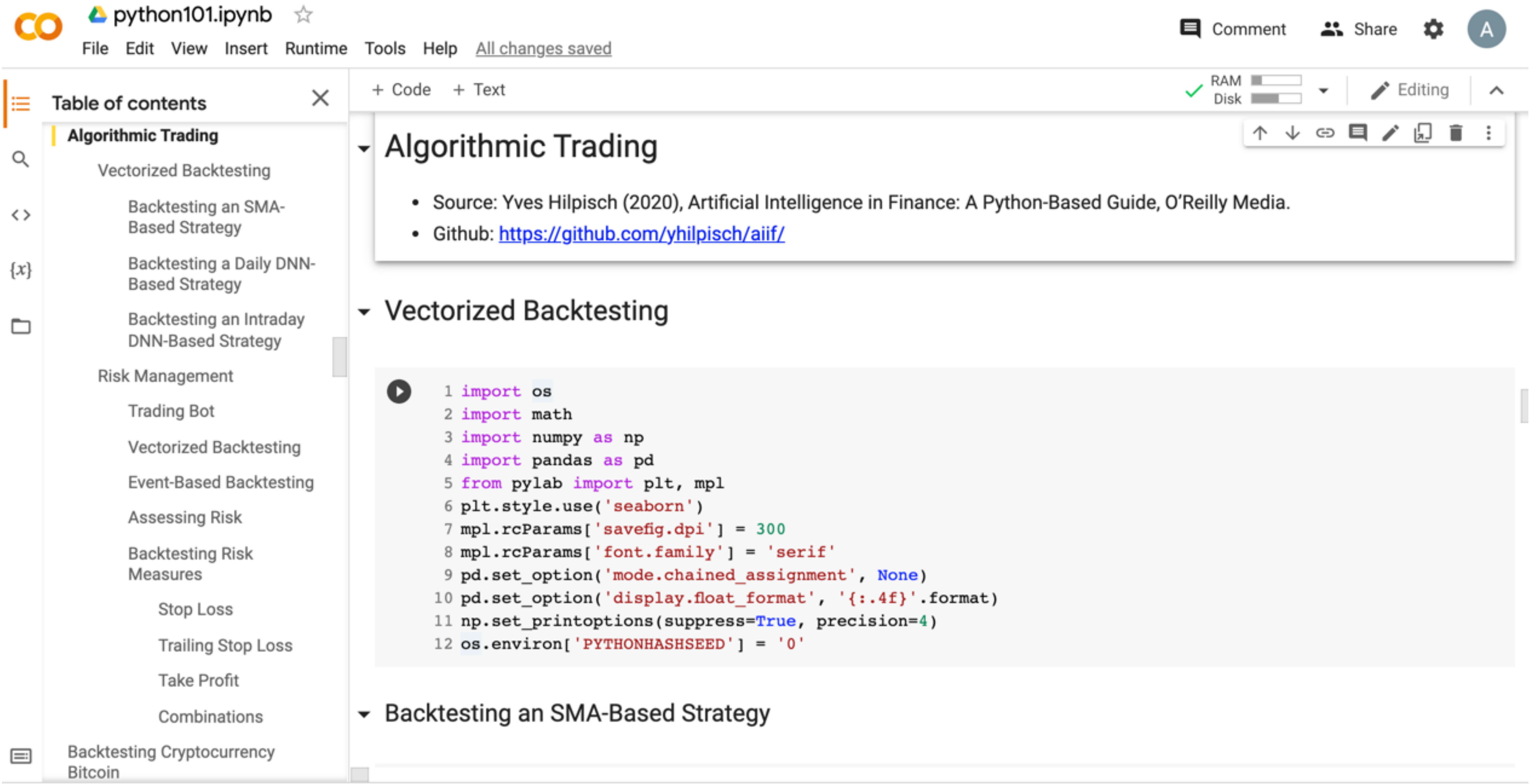
https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



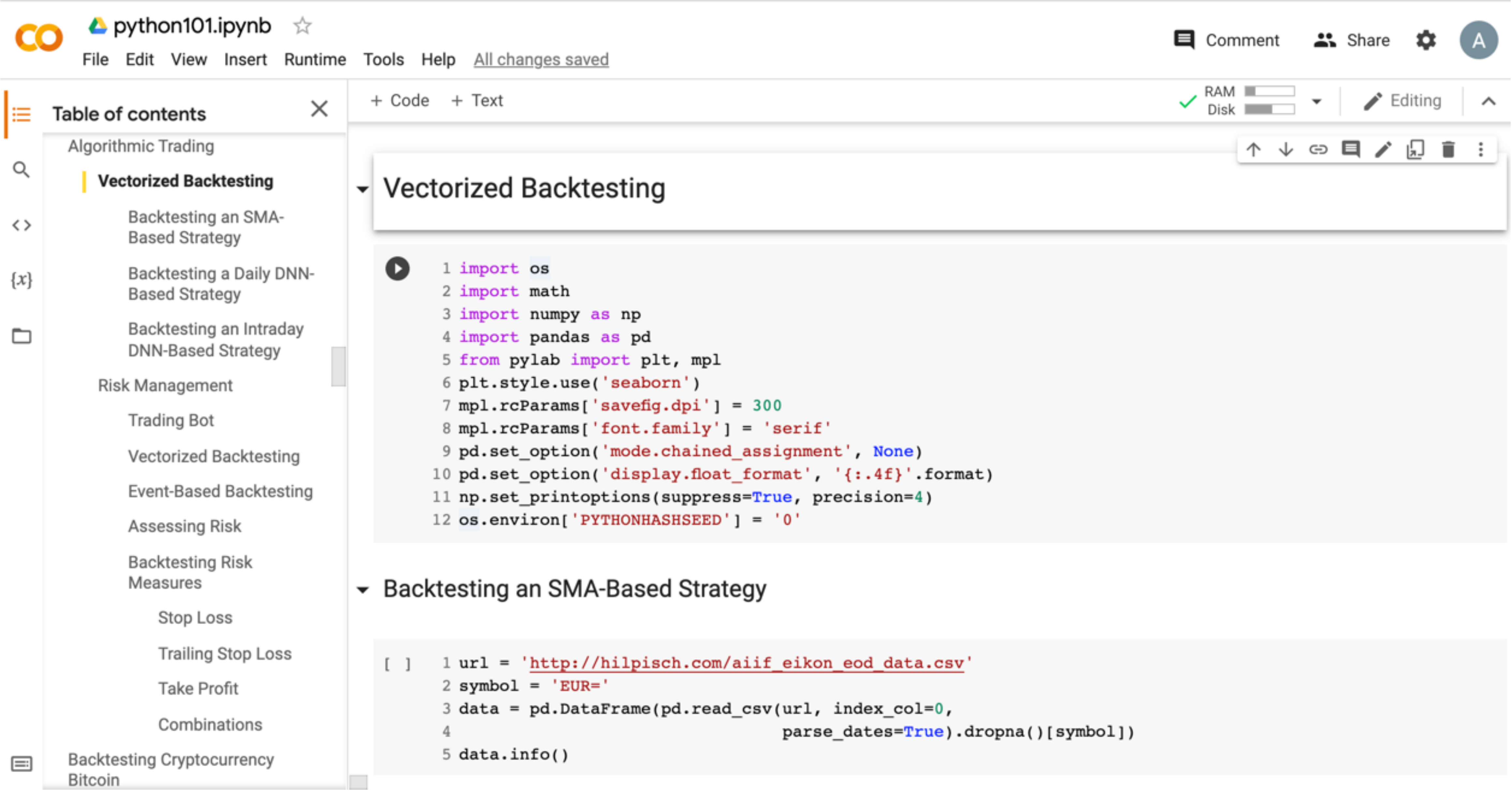
https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



https://tinyurl.com/aintpupython101

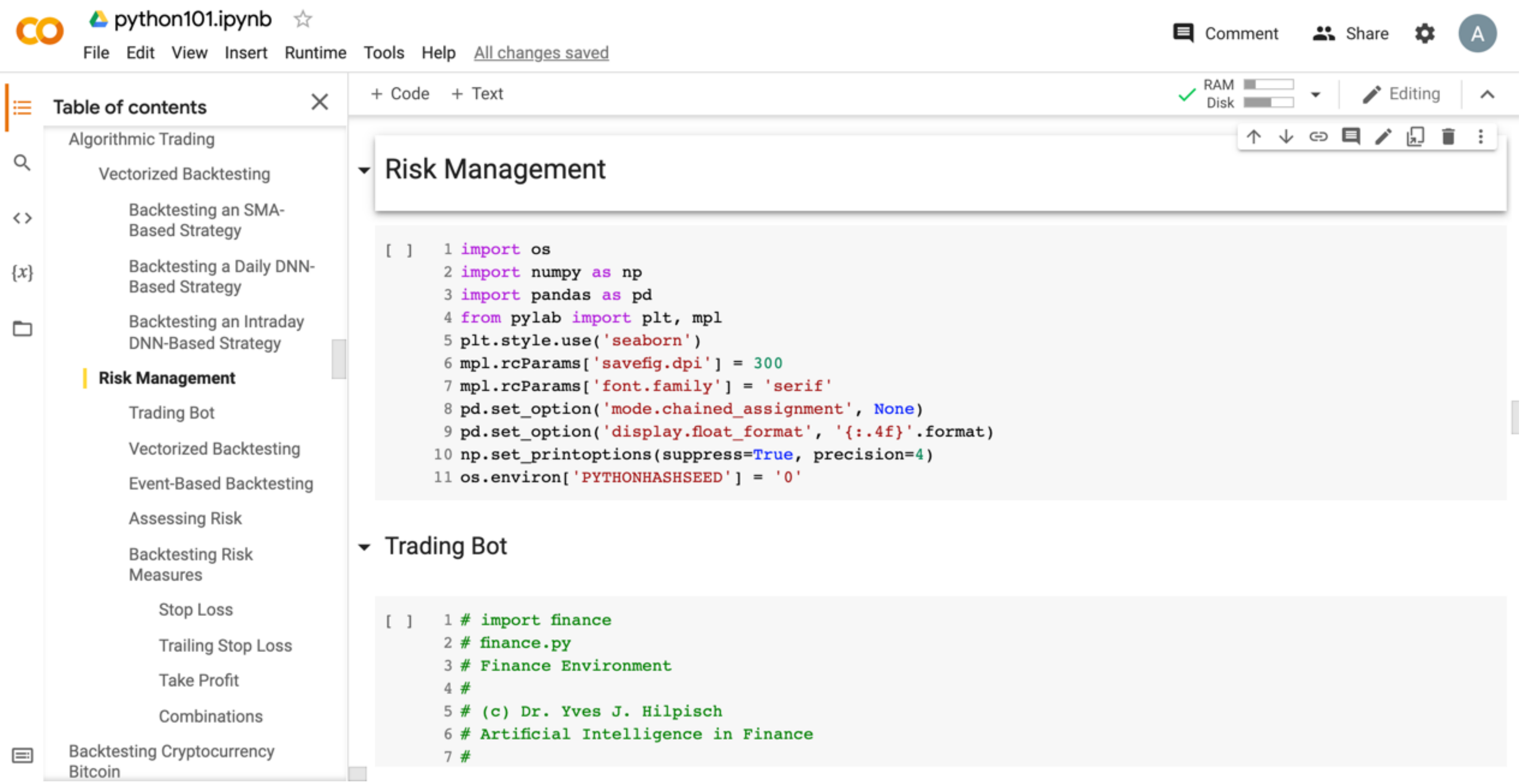
Python in Google Colab (Python101)

https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

python101.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

RAM Disk Editing

+ Code + Text

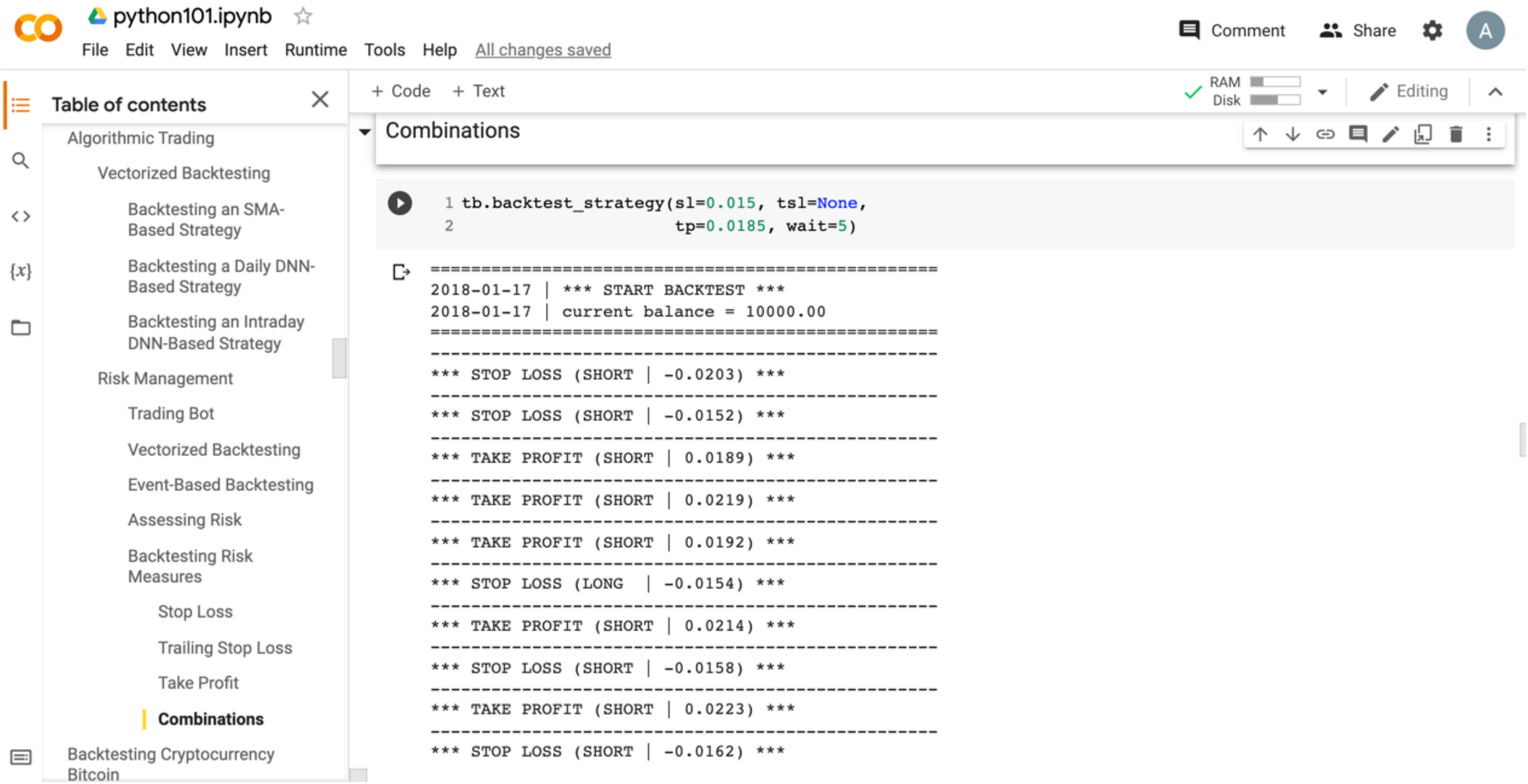
Table of contents

Event-Based Backtesting

```python
#import backtesting as bt

# backtesting.py
# Event-Based Backtesting
# --Base Class (1)
#
# (c) Dr. Yves J. Hilpisch
# Artificial Intelligence in Finance
#

class BacktestingBase:
    def __init__(self, env, model, amount, ptc, ftc, verbose=False):
        self.env = env
        self.model = model
        self.initial_amount = amount
        self.current_balance = amount
        self.ptc = ptc
        self.ftc = ftc
        self.verbose = verbose
        self.units = 0
        self.trades = 0

    def get_date_price(self, bar):
        ''' Returns date and price for a given bar.
        '''
```

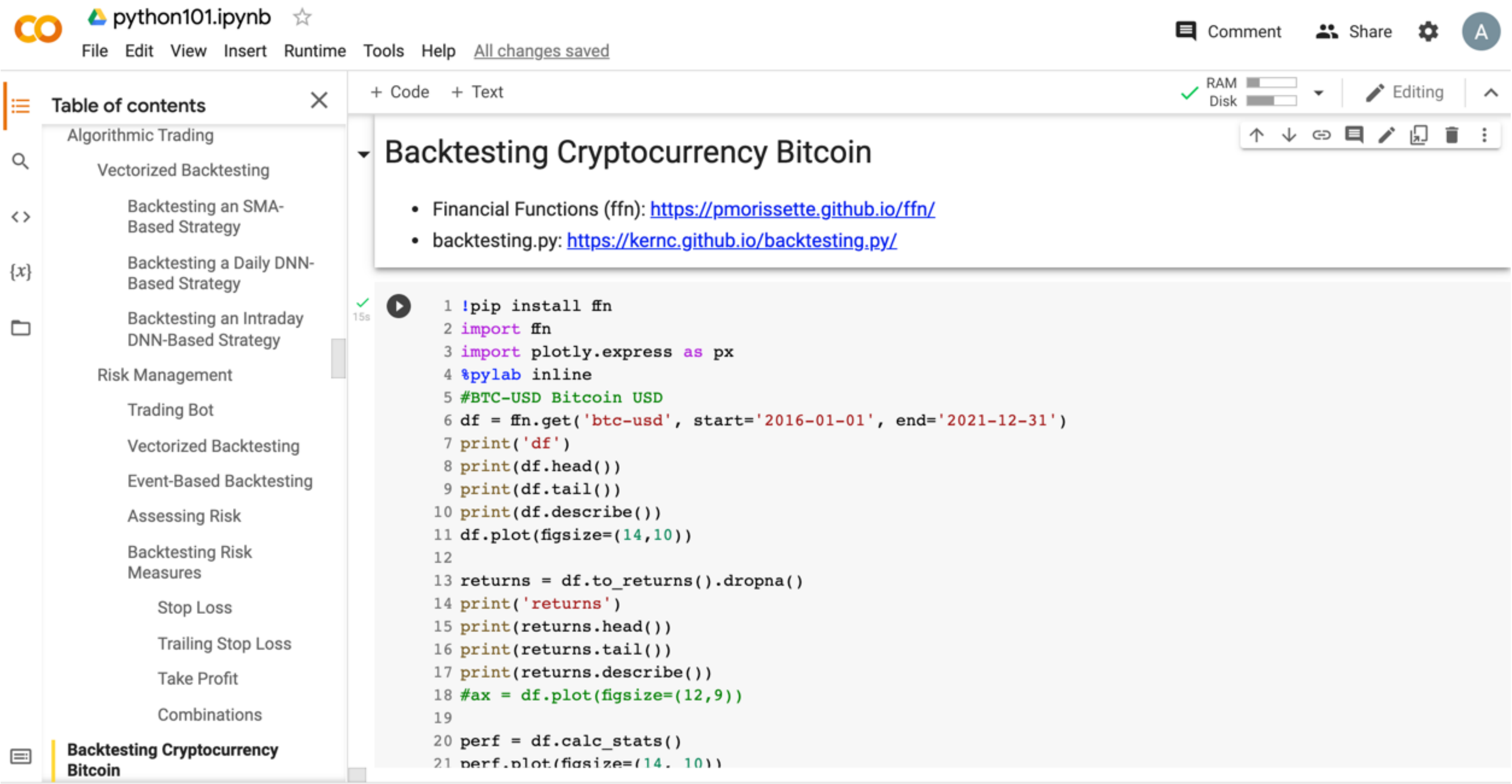
https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



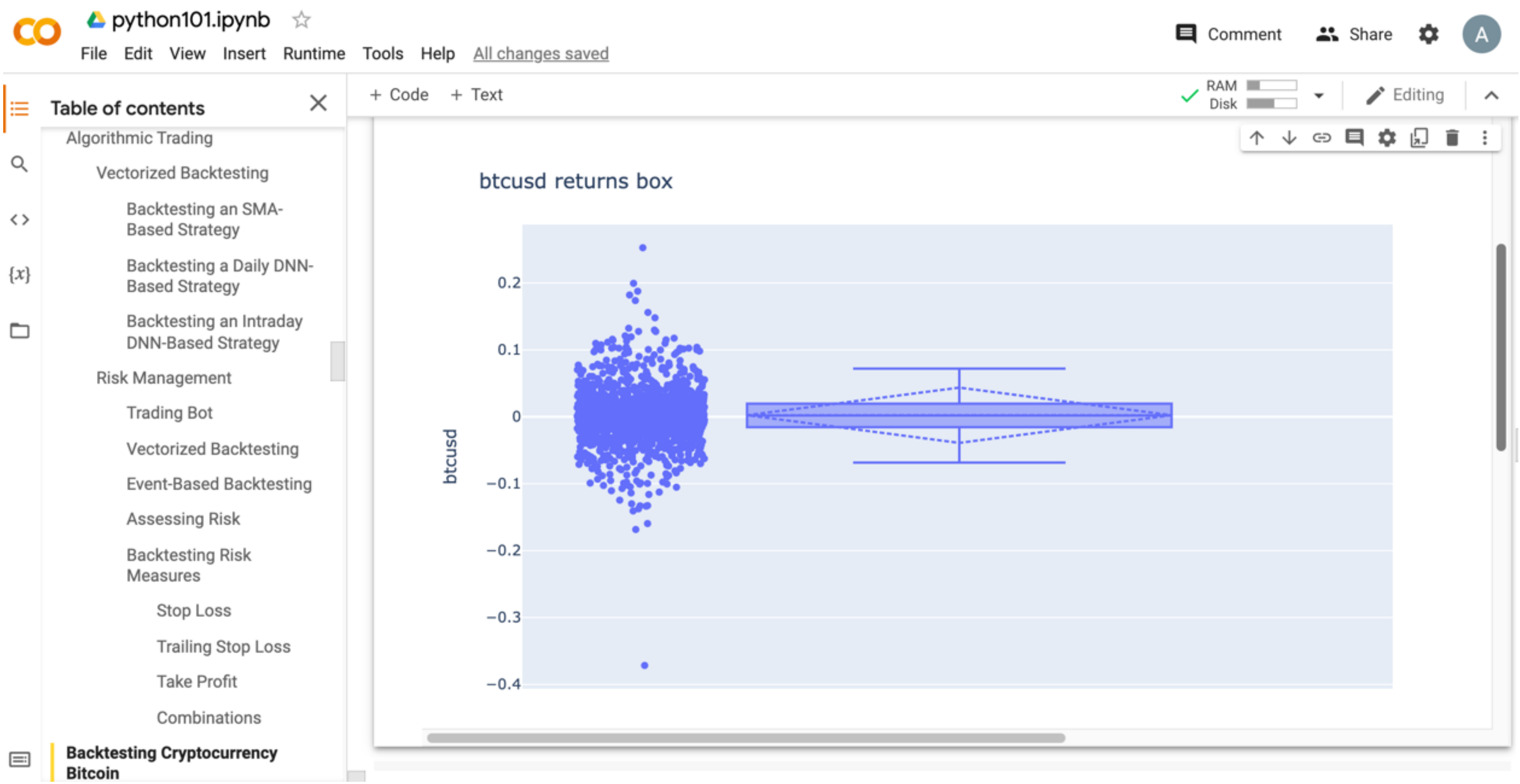
https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



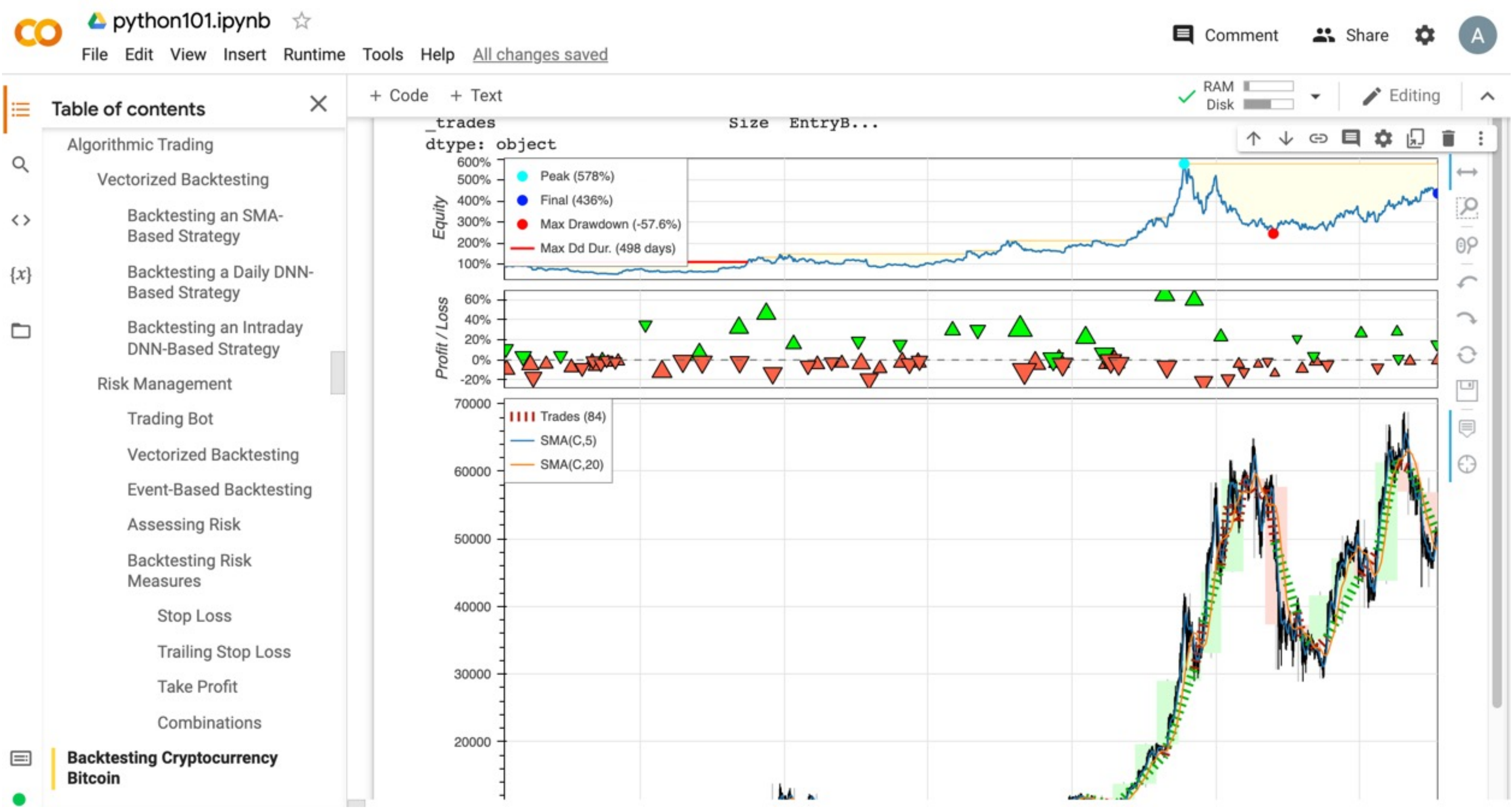
https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

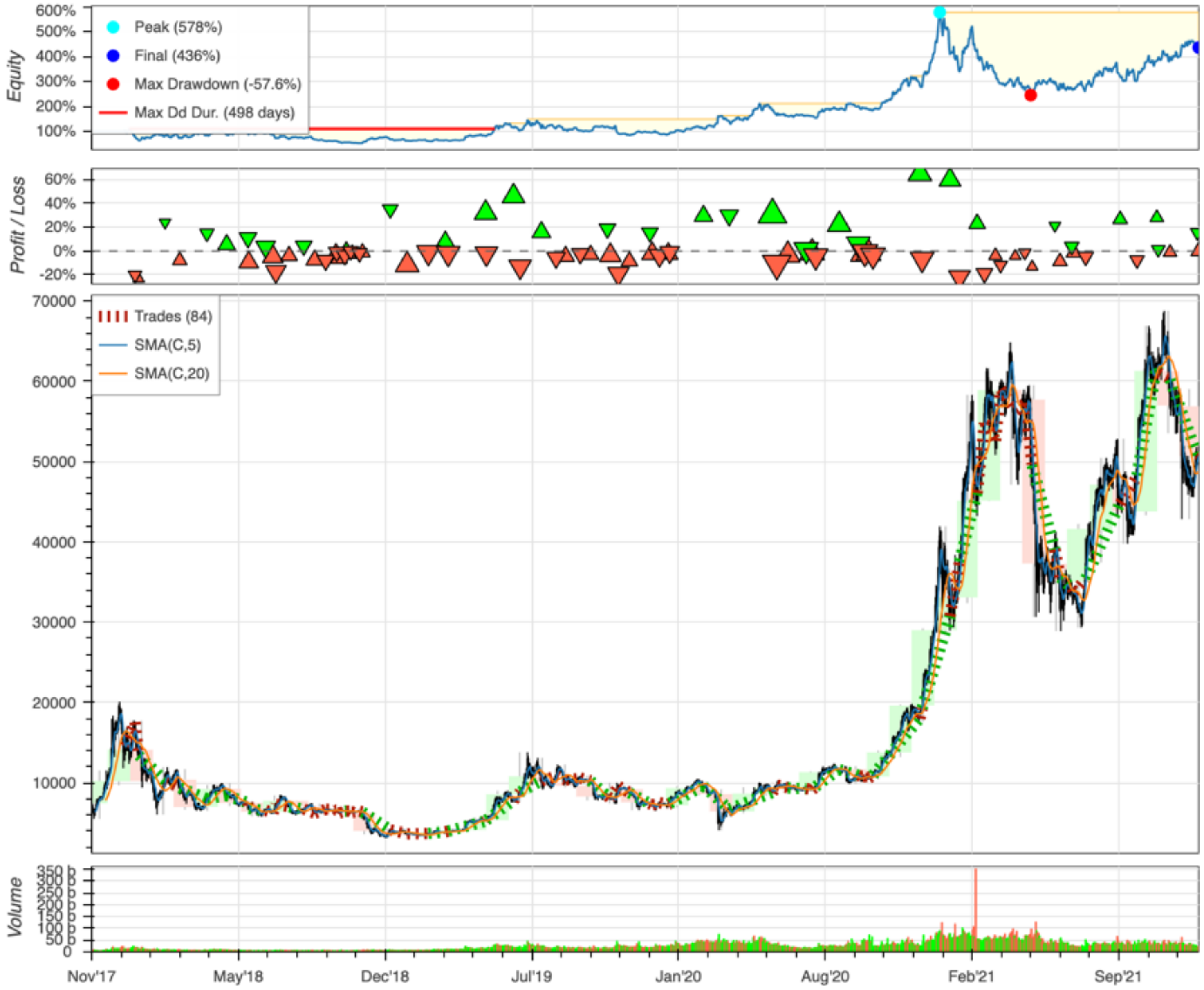


https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



https://tinyurl.com/aintpupython101



https://tinyurl.com/aintpupython101

Summary

- Algorithmic Trading
- Risk Management
- Trading Bot
- Event-Based Backtesting

Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

References

- Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O’Reilly Media, https://github.com/yhilpisch/aiif .
- Yves Hilpisch (2020), Python for Algorithmic Trading: From Idea to Cloud Deployment, O’Reilly Media.
- Stefan Jansen (2020), Machine Learning for Algorithmic Trading: Predictive models to extract signals from market and alternative data for systematic trading strategies with Python, 2nd Edition, Packt Publishing.
- Aurélien Géron (2019), Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 2nd Edition, O’Reilly Media.
- Hariom Tatsat, Sahil Puri, Brad Lookabaugh (2020), Machine Learning and Data Science Blueprints for Finance: From Building Trading Strategies to Robo-Advisors Using Python, O'Reilly Media
- Chris Kelliher (2022), Quantitative Finance With Python: A Practical Guide to Investment Management, Trading, and Financial Engineering, Chapman and Hall/CRC.
- Abdullah Karasan (2021), Machine Learning for Financial Risk Management with Python: Algorithms for Modeling Risk, O’Reilly Media.
- Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.
- Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu (2020), "Financial time series forecasting with deep learning: A systematic literature review: 2005–2019." Applied Soft Computing 90 (2020): 106181.
- Min-Yuh Day (2022), Python 101, https://tinyurl.com/aintpupython101