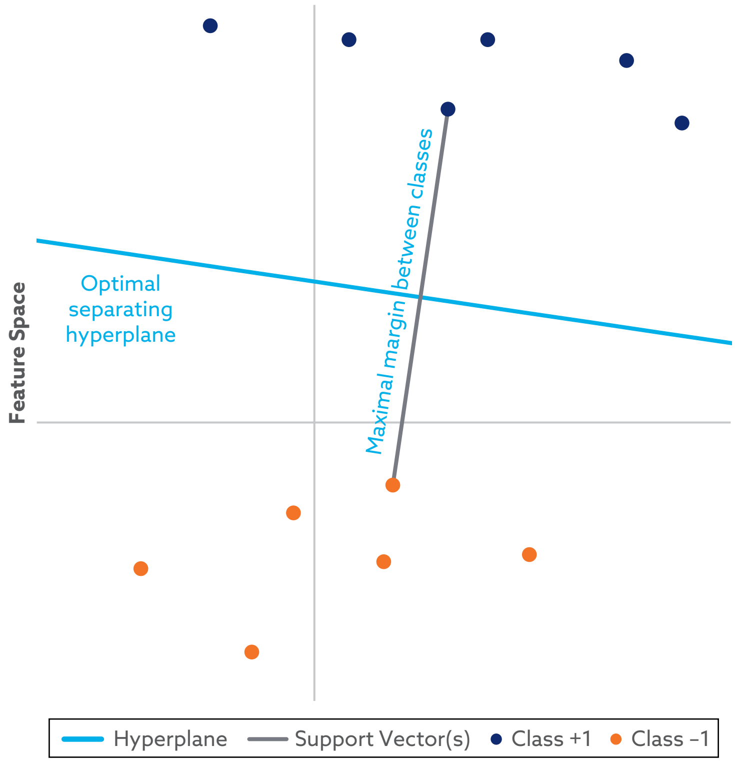
SUPPORT VECTOR MACHINES

Maxim Golts, PhD
VP, Multi-Asset Solutions, PGIM
Senior Advisor, Questrom School of Business, Boston University

Introduction

Support vector machines (SVMs) are a powerful machine learning tool developed by Vladimir Vapnik and Corinna Cortes in the 1990s as an alternative to neural networks (see Cortes and Vapnik 1995; Vapnik 1999, 2000). They were developed to address classification problems. The basic idea is to separate the training data points corresponding to two different classes, often labeled +1 and −1, by an *affine hyperplane* in the feature space (see **Exhibit 1**). According to Vapnik (1999, p. 996), "We say that this set of vectors is separated by the *optimal hyperplane (or the maximal margin hyperplane)* if it is separated without error and the distance between the closest vector and the hyperplane is maximal."

Exhibit 1. SVM Setup: Data in the Feature Space, Support Vectors, and the Optimal Separating Hyperplane



Note: This is a generic SVM illustration with random data generated by the author.

Technical Concepts

Let us start by considering a classification problem with training data of the following form:

$$(x_1,y_1),\ldots,(x_l,y_l),\ x\in R^n,\ y\in\{+1,-1\}.$$

In other words, vectors $x \in R^n$, often called *inputs* or *features*, in the training data belong to two classes: those paired with $y = +1$ and those paired with $y = -1$. The goal of any learning machine is to find a functional form for this classification, fitting the training data as well as possible but not overfitting.

One simple case is when the two classes of $x \in R^n$ can be separated by a *hyperplane:* an affine subspace of R^n of codimension 1. Then the class of $x \in R^n$ corresponding to $y = +1$ will simply be on one side of the hyperplane, and the class corresponding to $y = -1$ will be on the other side.

In practice, such a separating hyperplane does not exist for many real-life problems. Following Cortes and Vapnik (1995) and Vapnik (1999, 2000), there are two major approaches to constructing learning machines for the classification problems:

- Use functional approximations by nonlinear (e.g., *sigmoid*) functions. This approach leads to neural networks, which are beyond the scope of this chapter.
- Map (nonlinearly) the input vectors $x \in R^n$ into a *higher-dimensional feature space*, and construct a separating hyperplane in this higher-dimensional space. This approach leads to SVMs, which we will consider here in more detail.

An SVM linear model developed in such a higher-dimensional feature space corresponds to a nonlinear class separation model in the original feature space. Support vector machines (and neural networks) can be used in machine learning beyond classification problems, such as regression and density estimation problems (see, e.g., Vapnik, Golowich, and Smola 1997; Smola and Schölkopf 2004; Awad and Khanna 2015).

Figure 1. Hyperplane-Separable Data

Support vector machines were introduced by Vladimir Vapnik and Corinna Cortes (see Cortes and Vapnik 1995; Vapnik 1999, 2000). Suppose the training data are of the form

$$(x_1,y_1),\ldots,(x_l,y_l),\ x\in R^n,\ y\in\{+1,-1\}$$

and the two classes corresponding to $y = +1$ and $y = -1$ *can be separated* by an (affine) hyperplane, called an "optimal separating hyperplane" or a "maximal margin hyperplane":

$$(w\cdot x)-b=0,$$

where w is the vector orthogonal to the optimal hyperplane and b is a scalar. Vapnik (1999) describes the optimal hyperplane using the following inequalities:

$$(w \cdot x_i) - b \geq 1 \text{ if } y_i = 1.$$

$$(w \cdot x_i) - b \leq -1 \text{ if } y_i = -1.$$

These inequalities can be expressed compactly as follows:

$$y_i \left[(w \cdot x_i) - b \right] \geq 1,\ i = 1, \ldots, l.$$

The maximal margin hyperplane is obtained by solving the following optimization problem:

$$\min_{w,b} (w, w)$$

$$\text{subject to } y_i \left[(w \cdot x_i) - b \right] \geq 1,\ i = 1, \ldots, l.$$

Following Vapnik (1999), this optimization problem can be set up with Lagrange multipliers α_i:

$$\min_{\alpha} \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$

$$\text{subject to} \sum_{i=1}^{l} \alpha_i y_i = 0 \text{ and } 0 \leq \alpha_i,\ i = 1, \ldots, l.$$

The optimal solution vector w_0 is a linear combination of the vectors in the training set:

$$w_0 = \sum_{i=1}^{l} y_i \alpha_i^0 x_i,\ \alpha_i^0 \geq 0,\ i = 1, \ldots, l.$$

Moreover, Vapnik shows that only some training vectors x_i, called the *support vectors*, have nonzero coefficients in the expansion of w_0. In other words, we have

$$w_0 = \sum_{\text{support vectors}} y_i \alpha_i^0 x_i,\ \alpha_i^0 \geq 0.$$

Figure 2. Generalization for Linearly Nonseparable Data

Dealing with the case when the data are linearly nonseparable, following Vapnik (1999), we set

$$\min_{w,b}(w,w)+C\sum_{i=1}^{l}\xi_i$$

$$\text{subject to } y_i\left[(w\cdot x_i)-b\right]\geq 1-\xi_i,\ i=1,\ldots,l.$$

The idea is to allow some points to be on the "wrong" side of the optimally separating hyperplane. The optimization problem can be formulated as follows:

$$\min_{\alpha}\sum_{i=1}^{l}\alpha_i-\frac{1}{2}\sum_{i,j}^{l}\alpha_i\alpha_j y_i y_j (x_i\cdot x_j)$$

$$\text{subject to } \sum_{i=1}^{l}\alpha_i y_i=0 \text{ and } 0\leq\alpha_i\leq C,\ i=1,\ldots,l.$$

More generally, we can map the input vectors $x\in R^n$ into a very higher-dimensional feature space through some nonlinear mapping chosen a priori and construct an optimal separating hyperplane in this high-dimensional feature space. It turns out we do not need to know the explicit form of this nonlinear map; all we need is the inner product in the higher-dimensional space, represented by a two-variable function $K(x_i, x_j)$ for $x_i, x_j\in R^n$, called the *kernel*.

Then, the most general optimization problem can be formulated as follows:

$$\min_{\alpha}\sum_{i=1}^{l}\alpha_i-\frac{1}{2}\sum_{i,j}^{l}\alpha_i\alpha_j y_i y_j K(x_i\cdot x_j)$$

$$\text{subject to } \sum_{i=1}^{l}\alpha_i y_i=0 \text{ and } 0\leq\alpha_i\leq C,\ i=1,\ldots,l.$$

One of the most important concepts in learning theory is the *trade-off* between the quality of the model fit on the training data and the complexity of the model. In fact, support vector machines were inspired by the so-called structural risk minimization principle, arising in statistical learning theory to address this kind of trade-off.

Intuitively, one can achieve a great model fit in the training data, but the confidence interval will be large, resulting in *overfitting*—fitting the spurious patterns in the training data too well, at the expense of poor performance on the data the model has not seen in training. Vapnik (1999) points out that SVMs allow the control of both model fit and the confidence interval. In the most general case, the unique optimization solution is obtained when one chooses the value of the *trade-off* parameter, C.

Simple Example: Classifying Stocks vs. Bonds

In this section, I illustrate the support vector machine concepts via a simple example. For a set of equity and bond indexes, we want to separate the set in a two-dimensional feature space: the correlation between an index return and inflation (Consumer Price Index, or CPI, change) and the correlation between the index return and real GDP growth. We calculate quarterly correlations using year-over-year inflation, year-over-year GDP growth rates, and (the prior quarter's) equity and bond index year-over-year (YoY) returns. We use quarterly data between 1998 and 2025.

For example, for US large caps (represented by the MSCI USA Gross Total Return USD Index), the correlation with year-over-year inflation is 20% and the correlation with GDP growth is 60%. Thus, US large caps are represented in our two-dimensional feature space by a vector:

$$x = (0.20, 0.60) \in R^2.$$

Similarly, US Treasuries (represented by the Bloomberg US Treasury Total Return Index), are −39% correlated with the subsequent quarter's inflation and −42% correlated with the subsequent quarter's GDP growth. Thus, US large caps are represented in our two-dimensional feature space by a vector:

$$x = (-0.39, -0.42) \in R^2.$$

We train the model on three equity indexes (the MSCI USA Gross Total Return USD Index, MSCI USA Small Cap Gross Total Return USD Index, and MSCI USA IMI High Dividend Yield Gross Total Return USD Index), setting $y = +1$, and three bond indexes (the Bloomberg US Treasury Total Return Index, Bloomberg US Corporate Investment Grade Total Return Index, and Bloomberg US Treasury Inflation-Linked Total Return Index), setting $y = -1$. Not surprisingly, all equity indexes are quite significantly positively correlated with real GDP growth, and US Treasuries are negatively correlated with real GDP growth. See **Exhibit 2** for more data details.

According to Vapnik (1999), optimizing the SVM's Lagrangian on our six training data points reduces to the following problem:

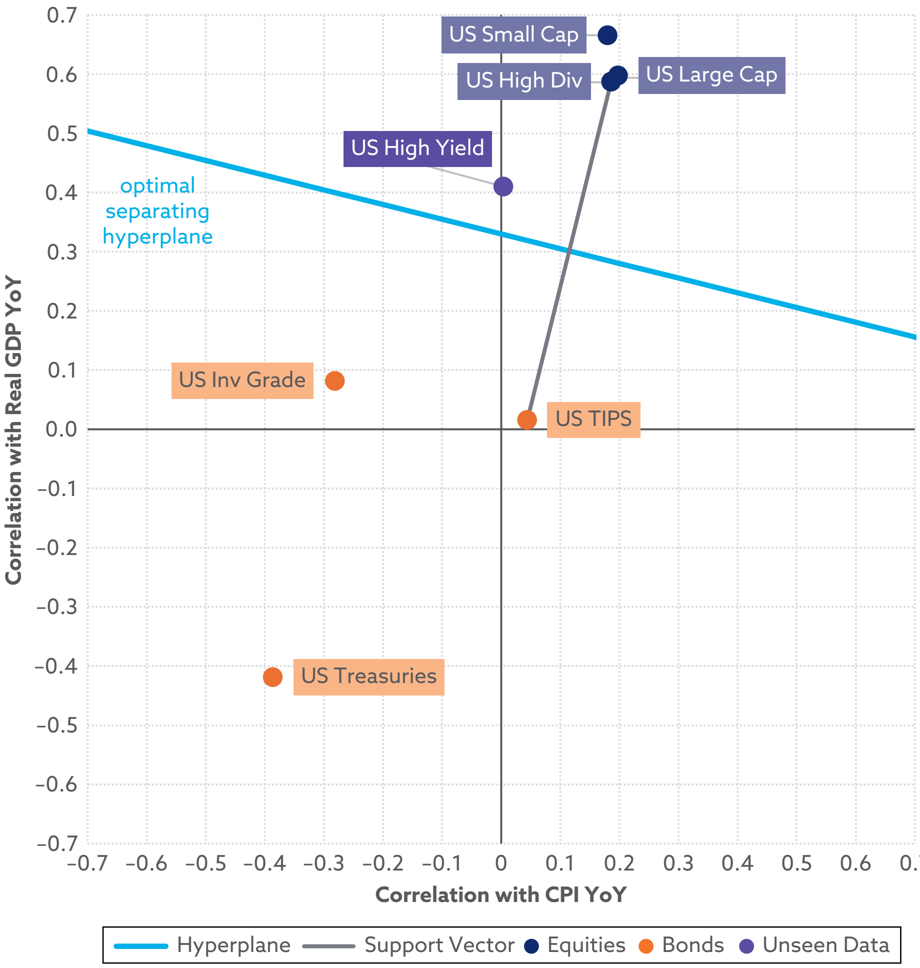
$$\min_{\alpha} \sum_{i=1}^{6} \alpha_i - \frac{1}{2} \sum_{i,j}^{6} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$

$$\text{subject to } \sum_{i=1}^{6} \alpha_i y_i = 0 \text{ and } 0 \le \alpha_i,\ i = 1, \ldots, 6,$$

where α_i are Lagrange multipliers. Refer back to Figure 1 for the general optimization setup.

Exhibit 2. Optimal Hyperplane Separating Stock and Bond Indexes in a Two-Dimensional Feature Space of Correlations with Real GDP and CPI Year over Year, 1998–2025

		Training Data						Unseen Data
	Correlations	US Large Cap	US Small Cap	US High Div	US Treasuries	US Inv Grade	US TIPS	US High Yield
Features	CPI YoY	0.20	0.18	0.19	−0.39	−0.28	0.04	0.00
	Real GDP YoY	0.60	0.67	0.59	−0.42	0.08	0.02	0.41
	y: stock (+1) or bond (−1)	1	1	1	−1	−1	−1	
	α^0: Lagrange multipliers	0	0	5.7639	0	0	5.7639	
	$(w \cdot x) - b$	1.045	1.253	1.000	−2.784	−1.048	−1.000	0.267



Notes: Quarterly correlations of CPI year-over-year change and real GDP year-over-year growth with the prior quarter's year-over-year return of equity and bond indexes. Author's calculations for illustration and educational purposes only.

Sources: Data are from Bloomberg (US Large Cap: MSCI USA Gross Total Return USD Index; US Small Cap: MSCI USA Small Cap Gross Total Return USD Index; US High Div: MSCI USA IMI High Dividend Yield Gross Total Return USD Index; US Treasuries: Bloomberg US Treasury Total Return Index; US Inv Grade: Bloomberg US Corp Investment Grade Total Return Index; US TIPS: Bloomberg US Treasury Inflation-Linked Total Return Index; US High Yield: Bloomberg US Corp High Yield Total Return Index).

After solving this problem, we can see that the optimal separating hyperplane is defined by two support vectors: the US high-dividend index and the US Treasury Inflation-Protected Security index. We have the optimized solution vector:

$$w_0 = (+1)\alpha^0_{US\,High\,Div} x_{US\,High\,Div} + (-1)\alpha^0_{US\,TIPS} x_{US\,TIPS},$$

where the (optimized) Largange multipliers, $\alpha^0_{US\,High\,Div} = \alpha^0_{US\,TIPS} = 5.7639$, are corresponding to the support vectors, $x_{US\ High\ Div}$ and $x_{US\ TIPS}$, and the Lagrange multipliers corresponding to the four nonsupport vectors are zero.

Further, we have

$$b = \frac{1}{2}(w_0 \cdot x_{US\,High\,Div} + w_0 \cdot x_{US\,TIPS}),$$

so that

$$(w_0 \cdot x_{US\,High\,Div}) - b = 1 \text{ and } (w_0 \cdot x_{US\,TIPS}) - b = -1.$$

Also note that the US large-cap index is fairly close to being a support vector:

$$(w_0 \cdot x_{US\,Large\,Cap}) - b = 1.045.$$

We can now use our SVM model to classify a data point the model has never seen: US high-yield bonds. The correlations of the Bloomberg US Corporate High Yield Total Return Index with CPI year-over-year change and real GDP year-over-year growth (features $x \in R^2$) put the US high-yield index on the side of equities, albeit close to the optimal separating hyperplane:

$$(w_0 \cdot x_{US\,High\,Yield}) - b = 0.267.$$

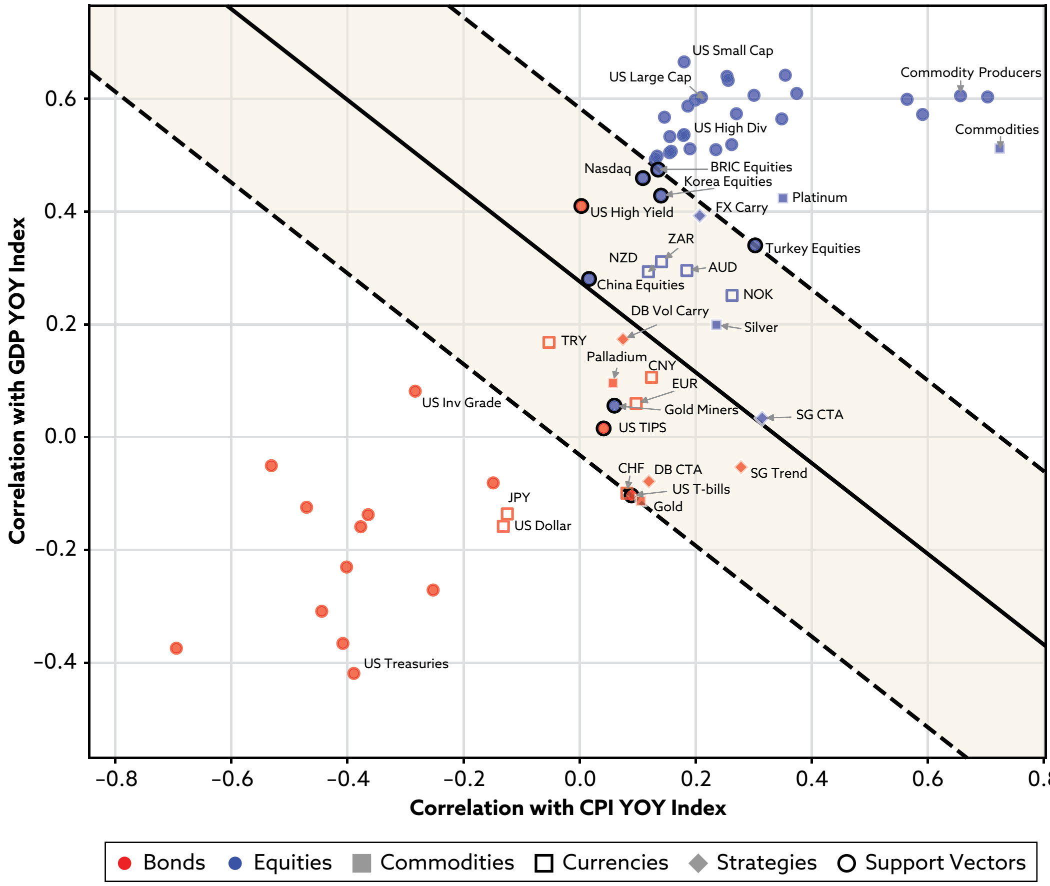
In this sense, the SVM model classifies the US high-yield index as equities rather than bonds.

For **Exhibit 3**, arising from the asset allocation research conducted in collaboration with Jiahui "Joy" Yu, PhD, we trained the data on the correlations of real GDP and CPI YoY with the prior quarter's year-over-year returns of 32 equity indexes and 15 bond indexes. In this case, stocks and bonds are not linearly separable and the trade-off parameter is set as $C = 1$, as described in Figure 2.

The support vectors in this case are six equity indexes (NYSE Arca Gold Miners Index, MSCI China Gross Total Return Local USD Index, Nasdaq-100 Total Return Index, MSCI Emerging Korea Gross Total Return Local Index, MSCI Emerging Turkey Gross Total Return Local Index, and MSCI EM BRIC Gross Total Return Local Index) and three bond indexes (Bloomberg US Treasury Inflation-Linked Total Return Index, Bloomberg US Corporate High Yield Total Return Index, and Bloomberg US Treasury Bill Index).

The trained model is then used to classify 21 other assets. Note that gold miners (NYSE Arca Gold Miners Index), gold, Japanese yen, the US Dollar Index, and CTA strategies (DB Cross Asset CTA Trend Index and SG Trend Index) are classified as bond-like assets, albeit "within the margin." US high-yield bonds (Bloomberg US Corporate High Yield Total Return Index) are classified as equity-like, similarly to the model in Exhibit 2, albeit "within the margin" in this model with regularization penalty.

Exhibit 3. Classifying Assets as Stock- or Bond-Like Using a Support Vector Machine with Regularization Penalty, 1999–2025



Notes: Quarterly correlations of CPI YoY change and real GDP YoY growth with the prior quarter's YoY return assets. The SVM model with trade-off parameter $C = 1$ is trained on 32 equity indexes and 15 bond indexes and is used to classify 21 other assets (4 metals, 1 commodity index, 11 currencies, and 5 strategy indexes) as equity-like and bond-like.

Labeled points: US Large Cap: MSCI USA Gross Total Return USD Index; US Small Cap: MSCI USA Small Cap Gross Total Return USD Index; US High Div: MSCI USA IMI High Dividend Yield Gross Total Return USD Index; US Treasuries: Bloomberg US Treasury Total Return Index; US Inv Grade: Bloomberg US Corporate Investment Grade Total Return Index; US TIPS: Bloomberg US Treasury Inflation-Linked Total Return Index; US High Yield: Bloomberg US Corporate High Yield Total Return Index; Gold: XAUUSD Spot Exchange Rate - Price of 1 XAU in USD (US dollars per troy ounce); Silver: XAGUSD Spot Exchange Rate - Price of 1 XAG in USD (US dollars per troy ounce); Platinum: XPTUSD Spot Exchange Rate - Price of 1 XPT in USD (US dollars per troy ounce); Palladium: XPDUSD Spot Exchange Rate - Price of 1 XPD in USD (US dollars per troy ounce); Commodities: Bloomberg Commodity Index Total Return; Gold Miners: NYSE Arca Gold Miners Index; China Equities: MSCI China Gross Total Return Local USD Index; Nasdaq: Nasdaq-100 Total Return Index; Korea Equities: MSCI Emerging Korea Gross Total Return Local Index; Turkey Equities: MSCI Emerging Turkey Gross Total Return Local Index; US T-bills: Bloomberg US Treasury Bill Index; Commodity Producers: MSCI World Commodity Producers Gross Total Return USD Index; ZAR: ZARUSD Total Return Long - Long ZAR, Short USD; NZD: NZDUSD Total Return Long - Long NZD, Short USD; AUD: AUDUSD Total Return Long - Long AUD, Short USD; NOK: NOKUSD Total Return Long - Long NOK, Short USD; GBP: GBPUSD Total Return Long - Long GBP, Short USD; TRY: TRYUSD Total Return Long - Long TRY, Short USD; CNY: CNYUSD Total Return Long - Long CNY, Short USD; EUR: EURUSD Total Return Long - Long EUR, Short USD; CHF: CHFUSD Total Return Long - Long CHF, Short USD; JPY: JPYUSD Total Return Long - Long JPY, Short USD; US Dollar: US Dollar Index; SG CTA: SG CTA Index; SG Trend: SG Trend Index; FX Carry: Bloomberg GSAM FXCarry Index; DB Vol Carry: Deutsche Bank Volatility Carry (US Large Cap); DB CTA: DB Cross Asset CTA Trend Index.

Sources: Author's calculations for illustration and educational purposes only. Data are from Bloomberg [Bloomberg US Agg Total Return Index; Bloomberg US Corporate High Yield Total Return Index; Bloomberg 1-3 Yr Gov/Credit Total Return Index; Bloomberg US Treasury Total Return Index; Bloomberg 10+ Yr Gov/Credit Total Return Index; Bloomberg US Treasury Inflation-Linked Total Return Index; Bloomberg US Treasury Bill Index; Bloomberg US Mortgage Backed Securities Index; Bloomberg US Corporate Investment Grade Total Return Index; Bloomberg Long US Treasury with 10+Y Total Return Index; Bloomberg US Treasury Inflation Notes 10+Y Total Return Index; Bloomberg Global Agg - Australia Total Return Index Unhedged AUD; Bloomberg Global Agg - Canadian Total Return Index Unhedged CAD; Bloomberg Global Agg - Japanese Total Return Index Unhedged JPY; Bloomberg EuroAgg Index; MSCI USA Value Gross Total Return USD Index; MSCI USA Growth Gross Total Return USD Index; MSCI USA Gross Total Return USD Index; MSCI USA Quality Gross Total Return USD Index; Nasdaq-100 Total Return Index; MSCI USA Minimum Volatility Gross Total Return USD Index; S&P 500 Total Return Index; MSCI USA Small Cap Gross Total Return USD Index; MSCI USA IMI High Dividend Yield Gross Total Return USD Index; MSCI Australia Gross Total Return Local Index; MSCI Canada Gross Total Return Local Index; MSCI UK Gross Total Return Local Index; MSCI EU Gross Total Return Local Index; MSCI Japan Gross Total Return Local Index; MSCI EM Gross Total Return Local Index; MSCI Brazil Gross Total Return Local Index; MSCI EM BRIC Gross Total Return Local Index; MSCI China Gross Total Return Local USD Index; MSCI India Gross Total Return Local Index; MSCI Hong Kong Gross Total Return Local Index; MSCI Emerging Korea Gross Total Return Local Index; MSCI Emerging South Africa Gross Total Return Local Index; MSCI Emerging Turkey Gross Total Return Local Index; MSCI Emerging Taiwan Gross Total Return Local Index; S&P Global Natural Resources Total Return Index; MSCI World Energy Gross Total Return USD Index; MSCI World Commodity Producers Gross Total Return USD Index; MSCI World Commodity Producer Sector Capped Gross Total Return USD Index; NYSE Arca Gold Miners Index; Dow Jones Equity REIT Total Return Index; MSCI US REIT Gross Total Return Index; S&P Global Infrastructure Total Return Index; XAUUSD Spot Exchange Rate - Price of 1 XAU in USD (US Dollars per Troy Ounce); XAGUSD Spot Exchange Rate - Price of 1 XAG in USD (US Dollars per Troy Ounce); XPTUSD Spot Exchange Rate - Price of 1 XPT in USD (US Dollars per Troy Ounce); XPDUSD Spot Exchange Rate - Price of 1 XPD in USD (US Dollars per Troy Ounce); Bloomberg Commodity Index Total Return; ZARUSD Total Return Long - Long ZAR, Short USD; NZDUSD Total Return Long - Long NZD, Short USD; AUDUSD Total Return Long - Long AUD, Short USD; NOKUSD Total Return Long - Long NOK, Short USD; GBPUSD Total Return Long - Long GBP, Short USD; TRYUSD Total Return Long - Long TRY, Short USD; CNYUSD Total Return Long - Long CNY, Short USD; EURUSD Total Return Long - Long EUR, Short USD; CHFUSD Total Return Long - Long CHF, Short USD; JPYUSD Total Return Long - Long JPY, Short USD; US Dollar Index; SG CTA Index; SG Trend Index; Bloomberg GSAM FXCarry Index; Deutsche Bank Volatility Carry (US Large Cap); DB Cross Asset CTA Trend Index].

Support Vector Machine Applications in Investments

Support vector machines can be used for descriptive classification, prediction, and optimization. I showed a classification example of stocks versus bonds in the previous section.

Nazareth and Reddy (2023) conducted a large-scale review of literature on financial applications of machine learning. They found that SVM models are the most frequently studied, especially in insolvency and bankruptcy prediction, mainly because of their effectiveness in dealing with two-group classification problems. There are also many applications of SVMs in stock market and cryptocurrency studies.

Ryll and Seidens (2019) conducted a systematic meta-analysis of existing works on ML-based trading algorithms and found that SVMs significantly "outscore" some neural networks that have similar objectives in classification.

As with any artificial intelligence (AI) or machine learning (ML) applications in investing, a major risk is overfitting the SVM to the training data; see Simonian (2024) for a comprehensive model validation review.

Prediction

In this section, I discuss the application of SVM to predicting future asset returns and credit ratings. For example, an investor may want to use SVMs with the following objectives:

- Given an asset universe, select assets more likely to have higher expected returns or outperform a benchmark over a subsequent time period.
- Reconstruct corporate credit ratings with fundamental accounting variables.

Given an asset universe, SVMs can be used for separating assets into two groups: those likely to have higher expected returns and those likely to have lower expected returns. Portfolios can then be formed by going long assets likely to outperform and/or short assets likely to underperform.

Fan and Palaniswami (2001) used SVMs to predict which stocks trading on the Australian Stock Exchange would be likely to outperform the market. They considered 37 firm accounting indicators, grouped them into eight groups (return on capital, profitability, leverage, investment, growth, short-term liquidity, return on investment, and risk), and used principal component analysis for dimension reduction, obtaining eight-element input (feature) vectors prior to training, with each element representing a single extracted principal component from each group. Then, they applied SVM techniques to this eight-dimensional feature space in order to select the top-returning 25% of the stock returns every year, labeled as exceptional high-return stock (Class +1), while the others were labeled as unexceptional return (Class −1). The equally weighted portfolio of exceptional high-return stock formed by SVM had a total return of 207% over a five-year out-of-sample period, 1995–99, outperforming the benchmark return of 71%.

Kim (2003) used SVMs to predict the direction of change in the daily KOSPI, the South Korean composite stock price index, using 12 technical indicators (including momentum, price oscillator, and relative strength index) as features. The KOSPI data sample has 2,928 trading days, from January 1989 to December 1998. About 20% of the data were used for holdout and 80% for training. The study used both the polynomial kernel and the Gaussian radial basis kernel for SVM. Empirically, SVM outperformed the back-propagation neural network and case-based reasoning by 3.10% and 5.85%, respectively, for the out-of-sample ("holdout") data. Kim (2003, p. 318) concluded that "SVM provides a promising alternative for financial time-series forecasting."

Huerta, Corbacho, and Elkan (2013) used SVM to identify stocks whose volatility-adjusted price change falls within the highest or lowest quantile (e.g., the highest or lowest 25%). The highest-ranked stocks were used for long positions, and the lowest-ranked stocks were used for short sales. The data sample is the US equity universe available from the merged CRSP/Compustat database between 1981 and 2010. The study examined 44 fundamental and 7 technical features. Empirically, the constructed portfolios achieved annual returns of 15% (not counting transaction costs), with volatilities under 8%. The authors also discussed the process of choosing SVM meta-parameters to mitigate overfitting.

An interesting application to credit ratings is studied in Huang, Chen, Hsu, Chen, and Wu (2004). They examined two datasets: a Taiwanese dataset of 74 cases of bank credit ratings and 21 financial variables, which covered 25 financial institutions from 1998 to 2002, and a US dataset with 265 cases of bank credit ratings for 36 commercial banks from 1991 to 2000. Five rating categories appeared in the US dataset: AA, A, BBB, BB, and B. The authors developed several SVM-based models for credit ratings with accounting ratios' feature spaces of dimensions ranging from 7 to 21. SVMs achieved the best performance (compared with benchmark neural networks) in three of the four models tested, and SVM and neural network models outperformed the logistic regression model consistently.

Portfolio Construction and Optimization

In this section, I discuss applications of SVMs to portfolio optimization, such as

- using SVMs to preselect assets before portfolio construction,
- integrating SVMs with portfolio construction and optimization, and
- using SVMs for feature selection in portfolio construction.

Gupta, Mehlawat, and Mittal (2012) used SVMs for classifying financial assets in three predefined classes and then solving a portfolio selection problem incorporating investor preferences.

Silva, Felipe, de Andrade, da Silva, de Melo, and Tonelli (2024) found that preselecting Brazilian stocks using the SVM model and subsequently optimizing them by maximizing the Sharpe ratio resulted in a superior return and faster recovery after drawdown periods compared with the benchmark.

Islip, Kwon, and Kim (2025) proposed integrating SVMs and cardinality-constrained mean–variance optimization into one procedure (rather than preselecting assets before portfolio optimization), called SVM-MVO. Their joint selection of a portfolio and a separating asset-screening hyperplane optimized the trade-off between risk-adjusted returns, hyperplane margin, and classification errors that were made by the hyperplane. The authors observed that "SVM-MVO models are equivalent to regularization that penalizes portfolios with eligibility decisions that cannot be well explained by a low-dimensional hyperplane" (Islip et al. 2025, p. 1056). The study "demonstrates the effectiveness of the SVM-MVO models in constructing portfolios with lower risk, higher returns, and higher Sharpe ratios" (Islip et al. 2025, p. 1057). In particular, they outperformed their cardinality-constrained MVO counterpart during major financial events.

Integration of various ML techniques into the framework of Markowitz's portfolio selection is discussed in López de Prado, Simonian, Fabozzi, and Fabozzi (2025). In particular, the study found that SVMs can be helpful in feature selection by pinpointing variables most relevant to asset price movements.

The next chapter examines another widely used group of supervised learning tools known as *supervised ensemble methods*.

References

Awad, Mariette, and Rahul Khanna. 2015. "Support Vector Regression." In *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*, 67–80. Berkeley, CA: Apress. doi:10.1007/978-1-4302-5990-9_4.

Cortes, Corrina, and Vladmir Vapnik. 1995. "Support-Vector Networks." *Machine Learning* 20 (3): 273–97. doi:10.1023/A:1022627411411.

Fan, A., and Marimuthu Palaniswami. 2001. "Stock Selection Using Support Vector Machines." *International Joint Conference on Neural Networks Proceedings (Cat. No.01CH37222)*, vol. 3: 1793–98. doi:10.1109/IJCNN.2001.938434.

Gupta, Pankaj, Mukesh Kumar Mehlawat, and Garima Mittal. 2012. "Asset Portfolio Optimization Using Support Vector Machines and Real-Coded Genetic Algorithm." *Journal of Global Optimization* 53: 297–315. doi:10.1007/s10898-011-9692-3.

Huang, Zan, Hsinchun Chen, Chia-Jung Hsu, Wun-Hwa Chen, and Soushan Wu. 2004. "Credit Rating Analysis with Support Vector Machines and Neural Networks: A Market Comparative Study." *Decision Support Systems* 37 (4): 543–58. doi:10.1016/S0167-9236(03)00086-1.

Huerta, Ramon, Fernando Corbacho, and Charles Elkan. 2013. "Nonlinear Support Vector Machines Can Systematically Identify Stocks with High and Low Future Returns." *Algorithmic Finance* 2 (1): 45–58. https://journals.sagepub.com/doi/abs/10.3233/AF-13016.

Islip, David, Roy H. Kwon, and Seongmoon Kim. 2025. "Integration of Support Vector Machines and Mean–Variance Optimization for Capital Allocation." *European Journal of Operational Research* 322 (3): 1045–58. doi:10.1016/j.ejor.2024.11.022.

Kim, Kyoung-jae. 2003. "Financial Time Series Forecasting Using Support Vector Machines." *Neurocomputing* 55 (1–2): 307–19. doi:10.1016/S0925-2312(03)00372-2.

López de Prado, Marcos, Joseph Simonian, Francesco A. Fabozzi, and Frank J. Fabozzi. 2025. "Enhancing Markowitz's Portfolio Selection Paradigm with Machine Learning." *Annals of Operations Research* 346: 319–40. doi:10.1007/s10479-024-06257-1.

Nazareth, Noella, and Yeruva Venkata Ramana Reddy. 2023. "Financial Applications of Machine Learning: A Literature Review." *Expert Systems with Applications* 219 (1 June). doi:10.1016/j.eswa.2023.119640.

Ryll, Lukas, and Sebastian Seidens. 2019. "Evaluating the Performance of Machine Learning Algorithms in Financial Market Forecasting: A Comprehensive Survey." Working paper (6 July). doi:10.48550/arXiv.1906.07786.

Silva, Natan Felipe, Lélis Pedro de Andrade, Washington Santos da Silva, Maísa Kely de Melo, and Adriano Olímpio Tonelli. 2024. "Portfolio Optimization Based on the Pre-selection of Stocks by the Support Vector Machine Model." *Finance Research Letters* 61 (March). doi:10.1016/j.frl.2024.105014.

Simonian, Joseph. 2024. *Investment Model Validation: A Guide for Practitioners*. Charlottesville, VA: CFA Institute Research Foundation. doi:10.56227/24.1.15.

Smola, Alex J., and Bernhard Schölkopf. 2004. "A Tutorial on Support Vector Regression." *Statistics and Computing* 14 (3): 199–222. doi:10.1023/B:STCO.0000035301.49549.88.

Vapnik, Vladmir N. 1999. "An Overview of Statistical Learning Theory." *IEEE Transactions on Neural Networks* 10 (5): 988–99. doi:10.1109/72.788640.

Vapnik, Vladmir N. 2000. *The Nature of Statistical Learning Theory*, 2nd ed. New York: Springer-Verlag.

Vapnik, Vladmir N., Steven Golowich, and Alex Smola. 1997. "Support Vector Method for Function Approximation, Regression Estimation, and Signal Processing." In *Advances in Neural Information Processing Systems 9 (NIPS 1996)*, edited by M. Mozer, M. Jordan, and T. Petsche, 281–87. Cambridge, MA: MIT Press.