Machine Learning in Financial Market Risk: VaR Exception Classification Model

Wei XIONG[1] [2]
Supervision: Antonella Bucciaglia
J.P. Morgan, Quantitative Research Market Risk
September 10, 2018

Abstract

Value-at-risk (VaR) is an important risk measure now widely used by financial institutions and regulators to quantify market risk and compute regulatory capital charge. The performance of VaR model can be examined by back-testing. Based on back-testing information, this paper develops a Machine Learning model to classify market risk VaR exceptions into "market move" and "VaR model issue" by SVM classifier. TSNE data visualization algorithm is used to study the separability of the two categories. Model parameters are selected by cross validation, and good prediction results are achieved.

Besides classification, we propose a numerical method to approximate VaR model predicted P&L and prove an asymptotic convergence property. The P&L attribution is studied by Lasso regression which selects the most significant components in a portfolio.

Keywords Back-testing, Classification, Cross validation, Lasso, SVM, tSNE, Value-at-risk

[1]Master 2nd year, M2MO Random Modelling, Université Paris Diderot
[2]Ingénieur 3nd year, Department of Geostatistics and Applied Probability, Mines ParisTech

Acknowledgement

I would like to sincerely thank all the members of QR Market Risk for building up an amazing team. From the first day I have felt integrated in the team and motivated by the positive atmosphere. I am very grateful for the mentorship and kindness of my advisor, Antonella Bucciaglia. Her careful and detailed revision on this paper greatly contributes to its amelioration. I am thankful for the kind and helpful advices from Brahim Bellahsayn, Mohamed Sakka and Roberts Nedovis. I have learnt and progressed a lot working with them.

I would like to thank Mines ParisTech for providing me with a high-quality education environment as well as an atmosphere fitted with flexibility. I'd like to thank the Director of Geostatistics at Mines ParisTech, Prof. Hans Wackernagel, for tutoring and advising me during my study years at Mines ParisTech. His kind academic support also allows me to balance between Mines ParisTech's ingénieur program and M2MO master program in parallel.

I would like to thank especially the M2MO master's program in Paris Diderot University for providing top-level courses in financial mathematics. Thank Prof. Huyên Pham and Prof. Aristide Ouattara, for their risk management course which has greatly prepared me for succeeding this internship. I would like to thank all professors and staffs I have interacted with during my study: Prof. Jean-François Chassagneux, Prof. Noufel Frikha, Prof. Peter Tankov, Prof. Claudio Fontana, Prof. Francis Comets, Prof. Fabien Viger, Prof. Benjamin Bruder, Prof. Zorana Grbac, Prof. Olivier Guéant, Prof. Stéphane Gaïffas, Prof. Jean-Marc Bardet, Prof. Aurélie Fischer, Prof. Svetlana Gribkova and Ms. Kuntzman Virginie.

Last but not least, I would like to thank my family and my friends for supporting me during all these years.

Contents

1 Introduction

Value-at-risk (VaR) is a widely-used risk measure to quantify market risk of portfolios and to compute regulatory capital charge by financial institutions. Under a time horizon T and a given probability α (typically $\alpha = 1\%, 5\%$), the α VaR is defined as the minimum value that negative loss can reach during the time horizon $[0, T]$ with probability $1-\alpha$. The VaR model needs to capture the risk of portfolios as accurately as possible, so that the value-at-risk predicted by model aligns with actual profit and loss (P&L). This is to be examined by back-testing, which often look into occurrences of VaR exceptions.

A VaR exception happens when the P&L of that day exceeds the daily VaR. The frequency and magnitude of VaR exceptions are important indicators to evaluate the performance of a VaR model. Kupiec[1995] proposed one of the earliest backtests for testing whether the frequency of VaR exceptions is close to (α). Kupiec[1995] constructed test statistics with a VaR exception 'hit' sequence, $[I_{t+1}(\alpha)]_{t=1}^{T}$, where $I_{t+1}(\alpha)$ is defined as the indicator of whether the actual P&L on day $t+1$ exceeds the VaR predicted on day t.

$$I_{t+1}(\alpha) := \mathbb{I}\left(P\&L_{t+1} \leq VaR_t(\alpha)\right)$$

Christoffersen[1998] then built the theoretical framework for back-testing by introducing two properties of exception sequence to test the accuracy of VaR model: **unconditional coverage** and **independence**, tested respectively by a likelihood ratio test and a Markov contingency table test. Combining the two properties, the exception sequence $I_t(\alpha)$ is also compared to a Bernoulli distribution $B(\alpha)$ to test its unconditional coverage and independence properties.

All the tests above focus on a single $\alpha-$quantile. By considering multiple-level quantiles we are able to obtain more information about the accuracy of VaR model accross its entire probability support. The mathematical foundation for distributional tests was given by Rosenblatt[1952]: Given a sequence of *ex post* clean P&L $\{clnPL_{t+1}\}_{t=1}^{T}$ and *ex ante* P&L distribution functions $\{D_t\}_{t=1}^{T}$ given by a VaR model, define the transformation $clnPV_t = D_t(clnPL_{t+1})$. If the *ex ante* distribution captures all risk factors of portfolio, then the distribution of $clnPL_{t+1}$ is exactly D_t. As a direct application of Rosenblatt[1952], the clean p-value sequence $\{clnPV_{t+1}\}_{t=1}^{T}$ are independently distributed uniform. In risk management terms, a VaR model backtests perfectly, we can reasonably assume the P&L process predictive distribution aligns with the actual distribution of that P&L process. Diebold et al.[1997] suggested tests on the clean p-value series with density graphs. The shape of the p-value histograms reveals information about the overstatement or understatement of the VaR model. Berkowitz[2001] applies inverse transform of Gaussian cumulative distribution function to a clean p-value sequence: $z_{t+1} = \Phi^{-1}(clnPV_{t+1})$. Under the null hypothesis, the transformed quantile $\{z_{t+1}\}_{t=1}^{T}$ is independent standard Gaussian, on which Berkowitz[2001] constructs likelihood ratio tests, as Gaussian likelihood ratio tests have better properties, such as uniformly most powerfulness, compared to distributional tests on uniformly distributed sequence. There exist numerous other kinds of back-testing methodologies for VaR model. Campbell[2005] summarizes these back-testing methods and concludes that tests on several quantiles are most successful in identifying inaccurate VaR models when the risk is systematically under reported.

Once the back-testing has highlighted an inaccuracy, it is important for the bank to identify the root cause. In case of a VaR exception event, market risk teams often label the event into different categories for more efficient risk reporting. Based on back-testing multi-level quantiles, this paper will explore further the classification power of the p-values. Building on the idea from distributional back-testing, we will specify a classification model that combines the daily actual P&L and VaR model predicted P&L. We also implement a P&L attribution which is able to detect the negative P&L driver in case of an exception. This is achieved by Lasso regression that is powerful for feature selection. Both statistical and numerical methods are applied. For the numerical method, a convergence result is also proved in this paper.

Before proceeding to the data modeling, we apply the t-distributed Stochastic Neighbour Embedding(tSNE) algorithm to visualize the data. TSNE is a powerful and recent dimension reducing technique by van der Maaten and Hinton[2008]. It maps high-dimensional features into low-dimensional spaces (2D or 3D) by projecting original inter similarities onto low-dimensional space, so that the data points can be visualized. Our visualisation helps up justifying that the exception categories are separable, yet with some overlapping points on 2D plane.

Support Vector Machine (SVM) divides the kernel-mapped feature space by hyperplanes. It was initially proposed to deal with binary classification problem by Cortes and Vapnik[1995]. In our classification problem, the available categorical data points overlap visually on 2D plane, that's why a kernel is needed to map the original features into new feature space so that SVM can separate different categorical data points. The kernel we use in this paper is the radial basis function (RBF) kernel, of which the parameters are chosen based on 10-fold cross validation. Cross validation is proposed by Kohavi[1995] to select optimal parameters while reducing the cost of overfitting.

Finally in order to attribute portfolio P&L, we will apply a Lasso regression by fitting portfolio P&L in function of its component P&Ls. Introduced systematically by Tibshirani[1994], Lasso regression refers to a linear regression with L_1-penalized least square. The L_1-penalization on linear coefficients is able to achieve variable selection and shrinkage at the same time. Coefficients before the least important variables first shrinks to zero while increasing the L_1-penalization constant. The significance hierarchy component P&Ls will be sorted by Lasso regression, then we can attribute the exception drivers with this hierarchy and the P&L magnitudes.

The remainder of this paper is organized as follows:

Section 2 gives a brief introduction to our risk model and basic definitions in risk management: value-at-risk and p-values. Then section 3 presents the machine learning algorithms and strategies used in this paper: tSNE, SVM, cross validation and Lasso. Section 4 is a summary of visualisation and classification methodology. Section 5 introduces the implied p-value approach with a result on asymptotic convergence. Section 6 outlines how to attribute P&L to its component portfolios by Lasso regression. Finally section 7 summarizes the paper and outlook future developments of this topic.

2 Risk Model and P-values

This section introduces some basic definitions used in risk management, as well as the p-value methodology to determine a P&L exception. First of all it is important to understand the basic classification criteria of VaR exception by p-values, since all the machine learning algorithms we will subsequently use will utilise p-values as features.

2.1 P&L-based Risk Model

Let X_{t+1} denote vector of daily portfolio returns between the end of day t and the end of day $t+1$, and Δ_t the risk captured by a value-at-risk model at the beginning of day $t+1$[3]. Then according to the VaR model, daily P&L between the end of day t and the end of day $t+1$ equals the product of risk and return.

$$varPL_{t+1} = \Delta_t \cdot X_{t+1} \tag{1}$$

At the end of day $t+1$, one can observe the realized clean P&L(which excludes the intraday trades and is as defined by the Basel Committee), denoted by $clnPL$. Compared with VaR-predicted P&L, the clean P&L may contain a part that is not explained by the VaR model, denoted as $unxPL$. Then the clean P&L can be decomposed in the following way:

$$clnPL_{t+1} = \Delta_t \cdot X_{t+1} + unxPL_{t+1} \tag{2}$$

In order to calculate VaR of date t, a predictive model(namely the VaR model) calculates a risk-synchronised P&L according to some distribution D_t. As an example, historical VaR models use historical returns back to T days and the computable risk of day t:

$$D_t = \{\Delta_t X_t, \quad \Delta_t X_{t-1}, \quad ..., \quad \Delta_t X_{t-T}\} \tag{3}$$

Also note that on day t the return of day $(t+1)$ X_{t+1} is not yet available. One needs to wait until the end of day $t+1$ to observe this daily return.

[3]Here we assume the VaR model is sensitivity-based

2.2 Value-at-risk and P-values

Given a confidence level $1-\alpha$ (typically $\alpha = 1\%$ or 5%), as well as a time horizon interval $[t, t+h]$, the value-at-risk $VaR_t^{t+h}(\alpha)$ is defined as the minimum negative value that the accumulated loss could reach between time $[t, t+h]$ with a probability $1-\alpha$. If the P&L between $[t, t+h]$ is continuously distributed, then $VaR_t^{t+h}(\alpha)$ is the $\alpha-$quantile of the cumulative distribution function.

Definition 2.1. Between time interval $[t, t+h]$, the VaR of a portfolio P&L PL_t^{t+h} at confidence level $1-\alpha \in (0,1)$ is the minimum negative value y such that the PL_t^{t+h} could reach with probability $(1-\alpha)$:

$$\mathrm{VaR}_t^{t+h}(\alpha) := \sup\{y : \mathbb{P}(\mathrm{PL}_t^{t+h} \leq y) \leq \alpha\} \tag{4}$$

We define the clean P&L p-value as the value implied from VaR model cumulative probability function of the occurrence of a loss of magnitude $clnPL_{t+1}$:

$$clnPV_{t+1} = D_t(clnPL_{t+1}) \tag{5}$$

Similarly the VaR P&L p-value is defined as:

$$varPV_{t+1} = D_t(varPL_{t+1}) \tag{6}$$

A good property of p-values is that the $\mathrm{VaR}_t^{t+1}(\alpha)$ is contained in the distribution, so that we can directly determine if a P&L exception event happens by comparing the p-value with α. The p-value space ranges from 0 to 1, therefore the P&Ls are 'normalized' automatically when projected onto the p-value spaces. The auto-normalization will be very helpful for statistical learning when all the features range in $[0, 1]$. For example, when $\alpha = 5\%$, we will know there is a VaR exception on day $(t+1)$ if $clnPV_{t+1} \leq 5\%$, as this is equivalent to $clnPL_{t+1} \leq \mathrm{VaR}_t^{t+1}(5\%)$.

The VaR-predicted p-value of day $t+1$ is essentially determined by the risk-return $\Delta_t X_{t+1}$ projected into historical risk-return sequence $\{\Delta_t X_t, \Delta_t X_{t-1}, ..., \Delta_t X_{t-T}\}$. However we are not able to obtain X_{t+1} at the beginning of day $t+1$, as X_{t+1} is not contained in available distribution D_t. Moreover, the risk vector Δ_t may evolve as a function of t, so on a forward date $(t+k)$, the historical vector P&L of day $(t+1)$ $\Delta_{t+k}X_{t+1}$, is likely to evolve as k moves forward. We then define $k-$forward vector p-value.

Definition 2.2. The $k-$forward vector p-value of day $t+1$ is the p-value calculated by the k^{th} element $\Delta_{t+k}X_{t+1}$ in D_{t+k} against D_{t+k}, the distribution of day $t+k$:

$$PV_{t+1}^{(k)} = D_{t+k}(\Delta_{t+k}X_{t+1}) \tag{7}$$

Generally speaking, there are various possible drivers for a clean P&L VaR exception: A downside market movements that drives the P&L downward the value-at-risk, or risk factors driving a loss but which are not captured by the VaR model.

We label them as “Market Move” and “Model Issues” respectively. The first loss arises from an ‘expected’ market move, where the VaR captures the unfavourable risk factor in one of dimensions in the risk vector Δ_t. In this case both the $varPL$ and $clnPL$ moves are excessively below the value-at-risk. The second category of loss arises from out-of-model risk factors that negatively impact the portfolio value. In this case, the unexplained P&L is relatively high, and $varPL$ doesn't have an exception, whereas clean P&L is actually below the value-at-risk.

On p-value spaces, we can define a simple decision criterion to classify a “Market Move” and a “Model Issue”:

$$\mathbb{I}(\text{Market move on day } t+1) = \mathbb{I}(clnPV_{t+1} \leq \alpha, varPV_{t+1} \leq \alpha) \tag{8}$$
$$\mathbb{I}(\text{Model issue on day } t+1) = \mathbb{I}(clnPV_{t+1} \leq \alpha, varPV_{t+1} > \alpha) \tag{9}$$

However even we have simple classification criteria, the VaR model is not able to compute the $varPL_{t+1}$, because the distribution D_t only contains returns prior to day $t+1$, thus we do not know the value $\Delta_t \cdot X_{t+1}$. This limitation leads us to the extensions: We will use forward p-values as an approximation to VaR-predicted P&L, with an introduction of Machine Learning models that are trained from historical exception data and then classifies exception category based on $clnPV$ and forward p-values.

2.3 P-values as Model Features

For the vector p-value $PV_{t+1}^{(k)}$, moving k forward from 1 to K, we obtain a K-dimensional forward p-value vector $\{PV_{t+1}^{(k)}\}_{k=1}^{K}$. We will then use it as approximation of VaR-predicted p-value. More detailed explanation why we choose this approximation can be found in section 4.1. Using both $clnPV_{t+1}$ and $\{PV_{t+1}^{(k)}\}_{k=1}^{K}$, and historically classified VaR exception data, we hope to train a model that automates the classification, as well as attributing the exception to the components of P&L.

Figure (1) is the parallel coordinate plot for average p-values of all real exception data points. P-values of market moves exceptions are on average much lower than model issues exceptions, which aligns with our p-value classification criterion. This suggests that using vector p-values as approximation to the VaR p-value is a plausible methodology to distinguish market moves from model issues.

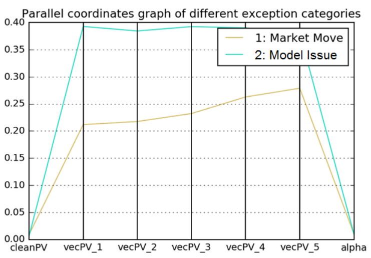


Figure 1: Parallel coordinates plot of average vector p-values between two categories

To summarize, in order to classify a VaR exception happening on day $t+1$, we wait for K days forward until we have collected the complete K-dimensional forward p-values. The chosen features include $clnPV_{t+1}, \{PV_{t+1}^{(k)}\}_{k=1}^{K}$ and α. We pick the VaR-level α as a feature because when the training data contains non-exception samples, the trained model should be able to distinguish exceptions and non-exceptions according to $clnPV$ and α' value. In this paper, we will only concentrate on binary classifiers rather than considering non-exception samples. We hope the model to be specialized in telling 'Market Move' from 'Model Issues', while non-exception samples can be distinguished by setting a hard boundary $\mathbb{I}(clnPV_{t+1} > \alpha)$. But for potential generalization $clnPV$ and α are still used as features.

Figure (2) presents the cumulative distribution curve of clean p-value and 5-forward p-values in a simulated portfolio.

Formally, we simulate VaR predicted P&L and unexplained P&L by two independent Gaussian distribution.

$$varPL_{t+1} \sim N(0,1), unxPL_{t+1} \sim N(0,1),$$
$$varPL_{t+1} \text{ and } unxPL_{t+1} \text{ are independent, } \forall t \in [1,500]$$

Then $clnPL_{t+1} = varPL_{t+1} + unxPL_{t+1} \sim N(0,\sqrt{2})$

When the risk profile $\{\Delta_{t+k}\}$ is stable (in 1-dimensional case this was simulated by using an invariant sign for Δ), the forward p-values perfectly coincides with $varPV$ for any date t, since the risk profile is reduced in the projection with only X_{t+1} affecting the p-value. In the chart the forward p-values are uniformly distributed, because $\{X_{t+1}, \forall t \in [1,500]\}$ is a set of *i.i.d.* Gaussian random numbers. The Gaussian unexplained P&L in clean P&L drives the CDF of clean P&L to move out of uniform distribution range. In this case, the distribution vectors tend to underestimate the VaR. In general the VaR model is only globally appropriate when the clean p-value follows an uniform distribution.

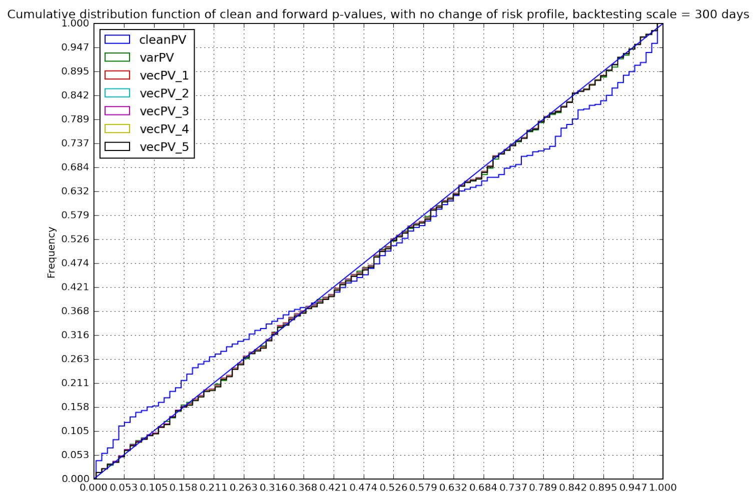


Figure 2: P-value chart in the case of a simulated portfolio: Gaussian clean P&L and Gaussian unexplained P&L

3 Machine Learning Algorithms Introduction

In this section, we summarize the machine learning tools and algorithms used in exception classification. In each subsection, we will briefly outline their mathematical basis and selected optimization algorithms.

3.1 T-distributed Stochastic Neighbor Embedding

T-distributed Neighbor Embedding is a powerful non-linear visualization technique. It maps high-dimensional data onto a low-dimensional space(2D or 3D) while keeping the original similarity structure. Proposed by van der Maaten and Hinton[2008], it outperforms most other visualization techniques on various problems.

If we denote the original high-dimensional data points by $\mathcal{X} = \{x_i\}_{i=1}^n$, the tSNE algorithm is essentially a map $\mathcal{Y} = \text{tSNE}(\mathcal{X})$, where $\mathcal{Y}$ is the set mapped data points in low-dimensional space. The tSNE minimizes the similarity error between $\mathcal{Y}$ and $\mathcal{X}$, so that $\mathcal{Y}$ keeps the same point-wise similarity as $\mathcal{X}$.

In the high-dimensional space, the similarity of two data points is quantified using a Gaussian probability. Specifically, a similarity of x_j given x_i is defined as the conditional probability that the point x_i will pick x_j as its neighbor:

$$p_{j|i} = \frac{\exp(-\frac{||x_j - x_i||^2}{2\sigma_i^2})}{\sum_{k \neq i} \exp(-\frac{||x_k - x_i||^2}{2\sigma_i^2})} \tag{10}$$

where σ_i is the Gaussian variance centered at x_i. For each point x_i, the corresponding variance σ_i is found by a binary search so that σ_i produces the probability distribution $P_i = \{p_{j|i}, j \neq i\}$ with a user-fixed perplexity:

$$Perp(P_i) = 2^{H(P_i)}$$

where $H(P_i)$ is the entropy of P_i

$$H(P_i) = -\sum_{j \neq i} p_{j|i} \log_2 p_{j|i}$$

The joint similarity of $\mathcal{X}$ is featured by the set of pairwise similarities, defined as

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

In the low-dimensional space, Student t-distribution is used. The law centered at y_i is assumed to follow a t-distribution with degree of freedom 1. The pairwise similarity between points in $\mathcal{Y}$ is then defined as

$$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\sum_{k \neq l}(1 + ||y_k - y_l||^2)^{-1}}$$

The tSNE uses Kullback-Leibler divergence between the joint probability P and Q:

$$C = KL(P||Q) = \sum_{i,j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

where the variables to optimize are the low-dimensional points $\mathcal{Y}$. The tSNE then applies a momentum-accelerated gradient descent to obtain a minimum $\mathcal{Y}$:

$$\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} - \eta \frac{\delta C}{\delta y} + \alpha(t)(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)})$$

where the gradient equals

$$\frac{\delta C}{\delta y} = 4 \sum_{j} (p_{ij} - q_{ij})(y_i - y_j)(1 + ||y_i - y_j||^2)^{-1} \tag{11}$$

The tSNE is a very powerful tool to visualize the data on a 2D plane without losing the similarity property of the original data. However, tSNE uses the whole datasets, and it is not capable of predicting out-of-sample points once the visualization is finalized. One can also add the out-of-sample new exception data point into the dataset, and re-run tSNE, then conduct the classification on embedded 2D dataset. The initialization of $\mathcal{Y}$ is from a Gaussian $N(0, 10^{-4}Id)$, so each visualization produces different results. This reduces the prediction reliability of using tSNE as a classifier. Therefore, tSNE is used here only as a reference to visualize the exceptions and test tentative models on embedded data.

3.2 K-nearest Neighbors

K-nearest Neighbors(KNN) is a simple non-parametric algorithm that constructs its decision function purely on input training data. Trained on labelled set of data points, the KNN classifier assigns a label to a new incoming data point x by looking at the majority of labels in its k nearest neighbors $\{\mathrm{nb}_i(x)\}_{i=1}^{k}$ from training set. The distance is defined in the high-dimensional feature space. Alternatively the 'majority voting' can be weighted by assigning each neighbor a weight $\omega_i = \frac{C}{\mathrm{dist}(x, \mathrm{nb}_i(x))}$, where C is the constant that insures $\sum_{i=1}^{k} \omega_i = 1$.

The model parameter k can be selected by cross validation, which will be introduced in following sections.

3.3 RBF Kernel Support Vector Machine

Support vector machine (SVM) is a supervised learning algorithm used for classification and regression. The idea behind SVM is to separate the data space by a hyperplane. Generally the raw feature space $\mathcal{X}$ is transformed into a new features space $\mathcal{Z}$ by a map $\varphi : \mathbb{R}^d \to \mathbb{R}^k$:

$$\mathcal{Z} = \varphi(\mathcal{X})$$

before entered as features into the SVM, in which $\mathcal{Z}$ is the linear feature space. When φ is the identity map in space $\mathbb{R}^d$, the kernel SVM is reduced to linear SVM, which applies when the label variable have a linear dependency with original features. In more general cases when the data points in original space $\mathbb{R}^d$ have no linear representation, then a non-linear mapping φ transforms the data into new features, in the hope that in projected space $\mathbb{R}^k$ the data points are linearly separable.

Given a labelled dataset $\{(x_i, y_i) : x_i \in \mathbb{R}^d, y_i \in \{-1, 1\}, \forall i \in \{1, 2, ..., n\}\}$, the SVM aims at estimating the hyperplane parameters $\omega \in \mathbb{R}^d$, $b \in \mathbb{R}$, such that the hyperplane

$$H = \{x \in \mathbb{R}^d : \omega^T \cdot \varphi(x) + b = 0\}$$

separates the data points by their labels.

The ideal case is when all points labelled by $y = 1$ and by $y = -1$ fall in different sides of the hyperplane. When the points of two labels overlap in space $\mathbb{R}^k$, "slack variables" s_i are introduced to train a hyperplane that distinguish most points. The mathematics of the SVM problem can be formalized as the following:

$$\begin{cases} \min\limits_{\omega,b} \frac{1}{2}||\omega||^2 + C\sum\limits_{i=1}^{n} s_i \\ \text{s.t. } y_i(\omega^T \cdot \varphi(x_i) + b) \geq 1 - s_i, , \forall i \in \{1, 2, ..., n\} \\ s_i \geq 0, \forall i \in \{1, 2, ..., n\} \end{cases}$$

From Lagrangian multiplier method and Karush-Kuhn-Tucker(KKT) conditions(Kuhn and Tucker[1951], Karush[1939]), it can be shown that the constraint optimization problem is equivalent to the following unconstraint optimization problem:

$$\min_{\omega,b} \frac{1}{2}||\omega||^2 + C\sum_{i=1}^{n} \left[1 - y_i(\omega^T \cdot \varphi(x_i) + b)\right]_+^2 \tag{12}$$

which can be solved by a gradient-descent approach.

In practice, we do not need to solve explicitly the mapping φ. Instead a kernel function $K(x, x') = \varphi(x)^T \cdot \varphi(x')$ is introduced to measure the pairwise similarity of data points. In this paper, the radial basis function kernel is used:

$$K(x, x') = \exp(-\gamma||x - x'||^2) \tag{13}$$

With the kernel function, one can write the dual problem

$$\begin{cases} \max\limits_{\alpha} \sum\limits_{i=1}^{n} \alpha_i - \frac{1}{2}\sum\limits_{i,j} \alpha_i\alpha_j y_i y_j K(x_i, x_j) \\ \text{s.t. } 0 \leq \alpha_i \leq C, \quad \forall i \in \{1, 2, ..., n\} \\ \sum\limits_{i=1}^{n} \alpha_i y_i = 0 \end{cases}$$

and when the parameter α in the dual problem is solved, the predicted label of point $x \in \mathbb{R}^d$ is directly computed by the kernel and α

$$\text{SVM}(x) = \sum_{j=1}^{n} \alpha_j y_j K(x, x_j) + b$$

where b is available from any support vector x_i, such that in the original space we have $y_i(\omega^T \cdot \varphi(x_i) + b) = 1$:

$$b = y_i - \sum_{j=1}^{n} \alpha_j y_j K(x_i, x_j)$$

RBF kernel SVM is powerful when the categories depend non-linearly on features, and can achieve much higher classification performance than the linear classifiers such as Perceptrons. In this paper we will use principally RBF kernel SVM as the classifier, and seek the model parameters which achieves optimal prediction power by cross validation.

3.4 Cross Validation

When several models are eligible to be applied on a prediction problem, it is necessary to define a methodology that assesses their predictive ability so that one can select an optimal model. Cross validation is suitable for model selection problems.

Cross validation is an out-of-sample testing methodology used to assess the performance of a statistical model when applied to an independent dataset. Generally a predictive model is evaluated by testing its ability to predict new data that are not used in training the model. The whole dataset is split into training set and testing set, while the model parameters are learned from the training set, and is then tested on testing set. The principle of cross validation is to repeat this split multiple times and test the overall performance of different testing datasets. On each testing set an accuracy score is given to the model, and the model's overall performance of predicting unseen data is evaluated by aggregating the scores.

The cross validation method used in this paper is $k-$fold cross validation. The original sample is randomly partitioned into k equal size subsets, where the $k - 1$ subsets are used to train the model, and the remaining one subset is used as testing dataset. The training-testing process is then repeated k times, with each of the k subsets used only once as testing data. The eventual score is the average of scores on the k subsets.

In our case we need to select optimal SVM parameters C and γ, where C is the penality for slack variables, and γ is the scale parameter in the RBF function. We will apply 10-fold cross validation with different sets of parameters (C, γ) to select optimal parameters.

3.5 Lasso Regression

Lasso solves the linear regression problem $y = X\beta + \epsilon$, where $\beta \in \mathbb{R}^p$ by minimizing the least square sum plus a L_1-regularization term.

$$\min_{\beta} ||y - X\beta||^2 + \alpha \sum_{j=1}^{p} |\beta_j| \tag{14}$$

The L_1 regularization can set coefficients to 0, because the problem is equivalent to minimizing $\min_{\beta} ||y - X\beta||^2$ with constraints $||\beta||_1 \leq t$. This constraint forms a hypercube in $\mathbb{R}^p$ space, so the minimum is more likely to be achieved at edges of the hypercube, where some coordinate components are zero.

Lasso regression is very useful for feature selection while controlling the magnitude of parameters(shrinkage). In the following sections, we will see its application in P&L attribution.

4 VaR Exception Classification

In section 2 we have introduced the mathematical formulation of our risk model. We have chosen a VaR model which estimates the value-at-risk based on the history of returns. Specifically on day t, the model assumes the following distribution vector $D_t = \Delta_t \cdot \{X_t, X_{t-1}, ..., X_{t-T}\}$. In the distribution vector D_t we have product of risk Δ_t and historical returns X_t back to X_{t-T}. An observed downward market move X_{t+1} on day $t+1$ will fall on tail among the historical return vectors $\{X_t, X_{t-1}, ..., X_{t-T}\}$. When X_{t+1} falls in the 'normal' range of distributional returns, the 'market factors' in the risk model do not strike any unfavourable moves.If an exception happens nonetheless, we can affirm that there are some unknown risks not in the VaR model that drive the exception.

The VaR-predicted P&L $varPL_{t+1} = \Delta_t \cdot X_{t+1}$ is mapped into D_t to compute the VaR p-value $varPV_{t+1} = D_t(\Delta_t X_{t+1})$. Comparing $varPV_{t+1}$ with α, we will know whether the exception is captured by risk model and thus driven by downward market move reflected by X_{t+1}, or whether there is a model deficiency driving the exception.

4.1 Risk Variation and Forward Vector P-values

The $varPV$ is an important factor for our classification. However it can be cumbersome to compute. When it is not available, we seek to approximate it by using only the information from distribution of different dates forward. Note that the return X_{t+1} is plugged into the distribution D_{t+k} when $k \geq 1$. It is essentially the k_{th} element of in the vector D_{t+k}:

$$D_{t+k} = \{\Delta_{t+k} \cdot X_{t+k}, ..., \boldsymbol{\Delta_{t+k} \cdot X_{t+1}}, ..., \Delta_{t+k} \cdot X_{t+k-T}\}$$

We can actually plug its k_{th} element in the same distribution of day $(t+k)$ to define a $k-$day forward vector p-value, $PV_{t+1}^{(k)} = D_{t+k}(\Delta_{t+k} \cdot X_{t+1})$. The information of return X_{t+1} is contained in $PV_{t+1}^{(k)}$, in spite of a different risk profile Δ_{t+k}. When k is small ($k \leq 5$), we expect $PV_{t+1}^{(k)}$ to be close with $varPV_{t+1}$, as the risk profile of $t+k$, Δ_{t+k}, is likely to align with that of day $t+1$, Δ_{t+1} in a relatively small time range of few days. An invariant risk profile case is demonstrated in figure (2), where Δ is uni-dimensional, and Δ_t doesn't change sign. In this case, all forward vector p-values match exactly with true VaR p-value.

However sometimes the effect of risk variation cannot be neglected. We observe sometimes large standard deviation among $\{PV_{t+1}^{(k)}, k = 1, ..., K\}$ for a date $t+1$, indicating significant differences between $\{\Delta_{t+k}, k = 1, ..., K\}$, so it is really unsure which k to choose to approximate the true $varPV$. Figure (3) presents an extreme case where significant risk changes are observed in the VaR risk model. For example, there are downside noises that drive the $2-$forward and $5-$forward p-value curves above the ideal back-testing line $y = x$.

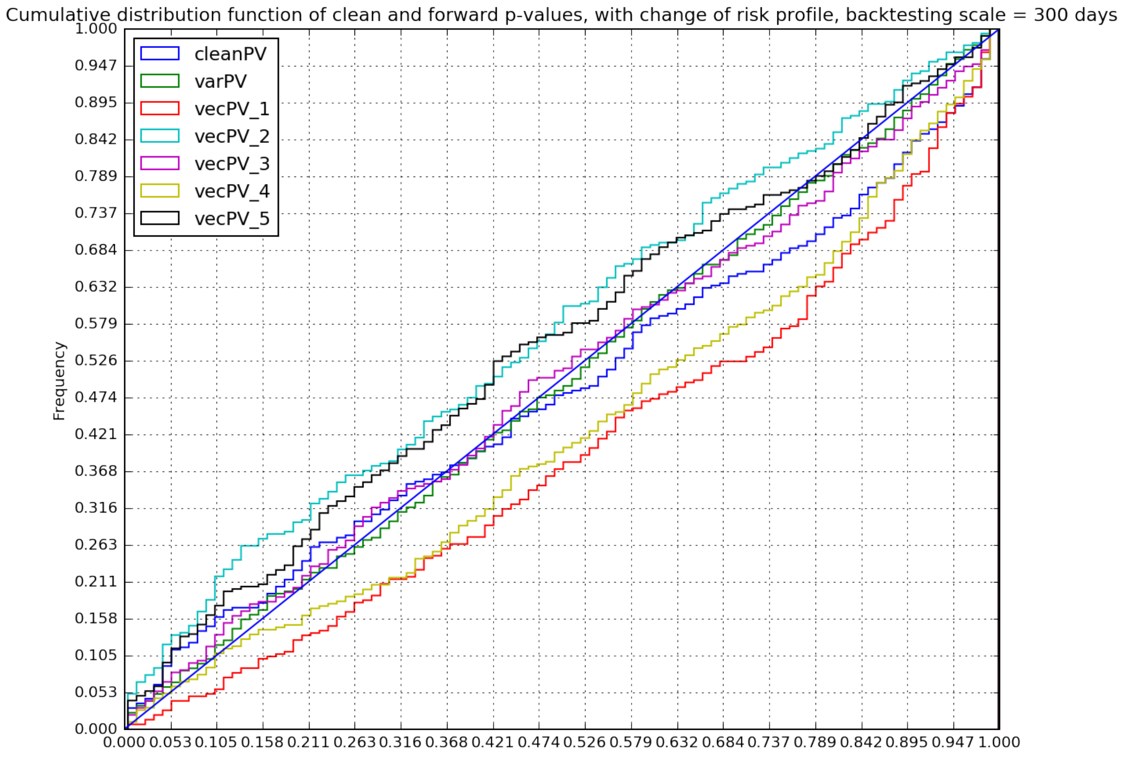


Figure 3: Clean p-value with noise and forward p-values with significantly unstable intra-day risk

Instead of replacing $varPV$ by forward p-values, we approach the problem by building a statistical model that implicitly estimates the $varPV$ using all the forward p-values' information. The model should categorize an exception when the clean p-value, $K-$forward p-values and the confidence level α is given. In practice, here is our strategy: On $t+1$ we know the $clnPV_{t+1}$ by which we can conclude if day $t+1$ has an exception. Then we wait until date $t+K$ to collect all the forward p-values $\{PV_{t+1}^{(k)}, k=1,...,K\}$. With these p-values known on K days later, we will classify date $(t+1)$'s exception into "Market Move" and "Model Issues".

To conclude, we need to build a machine learning model featuring p-values based on the fact that:

- The actual $varPL_{t+1}$ may not be available, therefore forward p-values $PV_{t+1}^{(k)}$ are calculated to simulate the $varPV_{t+1}$.
- For different $k(1 \leq k \leq K)$, the risk variation among $\{\Delta_{t+k}, k=1,...,K\}$ brings discrepancy among forward p-values, therefore we need to make use of implicit patterns in risk variation, which can be taken into account by a machine learning model.

4.2 Data Visualisation

The dimension of features is $K+2$. Before plugging these features into our model, we hope to get sense of how much the exception data aligns with our classification criteria (8) and (9). The idea is to reduce the dimensionality of features, and plot the data on a reduced dimension space. If there exist some structural differences between different exception classes, they are expected to be observed as points clustered in separated positions in 2D or 3D space. This can be achieved either by Principle Component Analysis(PCA), a linear dimension reduction technique, or by non-linear mapping. Here t-distributed Stochastic Neighbor Embedding is used and it achieved good visualisation results.

Figure (4) and (5) jointly illustrate how tSNE can distinguish different exception classes of a simulated portfolio daily P&L data. The original features are $(clnPV, PV^{(1)}, PV^{(2)}, PV^{(3)}, PV^{(4)}, ..., PV^{(K)}, \alpha)$. The simulated portfolio contains 5000 days' P&L data with their p-value empirical CDFs plotted in figure (4), therefore we have about $[\alpha(5000-K)] \approx 500$ data points with complete K forward p-values that can be used for training a classification model. The empirical CDF curves shows that the $1-$forward and $4-$forward p-values share similar risk profile while $2-$forward and $5-$forward p-values share similar risk profile of the opposite side. The $3-$forward p-value correspond with true $varPV$ along the theoretical uniform cumulative distribution function line. The clean P&L contains a Gaussian unexplained P&L, so the clean p-value CDF curve also deviates from the CDF of uniform distribution. The tSNE with perplexity$=40$ embeds the exception data points with original features onto the 2D plane. The embedded 2-dimensional coordinates keeps pointwise similarity structure of the original features. After visualization, a simple linear Perceptron classifier is tested featuring the two embedded dimensions. A Perceptron is a binary classifier in a neural network. It has similar decision function as linear SVM, but applies an online learning algorithm that updates parameters every time each sample is inserted into the model. The visualization result and Perceptron decision area are plotted and colored in the scatter plot of figure (4), where the color of each points represents its true exception category.

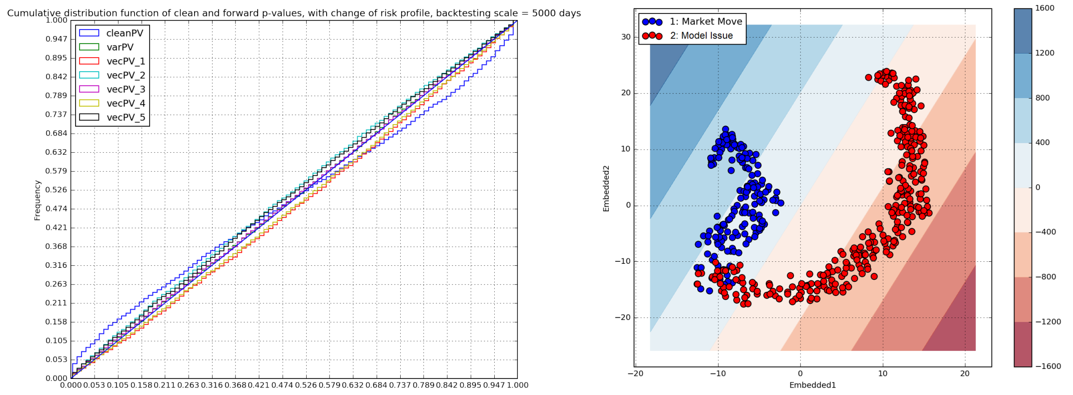


Figure 4: P-value CDF chart and TSNE visulization on simulated p-values, with linear Perceptron classifier

The linear Perceptron fits 92.88% accuracy on exceptions with embedded features. From figure (4), we can confirm that on a simulated dataset the p-values structure allows for a classifier to distinguish between 'Market Move' and 'Model Issue', since samples belonging to different categories are clustered in different positions, despite the overlapped points near the Perceptron classification boundary that causes confusion in Perceptron classification result. These overlapped points are due to noise in clean P&L that drives p-values of different categories to have similar structure. Figure (5) tells us more details abut how the tSNE method clusters the sample exception points with respect to mean and standard deviation of the forward p-values.

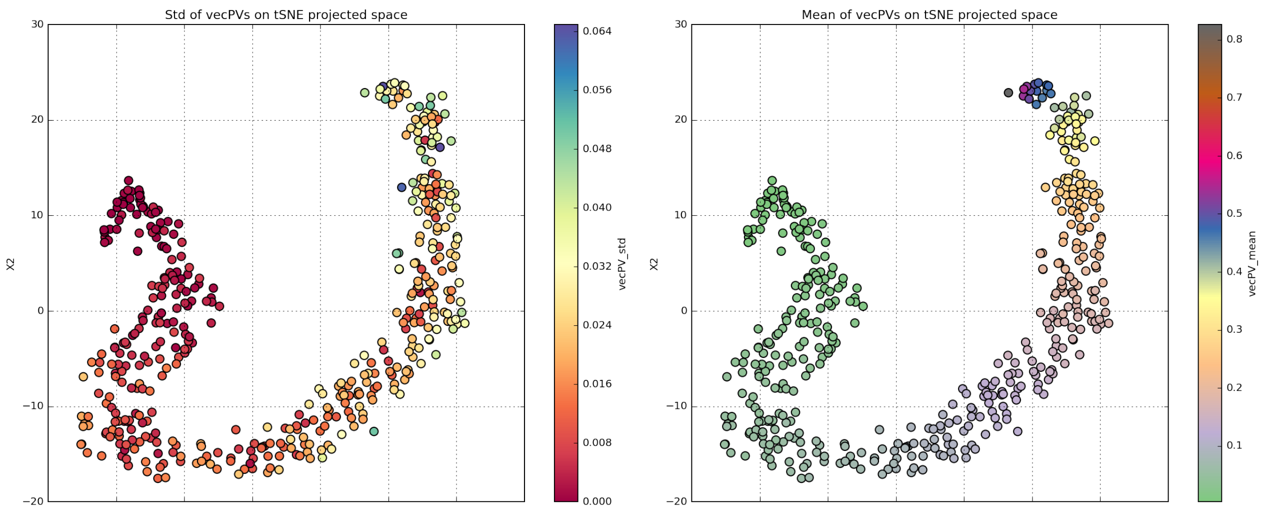


Figure 5: TSNE visulization on simulated p-values, colored by standard deviation and mean of forward p-values

Standard deviation of forward p-values represents the scale of daily risk profile change. The scatter points shapes like a band on the 2D plane. In figure (5), points with low forward p-value standard deviation are likely to be clustered on the center of the band, while higher standard deviation features the points at the side of the band. The mean of forward p-values indicates the possible true category of an exception. Low average value is likely to indicate a market move, while higher value suggests a model deficiency. The average of forward p-values follows an increasing trend from left to the right. This corresponds well with true category colors in figure (5). In conclusion, the results in figure (5) illustrate that tSNE has effectively clustered the exceptions by clustering nearby similar points.

We then fit the simulated data by an exploratory RBF kernel SVM model, to demonstrate its classification performance. We set the regularization parameter $C = 100$ in (12), and $\gamma = \frac{1}{K+2}$ in the kernel (13). Sampling 80% of the 464 data points as training dataset, the SVM model achieves AUC(area under ROC curve) of 99% on the remaining 20% testing datase, as is shown in figure (6).

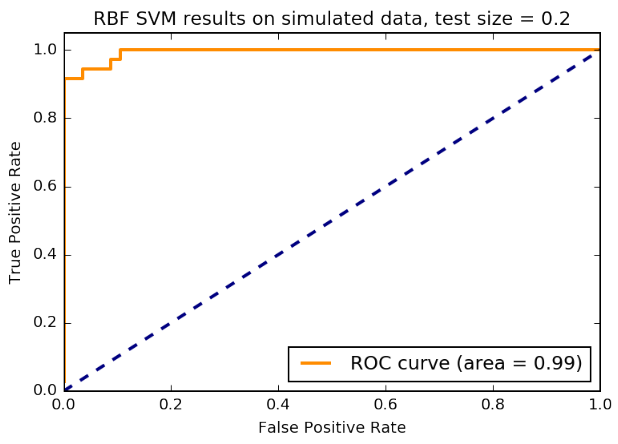


Figure 6: RBF kernel SVM model tested on simulated dataset, training size = 0.8

The confusion matrix on all data samples are included in figure (7). This confusion matrix reveals information about the overall performance of a classifier, where we are able to know the numbers of true-positives, false-positives, true-negatives and false-negatives. Meanwhile, the p-value scatter plots depicts the model's correspondence with the exact decision criteria in (8) and (9), if the $1-$forward p-value equals the true $varPV$. Here the shallow grey area represents 'Market Move' area if the horizontal axis were true $varPV$, with the deep grey representing 'Model Issue' area. In this case, the $1-$ forward p-value doesn't deviate significantly against true $varPV$, and the RBF kernel SVM is able to reproduce the true decision criteria on an artificially simulated dataset.

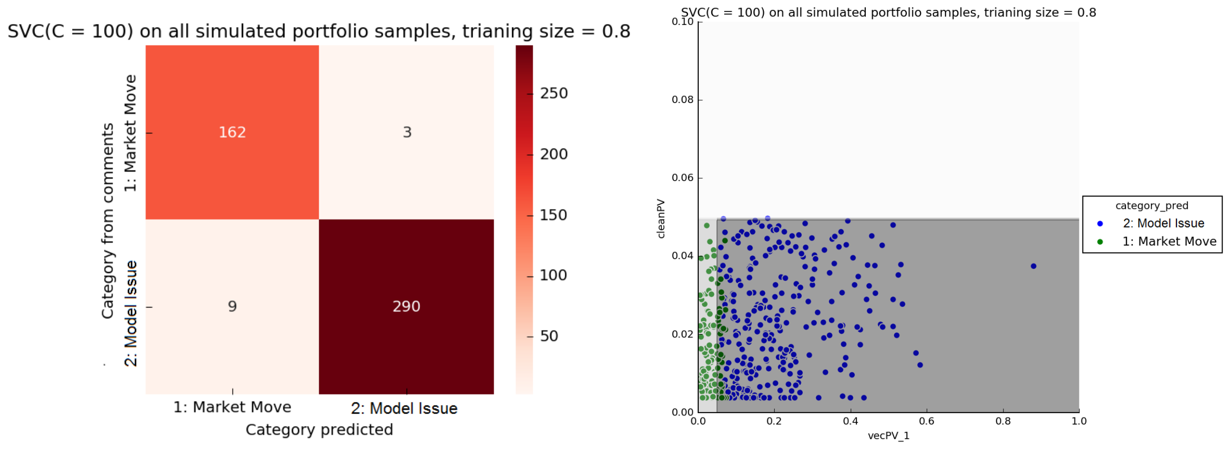


Figure 7: Left: RBF kernel SVM model confusion matrix on all sample points; Right: scatter plot of clean p-value v.s. 1−forward p-value

Finally we present in figure (8) the visualization results of two real datasets of VaR exceptions, respectively 99%-VaR and 95%-VaR. The market moves and model issues are clustered at different regions on the plane, in spite of several overlapping points. There are clear visual boundaries between the two clusters. These results further validates in addition to the parallel coordinates plot in figure (1) that the market moves and model issues are inherently distinguishable according to their vector p-values.

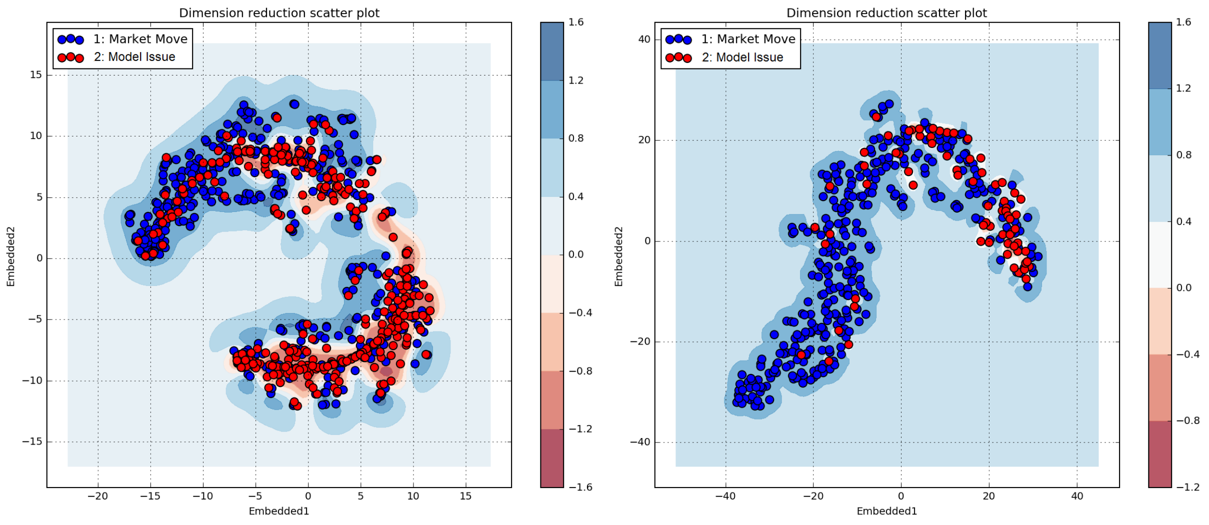


Figure 8: Left: 99%-VaR exceptions visualization result; Right: 95%-VaR exceptions visualization result;

4.3 Selection of Model Parameters by Cross Validation

In practice, we have exception data points from various portfolios, whose p-values are all available for the models to fit. We would like to choose among candidate models or parameters to achieve overall optimality as well as reducing overfitting.

For the given dataset, there might be several plausible models for classification. It is therefore important to select an optimal classifier for the proceeding analysis. Even for one specific classifier, different model parameters might lead to different performance of the model. 10-fold cross validation is used to select the optimal number of neighbors in KNN and the parameters C, γ in the kernel SVM. The whole dataset is divided randomly into 10 folds. At each time one fold is used as testing dataset, to test the model trained on the aggregation of remaining 9 folds. We calculate the average of the 10 accuracy scores on 10 testing folds, and the optimal model or optimal parameters can be selected by the highest average accuracy score.

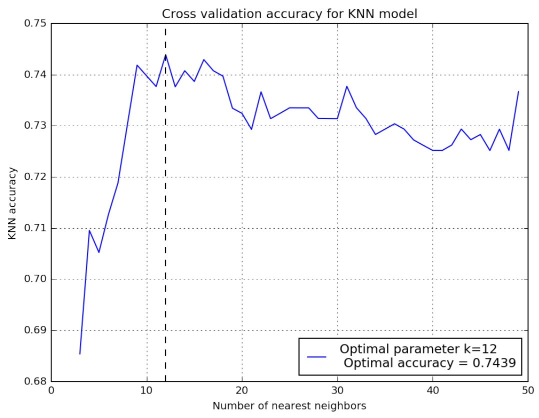


Figure 9: Cross validation results of KNN classifier

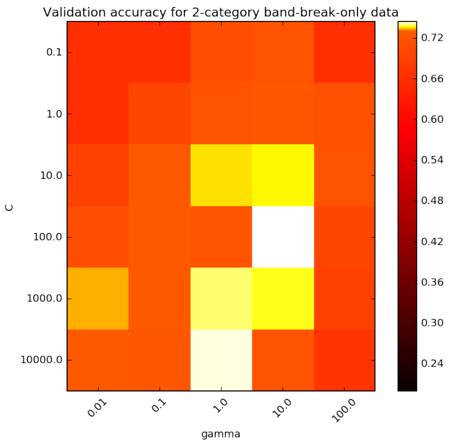


Figure 10: Cross validation results of RBF kernel SVM classifier

Both the KNN and SVM achieve optimal accuracy at around 74%. From figure (9) and figure (10) the optimal choice of nearest number for KNN is $k = 12$, while for SVM model, the optimal parameters are $C = 100, \gamma = 10$.

We can thus infer that the KNN and SVM are able to reduce overfitting by cross validation. Their mutually close accuracy score indicates consistently good performance. However KNN is strongly dependent on training data, as its decision function picks neighbors solely in training set. It doesn't learn any parameters as SVM does and the computation cost is high as the distances to all training samples are calculated. In conclusion, we will apply RBF kernel SVM with optimal parameters from cross validation for proceeding classification.

4.4 Classification by RBF Kernel SVM

Previous sections demonstrate the methodologies and some results on simulated dataset. We present the classification result using the optimal cross-validated parameters trained on real exceptions dataset. The real exception dataset consists of both 95% and 99% VaR exceptions of different portfolios. First we re-split the data into 60% training and 40% testing set, and output the confusion matrices and the ROC curves of the models on the testing dataset. These outputs compare the SVM model with optimal cross-validation parameters and the SVM model with default parameters. The optimal parameter admits lower false positive and false negative rates, which indicates that the optimal parameter successfully reduces overfitting effects.

	Predicted Market Move	Predicted Model Issue
Observed Market Move	246	51
Observed Model Issue	68	93

Table 1: Confusion matrix: SVM trained with optimal parameters (C=10, $\gamma = 10$)

	Predicted Market Move	Predicted Model Issue
Observed Market Move	242	65
Observed Model Issue	72	79

Table 2: Confusion matrix: SVM trained with default parameters (C=1, $\gamma =$ '$auto'$)

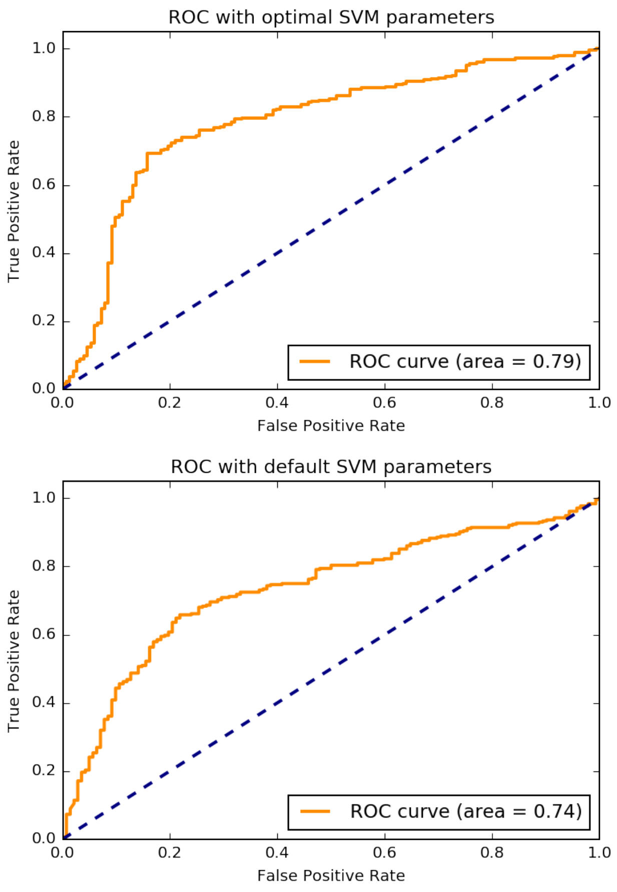


Figure 11: ROC curves of models by optimal and default parameters

Finally we train the SVM with optimal parameters on the whole dataset, and we obtain a ROC curve with area under curve being 0.90.

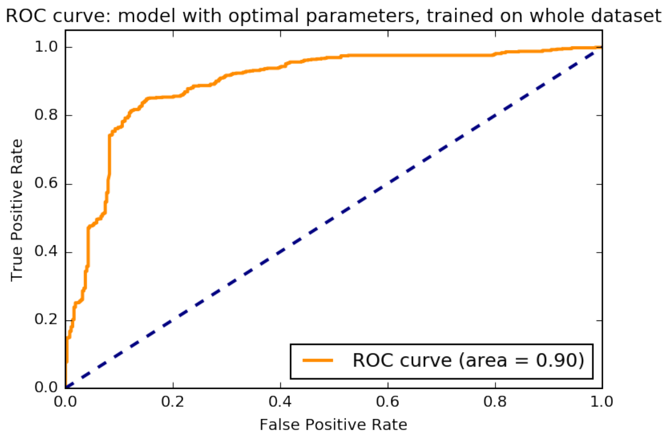


Figure 12: ROC curves of models by optimal parameters, trained on whole dataset

To better understand how the model performs on real exception data, we plot the classification confusion matrix and the scatter plot on all 99% VaR exceptions.

High accuracy rate is achieved by the model. Moreover from the scatter plot with $x-$axis being the first vector p-value, the model successfully predicts the exception points which do not obey exactly our p-value criterion. There is no clear boundary between predicted market moves and model issues, which indicates that the real datasets are for portfolios where the risk profile significantly changes. This can be directly observed from that many exception points have distinct values among their 5 vector p-values. However, our trained model has learned sufficiently from the risk change patterns, and when a new exception happens with some similar pattern as one in training dataset, the model could identify it and assign it into a category that follows the similar risk-changing pattern.

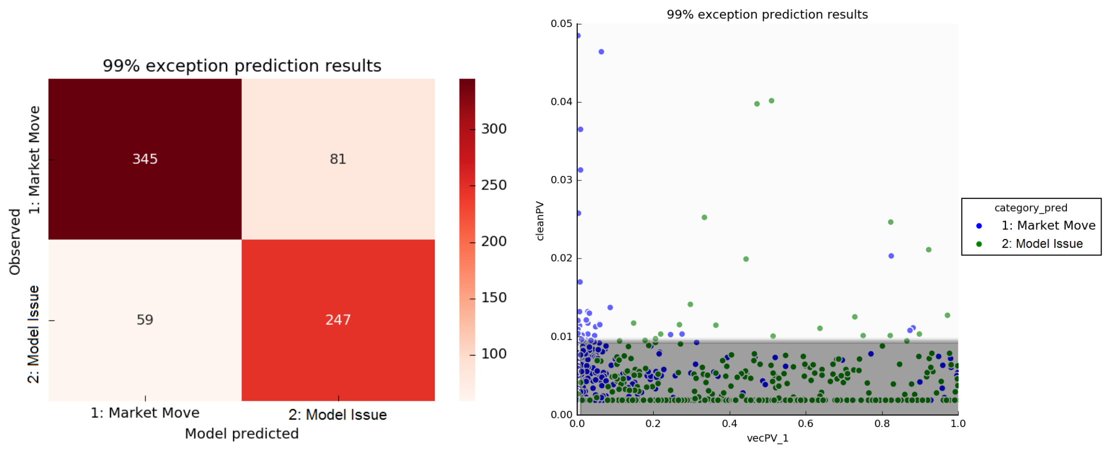


Figure 13: Prediction results on 99%-VaR exceptions

In order to further justify that the model is able to reproduce the exact classification criterion when the VaR p-value equals exactly all the 5 vector p-values, we validate the model on a simulated dataset containing over 2300 99%-exception samples with known simulated VaR p-values. All the samples have constant risk profile, so that their p-values equals their vector p-values. The scatter plot with coloured prediction results is shown in figure (14). We can see the vertical boundary between market moves and model issues. Although the boundary is not exactly 1% but around 6%, the exceptions are categorized according to a VaR p-value threshold. The difference might be due to a tolerance interval of the VaR p-value, in that when a VaR p-value is relatively low but above the VaR level, we still tend to classify it into market move. In conclusion, the SVM model is effective and we can use it for future predictions.

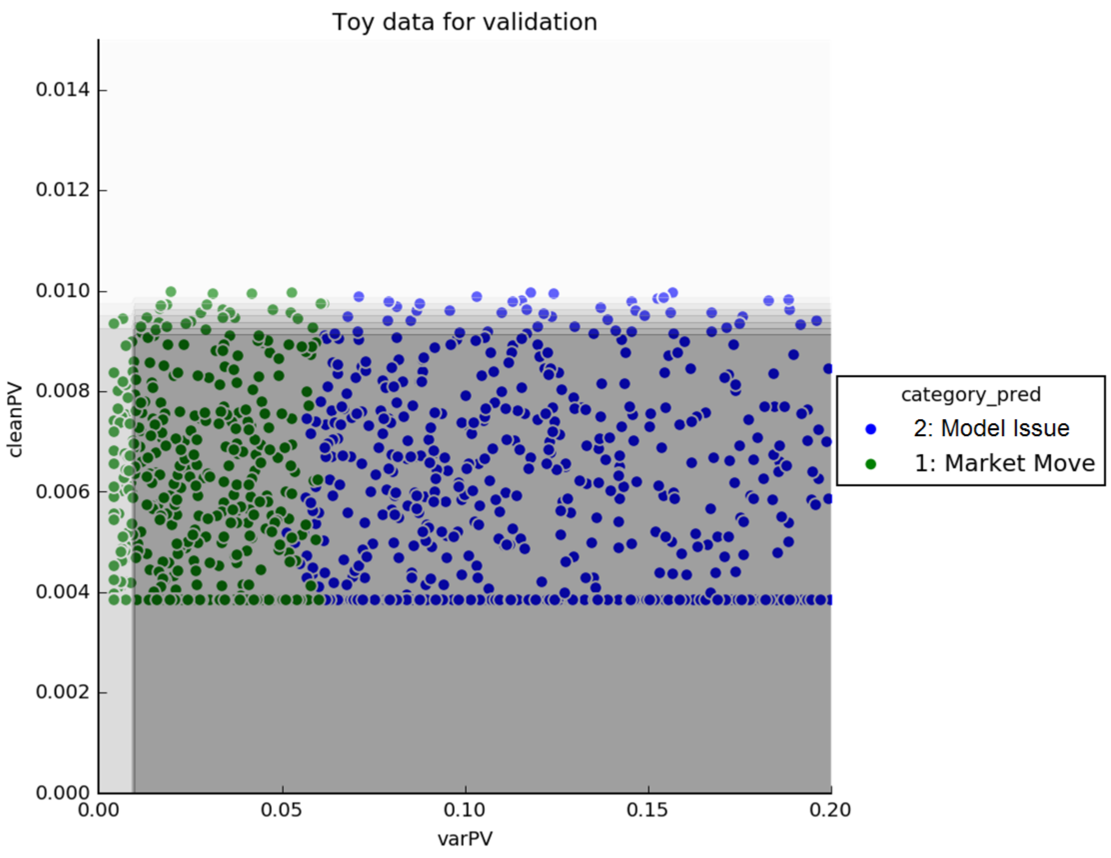


Figure 14: Prediction results on 99%-VaR exceptions

5 Alternative Approximation to VaR p-value: Numerical Implied P-values

Previously we explore the prediction power of SVM model featuring the forward vector p-values, this methodology is efficient as long as the VaR model generates a distribution vector in a daily basis, since vector p-values provide a good approximation to the VaR p-value. In this section, we will study a different approach to approximate the VaR p-value: implied VaR p-value method. The implied p-value is still in theoretical development, and this section will cover the idea and a basic convergence result.

5.1 Motivation

A perfect VaR risk model would generate a distribution vector that includes all the risk factors driving clean P&L. In this case, the clean P&L equals exactly the VaR P&L, and the unexplained P&L is 0. Recall the definitions of clean P&L and clean p-value in (2) and (5), if $(unxPL_{t+1} = 0, \forall t)$, then the clean p-value would only be dependent on the random variable return X_{t+1} plugged into the *i.i.d.*

random variables $X_t, X_{t-1}, ..., X_{t-T}$. Therefore, the clean p-value from a perfect VaR risk model should admit a uniform distribution, with the clean p-value CDF chart matching the uniform distribution CDF in figure (2). But in practice, the VaR risk model can be imperfect, and non-zero unexplained P&L causes the clean p-value CDF to deviate from the ideal straight line. The implied p-value method adjusts the clean p-value to match the ideal back-testing CDF to an implied p-value series, so that the implied p-value series passes the uniform distribution test, and also keeps high correlation with clean p-value series. The implied p-value $impPV_{t+1}$ is an approximation to the $varPV_{t+1}$ adjusted from the clean p-value. It also aims at fitting the unexplained P&L that minimizes the 'ideal v.s. fitted' VaR P&L error.

The problem is formulated as below:

For a given (clean p-value, distribution) sequence $\{(clnPV_{t+1}, D_t), t \in \mathcal{T}\}$, find a new p-value sequence $impPV = \{impPV_{t+1}, t \in \mathcal{T}\}$, such that $impPV$ follows a uniform distribution, and $(impPV_{t+1})$ is positively correlated to $(clnPV_{t+1})$.

From the formulation, one can write $impPV_{t+1}$ as a function of $(t, clnPV_{t+1})$. Then the implied unexplained P&L can also be written as a function of $(t, clnPV_{t+1})$, and we can express the implied unexplained P&L by a conditional expectation, as analogy to Dupire's local volatility:

$$\begin{aligned} unxPL_{imp}(t+1, clnPV_{t+1}) &= \mathbb{E}_t[unxPL_{true}(t+1)|clnPV_{t+1}] \\ &= clnPL_{t+1} - D_t^{-1}(impPV_{t+1}) \end{aligned}$$

To summarize, the implied p-value needs to have two main properties:

- $\{impPV_{t+1}, \forall t\}$ should pass the uniform distribution test, admitting an ideal back-testing chart.
- $\{impPV_{t+1}, \forall t\}$ and $\{clnPV_{t+1}, \forall t\}$ are 'highly correlated'.

5.2 Algorithm

The complete implied p-value series $(impPV_{t_1}, impPV_{t_2}, ..., impPV_{t_n})$ is bootstrapped from a pre-defined ordered uniform random series

$$(u_1, u_2, ..., u_M)$$

and the available clean p-value series

$$(clnPV_{t_1}, clnPV_{t_2}, ..., clnPV_{t_n})$$

The algorithm consists of n iterations where each iteration solves for one implied p-value.

1. Set the length M of the initial uniform random series. Then generate the random sequence $(u_1, u_2, ..., u_M)$.

2. Begin iterations from 1 to n. In iteration m, let $(\hat{PV}_i, i = 1, 2, ..., M+1)$ denote the ordered sequence consisting of previous $(m-1)$ calculated implied p-values

$$impPV_{t_1}, ..., impPV_{t_{m-1}}$$

and $(M - m + 1)$ random variables

$$u_m, ..., u_M$$

as well as x, the new $impPV_{t_m}$ to calculate. Then in iteration m, solve the optimization problem with the L_2-error between $(\hat{PV}_i, i = 1, 2, ..., M+1)$ and uniform quantiles $(\frac{k-0.5}{M+1}, k = 1, ..., M+1)$, plus a L_2-penalization of $(x - clnPV_{t_m})^2$;

$$impPV_{t_m} = \text{argmin}_{x \in [0,1]} \sum_{i=1}^{M+1} \left(\hat{PV}_i - \frac{i-0.5}{M+1} \right)^2 + \lambda (x - clnPV_{t_m})^2 \quad (15)$$

λ is the historical penalization constant. it penalizes the deviation of implied p-value from corresponding clean p-value. Large λ makes implied p-value closer to clean p-value, and a small λ will admit implied p-value series more like uniform distribution.

5.3 Numerical Examples

A simulated example on implied p-value is presented in figure (15). The green curve is the implied p-value empirical CDF, and its back-testing result is close to that of true VaR p-value, compared to the clean p-value. A comparison in daily basis between implied and VaR p-value is shown in figure (16). In this simulation case, the clean-implied correlation is above 90%, and we see from figure (16) that the implied P&L aligns with the VaR P&L with correlation= 0.56, at least the implied P&L captures the downside and upside peaks of the VaR P&L.

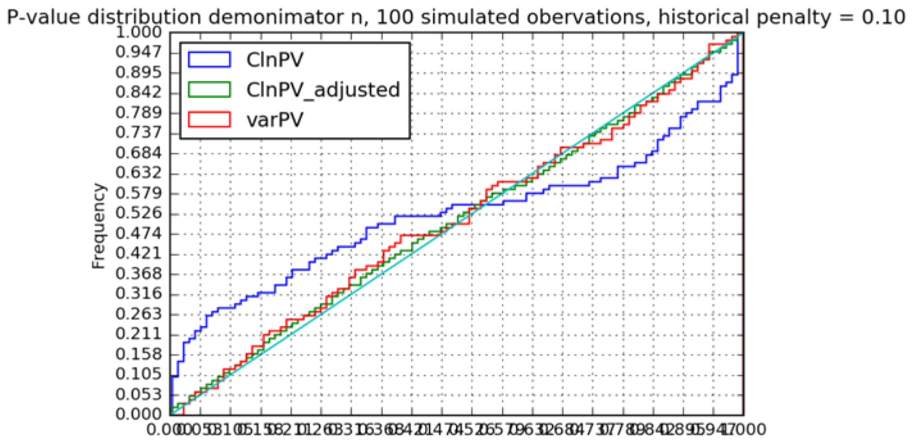


Figure 15: Back-testing chart of implied p-value and clean p-value

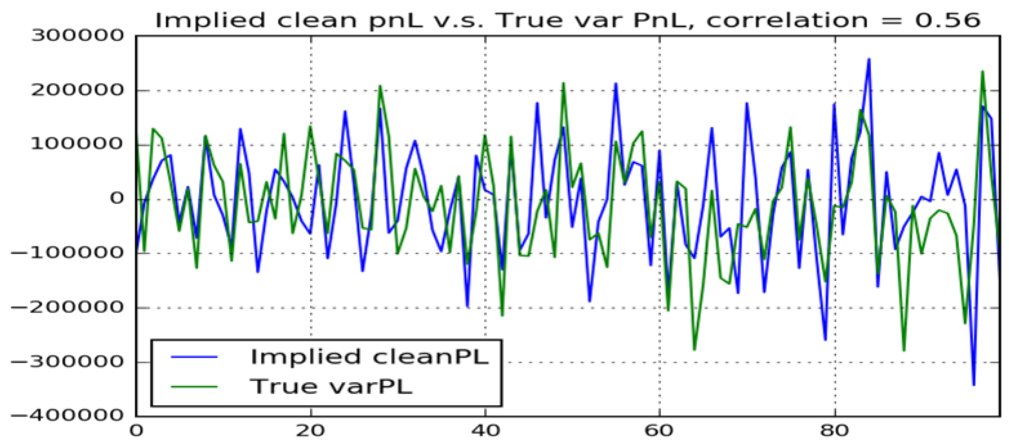


Figure 16: Time series plot of VaR p-value and implied p-value

Another simulation example features two cases, where the clean P&L experiences considerable shock due to large out-of-risk-model factors' movements. In figure (17), the risk out of model drives an important loss, while in figure (18), increased volatility is observed during the shock period. We apply the implied p-value approach on clean p-value series, then calculate unexplained P&L by $unxPL^{imp}_{t+1} = clnPL_{t+1} - D^{-1}_t(impPV_{t+1})$ and compare it with true unexplained P&L. These results demonstrate that implied p-value method is able to capture large downside movements and sudden risk volatility variations in unexplained P&L.

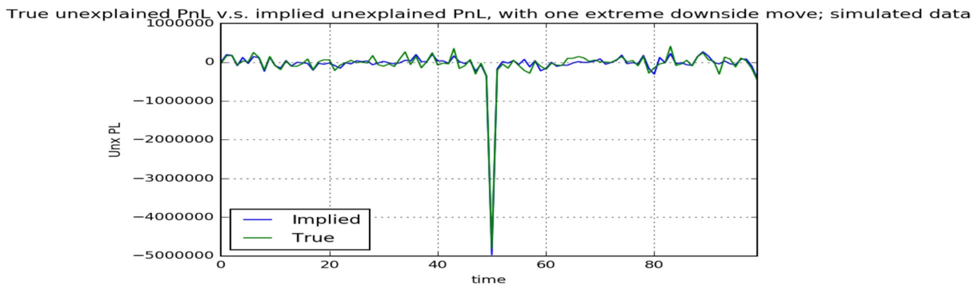


Figure 17: Unexplained P&L: large downside movement

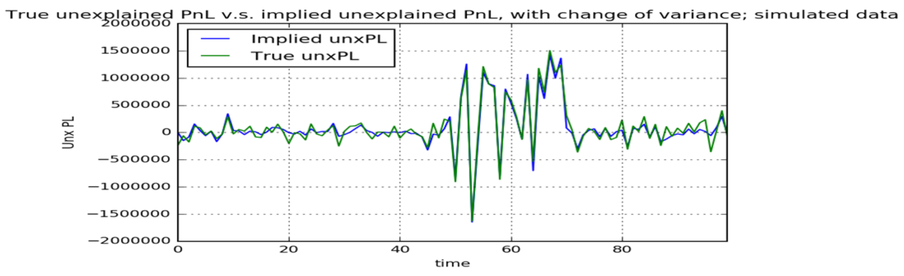


Figure 18: Unexplained P&L: large volatility change

Implied p-value approach also applies to P&L attribution, when the parent portfolio's clean P&L is decomposed to several component portfolios' clean P&L. In this example, the parent portfolio Π consists of portfolios A and B. We assume the risk model captures all the risk of A, but B has a large part of unexplained P&L missed by the model. More precisely we write:

$$\begin{aligned}
&X_A \sim N(0,1), X_B \sim N(0,1), \epsilon_B \sim N(0,3) \\
&clnPL_\Pi = clnPL_A + clnPL_B \\
&clnPL_A = X_A \\
&clnPL_B = X_B + \epsilon_B
\end{aligned}$$

First bootstrap implied p-value from Π's clean p-value and plot $unxPL_\Pi^{imp}$ against $unxPL_A^{real} + unxPL_B^{real}$, we can see that the scatter points are situated alongside the straight line, which reveals the compatibility of implied P&L attribution with the unexplained P&Ls.

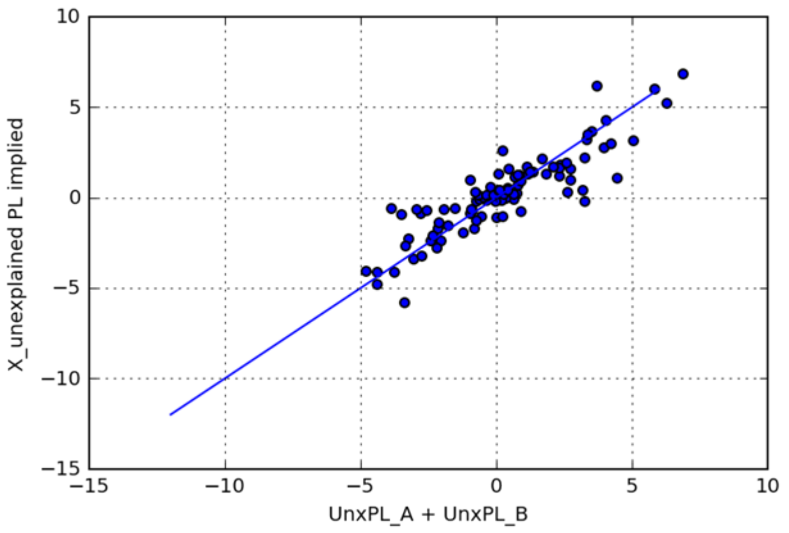


Figure 19: Scatter plot: portfolio Π's implied unexplained P&L against sum of components real unexplained P&L

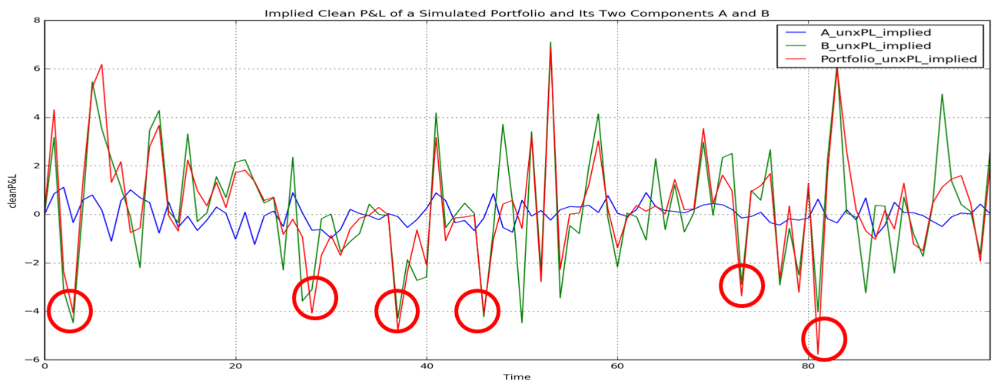


Figure 20: Time series: implied unexplained P&L of Π, A and B

Next we apply the implied p-value approach to A and B, then plot the time series of implied unexplained P&L in figure (20). The downside movements of $unxPL_{\Pi}^{imp}$ and $unxPL_{B}^{imp}$ coincide, especially when the movements have a large magnitude. Therefore we can conclude that implied p-value approach explains portfolio Π's unexplained P&L comes mainly from B's unexplained P&L.

The implied p-value approach has demonstrated its efficiency on simple simulated examples. It uses only a few assumptions in the risk model, and the implied p-values are bootstrapped solely from clean p-values, which is simple to implement, as long as the clean p-value series and daily distributions are available. The theoretical properties of implied p-value are still in research, and we will prove a convergence result in next subsection.

5.4 Convergence

Since implied p-values approximate the real VaR p-values and they are bootstrapped from a M-length uniform random sequence, one would like to ask when using two different uniform random series initially, how close the two implied p-value sequences will be. The materiality of the implied p-values will be validated if they converge to a unique p-value series at a reasonable computation complexity, when $M \to \infty$. We will see later that the converged p-value sequence are likely to be no other than the clean p-values. We first define the asymptotic property.

Definition 5.1 (l_2-consistency)**.** Let C be a clean p-value series of fixed length n. $u_M^{(1)}$ and $u_M^{(2)}$ are two initial uniform random series of length M, where $u_M^{(1)}$ and $u_M^{(2)}$ are generated independently. $X_M^{(1)}$ and $X_M^{(2)}$ are 2 implied p-value bootstrapped from C initialized respectively by $u_M^{(1)}$ and $u_M^{(2)}$ with penalization constant λ. Then the implied p-value algorithm is consistent, if there exists a function $f_\lambda(M)$, such that

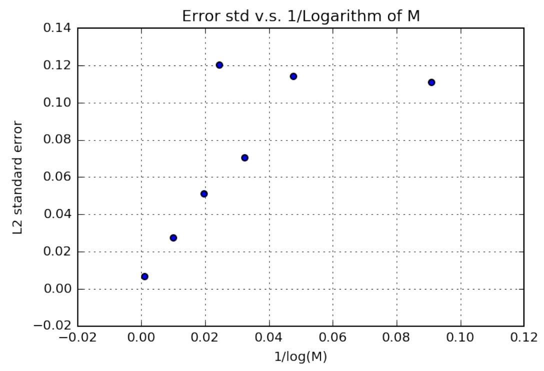
$$\lim_{M\to\infty} f_\lambda(M) = 0$$

and

$$\mathbb{E}_{l_2}\left[\left|X_M^{(1)} - X_M^{(2)}\right|^2\right] \leq f_\lambda(M) Var(u_M^{(1)} - u_M^{(2)})$$

We demonstrate the convergence property by reasoning: When M increases to very large values, the random initializer u_M becomes denser in interval $[0, 1]$. In the objective function (15) the first term will have less impact on its value as the sorted $\hat{PV}_i, i = 1, ..., M+1$ is 'close' to $\frac{i-0.5}{M+1}, i = 1, ..., M+1$. In this case, to minimize the objective the second term $\lambda(x - clnPV_{t_m})$ is more important, and x is more likely to be close to $clnPV_{t_m}$. Therefore the implied p-value series tends to converge to the clean p-value series.

Two numerical examples are shown in figure (21), with l_2-error between implied and clean p-value sequences against the functions $f_\lambda(M) = \frac{1}{\log(M)^2}$ and $f_\lambda(M) = \frac{1}{M}$. As $f_\lambda(M)$ approaches 0, we observe that the l_2-error tends to 0 as well.



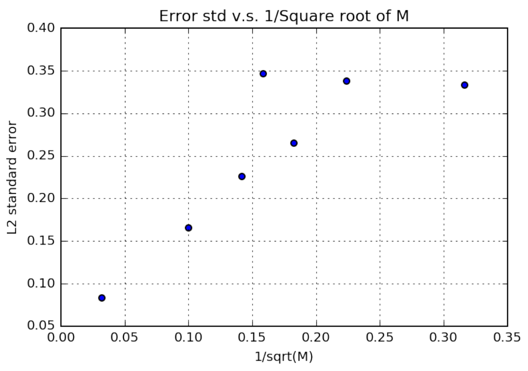


Figure 21: l_2-error between implied/clean p-value against $\frac{1}{\log(M)}$ and $\frac{1}{\sqrt{M}}$

We state the following proposition on the convergence property of implied p-values.

Proposition 5.1 (l_2-convergence to clean p-values)**.** *Let C be a clean p-value series of fixed length n. u_M is an uniform random series of length M. X_M is the implied p-value bootstrapped from C initialized by u_M with penalization constant λ. Then there exists a function $f_\lambda(M)$, such that*

$$\lim_{M\to\infty} f_\lambda(M) = 0$$

and

$$\mathbb{E}_{l_2}\left[|X_M - C|^2\right] \leq f_\lambda(M)$$

The randomized property of the initializer u_M brings about much complexity, we will begin by a special case, where u_M is fixed as discrete uniform sequence on $[0, 1]$.

Proposition 5.2. *Assumptions are the same with proposition 5.1, and u_M is fixed as*

$$u_M(i) = \frac{i - 0.5}{M}, \forall i \in \{1, 2, ..., M\}$$

Then the implied p-value sequence converges to clean p-values with $f_\lambda(M) = \frac{C_n}{M^2}$, where C_n is a constant in function of the p-values length n.

Proof. Since the length of clean p-values is fixed by n, it suffices to prove convergence of the first implied p-value

$$impPV_{t_1} \xrightarrow[M\to\infty]{} clnPV_{t_1}$$

and the convergence for the rest of implied p-values follows.

In the objective function (15), let i_1 be the index such that:

$$\frac{i_1 - 0.5}{M+1} \leq clnPV_{t_1} < \frac{i_1 + 0.5}{M+1}$$

then we have

$$M \cdot clnPV_{t_1} - 0.5 < i_1 \leq (M+1) \cdot clnPV_{t_1} + 0.5$$

Assume the interval for x:

$$u_M(i_x) = \frac{i_x - 0.5}{M} \leq x < \frac{i_x + 0.5}{M} = u_M(i_x + 1)$$

we can write the objective function $g(x)$:

$$g(x) = \sum_{i=1}^{M+1} \left(\hat{PV}_i - \frac{i - 0.5}{M+1} \right)^2 + \lambda (x - clnPV_{t_1})^2$$

$$= \sum_{i=1}^{i_x} \left(\frac{i-0.5}{M} - \frac{i-0.5}{M+1} \right)^2 + (x - \frac{i_x + 0.5}{M+1})^2 + \sum_{j=i_x+1}^{M} \left(\frac{j-0.5}{M} - \frac{j+0.5}{M+1} \right)^2 + \lambda (x - clnPV_{t_1})^2$$

$$= \sum_{i=1}^{i_x} \frac{(i-0.5)^2}{(M(M+1))^2} + (x - \frac{i_x+0.5}{M+1})^2 + \sum_{j=i_x+1}^{M} \frac{(M+0.5-j)^2}{(M(M+1))^2} + \lambda (x - clnPV_{t_1})^2$$

$$= \frac{\sum_{j=1}^{i_x} (j-0.5)^2 + \sum_{j=1}^{M-i_x} (j-0.5)^2}{M^2(M+1)^2} + (x - \frac{i_x+0.5}{M+1})^2 + \lambda (x - clnPV_{t_1})^2$$

$$= G(i_x) + (x - \frac{i_x+0.5}{M+1})^2 + \lambda (x - clnPV_{t_1})^2$$

Then we estimate that $G(i_x)$ has the order $O(\frac{1}{M})$. As

$$\sum_{j=1}^{[\frac{M}{2}]} (j-0.5)^2 \leq \sum_{j=1}^{i_x} (j-0.5)^2 + \sum_{j=1}^{M-i_x} (j-0.5)^2 \leq 2 \sum_{j=1}^{M} (j-0.5)^2$$

Then

$$\frac{\sum_{j=1}^{[\frac{M}{2}]} (j-0.5)^2}{M^2(M+1)^2} \leq G(i_x) \leq 2 \frac{\sum_{j=1}^{M} (j-0.5)^2}{M^2(M+1)^2}$$

By the square sum formula, we obtain that $G(i_x) \sim O(\frac{1}{M})$. As for $g(x)$, if x is restricted on $[\frac{i_x-0.5}{M}, \frac{i_x+0.5}{M}]$, then the minimum is achieved on the following values:

$$x^* = \frac{\frac{i_x+0.5}{M+1} + \lambda clnPV_{t_1}}{1+\lambda}, \frac{i_x - 0.5}{M}, \text{or } \frac{i_x+0.5}{M}$$

depending on the position of x^* and the interval. At x^*, the objective value is

$$g(x^*) = G(i_x) + \left[\lambda + \frac{\lambda^2}{(1+\lambda)^2} \right] \left(clnPV_{t_1} - \frac{i_x + 0.5}{M+1} \right)^2$$

where $G(i_x)$ can be considered as a constant in the same order as $O(\frac{1}{M})$. Then the minimum of g is achieved when $i_x = i_1$ or $i_x = i_1 - 1$, in which $clnPV_{t_1}$ and x^* are closest, thus minimizing the error. Then let's assume $i_x = i_1$, then

$$x^* = \frac{\frac{i_1+0.5}{M+1} + \lambda clnPV_{t_1}}{1+\lambda} \in [\frac{i_1 - 0.5}{M+1}, \frac{i_1 + 0.5}{M+1})$$

Then x^* is a global minimum for the objective function g on $[0, 1]$. Therefore

$$impPV_{t_1} = \frac{\frac{i_1+0.5}{M+1} + \lambda clnPV_{t_1}}{1+\lambda}$$

Using the restrictions on i_1 based on $clnPV_{t_1}$, we finally have

$$clnPV_{t_1} + \frac{0.5 - clnPV_{t_1}}{(1+\lambda)(M+1)} < impPV_{t_1} \leq clnPV_{t_1} + \frac{1.5}{(1+\lambda)(M+1)}$$

Then $|clnPV_{t_1} - impPV_{t_1}| \sim O(\frac{1}{M})$.

Since clean p-value sequence has a limited length n, using the same reasoning above, we will finally have

$$||impPV - clnPV||_{l_2}^2 \sim O(\frac{1}{M^2})$$

$\square$

The proposition 5.1 involves uniform random initial sequence u_M, which still requires a rigorous proof for convergence. But given the numerical results and observation, we believe that the proposition 5.1 holds, and *impPV* converges finally to *clnPV* in l_2 as the length of initializer tends to ∞.

To summarize, the implied p-value method provides a numerical perspective on observed clean p-values. The implied p-value converges eventually to the clean p-value, instead of VaR p-value. So the effect of implied p-value is similar to allowing a random adjustment on clean p-value for it to be uniformly distributed. We can also use the bootstrapped implied p-value to classify VaR exceptions, in which case we do not use any prediction models.

6 P&L Attribution by Lasso Regression

When a VaR exception occurs, it is important to understand the root cause. A first step in doing so, is the analysis of the portfolio's P&L attribution to its components. Lasso regression is a useful tool to select most important P&L contributors based on historical data. The learned contributors can provide a confidence level for the P&L attribution when an exception happens. Suppose that the portfolio Π_0 consists of n sub-portfolios $\Pi_1, ..., \Pi_n$, which yields

$$clnPL_t^{\Pi_0} = \sum_{i=1}^{n} clnPL_t^{\Pi_i} + clnPL_t^{res} \tag{16}$$

$clnPL_t^{res}$ is the residual P&L (if the clean P&L of parent portfolios may differ from the sum of its components' P&L). Then their VaR P&L and unexplained P&L should also follow the equations, when we use the same risk model for all portfolios:

$$varPL_t^{\Pi_0} = \sum_{i=1}^{n} varPL_t^{\Pi_i} \tag{17}$$

$$unxPL_t^{\Pi_0} = \sum_{i=1}^{n} unxPL_t^{\Pi_i} + unxPL_t^{res} \tag{18}$$

The $varPL$ is based on estimated $varPV$, which can either be approximated by taking the average of forward vector p-values, or by bootstrapping directly implied p-values of Π_0 $impPV^{\Pi_0}$, and then plugging this implied p-value into joint vector distributions of component portfolios to select the marginal p-values. Concretely for the implied approach, in the joint distribution vectors

$$(vecPL_t^{\Pi_0}, vecPL_t^{\Pi_1}, ..., vecPL_t^{\Pi_n}), \forall t$$

Find the t_0 such that $P\&L_{t_0}^{\Pi_0}$ yields the closest value to $impPV^{\Pi_0}$. Then the components' marginal implied p-values are computed using $\{P\&L_{t_0}^{\Pi_i}, i \geq 1\}$

The training data includes all historical clean P&L vectors with Π_0 and its n sub-portfolios.

$$(clnPL_t^{\Pi_0}, clnPL_t^{\Pi_1}, ..., clnPL_t^{\Pi_n}), \forall t$$

6.1 Coefficient Path from Lasso Regression

In a Lasso regression as we are trying to select the most relevant portfolios, we do not restrict coefficients to be 1 as in equation (16). Our regression cost function is the following:

$$\min_{\beta} ||clnPL^{\Pi_0} - \sum_{i=1}^{n} clnPL^{\Pi_i} \cdot \beta_i||^2 + \alpha \sum_{j=1}^{n} |\beta_j| \tag{19}$$

As α increases from 0, more β_i tends to be 0. We plot the path graph in which the value evolution of each feature is represented by a curve. Figure (22) is an example of Lasso regression applied on a simulated portfolio's $clnPL$. The path graph shows how the feature coefficients evolve as the regularization parameter α changes. When α remains at a lower level ($< 10^7$), all the sub-portfolios have coefficients to be 1, which matches the unitary sum equation in (16). But as α grows, more coefficient curves drop to 0, and remaining non-zero coefficients belong to the most relevant contributor sub-portfolios.

We also run Lasso on estimated $varPL$ and $unxPL$ respectively so that the VaR P&L and unexplained P&L are also attributed to their contributors.

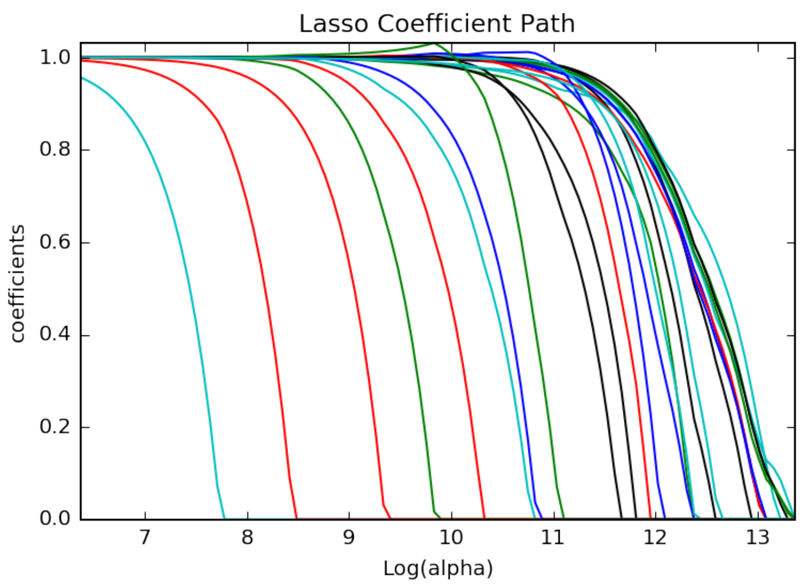


Figure 22: Lasso coefficients path on a simulated portfolio

6.2 Variance Attribution

Lasso regression provides helpful insights into the importance hierarchy of sub-portfolios. We can further quantify this hierarchy by linear regression re-fitting stepwise the sub-portfolios.

Suppose we denote $\Pi_{(1)}$ to $\Pi_{(n)}$ the sorted sub-portfolios by their vanishing order in Lasso regression, where $\Pi_{(1)}$ is the last sub-portfolio whose coefficient vanishes as α increases. Then we describe the attribution process by a for-loop as below.

For $i = 1$ to n :

- Fit $clnPL_{\Pi_0}$ in function of $(clnPL_{\Pi_{(1)}}, ..., clnPL_{\Pi_{(i)}})$ by linear regression.

$$cln\hat{P}L_{\Pi_0} = \text{LinearRegression}(clnPL_{\Pi_{(1)}}, ..., clnPL_{\Pi_{(i)}})$$

- Compute the residual vector $\hat{\epsilon_{(i)}} = clnPL_{\Pi_0} - cln\hat{P}L_{\Pi_0}$
- The score is defined by the residual variance divided by the variance of total portfolio Π_0

$$s_{(i)} = \frac{\text{var}(\epsilon_{(i)})}{\text{var}(clnPL_{\Pi_0})}$$

Therefore $1 - s_{(i)}$ is the R^2 score on parent portfolio Π_0 explained by sub-portfolios $(\Pi_{(1)}, ..., \Pi_{(i)})$. The bar plot of scores reveals information on quantified contribution from each sub-portfolios sorted by Lasso regression hierarchy. With $s_{(0)} = 1$, $s_{(i)} - s_{(i+1)}, i \geq 0$ represents the residual variance reduction effect with the introduction of portfolio $\Pi_{(i+1)}$.

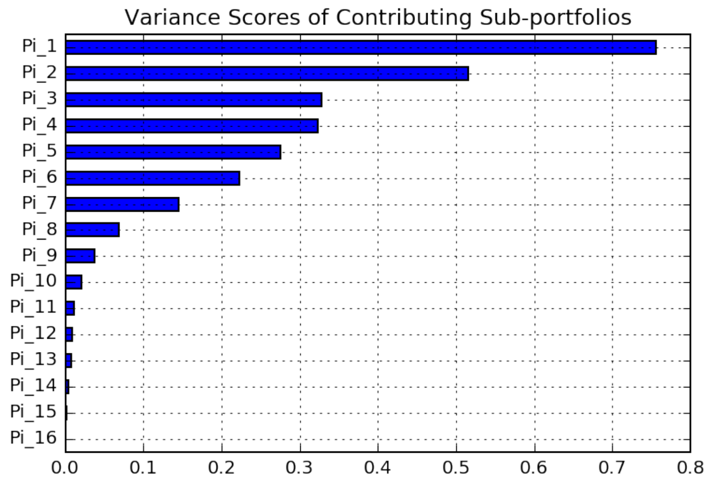


Figure 23: Variance scores sorted according to Lasso hierarchy of sub-portfolios

This analysis is based on a global scope where the model is fitted by historical P&L data frame. It provides useful references for sample-wise VaR exception P&L attribution. One day when a VaR exception happens in portfolio Π_0, we can attribute $varPL_{\Pi_0}$ and $unxPL_{\Pi_0}$ by equations (17) and (18). They are attributed in a single daily basis, where the $varPL$s are estimated for this single day. The global statistical attribution analysis reveals more empirical information on how consistent the exception day's P&L attribution aligns with respect to historical attribution.

7 Conclusion

In this report, we have presented applications of machine learning to classify VaR exceptions. Along with an introduction to the VaR risk model framework, we have visualized p-values data by tSNE and then built a RBF kernel SVM model whose parameters are obtained through cross validation. The prediction power of SVM is validated by the fact that it reproduces the exact classification rule on ideally simulated dataset. Aside from learning from p-values data, the implied p-value method is proposed to numerically bootstrap approximated VaR p-values. We have proved the convergence result in a special case for this method. We have further attributed P&L by Lasso regression, which provides statistical insights for exceptions' P&L attribution.

In the end, we list potential topics for further research:

- Multi-category classification model: Currently we have market move and model issues as the only two categories, but more categories can be added. For instance improper usage of a proxy time series is also a source of exceptions. This would involve more features and more complicated models.

- Direct approximation on the VaR P&L: Regression models could be used to fit directly the VaR P&L, in which case the exact category would be available and one would not need a classification model based on p-values.
- Completion of theoretical framework for implied p-value method: We proved the asymptotic convergence of implied p-values to clean p-values in a special case, where initial uniform series is fixed. In more general case with randomized initializer, we believe in the same convergence but still need to develop a proof for proposition 5.1.

References

[1] Jeremy Berkowitz. Testing density forecasts, with applications to risk management. *Journal of Business & Economic Statistics*, 19(4):465–474, 2001. doi: 10.1198/07350010152596718. URL https://doi.org/10.1198/07350010152596718.

[2] Sean D. Campbell. A review of backtesting and backtesting procedures. Finance and Economics Discussion Series 2005-21, Board of Governors of the Federal Reserve System (U.S.), 2005. URL https://EconPapers.repec.org/RePEc:fip:fedgfe:2005-21.

[3] Peter F. Christoffersen. Evaluating interval forecasts. *International Economic Review*, 39(4):841–862, 1998. ISSN 00206598, 14682354. URL http://www.jstor.org/stable/2527341.

[4] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, Sep 1995. ISSN 1573-0565. doi: 10.1007/BF00994018. URL https://doi.org/10.1007/BF00994018.

[5] Francis X Diebold, Todd A Gunther, and Anthony S Tay. Evaluating density forecasts. Working Paper 215, National Bureau of Economic Research, October 1997. URL http://www.nber.org/papers/t0215.

[6] William Karush. Minima of functions of several variables with inequalities as side conditions. Master's thesis, Department of Mathematics, University of Chicago, Chicago, IL, USA, 1939.

[7] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, pages 1137–1143, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-363-8. URL http://dl.acm.org/citation.cfm?id=1643031.1643047.

[8] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, Berkeley, Calif., 1951. University of California Press. URL https://projecteuclid.org/euclid.bsmsp/1200500249.

[9] Paul Kupiec. Techniques for verifying the accuracy of risk measurement models. Finance and Economics Discussion Series 95-24, Board of Governors of the Federal Reserve System (U.S.), 1995. URL https://EconPapers.repec.org/RePEc:fip:fedgfe:95-24.

[10] Murray Rosenblatt. Remarks on a multivariate transformation. *Ann. Math. Statist.*, 23(3):470–472, 09 1952. doi: 10.1214/aoms/1177729394. URL https://doi.org/10.1214/aoms/1177729394.

[11] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

[12] Laurens van der Maaten and Geoffrey E. Hinton. Visualizing data using t-sne. 2008. URL `https://lvdmaaten.github.io/publications/papers/JMLR_2008.pdf`.