Exercises

Portfolio Optimization: Theory and Application
Chapter 16 – Deep Learning Portfolios

Daniel P. Palomar (2025). *Portfolio Optimization: Theory and Application.*
Cambridge University Press.

portfoliooptimizationbook.com

Machine Learning

Exercise 16.1: Classification of spam in emails

Build a black-box model that can accurately classify whether an email is spam or not.

- Use a dataset of labeled emails to train and evaluate your model; for example, the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/spambase) or the Enron-Spam dataset (http://nlp.cs.aueb.gr/software_and_datasets/Enron-Spam).
- Experiment with different black-box models, such as logistic regression, decision trees, random forests, and support vector machines, and compare their performance.

Exercise 16.2: Regression of housing prices

Build a black-box model that can predict the price of a house based on its features such as square footage, number of bedrooms, and location.

- Use a dataset of labeled houses to train and evaluate your model; for example, the Boston Housing Dataset at Kaggle (www.kaggle.com/code/prasadperera/the-boston-housing-dataset) or the California Housing Prices at Kaggle (www.kaggle.com/datasets/camnugent/california-housing-prices).
- Experiment with different black-box models, such as linear regression, decision trees, random forests, and support vector machines, and compare their performance.

Deep Learning

Exercise 16.3: Image classification with CNNs

Use a CNN to classify images from the CIFAR-10 dataset (www.cs.toronto.edu/~kriz/cifar.html), which consists of 60 000 32×32 color images in 10 classes. Experiment with different architectures, such as varying the number of convolutional layers, pooling layers, and fully connected layers, to see which one performs best.

Exercise 16.4: Object detection with Faster R-CNN

Use a Faster R-CNN model (a deep learning model for object detection, developed by Microsoft Research in 2015) to detect objects in images from the COCO dataset (https://cocodataset.org), which consists of over 200 000 labeled images with 80 different object categories. Experiment with different backbone architectures, such as ResNet and VGG, and adjust the hyper-parameters to improve the detection accuracy.

Exercise 16.5: Language translation with sequence-to-sequence models

Use a sequence-to-sequence model with attention to translate text from one language to another. Use a dataset such as the Multi30k dataset (https://github.com/multi30k/dataset), which consists of about 30 000 parallel sentences in English, French, and German. Experiment with different encoder and decoder architectures, such as LSTM and transformer, and adjust the hyper-parameters to improve the translation accuracy.

Exercise 16.6: GANs for image synthesis

Use a GAN to generate realistic images that resemble a given dataset.

- Use an image dataset, such as the CIFAR-10 dataset (www.cs.toronto.edu/~kriz/cifar.html), which consists of 60 000 32×32 color images in 10 classes, or the MNIST dataset (https://github.com/mbornet-hl/MNIST/tree/master/IMAGES/GROUPS), which consists of a large database of handwritten digits with 60 000 training images and 10 000 testing images.
- Experiment with different GAN architectures, such as DCGAN and WGAN, and adjust the hyper-parameters to improve the image quality.

Exercise 16.7: Handwritten Digit Recognition with CNNs

Use a CNN to classify handwritten digits from the MNIST dataset (https://github.com/mbornet-hl/MNIST/tree/master/IMAGES/GROUPS), which consists of 60,000 training images and 10,000 test images of handwritten digits from 0 to 9. Experiment with different architectures, such as varying the number of convolutional layers, pooling layers, and fully connected layers, to see which one performs best.

Machine Learning for Finance

Exercise 16.8: Linear regression for stock prices

Implement a linear regression model to predict the stock price of a company based on its historical data. You can use data from financial databases such as Yahoo Finance. Evaluate the performance of the model using metrics such as mean squared error.

Exercise 16.9: Decision trees for default prediction

Build a decision tree model to classify whether a loan applicant is likely to default or not based on features such as income, credit score, and loan amount.

- Use some publicly available dataset for classifying loan defaults; for example, the German Credit Dataset or the UCI Credit Approval Dataset, both available at the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data) and https://archive.ics.uci.edu/ml/datasets/credit+approval, respectively).
- Evaluate the performance of the model using metrics such as accuracy, precision, and recall.

Exercise 16.10: Random forests for default prediction

Extend the previous decision tree model to a random forest model and compare the performance of both models. Use cross-validation to tune the hyper-parameters of the random forest model and evaluate it using the same metrics as the decision tree model.

Exercise 16.11: SVMs for stock direction forecast

Implement an SVM model to predict whether a stock will go up or down based on technical indicators such as moving averages and relative strength index (RSI). Use grid search to find the best hyper-parameters of the model and evaluate it using metrics such as accuracy, precision, and recall.

Deep Learning for Finance

Exercise 16.12: Comparison of LSTMs vs. transformers with synthetic data

Generate synthetic data to compare the long-term memory of LSTMs and transformers.

a. Create the inputs as 20-dimensional vectors (containing 20 samples over time) from two possible predefined sequences and the outputs as two possible labels corresponding to the two possible sequences.
b. Add random noise to the inputs for different values of noise variance, leading to different values of signal-to-noise ratio.
c. Train an LSTM network and a transformer network for different signal-to-noise ratios and compare them.

d. Then, repeat the experiment, but now using as inputs 100-dimensional vectors containing the previous 20-dimensional predefined sequences at the end of the vectors (more recent temporal observations) and zeros elsewhere. The networks should learn that only the more recent 20 samples contain the useful pattern, whereas the first 80 samples contain just noise.
e. Repeat the experiment with 100-dimensional input vectors, but now placing the 20-dimensional predefined sequences at the beginning of the vectors (earlier temporal observations). Again, the networks should learn the correct location of the useful 20 samples, but now they happen earlier in time. The transformer architecture should not be affected, whereas the LSTM may tend to "forget" the useful patterns as they happened earlier.
f. Finally, repeat the experiment with bigger dimensions until the difference between LSTMs and transformers becomes clear.

Exercise 16.13: Predicting stock prices using deep learning

Develop a deep neural network to predict the future prices of a stock based on historical time series data. In particular, consider the following architectures and compare their performance: LSTM, CNN, and transformer.

Exercise 16.14: Portfolio optimization using deep learning

Use deep learning to optimize a portfolio of investments. This could involve using historical time series data to develop a model that maximizes returns while minimizing risk. You may want to experiment with different loss functions, such as MSE or MAPE, and explore different optimization algorithms, such as SGD or Adam, to train your model. Additionally, you may want to explore the use of techniques such as attention mechanisms or reinforcement learning to further improve the performance of your model.