Hedging Optimisation with K-means Clustering on Cryptocurrencies

Chengkai Lu

Submitted for the Degree of Master of Science in

Data Science and Analytics with a Year in Industry

Department of Computer Science
Royal Holloway University of London
Egham, Surrey TW20 0EX, UK

August 11, 2018

Declaration

This report has been prepared on the basis of my own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Word Count:

Student Name:

Date of Submission:

Signature:

Abstract

Cryptocurrencies like Bitcoin and Ripple are becoming popular in these years. They can be obtained through mining or transactions. Every individual cryptocurrency can be transferred directly using a public key in digital wallets. Their price is mostly tied to supply/demand and hard to be interfered by governments. In addition, cryptocurrencies have some dependencies because some of them like Dogecoin can only be bought by some major cryptocurrencies such as Bitcoin and Ethereum in cryptocurrency exchanges.

Value at Risk is a popular traditional method in financial technical analysis for estimating potential total risks of a financial portfolio. A well-estimated risk can prevent an investor or a financial institution from losing more than their expectation. In reality, the financial markets sometimes are unpredictable. Investors are striving for putting all the risks under their control.

K-means clustering is a popular unsupervised learning algorithm for grouping unlabelled data. It aims on finding the natural way of separating observations into diffident clusters and is quite popular in the area of marketing for discovering potential customer behaviour. However, it might be also applicable to group stocks/cryptocurrencies having similar price movement together, which means a clustering algorithm may help diversify investors' investment. It states that a systematic way of hedging is achievable.

A modern machine learning algorithm for diversifying investment before a traditional technical analysis might improve the performance of risk estimation. A better hedging can prevent investors from suffering disastrous loss. Especially in the market of cryptocurrency, the price movement is more irrational and abnormal because of lacking regulation. This project will try to research if the optimisation of risk estimation performs well in the market of cryptocurrency.

Contents

List of Figures

List of Tables

Acknowledgement

1 Introduction

Hedging is an important investing strategy to avoid risks that will result in substantial losses or gains. Investors have been developing and implementing traditional technical analysis on evaluating risks of an investment, but it is still tricky to capture some events such as a financial crash which breaks the previous risk estimation. Some of the tasks are computationally expensive and human beings are hard to achieve, therefore, computer science and machine learning has been used for improving the performance of analysis.

Machine learning has become a popular area which focuses on enabling machines or computers to learn and discover the patterns. It can be applied to different domains as long as there exists data. Especially in the financial area, people are striving for using modern computing power to deal with problems efficiently because time is a crucial factor in finance, and this is called computational finance.

1.1 Motivation

When we talk about the risk of an investment, we are normally curious about the risk for the future. Prediction of risks can be tricky because a randomly unpredictable event will impact on the market movement. An unexpected financial event might collapse the market and result in a crash. All the products in the market will be influenced by the risks. These kinds of risks are called *Systematic Risk*, which cannot be avoided through a diversified investment portfolio. However, we can spread the risks over diffirent investments by generating a well-diversified investment portfolio.

Traditional risk management methods have been developed for centuries and they can produce reasonable result of hedges, but there still exists possibilities to improve the performance. Machine learning are also used for avoidance of risk or market prediction. We want to seek for a way of improving traditional risk management methods by applying machine learning algorithms on top.

1.2 Aims And Objectives

Value at Risk is a traditional method which can evaluate the risk of a investment portfolio or a single investment. It is widely used by a large number of financial institutions, especially banks because it provides a quantitative criterion for risk estimation. However, the risk might be overestimated or underestimated since the products in a portfolio might have varied volatil-

ities and activities. There also might be some events that have happened on one product, and another product might experience it in the future. For example, we might apply VaR of the combination of two products with similar activities and cover the worst/best cases of a product. Another product will then have the ability to consider the possible worst cases.

A diversification of investment before generating a portfolio can remove the concern of wrongly evaluating the risks. Well-diversified financial portfolios will ensure that products in a investment combination have similar risks or volatilities.

K-means clustering is an appropriate method for diversifying financial investments. It can create natural grouping on the financial products by putting those with similar price volatilities together. That is, it looks at the shape of each product along the timeline and estimates the similarity between two lines to decide if the two products should be in the same cluster.

This project aims on seeing if K-means clustering can diversify financial products with their daily volatilities and improve the result of risk estimation on cryptocurrencies. In addition, it will examine if the effect on cryptocurrencies is different from that on stocks.

1.3 Project Structure

The project includes several parts, experiment, implementation of the analysis and report. The experiment is for developing and adjusting the analysis. After receiving a satisfactory result from the experiment, the analysis will be implemented on a minimal production-level basis. This report will then present the most critical detail of the analysis.

1.3.1 Technologies

In order to facilitate the process of analysis, we use several technologies to develop this project. Scalability is also important for a proper project. As a result, we will build a well-structured programme that meet the minimal requirement of scalability and can be productionised in the future.

In this project, the following technologies are used for different purposes:

- Programming Languages
  - *Python* : A popular programming language for statistics, machine learning and data analysis due to its flexibility, powerful packages, ease of learning and data visualisation. We use Python for most of the analysis because it is handy for quick experiments.

  - *Java* : Many companies are using Java as the main languages for their servers and applications because of its efficiency, performance and stability. We use Java for the ETL pipeline and a preparation of productionising the result of analysis.

- Database
  - *PostgreSQL* : We use PostgreSQL as our database because it is well-developed and supports efficient quaries.

- Containerisation
  - *Docker* : An open source operating-system-level virtualisation tool to build an environment for running programmes. It not only provides an diffirent environment such as Linux on a local machine, but makes the same configuration and environment portable. We use Docker for a quick construction of PostgreSQL database.

- Version Control
  - *Git* : A popular tool for version control. The project uses it for managing records of development.
  - *GitHub* : All the records and resources of this project are stored into a private repository on GitHub which is a website for storing local Gits onto the cloud.

- Frameworks
  - *Apache Spark* : An open source framework for cluster computing. We will use Spark SQL to accelerate reading data from the database and the MLlib for analysis with machine learning.
  - *Spring Framework* : An open source Java framework for implementation of inversion of control. We use it to facilitate programming process and for preparation of building a web server in the future.

- Python Packages
  - *Numpy* : Array manipulation.
  - *Pandas* : Data manipulation and analysis.
  - *Scikit-Learn* : Machine learning and data mining.

  - *Tslearn* : Time series data processing and analysis.
  - *Matplotlib* : Data visualisation.

- Java Libraries
  - *Joda-Time* : Datetime and timestamps manipulation.
  - *Project Lombok* : Being used for simplifying code and logging.
  - *Gson* : Json format data processing.
  - *Apache HttpComponents* : Asynchronous http request.

1.3.2 Programme Structure

1.3.3 Report Structure

2 Background Research

2.1 Value at Risk

Value at Risk (VaR) is a method used to evaluate the risk of a financial portfolio. It summarises the degree of investment risk with a single number. Financial institutions have been widely using VaR as a metric to decide how much capital they should keep to bear with risks.

Financial specialists are usually interested in the statement which indicates a specific criterion that helps there decision making. As a result, VaR is normally interpreted as follow:

"I have X% of confidence that the loss of our investment will not be more than $V in the next N days."

where:

- X is the confidence level
- V is the VaR of the portfolio
- N is the time horizon

A proper VaR states that there will only be a (100 - X)% of chance that our loss of investment will be exceeded. For example, given V is $100,000, X is 95, and N is 10, there will only be 5% of scenarios that the loss of our portfolio will be more than $100,000 in the next 10 days based on our estimation.

An N-day VaR(2.1) is usually calculated on the basis of 1-day VaR:

$$\text{N-day VaR} = \text{1-day VaR} \times \sqrt{N} \tag{2.1}$$

There are several approaches for calculating VaRs. In this project, as our objective is to see if clustering algorithms can improve the VaR estimation, we will only talk about the simplest one - *historical simulation approach*[7].

2.1.1 Historical Simulation Approach

Historical simulation approach looks at historical data and simulate the past scenarios that might happen in the future. Then it chooses the worst (100 - X)% of scenarios as the worst cases of the loss of our investment where the $(100 - X)_{th}$ percentile is our VaR.

For example, we want to estimate the 1-day VaR with 95% of confidence for tomorrow, and there are 500 days of historical data. The data for each day is the return/loss from the previous day, which means there are 500 scenarios(2.2) that might happen tomorrow. We can then multiply the value of investment we are holding today by the 5_{th} least return rate (most loss rate) and gain the loss that might happen tomorrow as our 1-day VaR[7].

$$\text{Value under } i_{th} \text{ scenario } = v_n \frac{v_i}{v_{i-1}} \tag{2.2}$$

where:

- v_n is the value of today
- v_i is the value of Day i

2.2 Correlation Analysis

In statistics, covariance(2.3) defines the variability between two individual attributes, which means the level of influence from one feature to another. The numbers correspond to similarity/dissimilarity of the two variables. Positive numbers represent a similar behaviour between them, and vice versa[2].

$$\text{cov}(X, Y) = \text{E}[(X - \mu_X)(Y - \mu_Y)] \tag{2.3}$$

where:

- E is the expectation
- X, Y are vectors of all the samples
- cov is the covariance
- μ_X is the mean of X
- μ_Y is the mean of Y

However, if we want to measure the strength of the linear relationship in between, covariance is not enough. We also need to consider the variance in each feature to tell whether the linear relationship is strong. In this case, the correlation was introduced. Correlation is a normalised form of covariance. It restricts the numbers to a certain range which shows how strong the relationship is. The most commonly used correlation coefficient is Pearson correlation coefficient(2.4). it is calculated by considering the standard deviation of both groups. This can ensure that dispersion of either attribute does not interfere our identification on the strength of mutual linear relationships[5].

$$\rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y} \tag{2.4}$$

where:

- ρ is the Pearson correlation coefficient
- X, Y are vectors of all the samples
- cov is the covariance
- σ_X is the standard deviation of X
- σ_Y is the standard deviation of Y

The value of Pearson correlation coefficient is always between -1 and +1. A positive number means a positive linear correlation, and a negative number means a negative linear correlation. The closer the number towards the extremes, the stronger the relationship is. If the number is 0, it means there is no linear correlation among the pair(2.5).

$$\text{relationship} = \begin{cases} \text{total positive linear correlation} & \text{if } \rho = 1 \\ \text{positive linear correlation} & \text{if } \rho > 0 \\ \text{no linear correlation} & \text{if } \rho = 0 \\ \text{negative linear correlation} & \text{if } \rho < 0 \\ \text{total negative linear correlation} & \text{if } \rho = -1 \end{cases} \tag{2.5}$$

2.2.1 Correlation Matrix

Given a set of data with multiple attributes, we may want to tell people how these attributes interact with each other. In addition, the result of analysis, especially in a simple regression, may not be reasonable when those features are highly dependent.

To achieve this, we can create a matrix which contains all the correlation coefficient calculated from the expanded equation(2.6) with a set of given samples.

$$\rho_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{2.6}$$

where:

- ρ is the Pearson correlation coefficient
- n is the sample size
- x_i, y_i are the single samples indexed with i
- $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ (the sample mean)
- $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ (the sample mean)

A correlation matrix is a symmetirc matrix to its main diagonal. The values on the main diagonal always equal to 1 because the attributes are fully dependent on themselves. Table 1 gives an example of how a correlation matrix looks like.

Features	f1	f2	f3	f4	f5
f1	1	0.74	-0.38	0.12	0.43
f2	0.74	1	0.26	0.88	-0.57
f3	-0.38	0.26	1	0.61	0.59
f4	0.12	0.88	0.61	1	-0.22
f5	0.43	-0.57	0.59	-0.22	1

Table 1: Correlation Matrix

2.3 Principal Component Analysis

It is always challenging to analyse a dataset with high-dimensional data points. Due to the curse of dimensionality, which was discovered by Richard Ernest Bellman in 1957, higher-dimensional space increases the difficulties of analysing and organising data exponentially[8]. Especially in machine learning, given a certain number of samples, the accuracy of predictions on these samples will increase followed by the rising dimensions to a peak but then gradually drop. This is known as Hughes phenomenon[6]. Figure 1 shows how the dimensionality influence the accuracy of predictions.

In order to reduce the dimensionality, there are two approaches can be implemented:

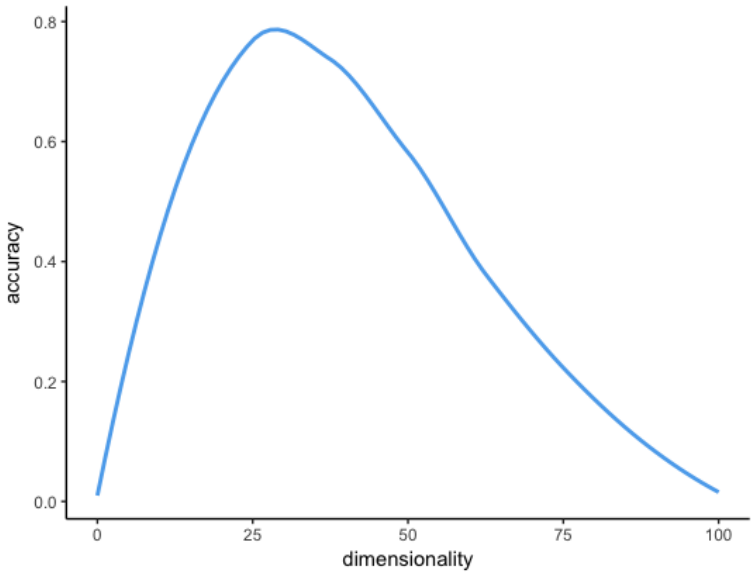


Figure 1: Hughes phenomenon

- Feature Selection: To select a subset that is more informative or relevant among all the attributes[10].
- Feature Extraction: To generate new features from the initial attributes of existing data[3].

The reasons and benefits of executing dimensionality reduction can be summarised as follow:

1. Computational efficiency: Fewer features mean less computation on dissimilarity between pairs of data points and lower arithmetic complexity. It also implies less storage usage as the variables in each sample decrease.
2. Statistical generalisation: By removing noise or irrelevant information from the inputs for building models, the prediction rules can be more general among the datasets.
3. Better explanation: Visualising a lower-dimensional space is much easier. We can effortlessly illustrate the structure of data when the dimension is lower than 3. Higher-dimensional space will be more challenging to visualise, explain and comprehend.

Principal component analysis(PCA) is a method for feature extraction. It projects features onto a lower-dimensional space. A traditional PCA is a kind of single representation approach as opposed to classification on revealing underlying information in a lower-dimensional space with a linear function.

In practice, an optimal mapping function is usually non-linear. In order to fit the data in a non-linear way, we can apply a kernel method on top of the traditional PCA, and this is called kernel PCA. It performs a linear PCA mapping in a higher dimensional kernel Hilbert space to provide a better classification. The kernel can be a polynomial function, a radial function or other functions[9]. However, in this project, we will assume that the relationship between the dimensions (cryptocurrencies) are linear and will only use a standard linear PCA to perform the dimensionality reduction.

2.3.1 Principal Components

The new features derived are called principal components (PCs). They represent new orthogonal axes in an order based on the amount of information it contains.

Imagine that we have a dataset X with data points $\{x_1...x_n\}$. Each point is a D-dimensional vector. The goal is to project the data points onto a M-dimensional space where $M < D$ and maximise the variance of data after projection. The dimensions will be the top M principal components which represent the most informative new features. M is generally determined by the following factors:

- Informativity: How much information of the original features has been involved?
- Interpretability: Are we able to visualise or make useful attribution among the principal components?
- Computational efficiency: The curse of dimensionality. A large number of dimensions will decrease the speed of computation in algorithms.

2.4 Clustering

Clustering, cluster analysis or data segmentation is a non-parametric algorithm in the subtree of unsupervised learning. It is used to separate data into different groups using their dissimilarities (similarities) or possible distributions. Unlike supervised learning, this type of learning algorithms does

not have any indicator for assessing the quality of results, and this means that it does not have any meaning or objective itself. Instead, it discovers the distribution of data and uses the definition given by people who have the specific domain knowledge. By giving the rules for partitioning data self-defined meanings, useful information can be obtained and utilised in different domains[10].

In general, clustering can be defined into two types, parametric and non-parametric. A parametric clustering groups the clusters with a a assumed density function which is usually a Gaussian, while a non-parametric one does not have any assumed distribution, it only aims on finding natural groupings within the givien dataset.

K-means clustering and hierarchical clustering are two of the most popular methods in the non-parametric cluster analysis. In K-means clustering, we specify the number of groups we want to classify into. In contrast, hierarchical clustering does not have an initial number of clusters that we want for the result. It instead shows all the possible clusters into a tree structure and allows us to choose the number of clusters we want at the end. In this project, we will only focus on the K-means algorithm in non-parametric methods.

2.4.1 K-means Clustering

K-means clustering is a intuitive approach which allows us to separate data points into distinct groups. To implement this, we first need to specify an initial number of clusters - K, and then randomly assign a number (cluster) from 1 to K to each object (data point). In this case, the clusters will have two features[4]:

1. $C_1 \cup C_2 \cup ... \cup C_K = \{O_1, O_2, ..., O_n\}$
2. $C_k \cap C_{k'} = \emptyset, \quad \text{for} \wedge k \neq k'$

where:

- C_k is the kth cluster
- O_n is the nth object

These properties mean that each single object will be in exactly one cluster and the clusters does not overlap. After this initial setting, we want to optimise(2.7) the grouping because the previous assignment is just a random initialisation. We want to make sure that the objects are concentrated

which means the data point fit the best in the assigned cluster.

$$\underset{\mathbf{C_1},\ldots,\mathbf{C_K}}{\text{minimize}} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j}) \right\} \tag{2.7}$$

where:

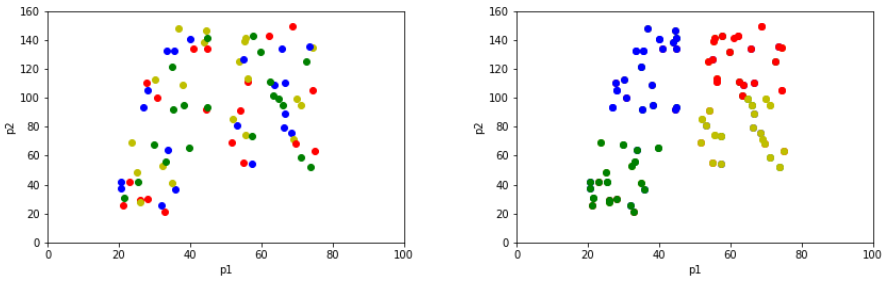
- C_k is the kth cluster
- x_{ij} is the jth attribute of the ith object

To fulfil the condition above which is (2.7), we can simplify the apporach as below to classify our data points into the multiple clusters[4]:

1. Randomly assign an initial number from 1 to K to each observation.
2. Iterate over the following steps until the assigned cluster of each observation stops changing:
   (a) for i in range(1, K):
   Compute the centroid of each cluster which is the mean of vectors with the same k (cluster) assigned.
   (b) Calculate the distance (usually Euclidean distance) between each object and each of the clusters.
   (c) Assign the nearest k (cluster) to each observation.

Figure 2 shows the difference after an optimisation of the cluster assignment. The colors indicate different clusters, and the groups are separated after the implementation of K-means clustering.

K = 4



(a) Before Iteration (b) After Iteration

Figure 2: K-means Clustering

2.4.2 Dynamic Time Warping Distance

Euclidean distance is a metric for evaluating the distance between sequences. It is useful as it produces non-negative result and has linear time complexity. However, it is restricted by the alignment of sequences, which means that the distance can only be calculated if the two sequences are in the same length. If we want to learn how similar the shapes of two objects/lines in time series are, this is not a good method to be used.

Dynamic time warping (DTW) is a popular shape-based algorithm based on dynamic programming in time series analysis. DTW is a time distortion method which adjusts the corresponding elements in two vectors and finds the minimal distance among the neighbours. The accumulative value will then become the final distance in between. It is widely used in temporal sequence data such as speech recognition[1].

Given two vectors t and r, and the lengths are M and N respectively, DTW aims on finding a path to minimise the distance between t and r.

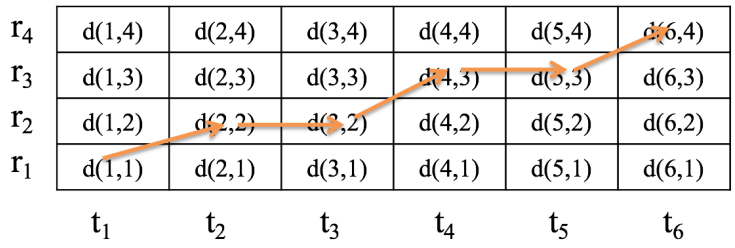


Figure 3: DTW Optimal Path

The calculation of the DTW distance needs to satisfy the following conditions[11]:

1. The first element from t must match the first element from r.
2. The last element from t must match the last element from r.
3. Given $d(i, j)$ is the distance of a point on the optimal path. The points that connect into $d(i, j)$ can only be $d(i-1, j)$, $d(i-1, j-1)$ and $d(i, j-1)$.
4. Given any element in t, there must be at least one corresponding element in r, and vice versa.

We can use a recursion(2.8) to find out the minimal accumulative distance $A(M, N)$ which is our goal of the DTW distance[11].

$$A(i,j) = d(i,j) + min \left\{ \begin{array}{c} \text{A(i-1,j)} \\ \text{A(i-1,j-1)} \\ \text{A(i,j-1)} \end{array} \right\} \tag{2.8}$$

where:

- $D(i, j)$ is the distance between t_i and r_j
- $A(i, j)$ is the accumulative distance from the starting point to (i, j)

3 Analysis of Cryptocurrencies

The data of cryptocurrencies is obtained from a RESTful API `https://min-api.cryptocompare.com/` which is free for non-profit purposes. We store the data into our local Docker based PostgreSQL database for development and convenience in case the internet or the API are unstable. Our Java programme will then update the latest data on a daily basis.

This analysis will use the historical daily OHLCV (Open price, High price, Low price, Close price, Volume) data of cryptocurrencies as the initial input obtained from the API and will only consider 29 cryptocurrencies that have complete data from 1st January 2016 to 31st July 2018. The reason why we choose the number 29 is because we want to compare the performance on cryptocurrencies with that on stocks and we will choose Dow Jones 30 as our counterpart. We want to make them have the same size of products and DWDP in Dow Jones does not have the data before 1st September 2017.

3.1 Data Preprocessing

The original data of cryptocurrencies contains the open price, highest price, lowest price, close price and volume changes of different cryptocurrencies for each day. The unit of the price is USD. There is no specific trading hours for cryptocurrencies. The metric the API used to split the values into days is based on 00:00 GMT time. For simplicity, we will use symbols to represent cryptocurrencies. Table 2 shows the corresponding cryptocurrency of each symbol.

Before we start our analysis, we need to preprocess the data to ensure that the data is clean enough and in the format which can be used as the input of the analysis. The historical daily OHLCV data requested from the API is on the crypto by crypto basis. In this case, we extract all the data, combine them into the format of Table 3 and store them into the database.

Symbol	Crypto	Symbol	Crypto	Symbol	Crypto
AEON	Aeon	BAY	BitBay	BLOCK	Blocknet
BTC	Bitcoin	BTCD	BitcoinDark	BURST	Burst
CRW	Crown	DASH	Dash	DGB	DigiByte
DOGE	Dogecoin	EMC2	Einsteinium	ETH	Ethereum
FCT	Factom	FTC	Feathercoin	GAME	GameCredits
GRS	Groestlcoin	LTC	Litecoin	MONA	MonaCoin
NAV	NavCoin	NLG	Gulden	POT	PotCoin
PPC	Peercoin	RDD	ReddCoin	SYS	Syscoin
VIA	Viacoin	VTC	Vertcoin	XCP	Counterparty
XMR	Monero	XRP	Ripple		

Table 2: Symbols of Cryptocurrencies

	AEON_open	AEON_high	AEON_low	...	XRP_close	XRP_volume
2016-01-01	0.01454	0.0217	0.01316	...	0.0055	0
2016-01-02	0.01498	0.01734	0.01388	...	0.005125	-496.94
2016-01-03	0.01378	0.01536	0.01379	...	0.0052	-23.8623
2016-01-04	0.01391	0.0143	0.01218	...	0.0051	0
...	...	...	...	...	...	...
2018-07-28	1.58	1.75	1.58	...	0.4576	-6853079.88
2018-07-29	1.67	1.78	1.58	...	0.4529	-5774418.55
2018-07-30	1.68	1.69	1.57	...	0.4458	-15453544.47
2018-07-31	1.55	1.55	1.42	...	0.4351	-16047905.37

Table 3: Historical OHLCV of Cryptocurrencies

Our objective is to evaluate the risk based on price volatility which is the daily price changes (g) or daily rate of return (r). We can calculate the daily price changes through dividing the close price by the open price. The rate of return (r) is then equal to $1 - g$. Table 4 shows the daily price changes after transformation from the historical OHLCV dataset.

	AEON	BAY	BLOCK	...	XCP	XMR	XRP
2016-01-01	0.030949	-0.296502	-0.265306	...	-0.038405	0.108141	0.056676
2016-01-02	-0.073431	0	0.180451	...	-0.033988	-0.153010	-0.001364
2016-01-03	0.002903	-0.085601	-0.036252	...	-0.025285	0.059783	0.014634
2016-01-04	-0.097052	0.124730	0.009491	...	0.048847	-0.035897	-0.019231
...	...	...	...	...	...	...	...
2018-07-28	0.056962	-0.024547	0.012285	...	-0.005362	-0.000714	0.006157
2018-07-29	0.011976	0.002037	-0.040519	...	-0.028340	-0.030859	-0.010487
2018-07-30	-0.023810	-0.016599	-0.078947	...	0	-0.025429	-0.015677
2018-07-31	-0.064516	-0.059292	-0.094542	...	-0.098820	-0.076615	-0.024002

Table 4: Historical Daily Changes of Cryptocurrencies

3.2 Risk Estimation

3.2.1 Historical Estimation of VaR

3.3 Risk Diversification

3.3.1 Product Correlations

3.3.2 Clustering on Price Movement

3.4 Performance

3.4.1 Back Test

3.4.2 Improvement After Risk Diversification

4 Analysis of Stocks

4.1 Data Preprocessing

4.2 Risk Estimation

4.2.1 Historical Estimation of VaR

4.3 Risk Diversification

4.3.1 Product Correlations

4.3.2 Clustering on Price Movement

4.4 Performance

4.4.1 Back Test

4.4.2 Improvement After Risk Diversification

5 Evaluation and Conclusion

5.1 Comparison Between Cryptocurrencis and Stocks

5.2 Trading Strategy

5.3 Further Work

6 Appendix

6.1 Programme Usage

References

[1] C.C. Aggarwal and C.K. Reddy. *Data Clustering: Algorithms and Applications*. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series. CRC Press, 2013.

[2] Wikipedia contributors. Covariance — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/wiki/Covariance`, 2018. [Online; accessed 26-May-2018].

[3] Wikipedia contributors. Feature selection — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/wiki/Feature_selection`, 2018. [Online; accessed 27-May-2018].

[4] Trevor Hastie Gareth James, Daniela Witten and Robert Tibshirani. *An Introduction to Statistical Learning, with Applications in R*. Springer, New York, sixth edition, 2015.

[5] G. Hall. Pearsons correlation coefficient. `http://www.hep.ph.ic.ac.uk/~hallg/UG_2015/Pearsons.pdf`, February 2015.

[6] G. Hughes. On the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory*, 14(1):55–63, January 1968.

[7] John C. Hull. *Options, Futures, and Other Derivatives*. Pearson, London, ninth edition, 2017.

[8] Eamonn Keogh and Abdullah Mueen. *Curse of Dimensionality*, pages 314–315. Springer US, Boston, MA, 2017.

[9] Hessam Mirgolbabaei and Tarek Echekki. Nonlinear reduction of combustion composition space with kernel principal component analysis. *Combustion and Flame*, 161(1):118 – 126, 2014.

[10] Robert Tibshirani Trevor Hastie and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, New York, second edition, 2009.

[11] Wikipedia contributors. Dynamic time warping — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Dynamic_time_warping&oldid=848932991`, 2018. [Online; accessed 11-August-2018].