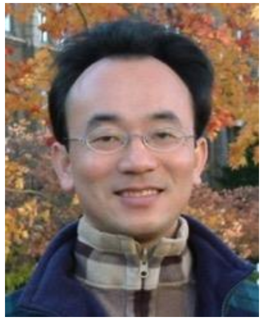
Artificial Intelligence in Finance and Quantitative Analysis

Financial Econometrics and Machine Learning

1132AIFQA06
MBA, IM, NTPU (M5147) (Spring 2025)
Tue 5, 6, 7 (13:10-16:00) (B3F17)



Min-Yuh Day, Ph.D, Professor

Institute of Information Management, National Taipei University

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

https://meet.google.com/miy-fbif-max

2025-04-15

Syllabus

Week	Date	Subject/Topics
1	2025/02/18	Introduction to Artificial Intelligence in Finance and Quantitative Analysis
2	2025/02/25	AI in FinTech: Metaverse, Web3, DeFi, NFT, Generative AI for Financial Innovation Applications
3	2025/03/04	Investing Psychology and Behavioral Finance
4	2025/03/11	Event Studies in Finance
5	2025/03/18	Case Study on AI in Finance and Quantitative Analysis I
6	2025/03/25	Finance Theory and Data-Driven Finance

Syllabus

Week Date Subject/Topics

7 2025/04/01 Self-Study

8 2025/04/08 Midterm Project Report

9 2025/04/15 Financial Econometrics and Machine Learning

10 2025/04/22 AI-First Finance

11 2025/04/29 Industry Practices of AI in Finance and Quantitative Analysis

12 2025/05/06 Case Study on AI in Finance and Quantitative Analysis II

Syllabus

Week	Date	Subject/Topics
13	2025/05/13	Deep Learning in Finance; Reinforcement Learning in Finance; Generative AI in Finance
14	2025/05/20	Algorithmic Trading; Risk Management; Trading Bot and Event-Based Backtesting
15	2025/05/27	Final Project Report I
16	2025/06/03	Final Project Report II

Financial Econometrics and Machine Learning

Outline

- Financial Econometrics
  - Financial Theories, OLS Regression
- Machine Learning
  - Learning, Evaluation, Bias and variance
  - Cross-validation

Financial Econometrics

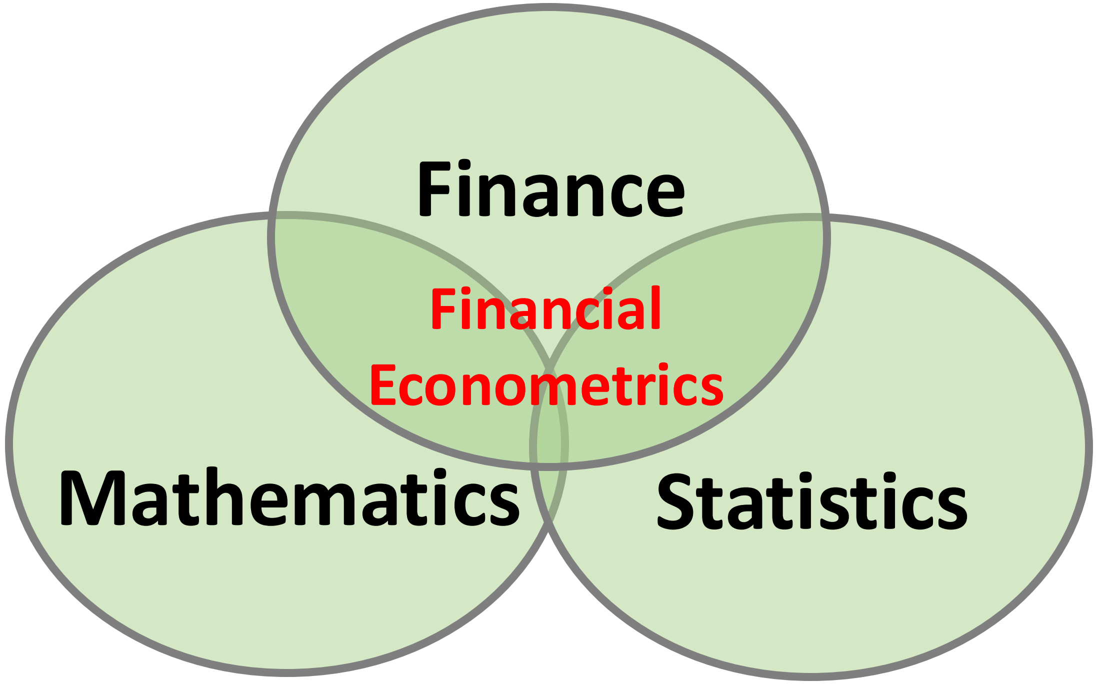
Financial Econometrics

- **Financial Theories**
- **OLS Regression**

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

Financial Econometrics

- **The discipline at the intersection of mathematics, statistics, and finance that applies such methods to financial market data is typically called financial econometrics.**



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

Financial Econometrics

(Chris Brooks, 2019)

- **Financial econometrics**
  - the application of statistical techniques to problems in finance
- Financial econometrics can be useful for
  testing theories in finance,
  determining asset prices or returns,
  testing hypotheses concerning the relationships between variables,
  examining the effect on financial markets of changes in economic conditions,
  forecasting future values of financial variables and for financial decision-making.

Source: Chris Brooks (2019), Introductory Econometrics for Finance, 4th Edition, Cambridge University Press

Financial Econometrics

- [Financial] econometrics is the quantitative application of statistical and mathematical models using [financial] data to develop financial theories or test existing hypotheses in finance and to forecast future trends from historical data.
- It subjects real-world [financial] data to statistical trials and then compares and contrasts the results against the [financial] theory or theories being tested.

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

Topics of Financial Econometrics

(Oliver Linton, 2019)

1. Econometric
2. Return Predictability and the Efficient Markets Hypothesis
3. Robust Tests and Tests of Nonlinear Predictability of Returns
4. Empirical Market Microstructure
5. Event Study Analysis
6. Portfolio Choice and Testing the Capital Asset Pricing Model
7. Multifactor Pricing Models

Source: Oliver Linton (2019), Financial Econometrics: Models and Methods, Cambridge University Press

Topics of Financial Econometrics

(Oliver Linton, 2019)

8. **Present Value Relations**
9. **Intertemporal Equilibrium Pricing**
10. **Volatility**
11. **Continuous Time Processes**
12. **Yield Curve**
13. **Risk Management and Tail Estimation**

Source: Oliver Linton (2019), Financial Econometrics: Models and Methods, Cambridge University Press

Applications of Financial Econometrics

(Chris Brooks, 2019)

1. **Testing whether financial markets are weak-form informationally efficient**
2. **Testing whether the capital asset pricing model (CAPM) or arbitrage pricing theory (APT) represent superior models for the determination of returns on risky assets**
3. **Measuring and forecasting the volatility of bond returns**
4. **Explaining the determinants of bond credit ratings used by the ratings agencies**
5. **Modelling long-term relationships between prices and exchange rates**

Source: Chris Brooks (2019), Introductory Econometrics for Finance, 4th Edition, Cambridge University Press

Applications of Financial Econometrics

(Chris Brooks, 2019)

6. Determining the optimal hedge ratio for a spot position in oil
7. Testing technical trading rules to determine which makes the most money
8. Testing the hypothesis that earnings or dividend announcements have no effect on stock prices
9. Testing whether spot or futures markets react more rapidly to news
10. Forecasting the correlation between the stock indices of two countries

Source: Chris Brooks (2019), Introductory Econometrics for Finance, 4th Edition, Cambridge University Press

Machine Learning and Financial Econometrics

- ML and DL methods are able to discover statistical inefficiencies and even economic inefficiencies that are not discoverable by traditional econometric methods, such as multivariate OLS regression.

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

Normative Financial Theories

- **Normative financial theories** mostly rely on assumptions and axioms in combination with deduction as the major analytical method to arrive at their central results.
  - **Expected utility theory (EUT)** assumes that agents have the same utility function no matter what state of the world unfolds and that they maximize expected utility under conditions of uncertainty.
  - **Mean-variance portfolio (MVP)** theory describes how investors should invest under conditions of uncertainty assuming that only the expected return and the expected volatility of a portfolio over one period count.

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

Normative Financial Theories

- **The capital asset pricing model (CAPM) assumes that only the nondiversifiable market risk explains the expected return and the expected volatility of a stock over one period.**
- **Arbitrage pricing theory (APT) assumes that a number of identifiable risk factors explains the expected return and the expected volatility of a stock over time; admittedly, compared to the other theories, the formulation of APT is rather broad and allows for wide-ranging interpretations.**

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

Financial Econometrics and Regression

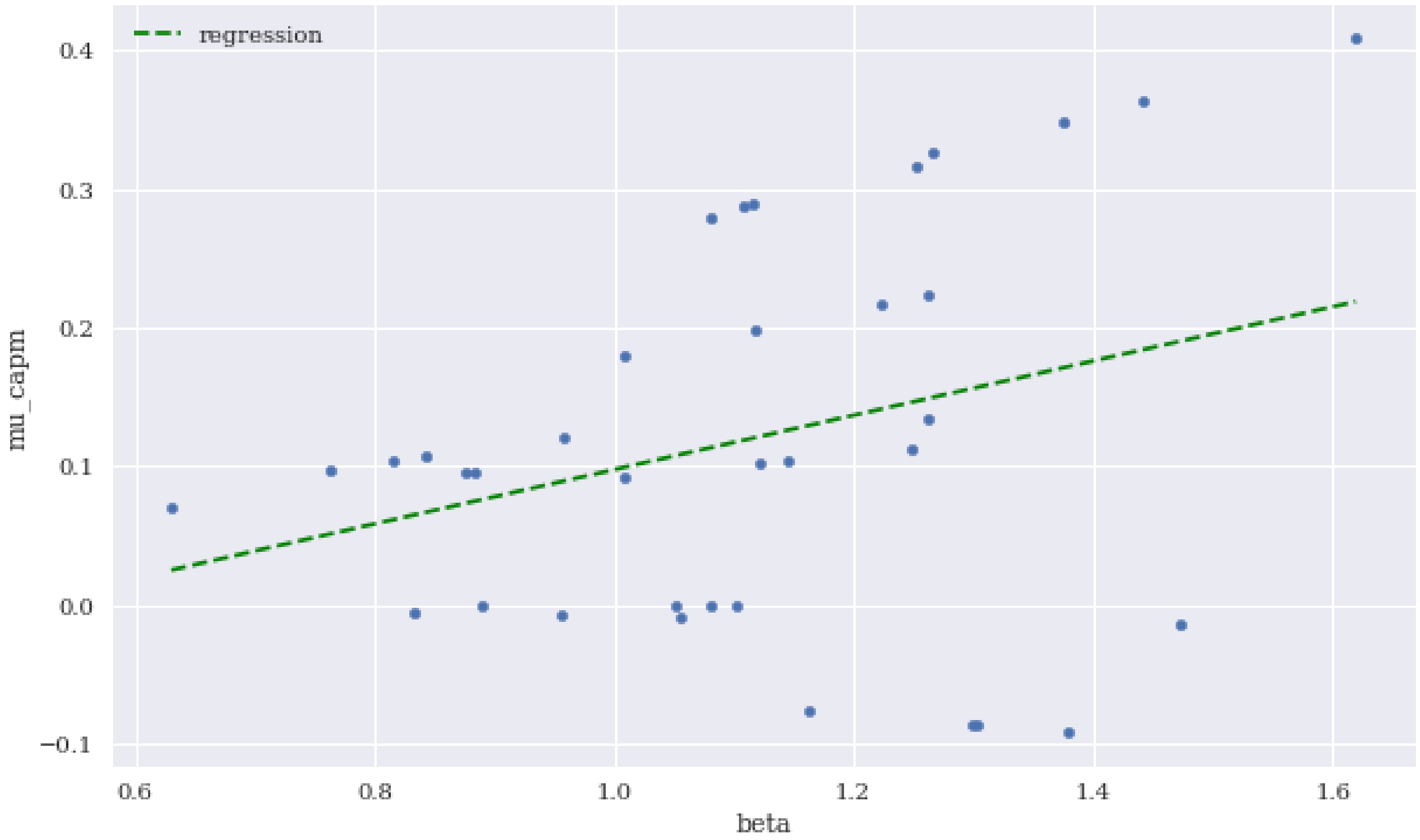
- One of the major tools in financial econometrics is regression, in both its univariate and multivariate forms
  - $y = \alpha + \beta\, x$
  - $y = \alpha + \beta_1\, x_1 + \beta_2\, x_2$
  - $y = \alpha + \beta_1\, x_1 + \beta_2\, x_2 + \beta_3\, x_3$
- Regression is also a central tool in statistical learning in general

CAPM and APT
OLS regression

- **Both the CAPM and the APT relate the output variables with the relevant input factors in linear fashion.**
- **From an econometric point of view, both models are implemented based on linear ordinary least-squares (OLS) regression.**
- **CAPM: univariate linear OLS regression**
- **APT: multivariate OLS regression**

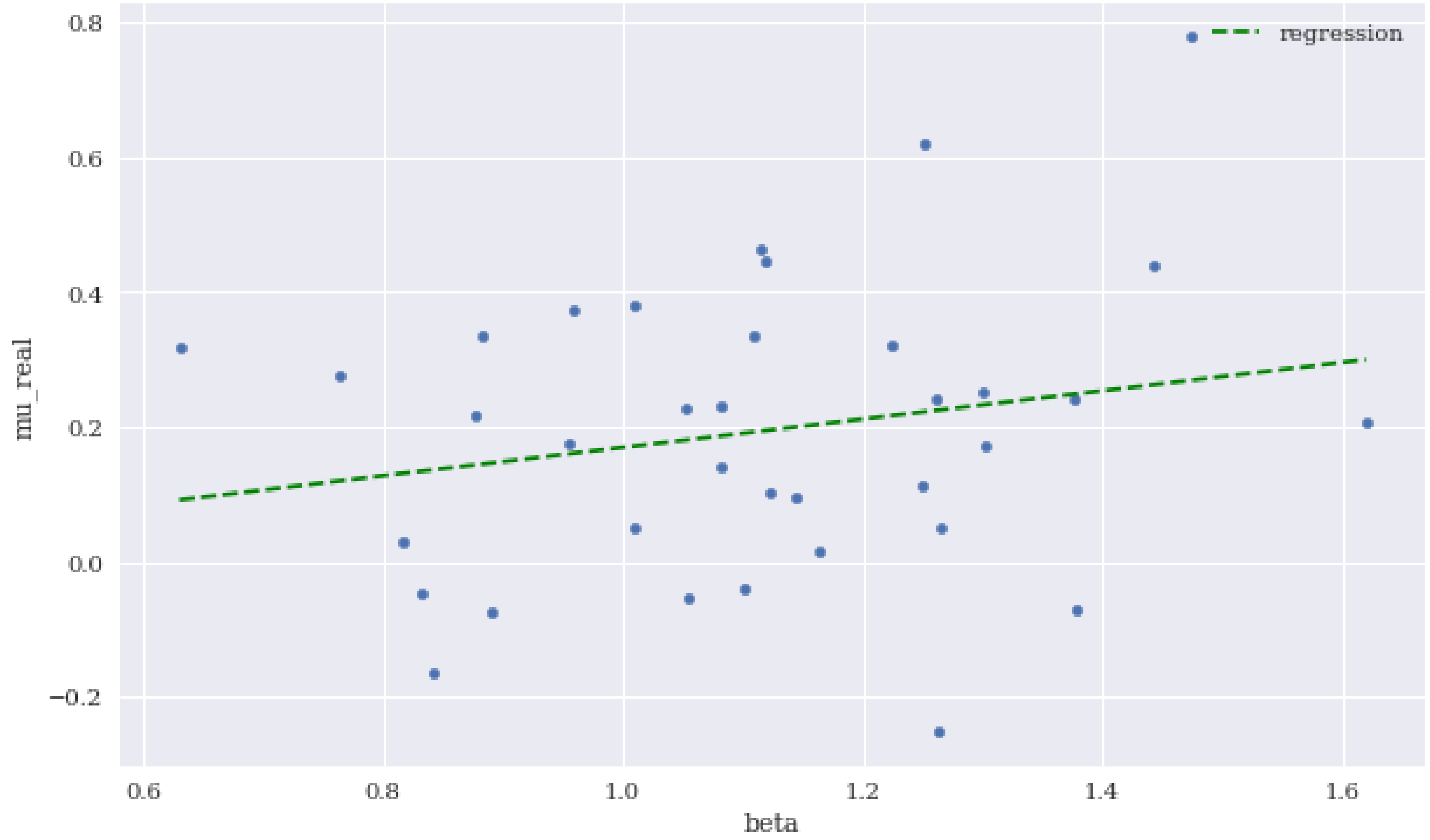
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Expected CAPM return versus beta (including linear regression)



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

Expected CAPM return versus beta (including linear regression)



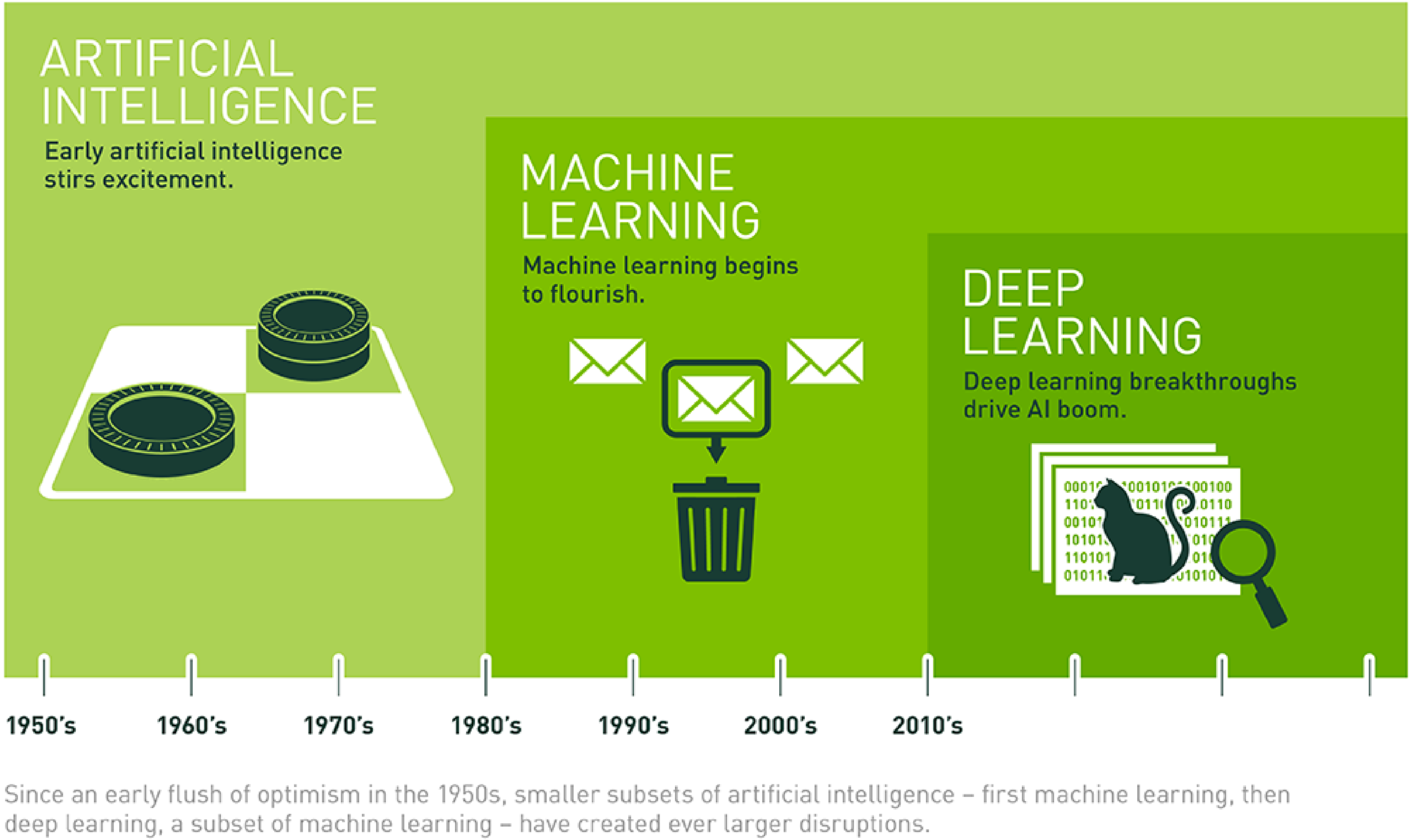
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Machine Learning

Machine Learning

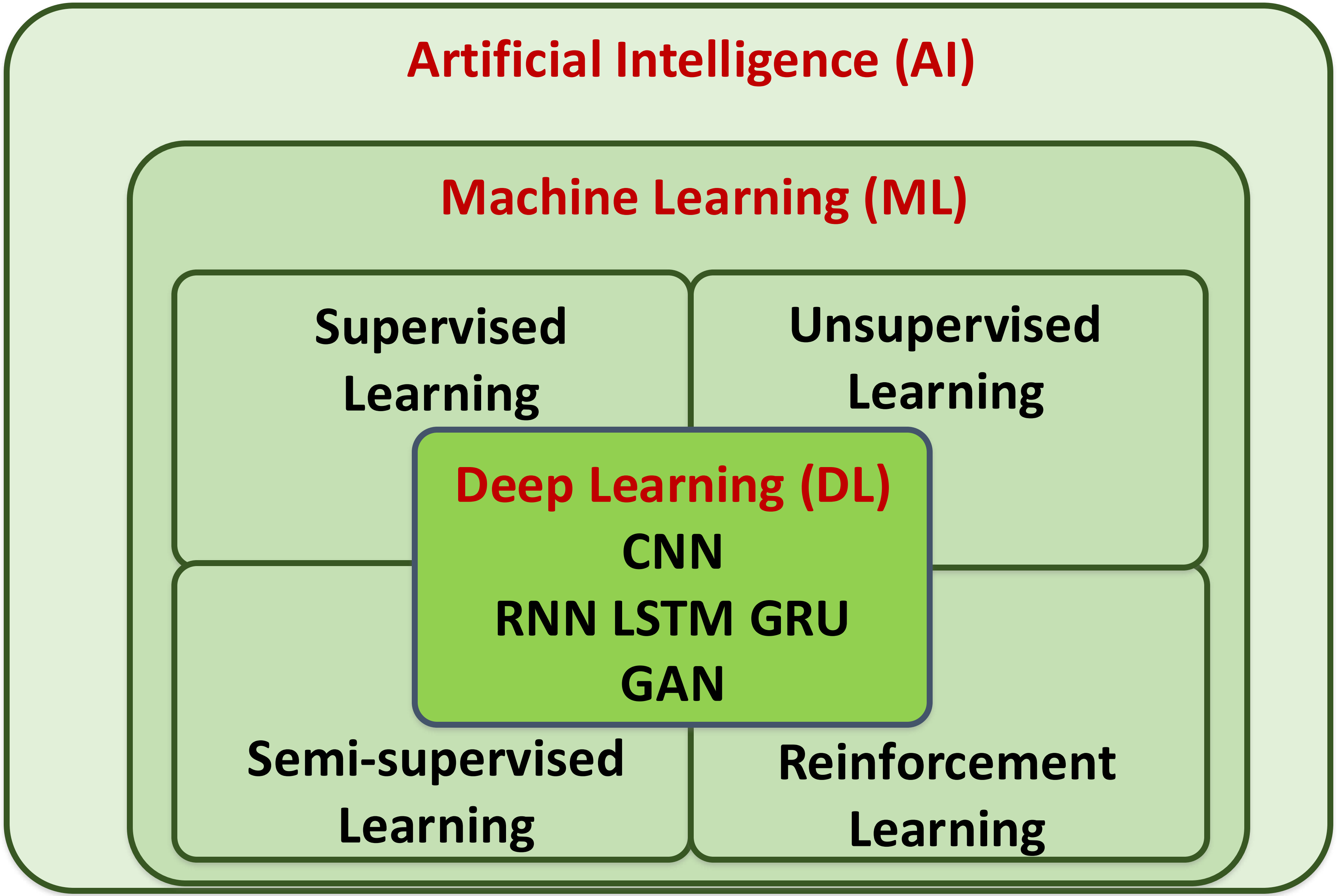
- Learning
- Evaluation
- Bias and variance
- Cross-validation

Artificial Intelligence
Machine Learning & Deep Learning



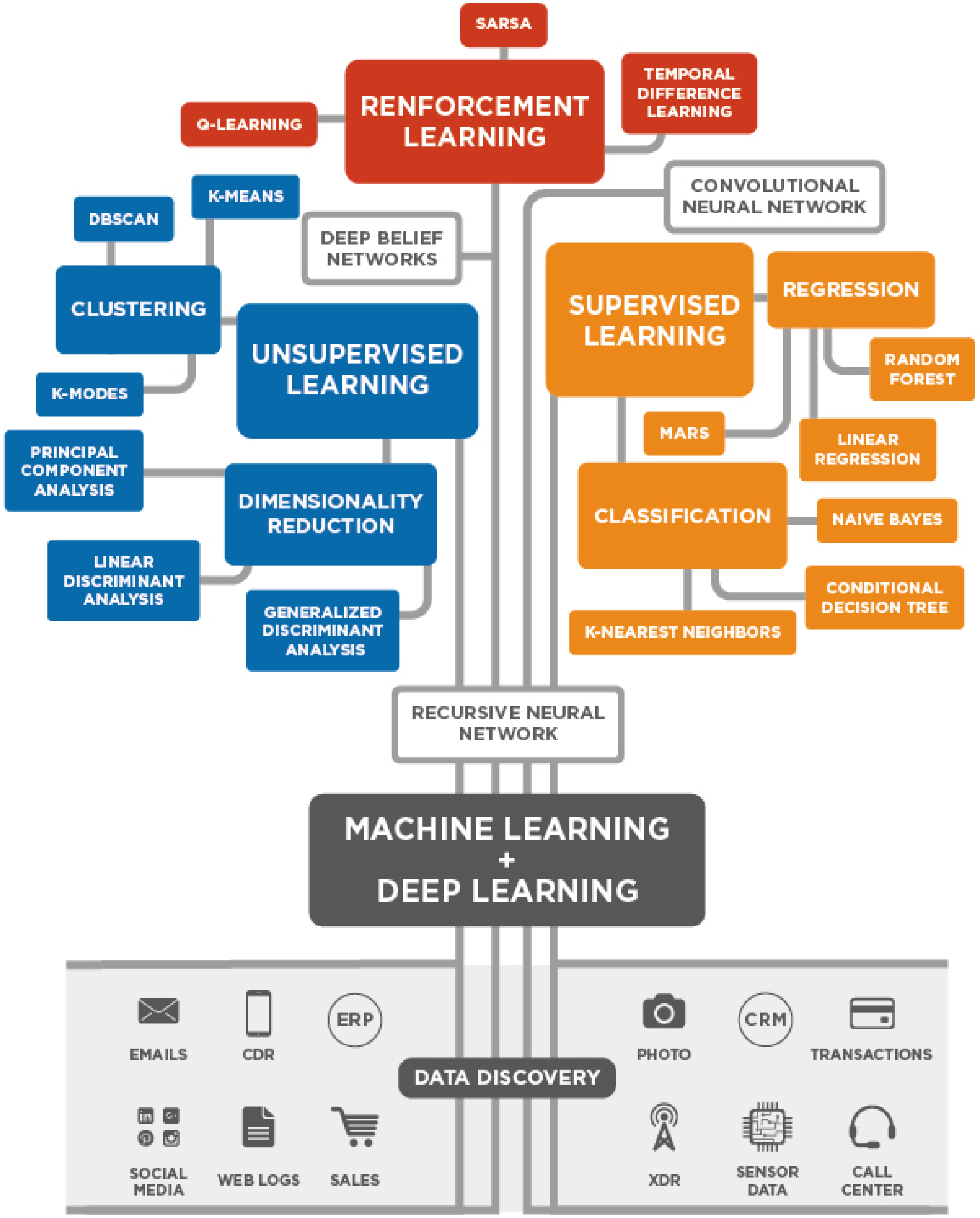
Source: https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/

AI, ML, DL



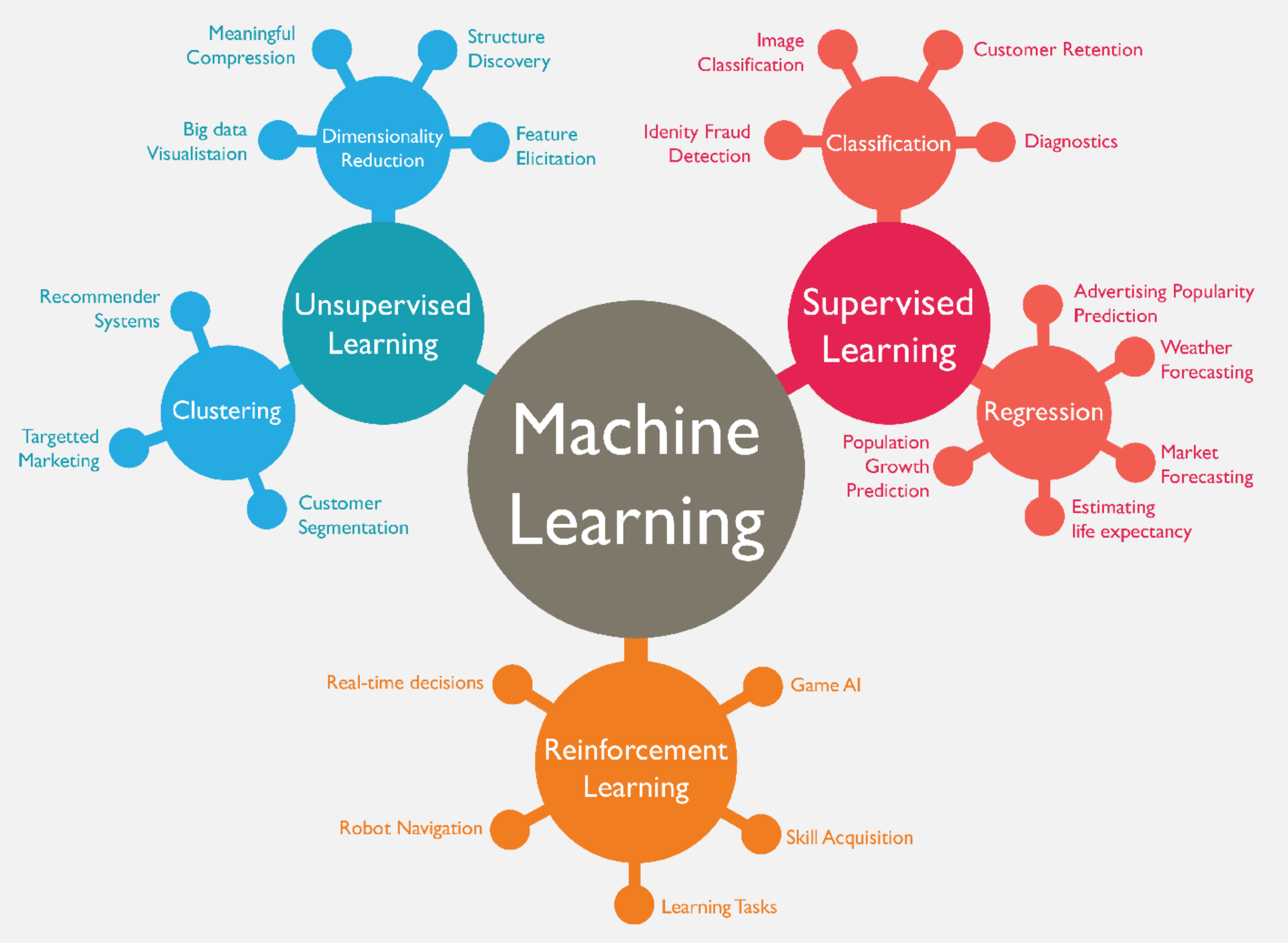
Source: https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/deep_learning.html

3 Machine Learning Algorithms



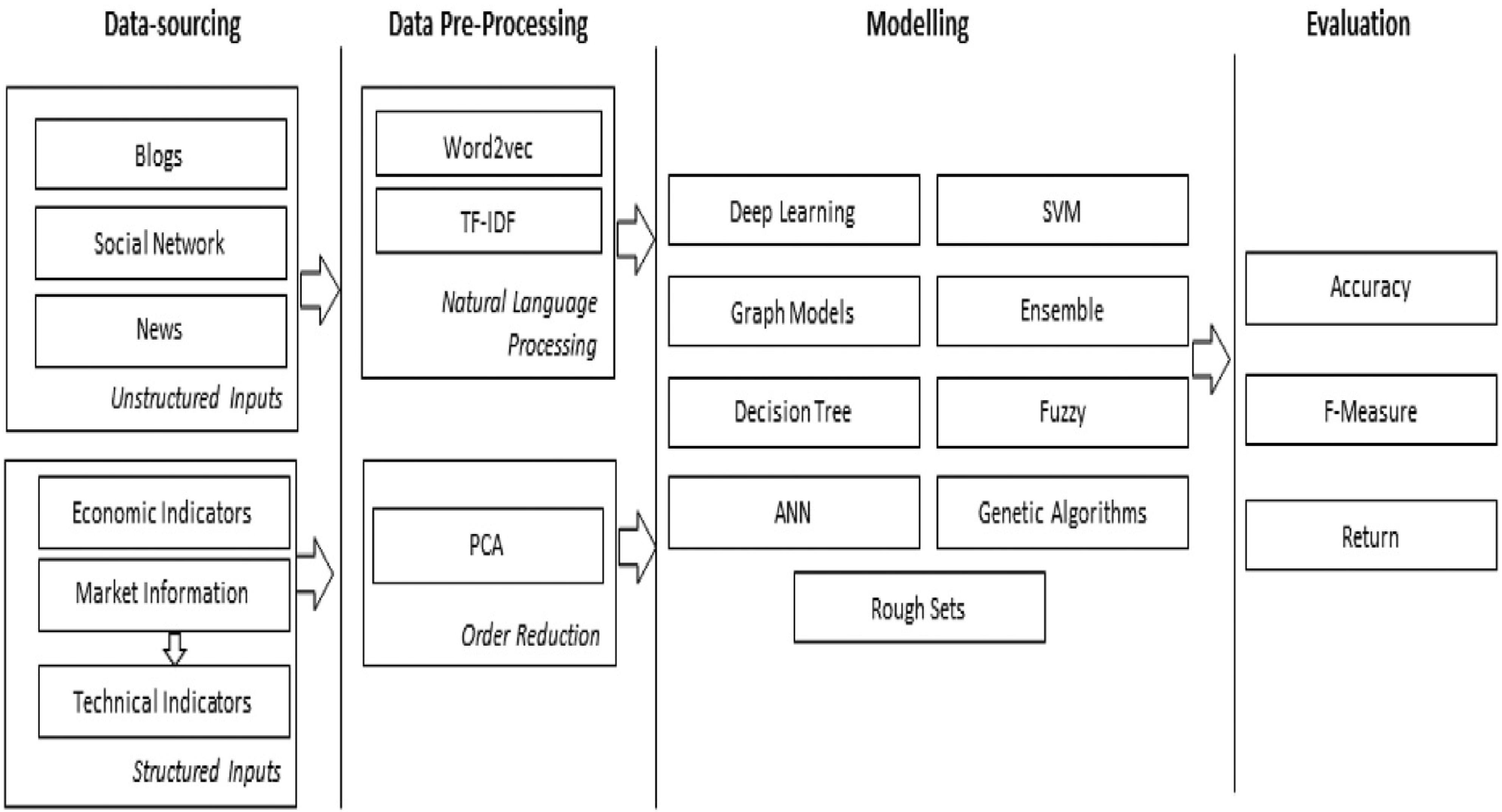
Source: Enrico Galimberti, http://blogs.teradata.com/data-points/tree-machine-learning-algorithms/

Machine Learning (ML)



Source: https://www.mactores.com/services/aws-big-data-machine-learning-cognitive-services/

Stock Market Movement Forecast:
ML 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.

Machine Learning

- Learning
- Data: Features, Labels
- Success (Loss Function): MSE
- Capacity (Model Fit)
- Evaluation
- Bias and variance
- Cross-validation

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

Learning (Mitchell, 1997)

- A computer program is said
  to learn from experience E
  with respect to some class of tasks T
  and performance measure P,
  if its performance at tasks in T,
  as measured by P,
  improves with experience E.

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

Machine Learning
Supervised Learning (Classification)
Learning from Examples

$$y = f(x)$$

x	y

Machine Learning
Supervised Learning (Classification)
Learning from Examples

$$y = f(x)$$

x	y
input	*Output*
	label

Machine Learning
Supervised Learning (Classification)
Learning from Examples

$$y = f(x)$$

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
6.9,3.1,4.9,1.5,Iris-versicolor
6.3,3.3,6.0,2.5,Iris-virginica
5.8,2.7,5.1,1.9,Iris-virginica
7.1,3.0,5.9,2.1,Iris-virginica
```

Machine Learning
Supervised Learning (Classification)
Learning from Examples

$y = f(x)$

Example

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
6.9,3.1,4.9,1.5,Iris-versicolor
6.3,3.3,6.0,2.5,Iris-virginica
5.8,2.7,5.1,1.9,Iris-virginica
7.1,3.0,5.9,2.1,Iris-virginica
```

Machine Learning
Supervised Learning (Classification)
Learning from Examples

$y = f(x)$

Example

x

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
6.9,3.1,4.9,1.5,Iris-versicolor
6.3,3.3,6.0,2.5,Iris-virginica
5.8,2.7,5.1,1.9,Iris-virginica
7.1,3.0,5.9,2.1,Iris-virginica
```

y

Time Series Data

[10, 20, 30, 40, 50, 60, 70, 80, 90]

X	Y
[10 20 30]	40
[20 30 40]	50
[30 40 50]	60
[40 50 60]	70
[50 60 70]	80
[60 70 80]	90

Time Series Data

[100, 110, 120, 130, 140, 150]

Y

X Y

[100 110 120 130 140] 150

X_{t1} X_{t2} X_{t3} X_{t4} X_{t5}

The Theory of Learning

- **How can we be sure that our learned hypothesis will predict well for previously unseen inputs?**
  - **How do we know that the hypothesis h is close to the target function f if we don’t know what is?**
- **How many examples do we need to get a good h?**
- **What hypothesis space should we use?**
- **If the hypothesis space is very complex, can we even find the best h or do we have to settle for a local maximum?**
- **How complex should h be?**
- **How do we avoid overfitting?**

Linear function

$$y = f(x)$$

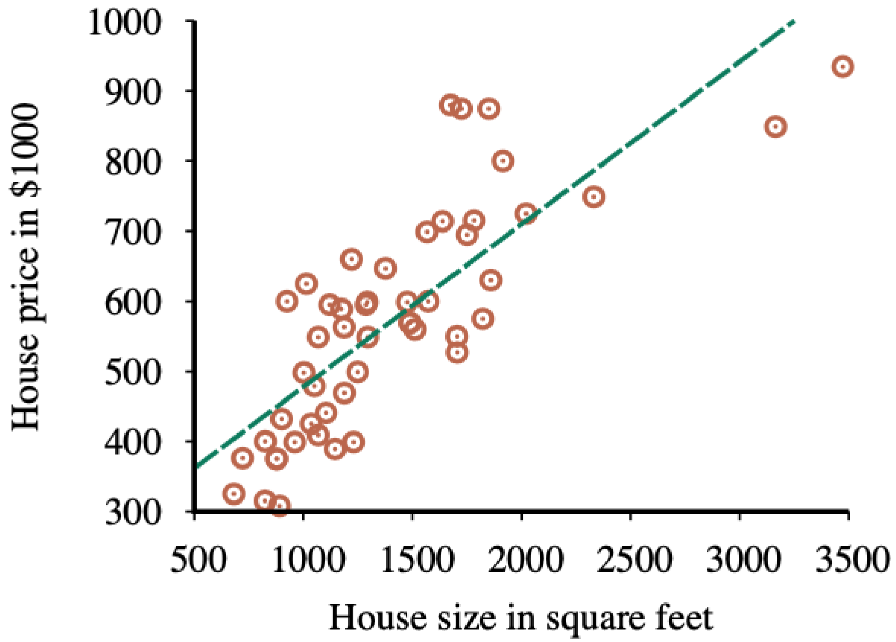
$$y = w_1 x + w_0$$

$$h_w(x) = w_1 x + w_0$$

Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

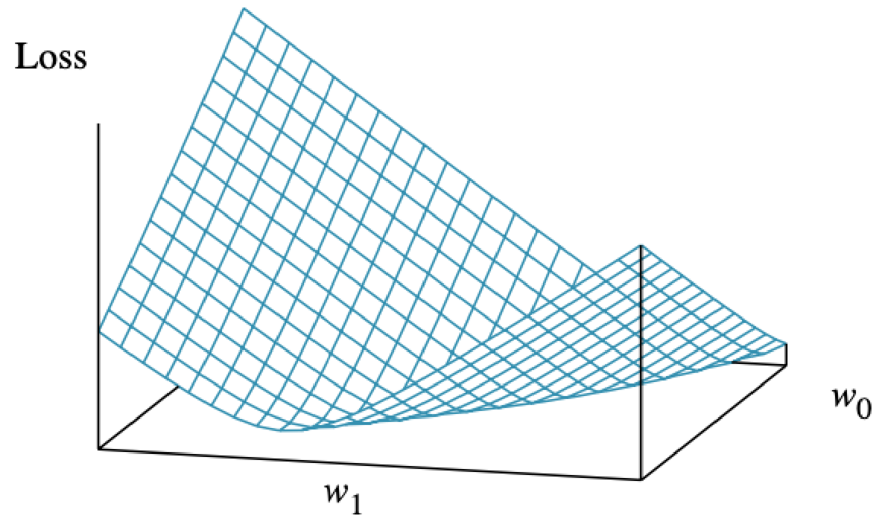
Linear Regression Weight Space

$$h_w(x) = w_1x + w_0$$



$y = 0.232\,x + 246$

$$w^* = \text{argmin}_w\ Loss(h_w)$$



Loss function for Weights (w_1, w_0)

Performance Measure

- The measure of success for estimation problems
  - mean-squared error (MSE)
- Classification problems
  - accuracy ratio

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

Evaluation
(Accuracy of Classification Model)

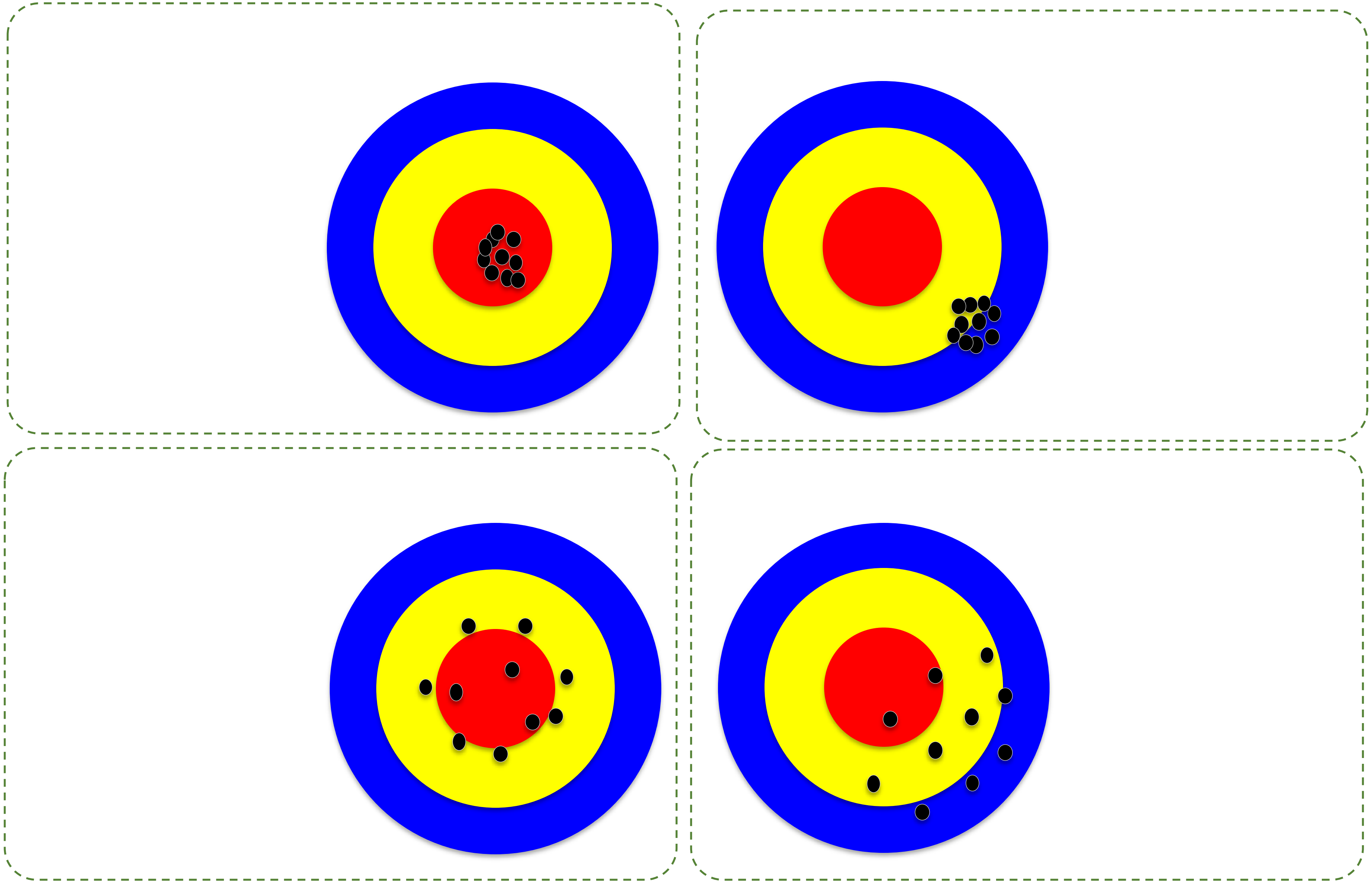
Assessing the Classification Model

- Predictive accuracy
  - Hit rate
- Speed
  - Model building; predicting
- Robustness
- Scalability
- Interpretability
  - Transparency, explainability

Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson

Accuracy Validity

Precision Reliability



Accuracy vs. Precision

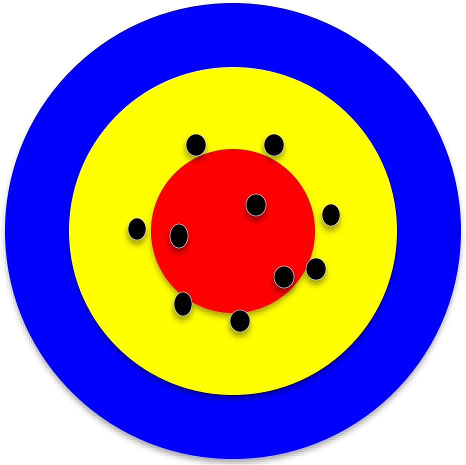
A

High Accuracy
High Precision

B

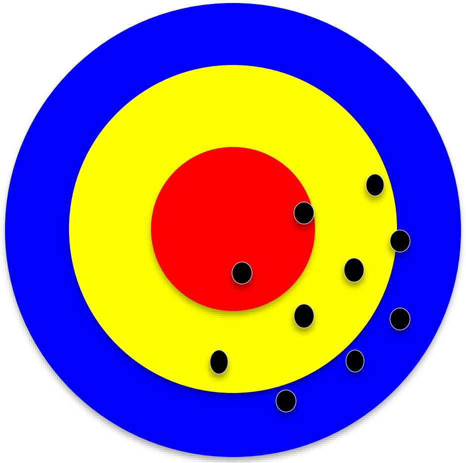
Low Accuracy
High Precision

C



High Accuracy
Low Precision

D



Low Accuracy
Low Precision

Accuracy vs. Precision

A

High Accuracy
High Precision

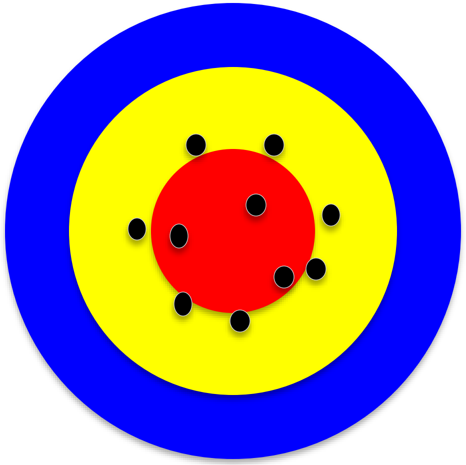
High Validity
High Reliability

B

Low Accuracy
High Precision

Low Validity
High Reliability

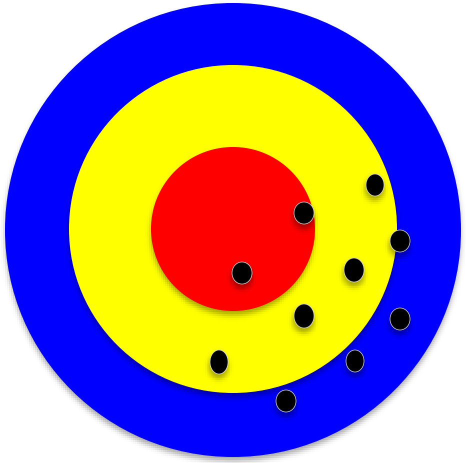
C



High Accuracy
Low Precision

High Validity
Low Reliability

D



Low Accuracy
Low Precision

Low Validity
Low Reliability

Accuracy vs. Precision

A

High Accuracy
High Precision

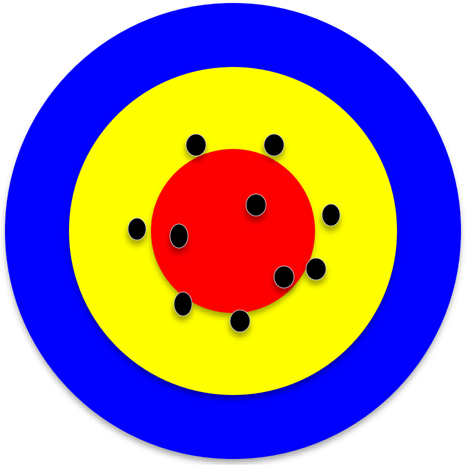
High Validity
High Reliability

B

Low Accuracy
High Precision

Low Validity
High Reliability

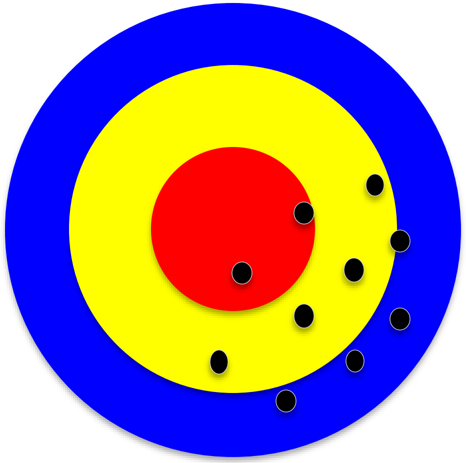
C



High Accuracy
Low Precision

High Validity
Low Reliability

D



Low Accuracy
Low Precision

Low Validity
Low Reliability

Confusion Matrix

for Tabulation of Two-Class Classification Results

		True/Observed Class	
		Positive	**Negative**
Predicted Class	**Positive**	True Positive Count (TP)	False Positive Count (FP)
	Negative	False Negative Count (FN)	True Negative Count (TN)

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$True\ Positive\ Rate = \frac{TP}{TP + FN}$$

$$True\ Negative\ Rate = \frac{TN}{TN + FP}$$

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

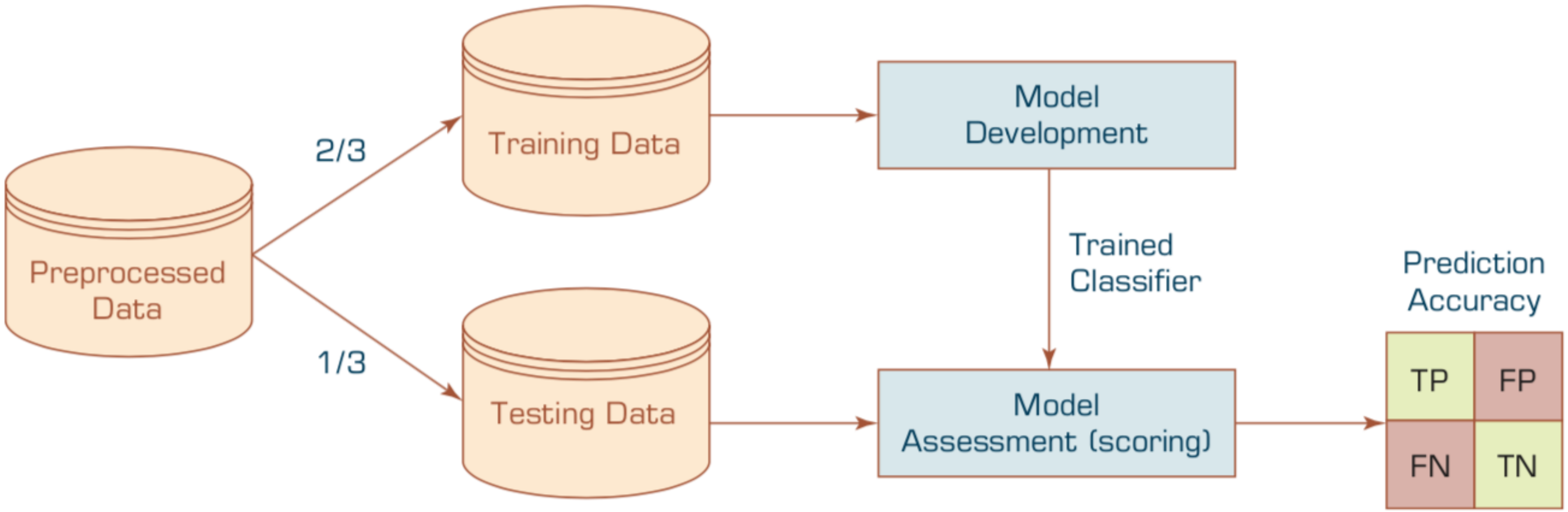
Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson

Sensitivity =True Positive Rate

Specificity =True Negative Rate

Estimation Methodologies for Classification

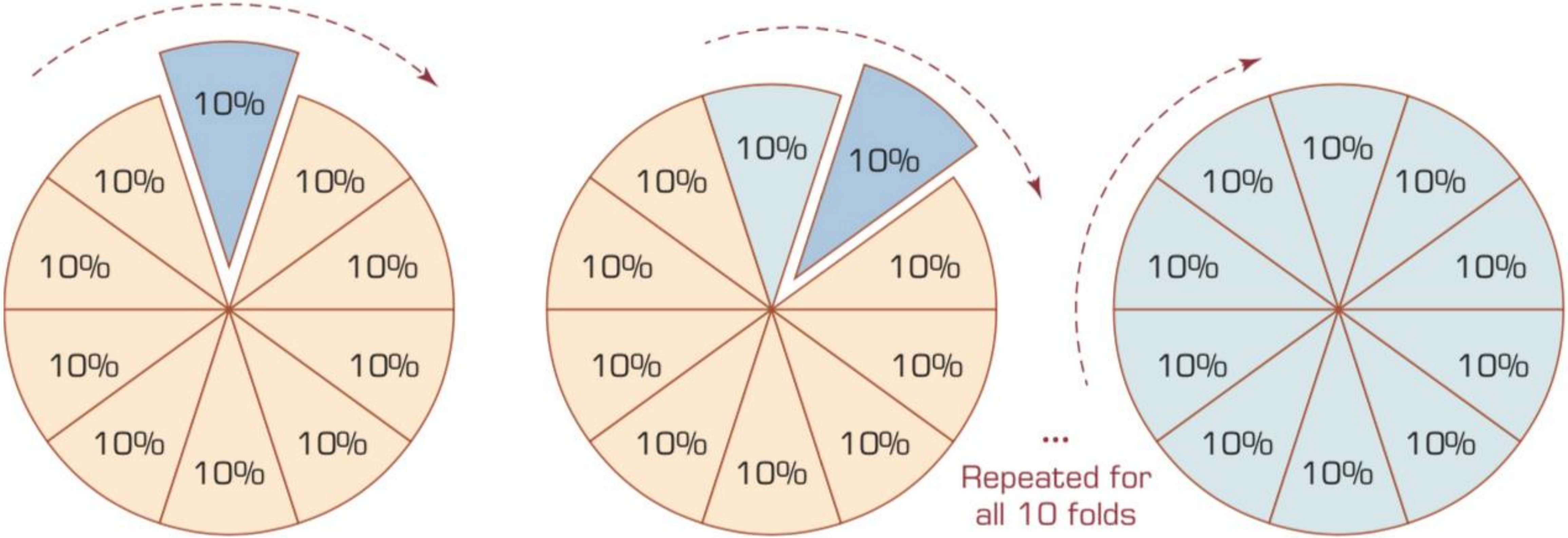
- **Simple split** (or holdout or test sample estimation)
  - Split the data into 2 mutually exclusive sets training (~70%) and testing (30%)



  - For ANN, the data is split into three sub-sets (training [~60%], validation [~20%], testing [~20%])

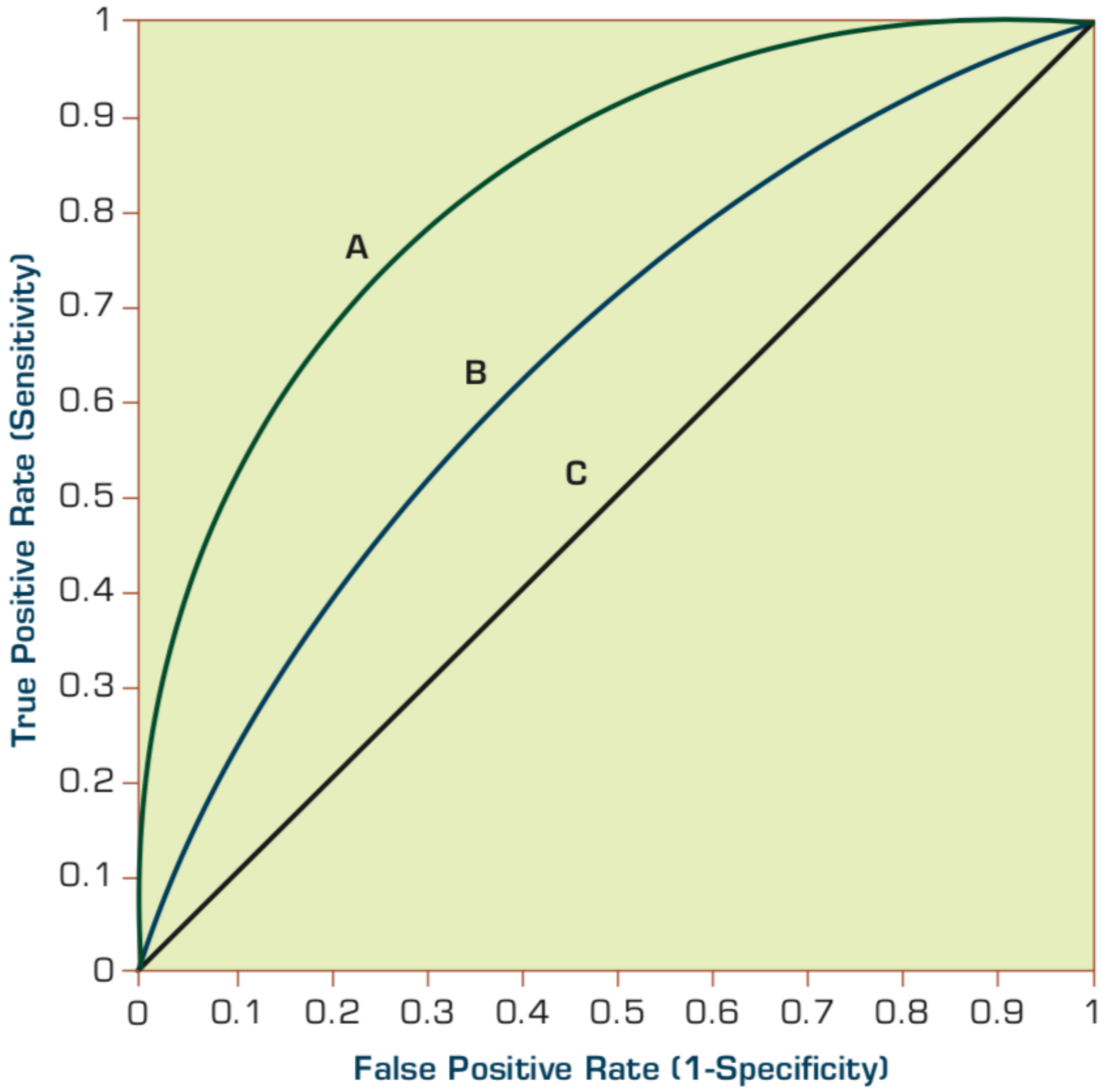
Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson

k-Fold Cross-Validation



Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson

Estimation Methodologies for Classification
Area under the ROC curve



Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson

		True Class (actual value)		
		Positive	Negative	total
Predictive Class (prediction outcome)	Positive	True Positive (TP)	False Positive (FP)	P'
	Negative	False Negative (FN)	True Negative (TN)	N'
total		P	N	

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$True\ Positive\ Rate = \frac{TP}{TP + FN}$$

$$True\ Negative\ Rate = \frac{TN}{TN + FP}$$

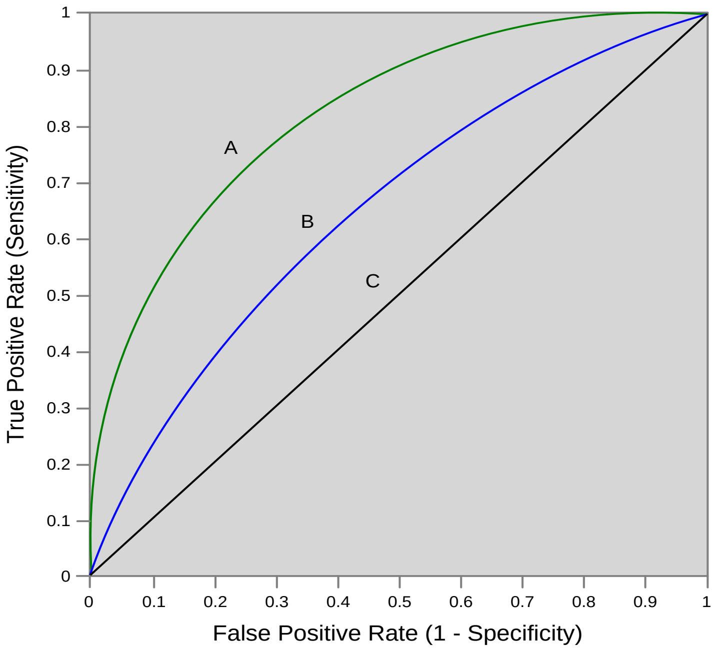
$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

$$True\ Positive\ Rate\ (\text{Sensitivity}) = \frac{TP}{TP + FN}$$

$$True\ Negative\ Rate\ (\text{Specificity}) = \frac{TN}{TN + FP}$$

$$False\ Positive\ Rate = \frac{FP}{FP + TN}$$

$$False\ Positive\ Rate\ (1 - \text{Specificity}) = \frac{FP}{FP + TN}$$



Source: http://en.wikipedia.org/wiki/Receiver_operating_characteristic

		True Class (actual value)		
		Positive	Negative	total
Predictive Class (prediction outcome)	Positive	True Positive (TP)	False Positive (FP)	P'
	Negative	False Negative (FN)	True Negative (TN)	N'
total		P	N	

$$True\ Positive\ Rate\ (\mathrm{Sensitivity}) = \frac{TP}{TP+FN}$$

Sensitivity

= True Positive Rate

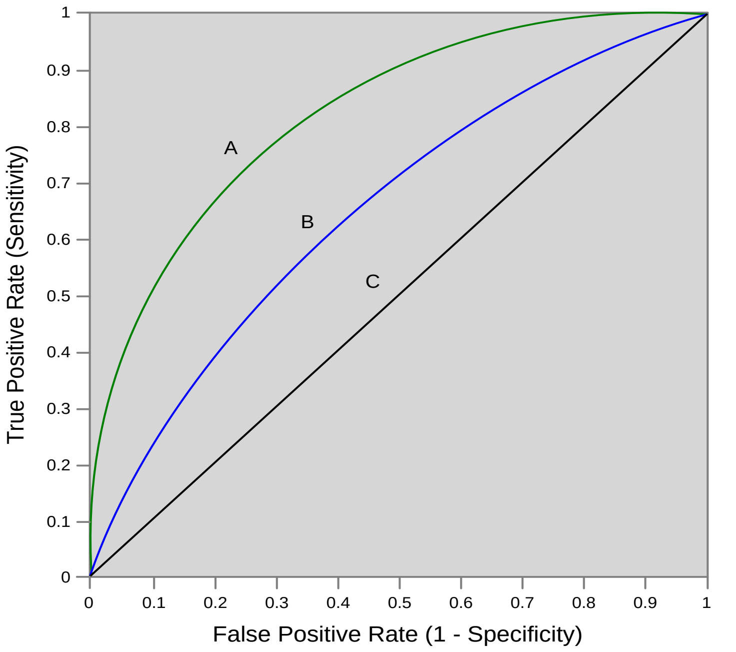
= Recall

= Hit rate

= TP / (TP + FN)

$$True\ Positive\ Rate = \frac{TP}{TP+FN}$$

$$Recall = \frac{TP}{TP+FN}$$



Source: http://en.wikipedia.org/wiki/Receiver_operating_characteristic

		True Class (actual value)		
		Positive	Negative	total
Predictive Class (prediction outcome)	Positive	True Positive (TP)	False Positive (FP)	P'
	Negative	False Negative (FN)	True Negative (TN)	N'
total		P	N	

$$True\ Negative\ Rate = \frac{TN}{TN + FP}$$

Specificity

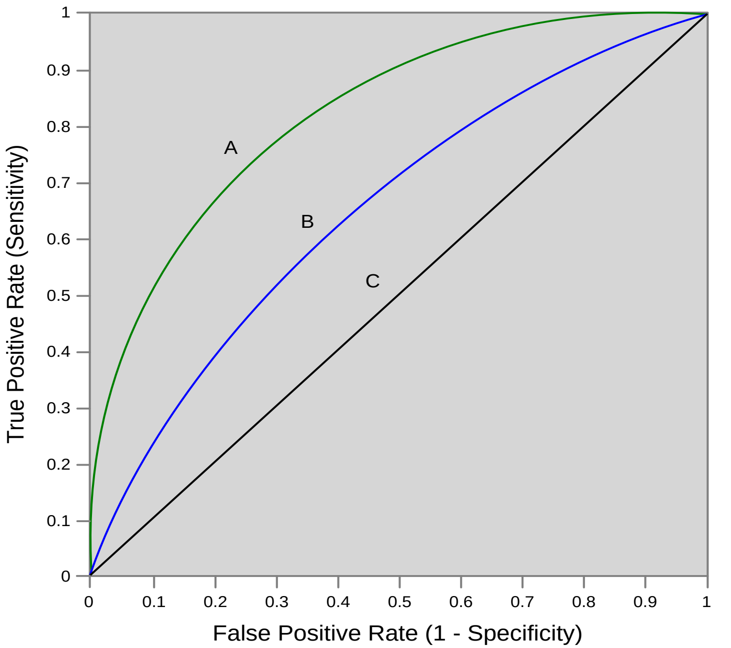
= True Negative Rate

= TN / N

= TN / (TN+ FP)

$$True\ Negative\ Rate\ (\text{Specificity}) = \frac{TN}{TN + FP}$$

$$False\ Positive\ Rate\ (1 - \text{Specificity}) = \frac{FP}{FP + TN}$$



Source: http://en.wikipedia.org/wiki/Receiver_operating_characteristic

		True Class (actual value)		
		Positive	Negative	total
Predictive Class (prediction outcome)	Positive	True Positive (TP)	False Positive (FP)	P'
	Negative	False Negative (FN)	True Negative (TN)	N'
total		P	N	

Precision

= Positive Predictive Value (PPV)

$$Precision = \frac{TP}{TP + FP}$$

Recall

= True Positive Rate (TPR)

= Sensitivity

= Hit Rate

$$Recall = \frac{TP}{TP + FN}$$

F1 score (F-score)(F-measure)

is the harmonic mean of precision and recall

= 2TP / (P + P')

= 2TP / (2TP + FP + FN)

$$F = 2 * \frac{precision * recall}{precision + recall}$$

Source: http://en.wikipedia.org/wiki/Receiver_operating_characteristic

A

63 (TP)	28 (FP)	91
37 (FN)	72 (TN)	109
100	100	200

Recall
= True Positive Rate (TPR)
= Sensitivity
= Hit Rate
= TP / (TP + FN)

Specificity
= True Negative Rate
= TN / N
= TN / (TN + FP)

TPR = 0.63

$$Recall = \frac{TP}{TP + FN}$$

$$True\ Negative\ Rate\ (\text{Specificity}) = \frac{TN}{TN + FP}$$

FPR = 0.28

$$False\ Positive\ Rate\ (1 - \text{Specificity}) = \frac{FP}{FP + TN}$$

PPV = 0.69
=63/(63+28)
=63/91

$$Precision = \frac{TP}{TP + FP}$$

Precision
= Positive Predictive Value (PPV)

F1 = 0.66
= 2*(0.63*0.69)/(0.63+0.69)
= (2 * 63) /(100 + 91)
= (0.63 + 0.69) / 2 =1.32 / 2 =0.66

$$F = 2 * \frac{precision * recall}{precision + recall}$$

F1 score (F-score)
(F-measure)
is the harmonic mean of precision and recall
= 2TP / (P + P')
= 2TP / (2TP + FP + FN)

ACC = 0.68
= (63 + 72) / 200
= 135/200 = 67.5

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Source: http://en.wikipedia.org/wiki/Receiver_operating_characteristic

A

63 (TP)	28 (FP)	91
37 (FN)	72 (TN)	109
100	100	200

TPR = 0.63

FPR = 0.28

PPV = 0.69
=63/(63+28)
=63/91

F1 = 0.66
= 2*(0.63*0.69)/(0.63+0.69)
= (2 * 63) /(100 + 91)
= (0.63 + 0.69) / 2 =1.32 / 2 =0.66

ACC = 0.68
= (63 + 72) / 200
= 135/200 = 67.5

B

77 (TP)	77 (FP)	154
23 (FN)	23 (TN)	46
100	100	200

TPR = 0.77
FPR = 0.77
PPV = 0.50
F1 = 0.61
ACC = 0.50

Recall
= True Positive Rate (TPR)
= Sensitivity
= Hit Rate

$$Recall = \frac{TP}{TP + FN}$$

Precision
= Positive Predictive Value (PPV)

$$Precision = \frac{TP}{TP + FP}$$

Source: http://en.wikipedia.org/wiki/Receiver_operating_characteristic

C

24 (TP)	88 (FP)	112
76 (FN)	12 (TN)	88
100	100	200

TPR = 0.24
FPR = 0.88
PPV = 0.21
F1 = 0.22
ACC = 0.18

C'

76 (TP)	12 (FP)	88
24 (FN)	88 (TN)	112
100	100	200

TPR = 0.76
FPR = 0.12
PPV = 0.86
F1 = 0.81
ACC = 0.82

Recall
= True Positive Rate (TPR)
= Sensitivity
= Hit Rate

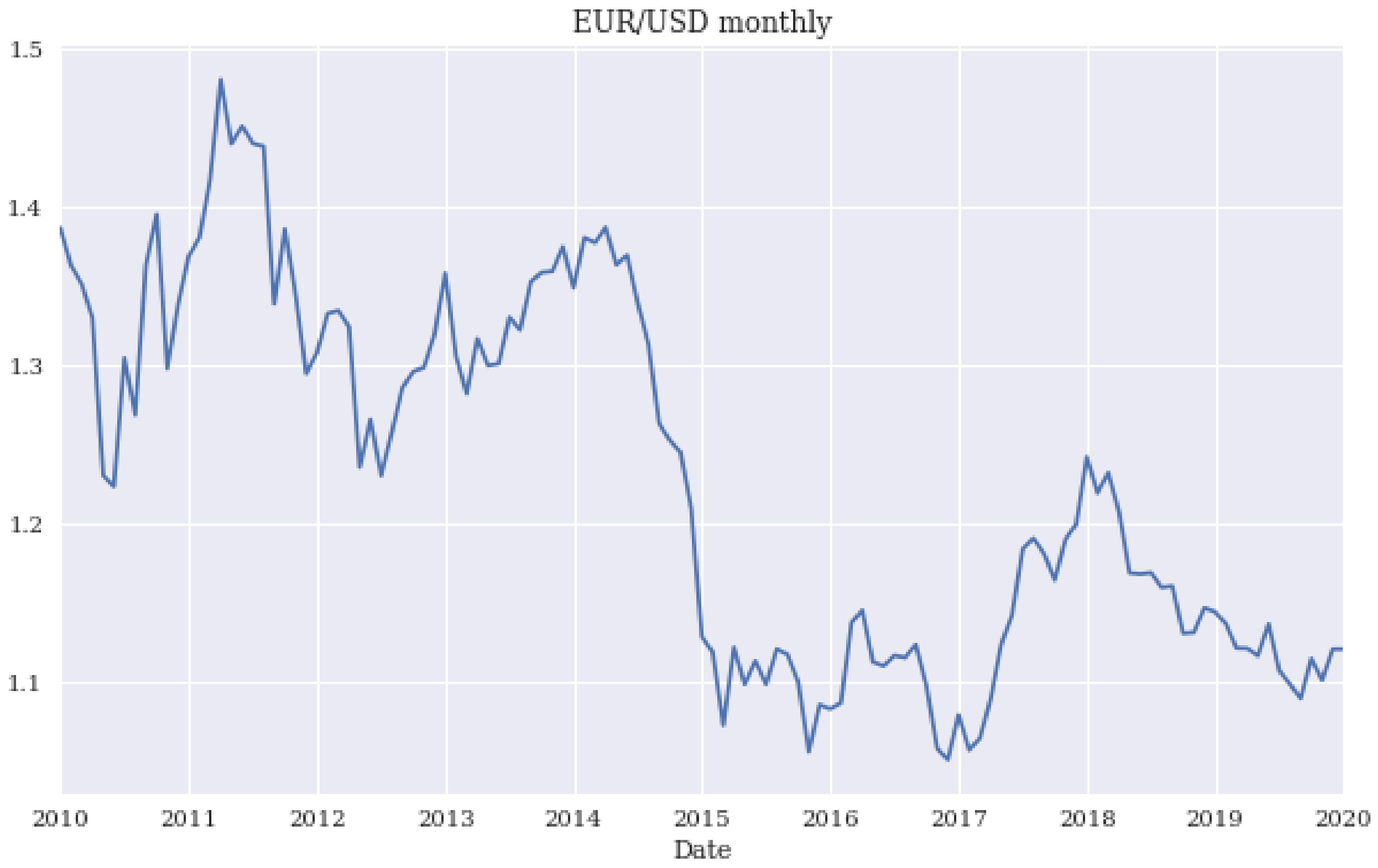
$$Recall = \frac{TP}{TP + FN}$$

Precision
= Positive Predictive Value (PPV)

$$Precision = \frac{TP}{TP + FP}$$

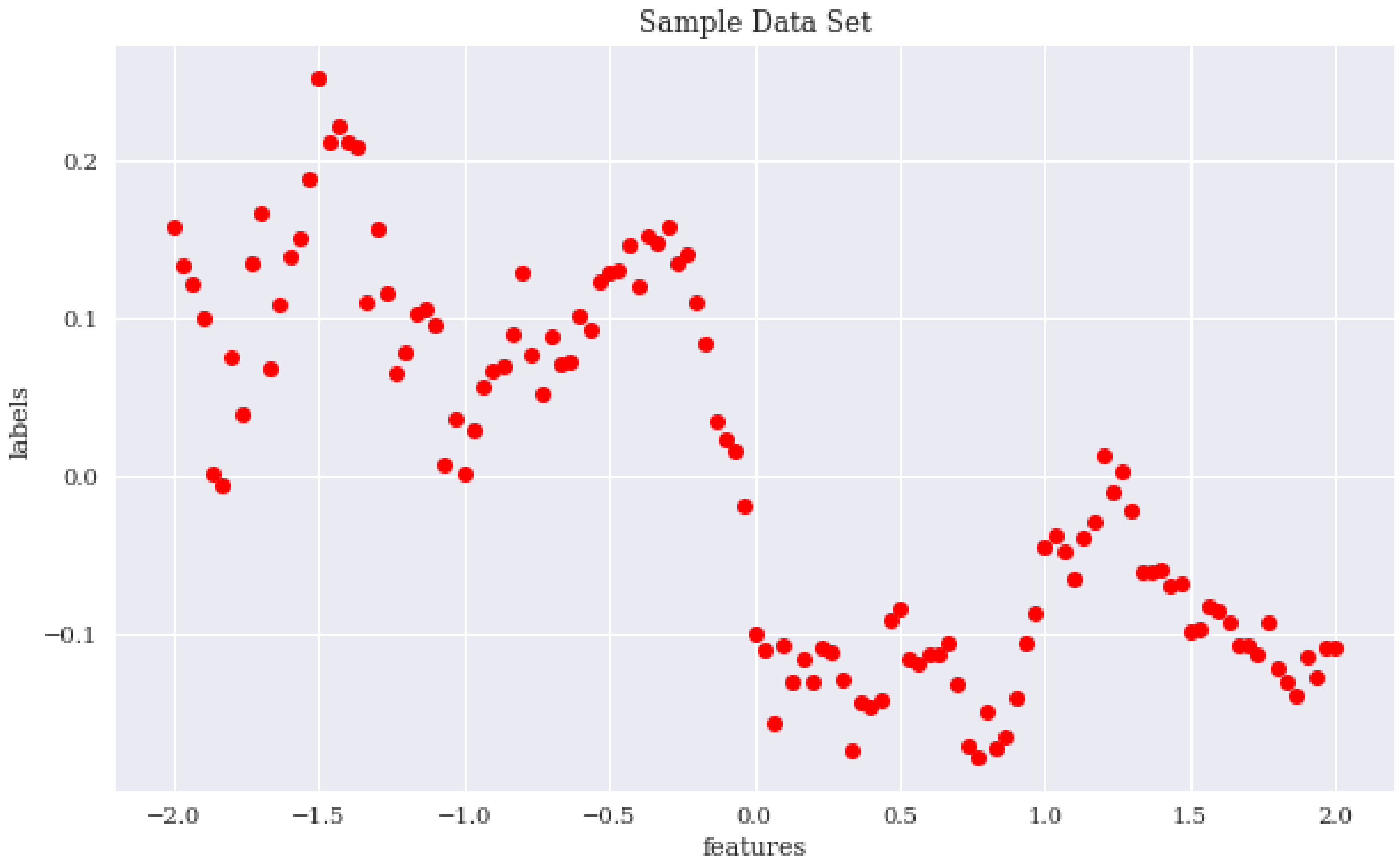
Source: http://en.wikipedia.org/wiki/Receiver_operating_characteristic

EUR/USD exchange rate as time series (monthly)



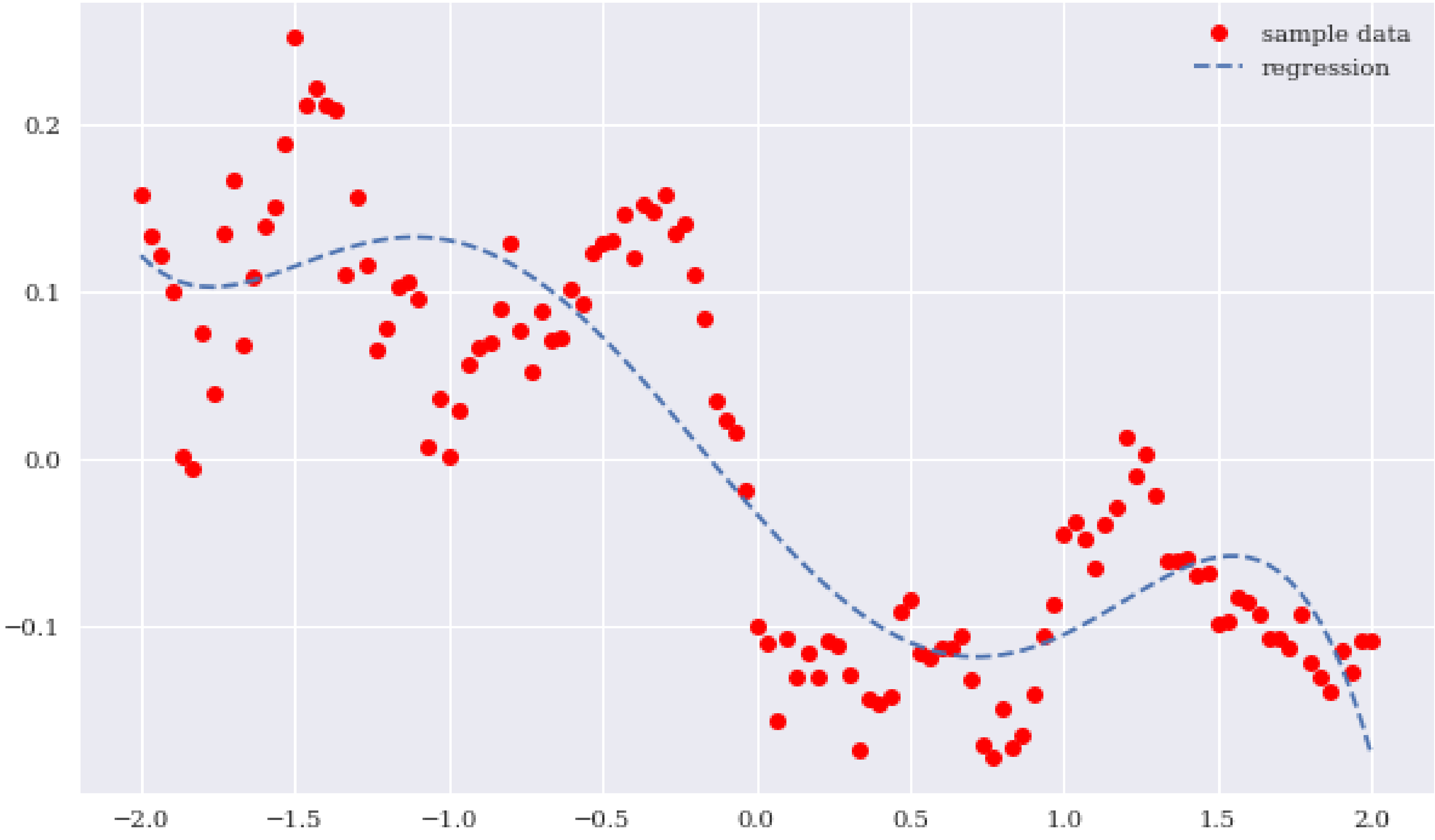
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Sample data set



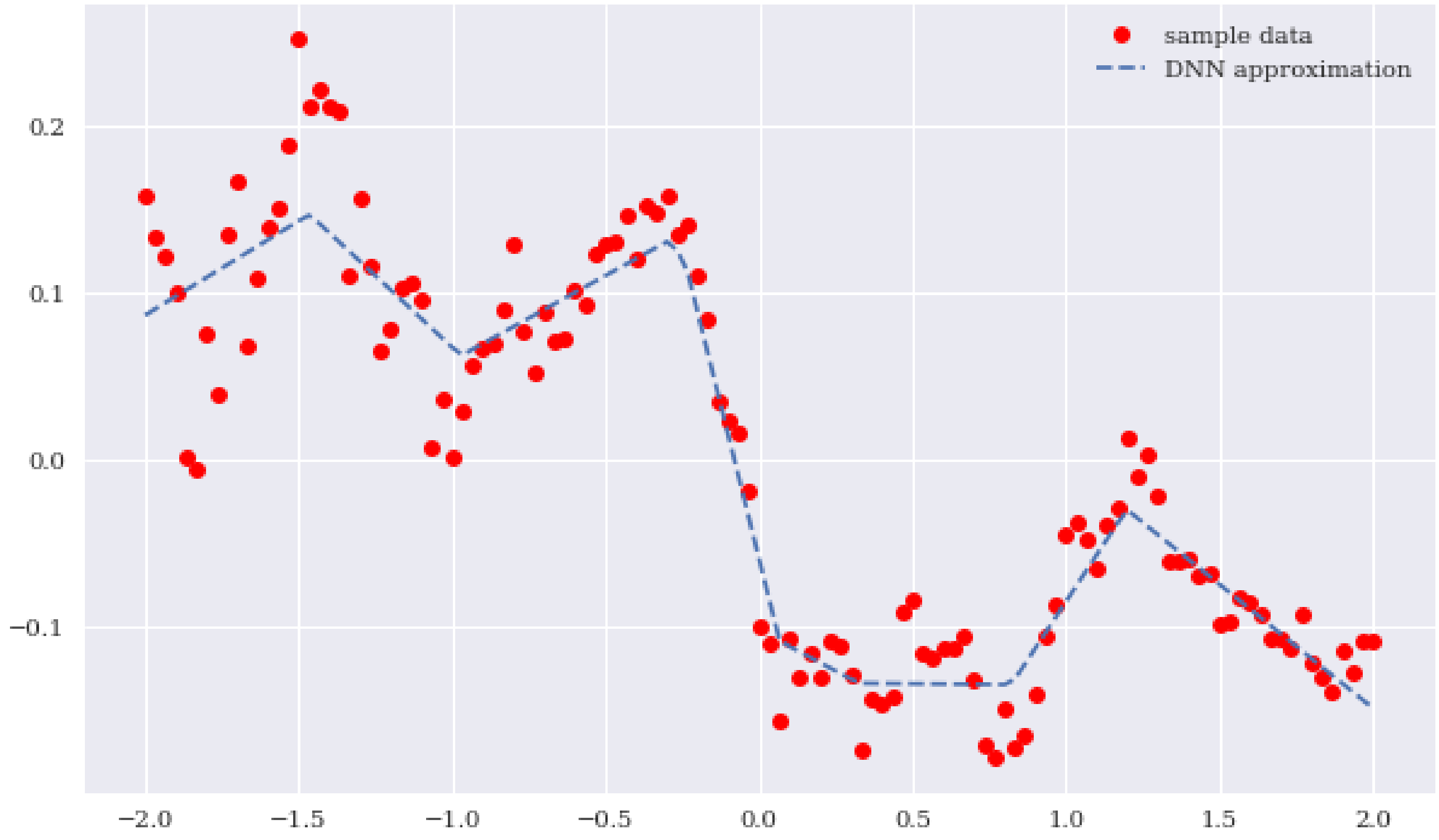
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O’Reilly Media.

Sample data and cubic regression line



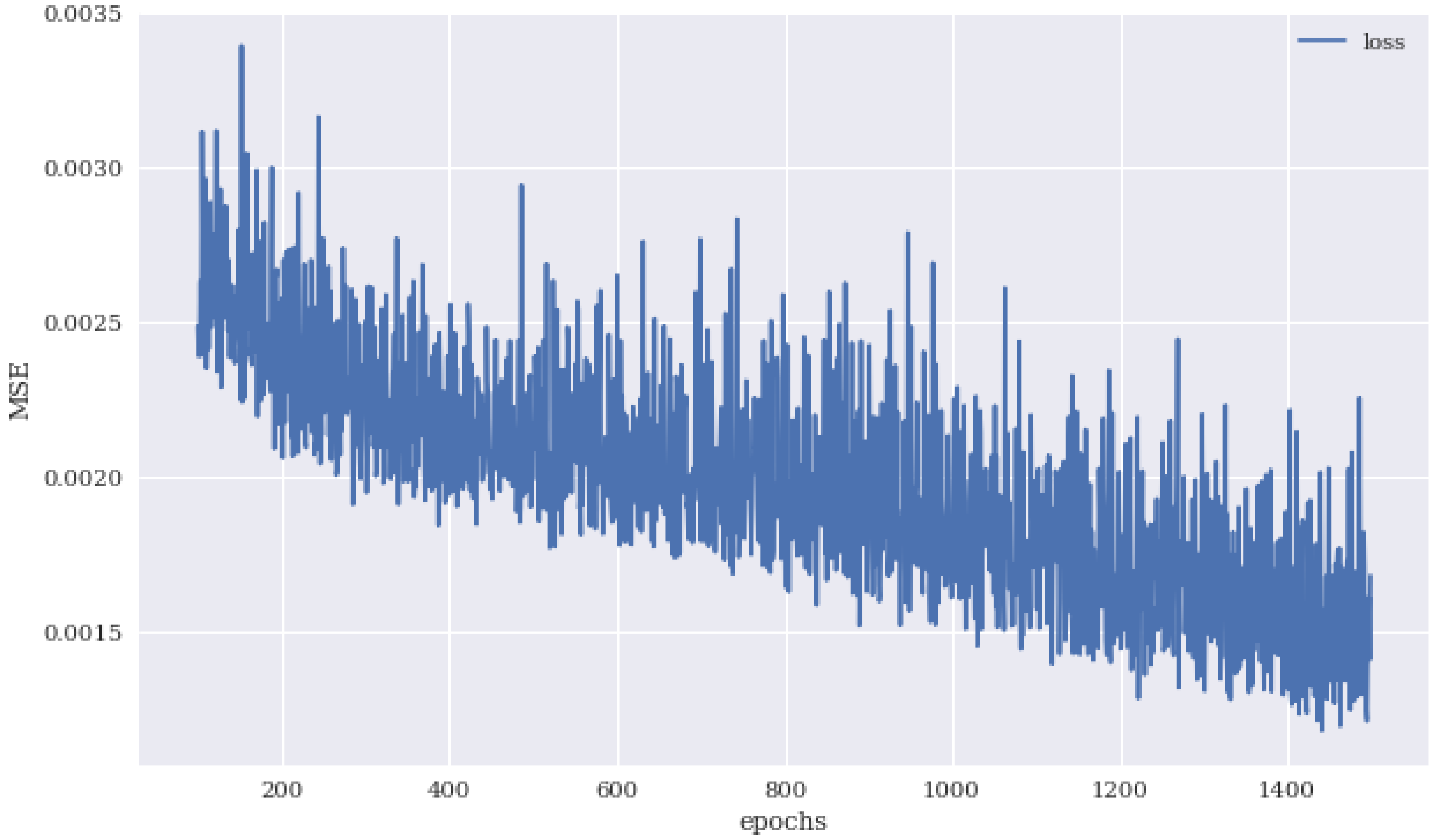
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Sample data and neural network approximation



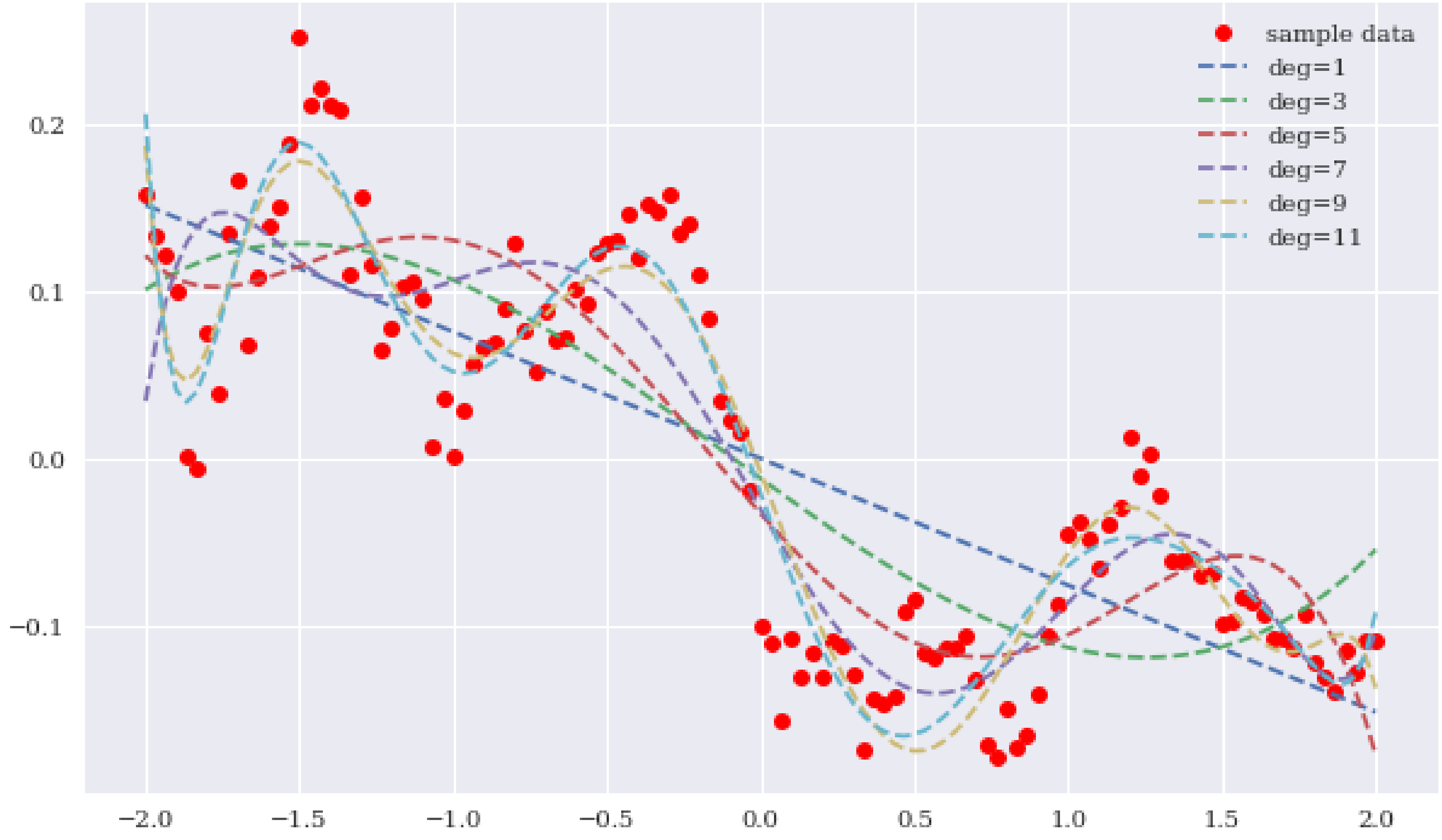
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

MSE values against number of training epochs



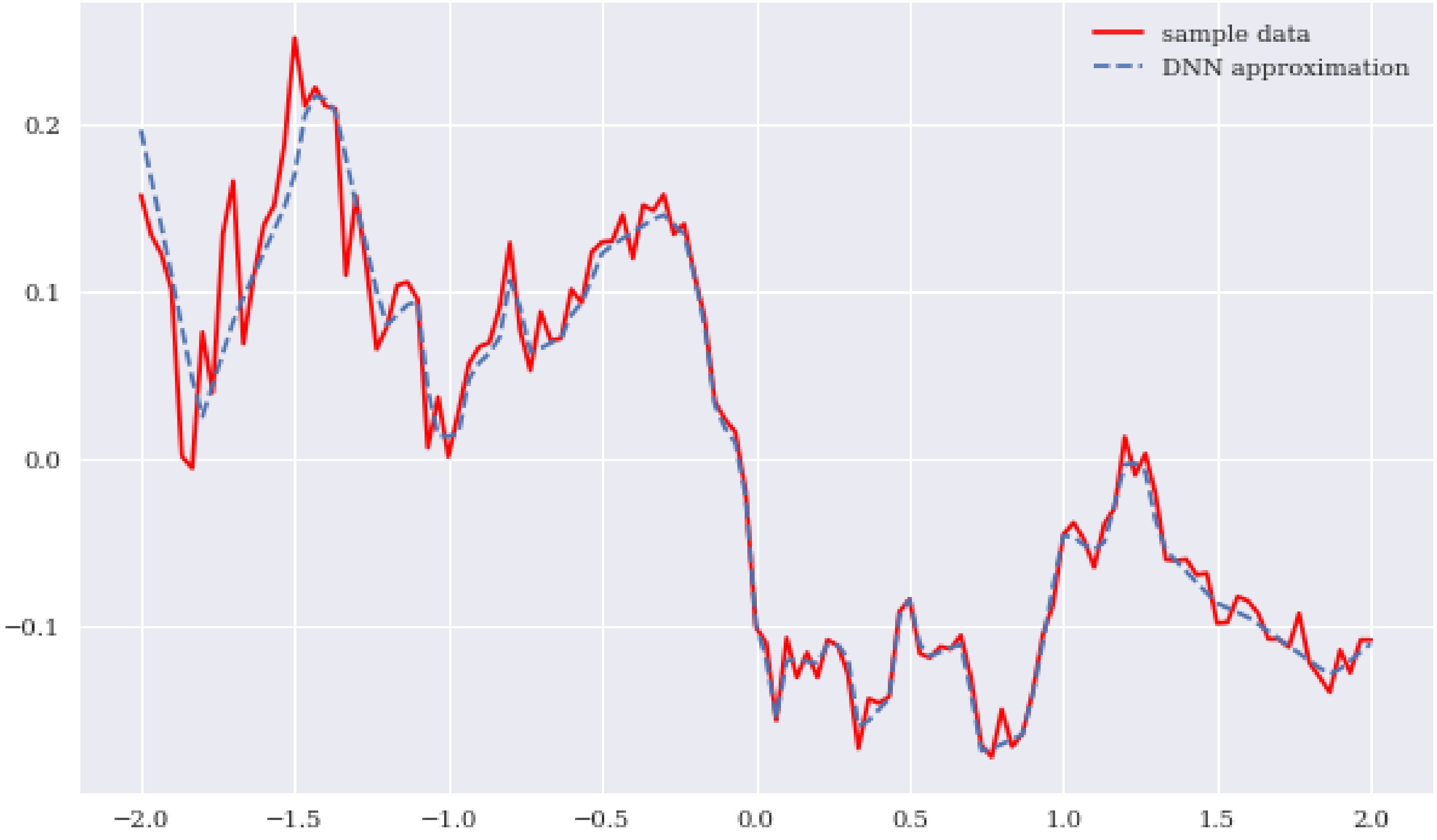
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Regression lines for different highest degrees

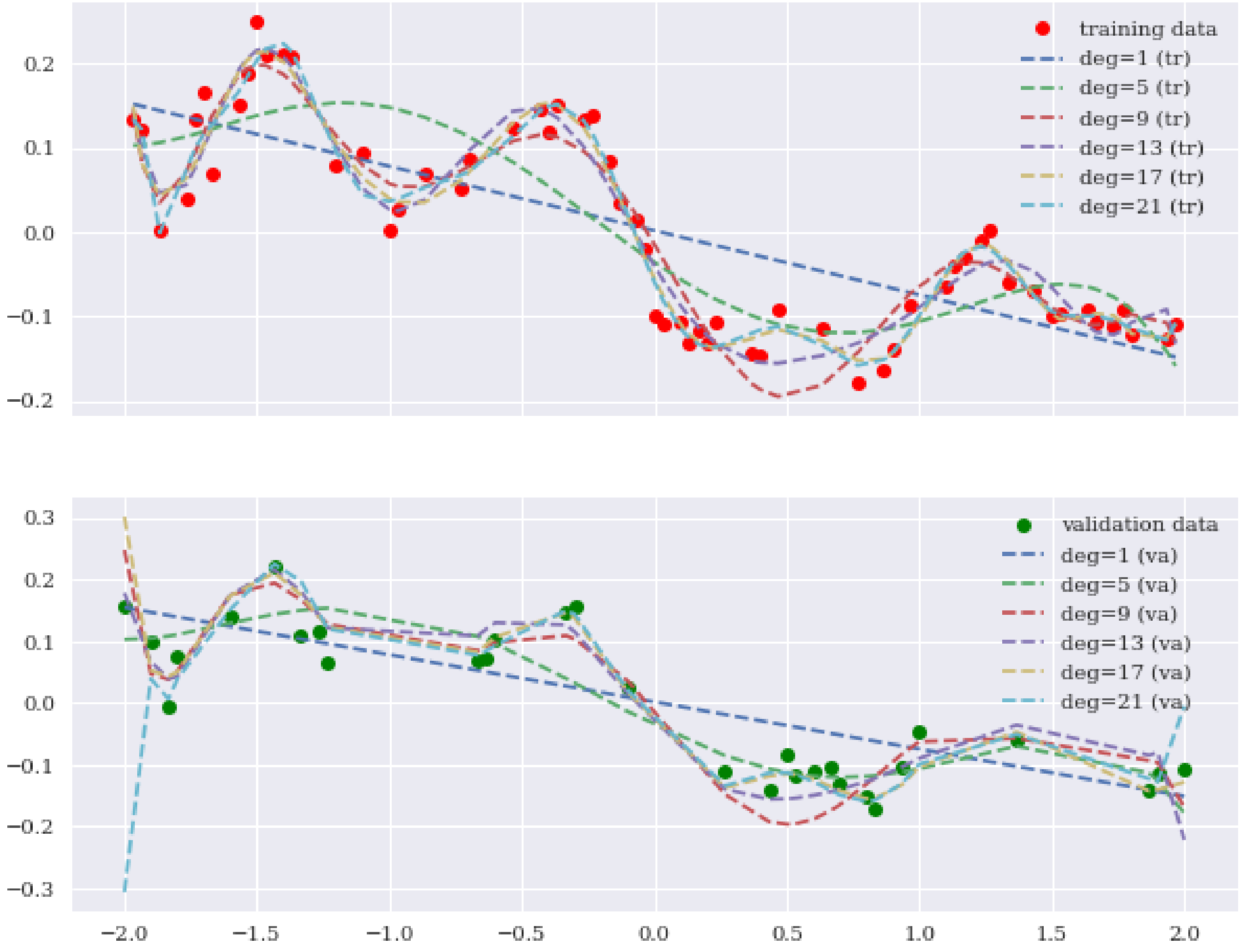


Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Sample data and DNN approximation (higher capacity)

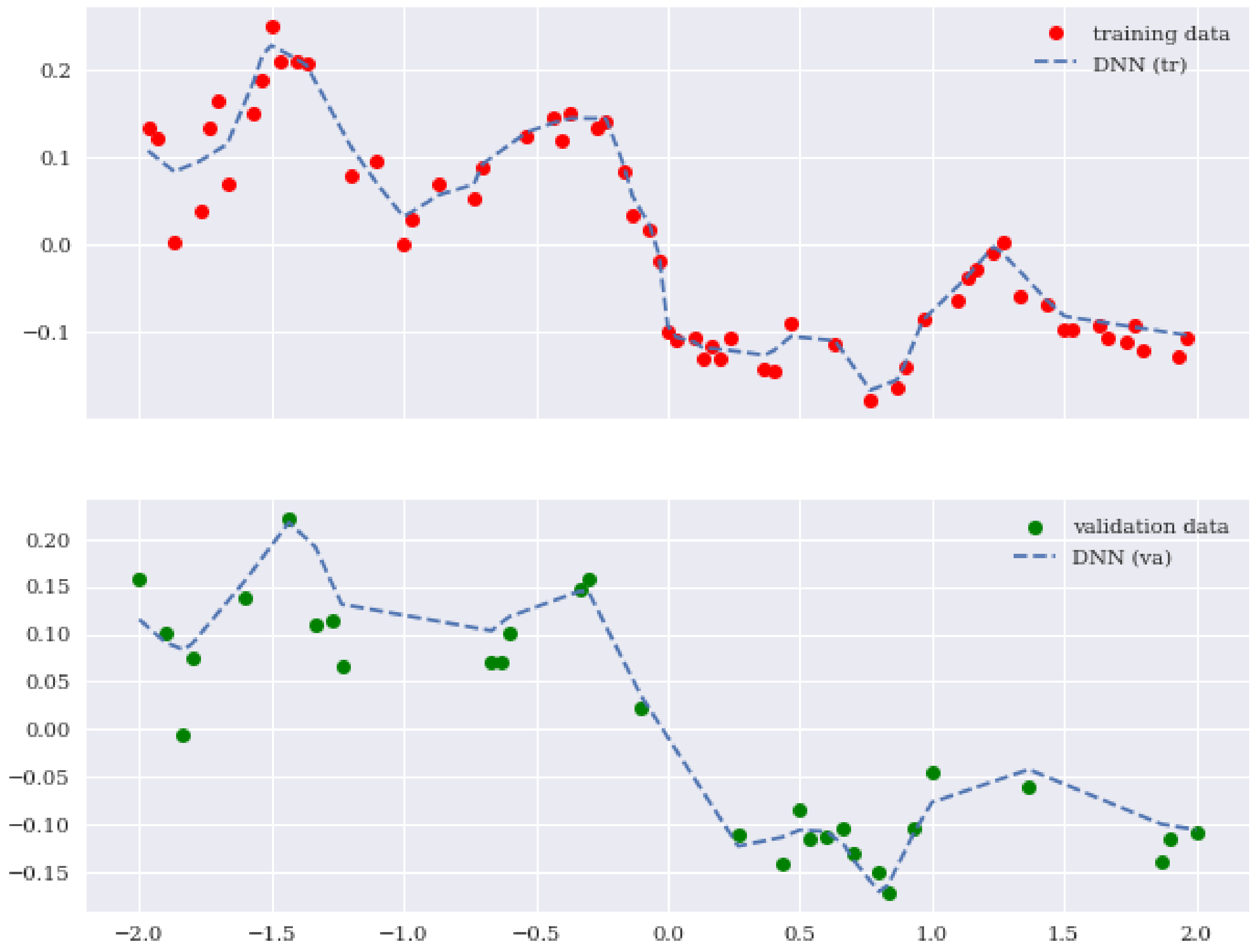


Training and validation data including regression fits



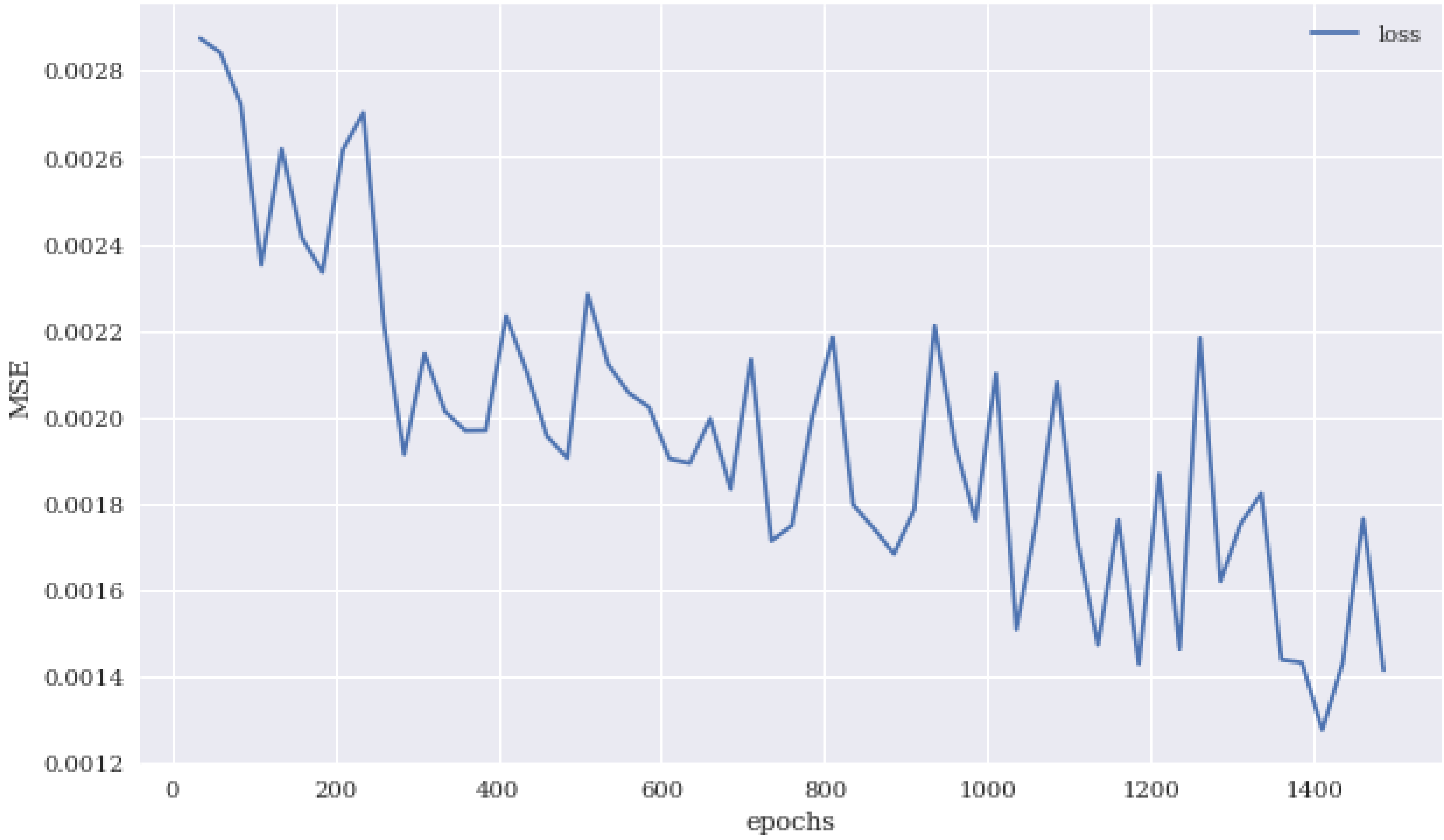
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Training and validation data including DNN predictions



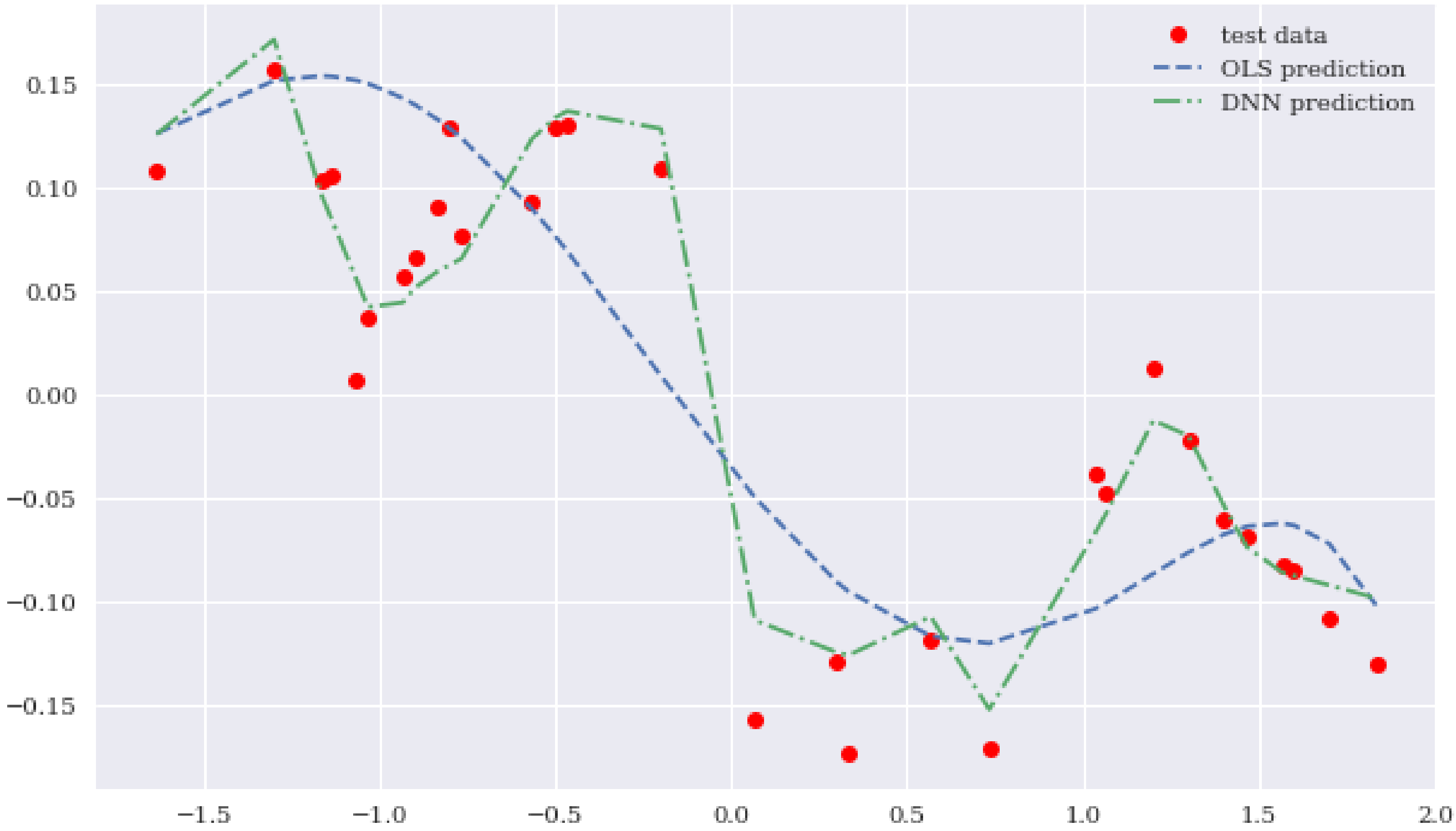
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

MSE values for DNN model on the training and validation data sets



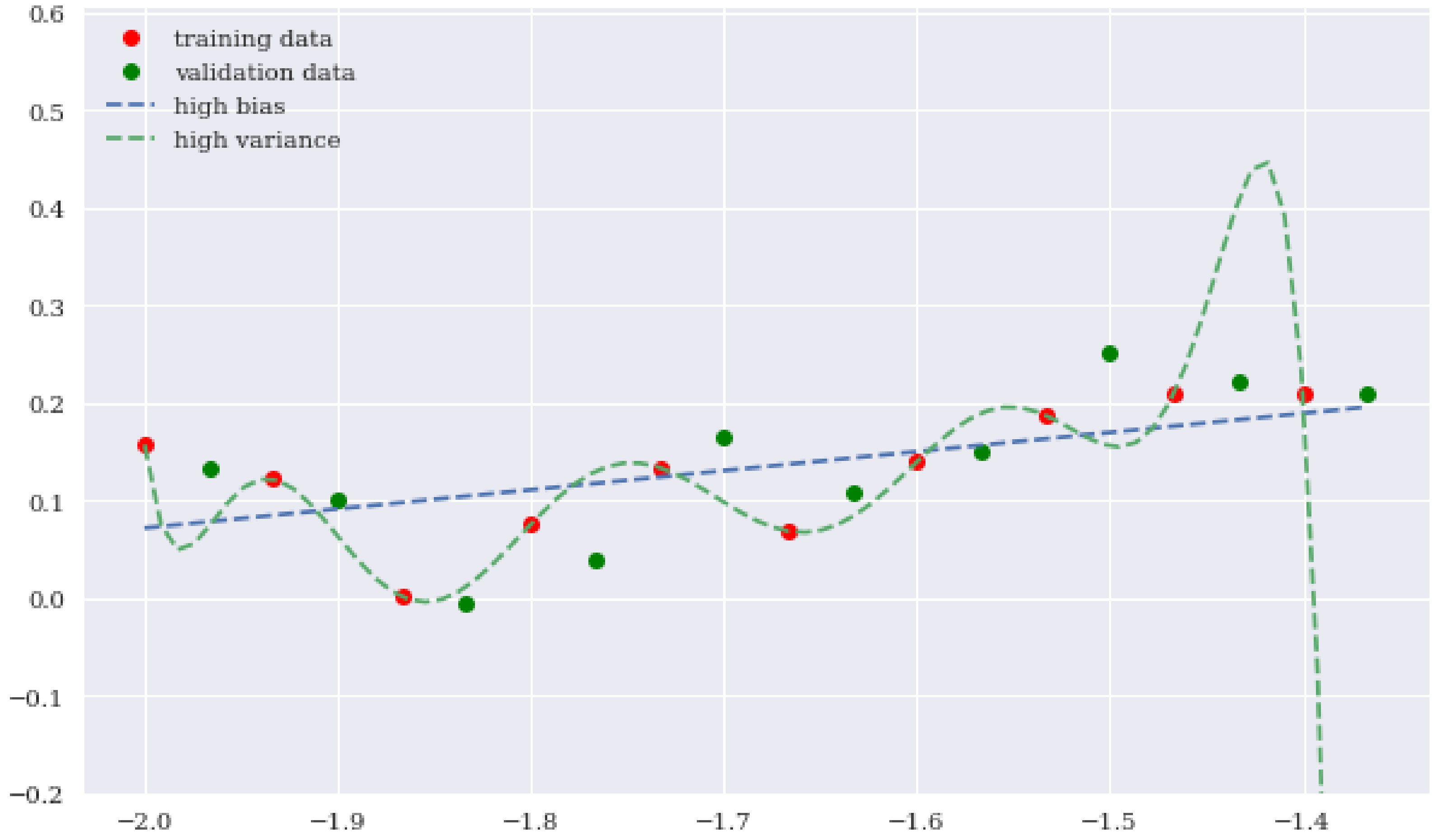
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Test data and predictions from OLS regression and the DNN model



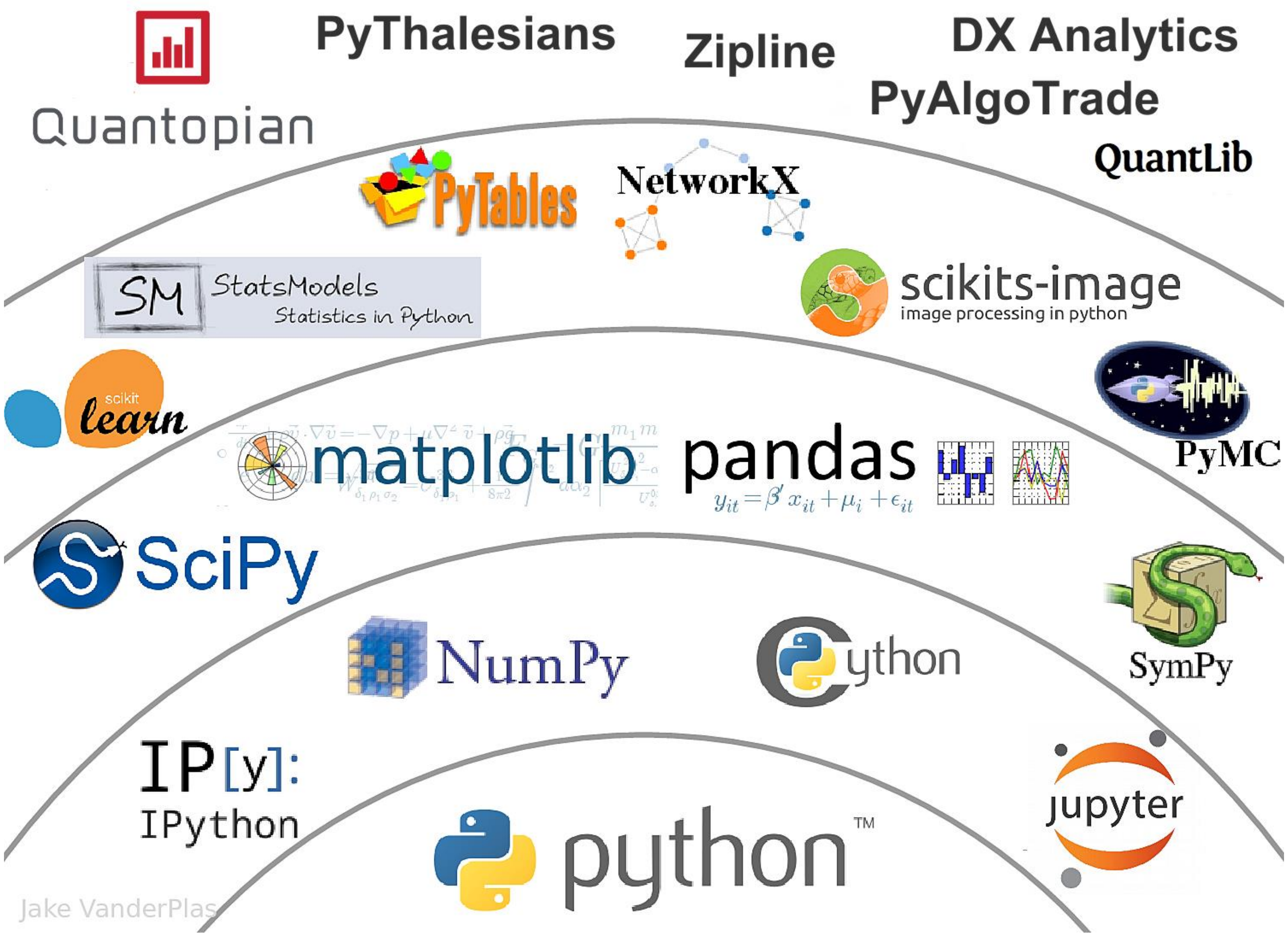
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

High bias and high variance OLS regression fits



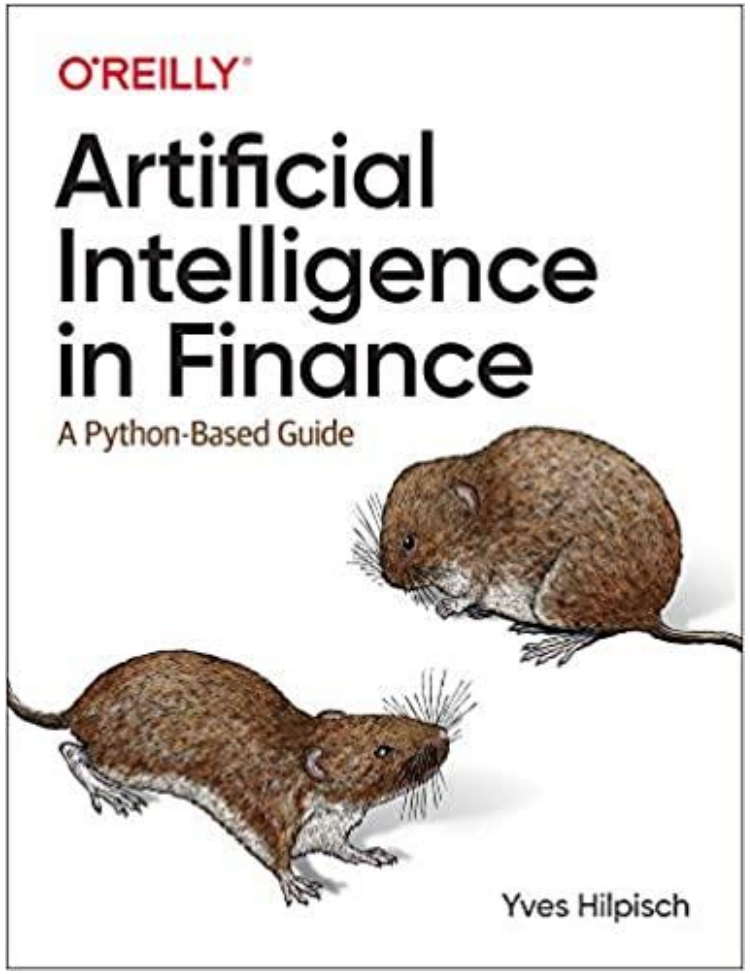
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

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), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly

yhilpisch / aiif Public

https://github.com/yhilpisch/aiif

Notifications | Star 98 | Fork 77

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights

main | 1 branch | 0 tags | Go to file | Code

yves Code updates for TF 2.3.	e334251 on Dec 8, 2020	4 commits
code	Code updates for TF 2.3.	11 months ago
.gitignore	Code updates for TF 2.3.	11 months ago
LICENSE.txt	Code updates.	11 months ago
README.md	Code updates.	11 months ago

README.md

Artificial Intelligence in Finance

About this Repository

This repository provides Python code and Jupyter Notebooks accompanying the **Artificial Intelligence in Finance** book published by O'Reilly.

O'REILLY

About

Jupyter Notebooks and code for the book Artificial Intelligence in Finance (O'Reilly) by Yves Hilpisch.

home.tpq.io/books/aiif

Readme

View license

Releases

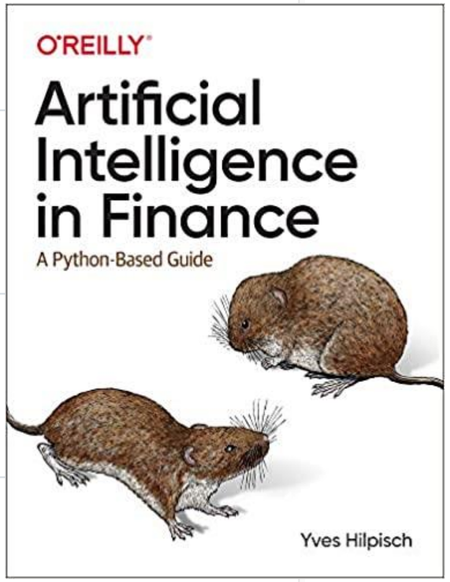
No releases published

Packages

No packages published

Languages

Jupyter Notebook 97.4% Python 2.6%



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

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

yhilpisch / aiif Public

Notifications | Star 98 | Fork 77

<> Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights

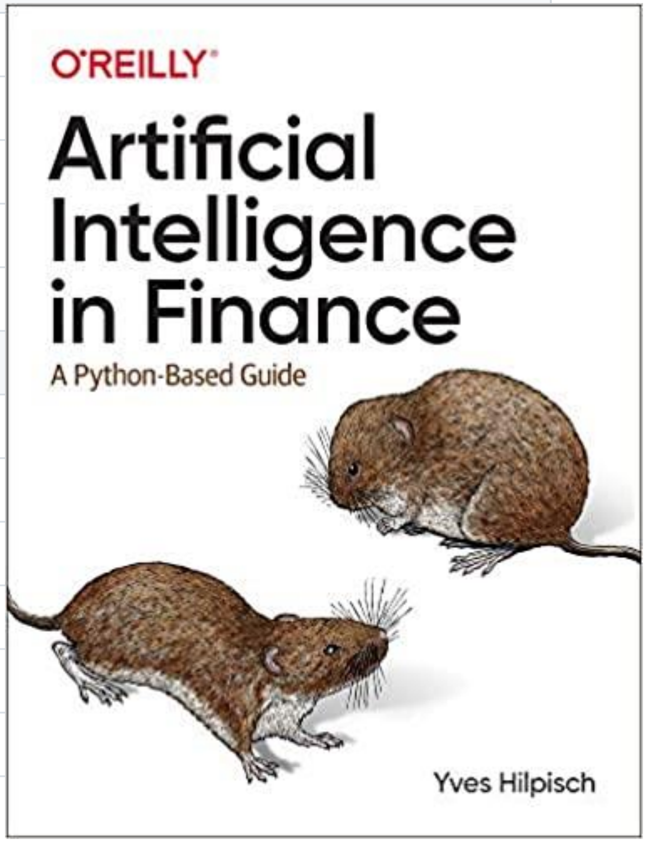
main | aiif / code /

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

Go to file

yves Code updates for TF 2.3. e334251 on Dec 8, 2020 History

..	
oanda	Code updates for TF 2.3.
01_artificial_intelligence.ipynb	Code updates for TF 2.3.
02_superintelligence.ipynb	Code updates for TF 2.3.
03_normative_finance.ipynb	Code updates for TF 2.3.
04_data_driven_finance_a.ipynb	Initial commit.
04_data_driven_finance_b.ipynb	Initial commit.
05_machine_learning.ipynb	Code updates for TF 2.3.
06_ai_first_finance.ipynb	Code updates for TF 2.3.
07_dense_networks.ipynb	Code updates for TF 2.3.
08_recurrent_networks.ipynb	Code updates for TF 2.3.
09_reinforcement_learning_a.ipynb	Code updates.
09_reinforcement_learning_b.ipynb	Code updates for TF 2.3.



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

python101.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

Table of contents

- AI in Finance
  - Normative Finance and Financial Theories
    - Uncertainty and Risk
    - Expected Utility Theory (EUT)
    - Mean-Variance Portfolio Theory (MVPT)
    - Capital Asset Pricing Model (CAPM)
    - Arbitrage Pricing Theory (APT)
- Deep Learning for Financial Time Series Forecasting
- Portfolio Optimization and Algorithmic Trading
  - Investment Portfolio Optimisation with Python
  - Efficient Frontier Portfolio Optimisation in Python
  - Investment Portfolio Optimization

AI in Finance

AI in Finance

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

Normative Finance and Financial Theories

Uncertainty and Risk

```python
import numpy as np

#The prices of the stock and bond today.
S0 = 10
B0 = 10
print('S0', S0)
print('S0', S0)

#The uncertain payoff of the stock and bond tomorrow.
S1 = np.array((20, 5))
B1 = np.array((11, 11))
print('S1', S1)
print('B1', B1)

#The market price vector
M0 = np.array((S0, B0))
```

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 RAM Disk Editing

Table of contents

Data Driven Finance

Data Driven Finance

Financial Econometrics and Regression

[18]

```python
import numpy as np

def f(x):
    return 2 + 1 / 2 * x

x = np.arange(-4, 5)
x
```

```
array([-4, -3, -2, -1,  0,  1,  2,  3,  4])
```

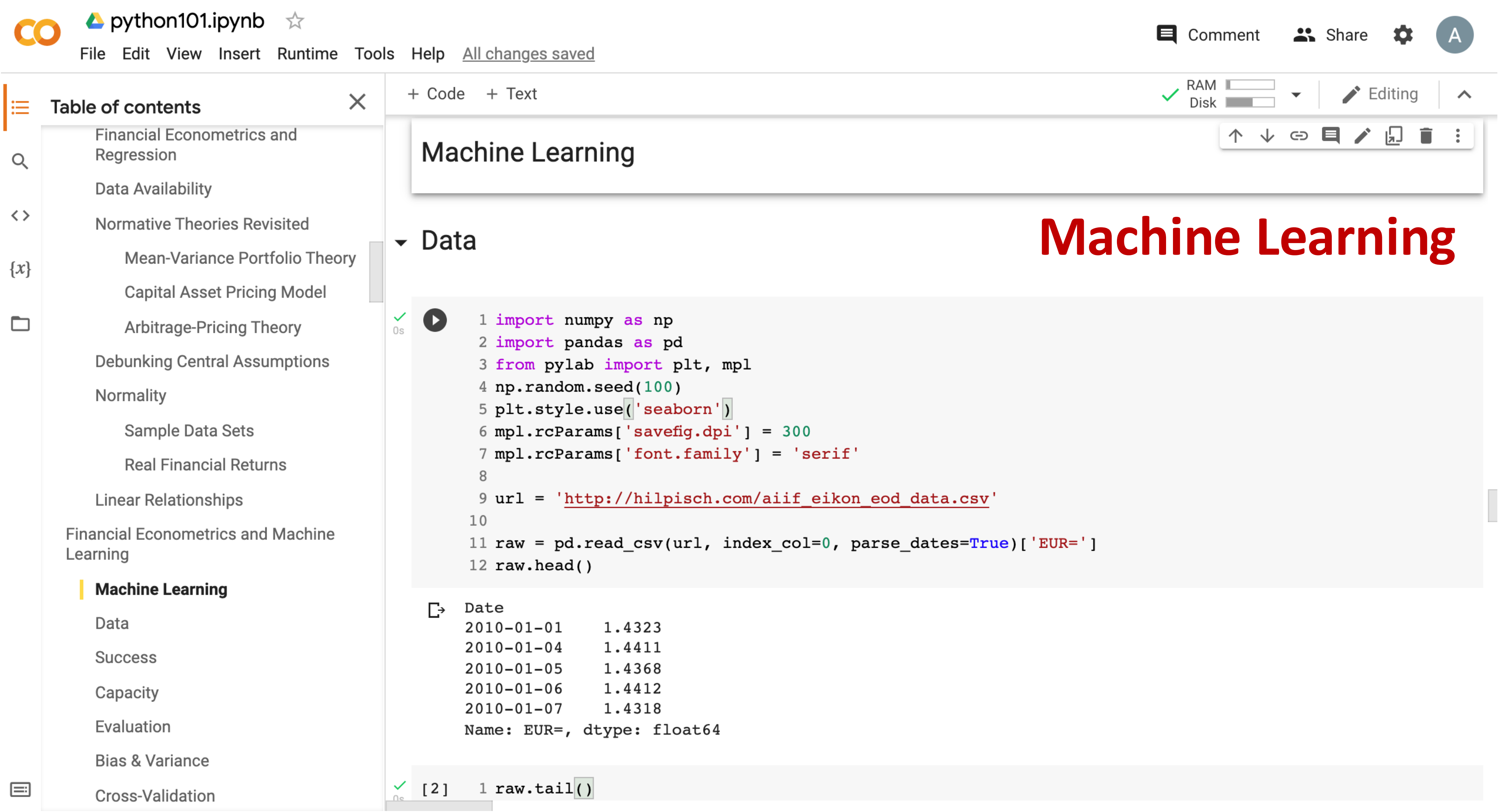
```python
y = f(x)
y
```

```
array([ 0.00,  0.50,  1.00,  1.50,  2.00,  2.50,  3.00,  3.50,  4.00])
```

```python
print('x', x)

print('y', y)

beta = np.cov(x, y, ddof=0)[0, 1] / x.var()
print('beta', beta)
```

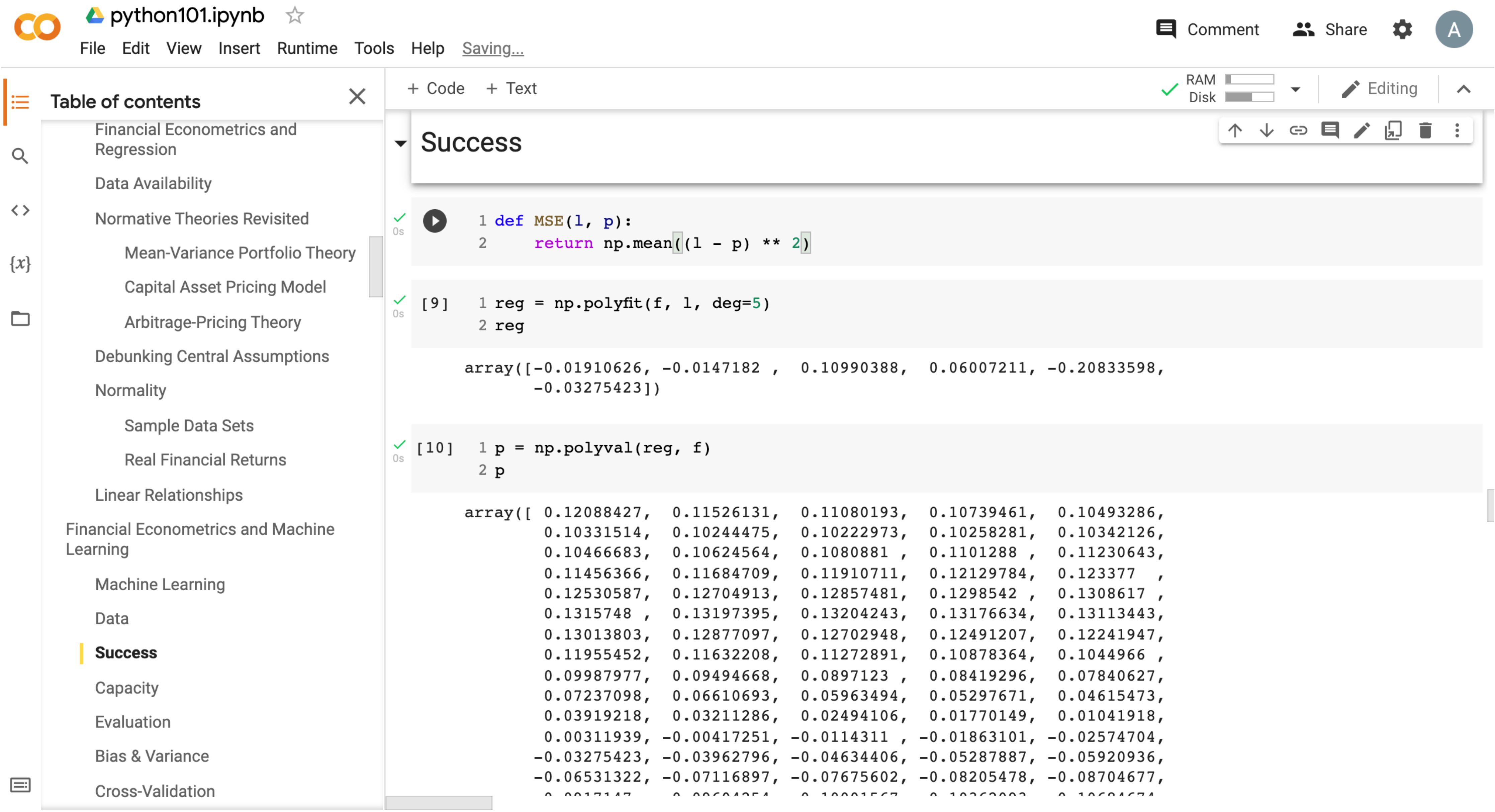
https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



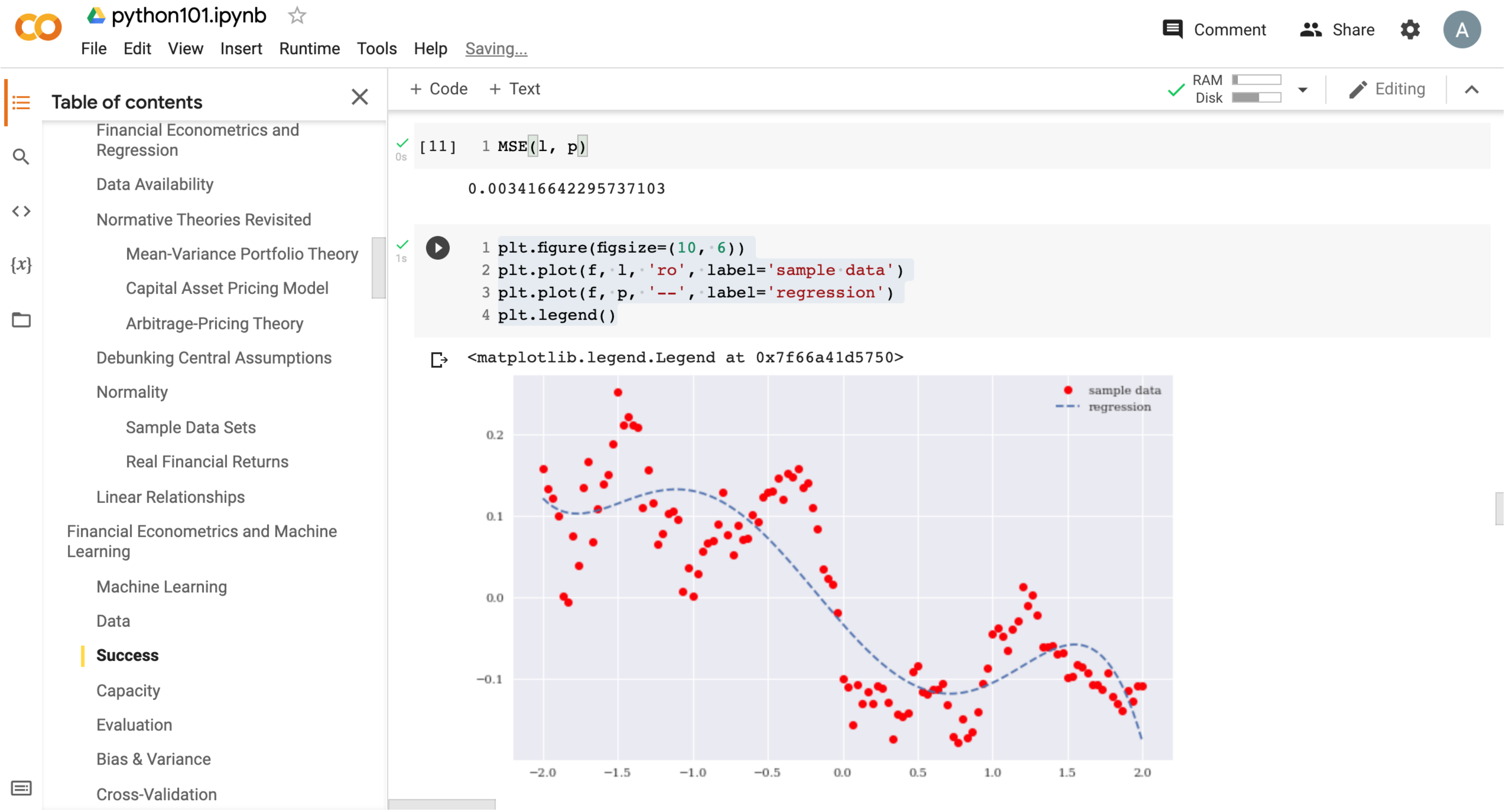
https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



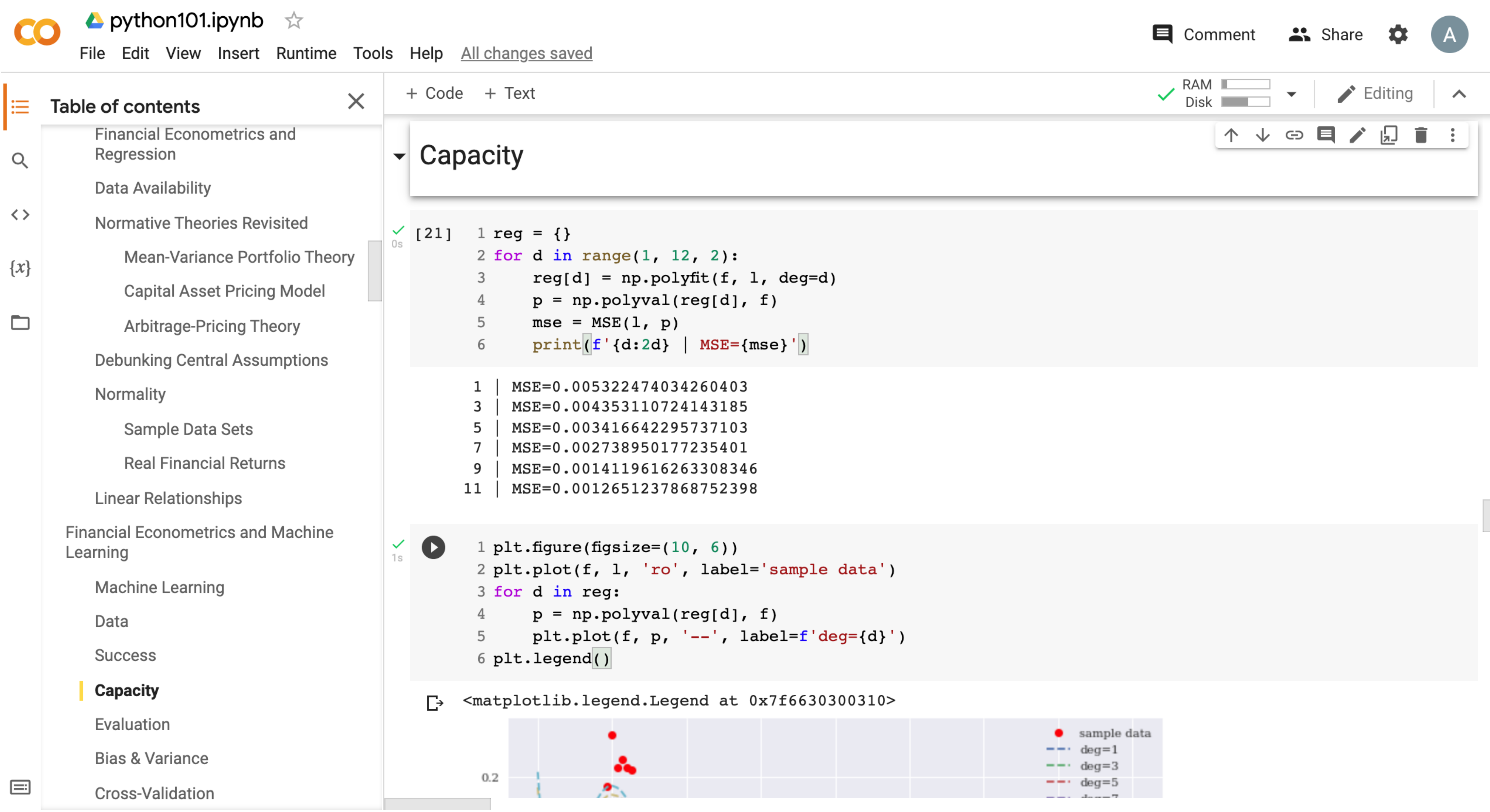
https://tinyurl.com/aintpupython101

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 Saving...

Comment Share

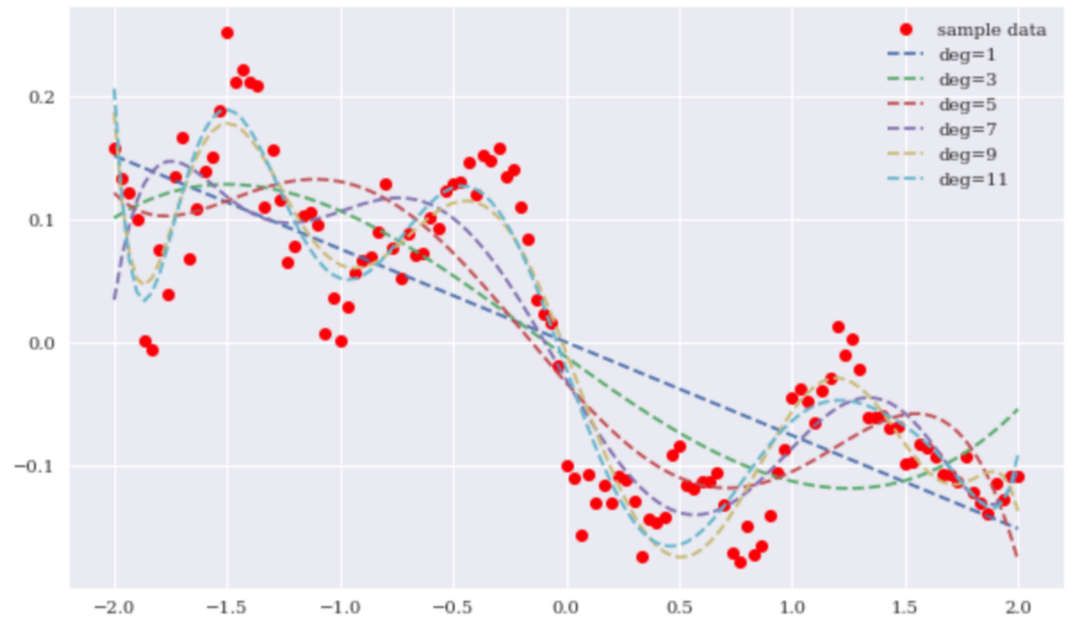
+ Code + Text RAM Disk Editing

Table of contents


```
 9 | MSE=0.0014119616263308346
11 | MSE=0.0012651237868752398
```

```python
plt.figure(figsize=(10, 6))
plt.plot(f, l, 'ro', label='sample data')
for d in reg:
    p = np.polyval(reg[d], f)
    plt.plot(f, p, '--', label=f'deg={d}')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7f6630300310>
```



https://tinyurl.com/aintpupython101

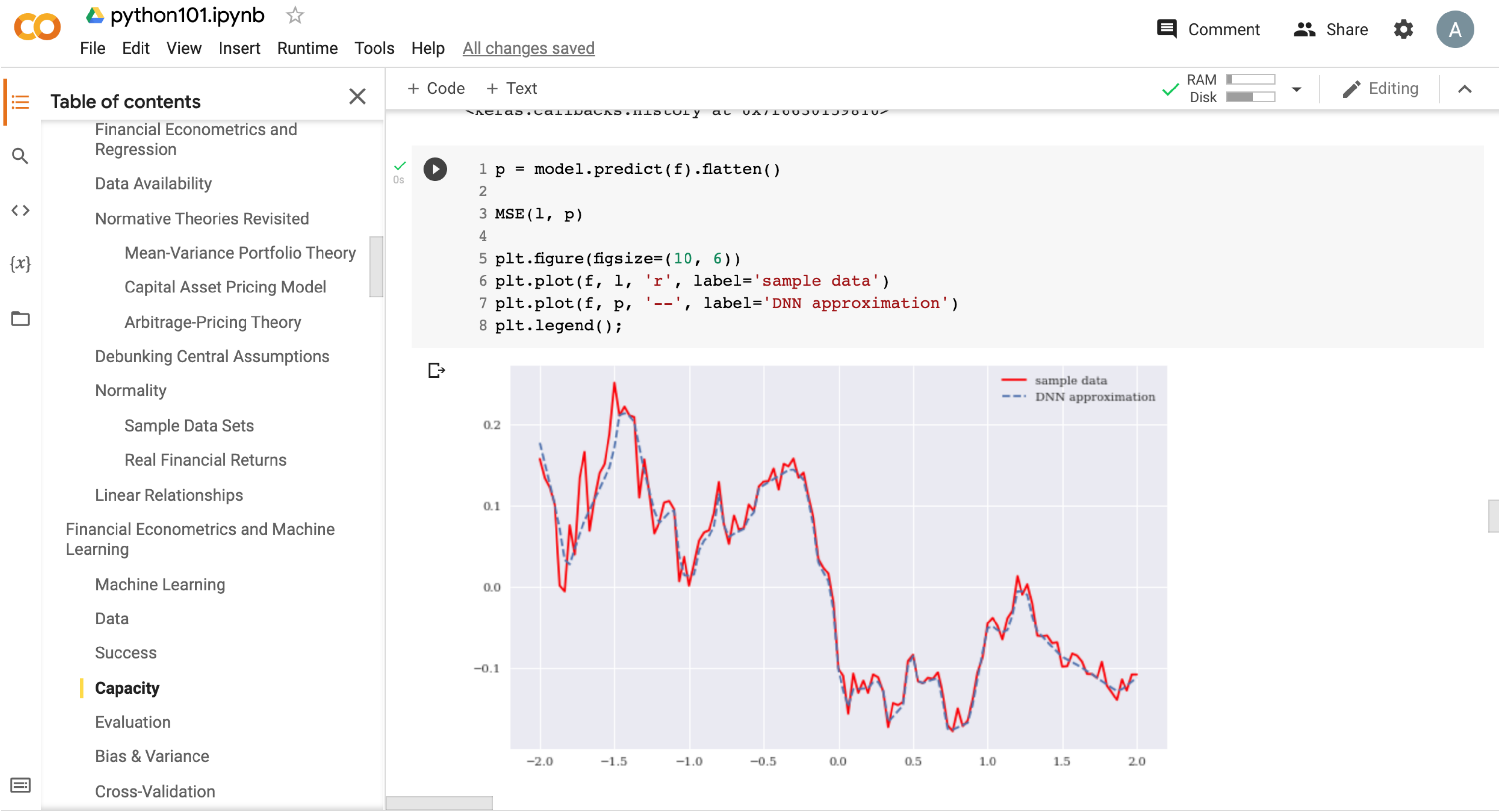
Python in Google Colab (Python101)

```
def create_dnn_model(hl=1, hu=256):
    ''' Function to create Keras DNN model.
    Parameters
    ==========
    hl: int
    number of hidden layers
    hu: int
    number of hidden units (per layer)
    '''
    model = Sequential()
    for _ in range(hl):
        model.add(Dense(hu, activation='relu', input_dim=1))
    model.add(Dense(1, activation='linear'))
    model.compile(loss='mse', optimizer='rmsprop')
    return model

model = create_dnn_model(3)

model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_2 (Dense)             (None, 256)               512

 dense_3 (Dense)             (None, 256)               65792

 dense_4 (Dense)             (None, 256)               65792

 dense_5 (Dense)             (None, 1)                 257

=================================================================
Total params: 132,353
Trainable params: 132,353
Non-trainable params: 0
_________________________________________________________________
```

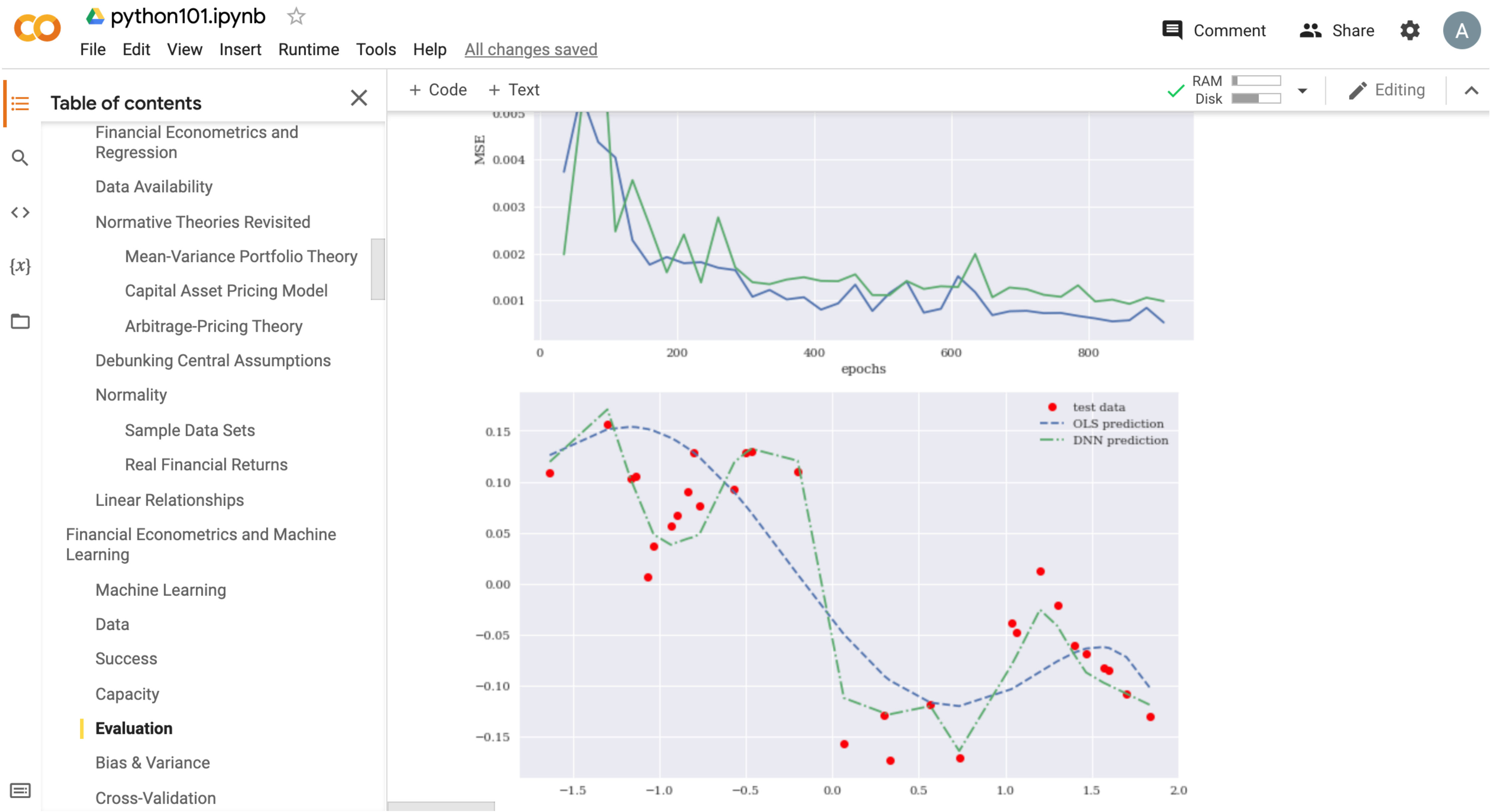
https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



https://tinyurl.com/aintpupython101

Summary

- Financial Econometrics
  - Financial Theories, OLS Regression
- Machine Learning
  - Learning, Evaluation, Bias and variance
  - Cross-validation

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 .
- Chris Brooks (2019), Introductory Econometrics for Finance, 4th Edition, Cambridge University Press
- Oliver Linton (2019), Financial Econometrics: Models and Methods, Cambridge University Press
- Tom Mitchell (1997), Machine Learning, McGraw-Hill.
- Max Tegmark (2017), Life 3.0: Being human in the age of artificial intelligence. Vintage.
- Ajay Agrawal, Joshua Gans, and Avi Goldfarb (2018). Prediction machines: the simple economics of artificial intelligence. Harvard Business Press.
- Eugene F. Fama (1995), "Random walks in stock market prices." Financial Analysts Journal 51, no. 1, 75-80.
- Ruey S. Tsay (2005), Analysis of financial time series, Wiley.
- Min-Yuh Day (2025), Python 101, https://tinyurl.com/aintpupython101