Master's Thesis

Century Analytics

Lund Institute of Technology, Dept. of Matematical Statistics

Tick data clustering analysis establishing support and resistance levels of the EUR-USD exchange market

Author: Karl Tengelin

Supervisors: Alexandros Sopasakis, Jimmy Carlsson & Hugo Langéen

May 11, 2020

LUND UNIVERSITY

Abstract

Our aim is to use clustering algorithms in order to compute support and resistance levels within an intra-day trading setting. To achieve this we use a tick data set from the EUR-USD exchange market during 2019 as a measure of market activity. Both the Gaussian Mixed Model (GMM) and an altered form of Kmeans clustering will be used as clustering methods where each method will be evaluated using a selection of common performance metrics. The computed support and resistance levels will then be put to the test by initiating mock trades during certain time windows from early 2019, which are specified by Century Analytics.

Both models that were used in this thesis managed to partition the data in a way that made it possible to create support and resistance levels that are comparable to traditional methods which do not rely on market activity. Although more research needs to be made the results look promising and we can, with some confidence, say that market activity in the form of ticks can be used as an indicator for support and resistance levels within the EUR-USD exchange market.

The support and resistance levels computed using GMM and Kmeans were quite similar but the GMM method performed better when examining the methods using mock trades. The GMM could predict support and resistance "bounces" with greater statistical significance compared to the Kmeans method.

Keywords: *Tick data, Support-and resistance levels, Clustering methods, Gaussian mixture model, Kmeans, EUR-USD exchange rates, Clustering performance metrics, Market activity*

Acknowledgements

I would like to thank...

My supervisor Associate Professor Alexandros Sopasakis for his guidance and support during this project and his extensive knowledge within statistics and machine learning that he has been willing to share.

My supervisors at Century Analytics, Hugo Langéen and Jimmy Carlsson, for providing the tick data necessary for the project and their expertise within trading and finance which has helped fill in the blanks.

My parents Kristofer and Malin and my brother Johan, for their constant support and their valuable inputs.

Contents

1 Introduction

Interpreting big volumes of data has become increasingly important for companies in the financial sector in order to make data driven decisions. More and more companies are therefore looking into machine learning methods such as clustering methods to aid them in their business.

In order for these companies to avoid risk and negate losses it is important to take into consideration as many variables as possible. One of the variables that have been used traditionally but perhaps not so much in a machine learning setting is tick-data. Tick-data is a way of measuring activity on a market, be it on the credit market or the currency exchange market.

In this thesis we want to examine if we can use clustering algorithms on provided tick-data in order to partition the price of EUR-USD exchange rates with respect to activity. The proposed setup of this thesis is shown in Figure 1.

This partitioning will then be used to investigate whether or not market activity can be an indicator for support-and resistance levels.

Support-and resistance levels is a well established phenomenon in the financial sector where a *support level* is a level which a price for an asset tend to stay **above** and a *resistance level* is a level which the price for an asset tend to stay **below**[1]. This is indicated by the price "touching" these price levels but then bouncing back up (in the case of support levels) or bounce down (in the case of resistance levels). If clustering methods could be used to establish these levels this would be a huge gain for anyone that works within the financial sector.

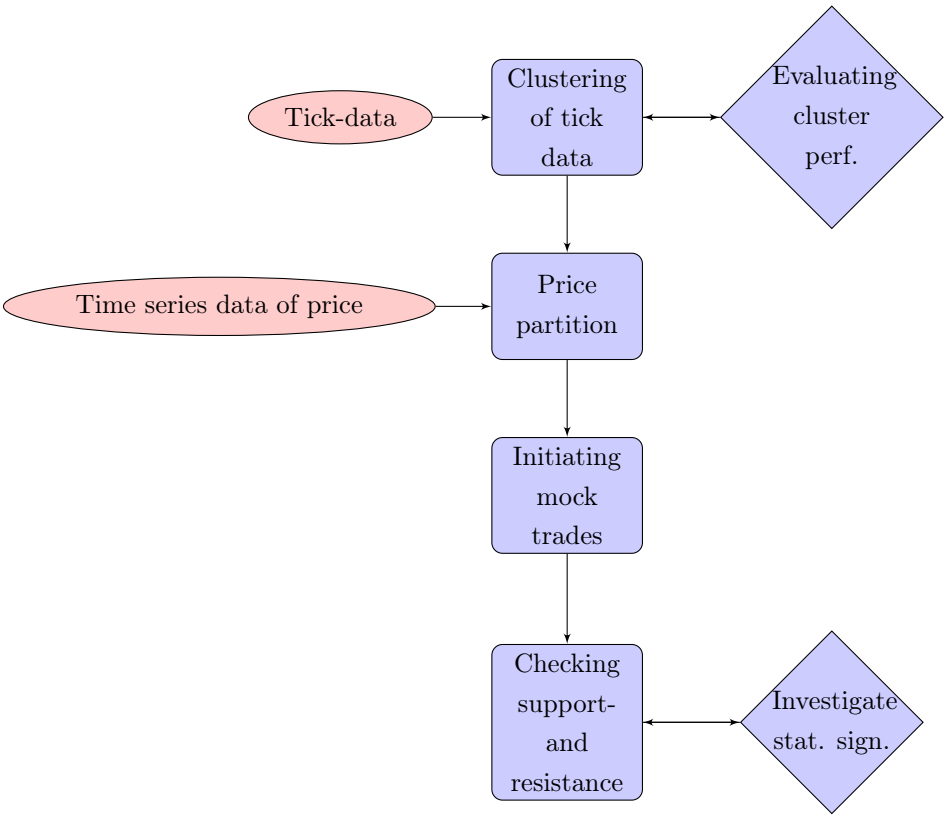

Figure 1: *Chart describing the proposed methodology for the method used in this thesis in order to investigate the occurrence of support and resistance levels in tick data and whether or not these levels can be found using a selection of clustering methods with statistical significance.*

The data used in this thesis comes from Century Analytics AB and consists of:

- tick data for EUR-USD exchange market.
- price data for EUR-USD exchange market.

These data sets are then used as described in Figure 1. There are however quite a few difficulties when handling and performing clustering algorithms on these types of data sets. Mainly:

- the volatile nature of the data.
- the problem of choosing a suitable number of clusters in an unsupervised setting.

These difficulties combined make it hard to evaluate the clustering methods. There are however some metrics that can be used to help this evaluation. A selection of them are described later in this thesis.

The clustering methods that will be used to solve this problem is a slightly altered form of Kmeans clustering as well as Gaussian Mixture Models (GMM) with Expectation Maximization (EM). These methods are well established and will be discussed in greater detail in Section 7.

2 Problem setting

This thesis will effectively be divided into two separate parts that seek to answer two different but connected questions:

1. **Can clustering methods such as Kmeans and GMM be used to partition tick data from EUR-USD exchanges efficiently?**
2. **Can this partitioning of tick data be used to establish support and resistance levels for EUR-USD exchanges with any statistical significance?**

3 Technologies

Python has been used extensively throughout this thesis due to its simplicity and its extensive library of useful packages. Most importantly the Pandas package and the SciKit-Learn toolkit has been used.

Pandas was used because of its capabilities to handle and manipulate large data sets in an efficient way.

SciKit-Learn was used since it contains many pre-made clustering algorithms such as GMM and Kmeans as well as a big library of performance metrics which have been used extensively throughout this thesis.

Jupyter notebook was used in order to run the Python code via a web browser.

4 Conventional methods to determine support and resistance levels

The use of support and resistance levels in intra-day trading is, to this day, of interest for financial institutions when conducting technical analysis [2]. Financial institutions such as foreign exchange trading companies publish technical data daily as an aid for their customers so that they can make informed trading decisions. Although these technical analysis reports vary from company to company the majority of them contain information about support and resistance levels [3]. Also, as Carol Osler describes in [3], the support and resistance levels for the same time period can vary a bit between companies where some companies appear to perform better than others. Perhaps indicating that the methods they use to determine support and resistance vary.

The algorithms used to determine support and resistance levels are naturally kept secret by these companies so we don't exactly know if it is common to use machine learning methods or not. On the other hand we do know that there are a number of none-machine learning methods that can be used to calculate support and resistance levels. Some of these methods include; *Trendlines*, *Moving averages* and *Fibonacci sequences/golden ratio* [1].

Trendlines is a method where one simply looks at the historical data of the asset of interest, for example EUR-USD exchange rates, and draws a line connecting the lowest price points in order to create a support level and a line connecting the highest price points to establish a resistance level [1].

Moving averages (MA) is a method which generates output based on an average of subsets of historical data. This creates a lag in the price movement produced by the MA and thus the output will act as a support level for price or exchange rate trending up and a resistance level for price or exchange rate trending downwards [4].

The Fibonacci method is, as the name suggests, based on Fibonacci sequences. There are several forms of Fibonacci sequencing of financial data but they all rely on Fibonaccis famous sequence of a progression of numbers where each new number is the sum of the two previous. The idea behind the method is to use the golden ratio which stems from the Fibonacci sequence of approximately 1.618 in order to produce support and resistance levels using the maximal and minimal price from historical data as anchor points [5].

There have been attempts to use machine learning methods to define support and resistance levels such as the one a user named "saturdayquant" on GitHub has implemented. Here a mean shift method together with the elbow method (which will be discussed later in this thesis) is used to determine support and

resistance levels in rate data [6]. However that method is not used on tick data but rather on rates.

Another case where a machine learning method was used to examine support and resistance levels is described by an author under the user name ”judopro” on the online publishing platform Medium [7] where a Kmeans (a clustering model we will investigate further in this thesis) was used to classify rate data. Again using the elbow method as performance metric for number of clusters. The difference with this method compared to the ones described in this thesis is once again the use of rate data rather than tick data.

5 Tick data and Timeseries

There are a few definitions and clarifications that have to be made before delving deeper into the mathematical aspects of the problem.

Definition 5.1 ***”Ask”*** *is the price asked by the* ***seller*** *to be paid for a certain asset*

Definition 5.2 ***”Bid”*** *is the price offered by the* ***buyer*** *for a certain asset*

5.1 Tick data

Tick data can have different meanings in different contexts, but in this thesis tick data describes the flow of orders that are registered in what is called the ”order book”. This can be interpreted as the activity on the market.

An order book stores the time stamp, current bid-price and current ask-price for the given asset whenever an order is received due to someone either wanting to buy the asset or sell the asset. Note here that the tick value does not distinguish between an order to sell and an order to buy, tick data can only be used as a measure of total activity within a given time interval.

In this thesis we are especially focusing on what is known as top of book. Top of book consists of two layers:

1. The *lowest* ask-level at the current time
2. The *highest* bid-level at the current time

These levels are also arguably the most interesting since they represent the price levels for which a trade will be substantiated [8]. The difference between these levels is known as the bid-ask spread and the average of the bid-ask spread is what we will use as price throughout this thesis.

5.1.1 Difference between ”Tick” and Pip”

The word ”tick” may in some cases refer to a minimal incremental change in value of a security or a currency exchange. However this is **not** what is meant in this thesis where the minimal incremental change of a currency exchange is referred to as a ”pip” or even in some cases fractions of pips. Tick will instead be used to describe activity on a market (as described in previous section).

In this thesis a pip has the value 0.0001 and we will mostly be working with price changes in the range of 10 pips, i.e an incremental change in price of 0.001.

5.2 Time series

Time series data is data that is indexed with respect to time and these types of data sets are very commonly used in order to spot trends in, for example, financial data. In this master thesis the primary subject of investigation consists of time series data of ticks that have been gathered during the first half of 2019 from the EUR-USD exchange market.

5.3 Target and stop loss

Target and stop loss are two terms that will be used extensively throughout this thesis and they are connected to how traders operate when trading within short time spans.

5.3.1 Target

If a trade is initiated where the trader aims to alter their position within a short time he or she will want the price to reach a certain level before the position is changed. An example would be if a trader buys an assets he or she might set a level 0.01 price units higher than the purchase price, if the price then reaches this *target* the trader sells the asset for a profit.

5.3.2 Stop loss

Stop loss (SL) represents the reversed case compared to target. If a trader initiates a trade, for example buying an asset with the intention to sell later, he or she might set a price level lower than the purchase price for which the asset will be sold if that level is reached. If the price reaches this level the asset is sold in order to avoid big losses.

6 Correlation metrics

A quick and easy estimate to find any connection between two variables, for example activity and price, is the cross-correlation metric. We chose to use two different types of correlation metrics: Pearson correlation and Spearman correlation. Pearson correlation is the more common of these two.

6.1 Pearson correlation

The Pearson correlation is named after Karl Pearson and is perhaps the more common form of correlation measure between two variables X and Y. The measure is defined as shown below,

$$\rho_{X,Y} = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{\mathrm{E}\left[(X-\mu_X)(Y-\mu_Y)\right]}{\sigma_X \sigma_Y}. \tag{1}$$

Where σ denotes the standard deviation of the variables and μ is the respective mean.

This formula will produce a measure between -1 and $+1$ where a value of -1 implies total negative correlation between the variables, i.e if one variable increases in value the other variable decreases in value.

A value of $+1$ means that if one variable increases the other also increases and a value close to zero indicates no correlation between the variables.

6.2 Spearman rank correlation

The Spearman correlation is based on a ranking system where variable values are ranked from smallest to largest and the sum of the difference between these ranks are measured and then normalized with respect to the sample size [9]. This is explained using clear examples in [10].

The formula for Spearman correlation differs slightly when having shared ranks in Equation (2) versus having no shared ranks in Equation (3),

$$\rho = 1 - \frac{6\sum d_i^2}{n\left(n^2-1\right)}, \tag{2}$$

$$\rho = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}}. \tag{3}$$

Here d_i describes the difference in paired ranks and i is the index of the paired score [10].

As was the case with Pearson correlation the ρ-value of the Spearman rank correlation ranges in between -1 and $+1$. A -1 indicates negative correlation while $+1$ indicates positive correlation and 0 indicates no correlation.

The difference between Pearson and Spearman-correlation is in essence that Pearson only detects linear correlation between variables whereas the Spearman correlation can detect more intricate, nonlinear relations. By using both methods we can get a better insight in how activity and price movement correlates than by just using one correlation metric.

7 Machine learning

7.1 Supervised versus unsupervised machine learning

Machine learning can be divided into several different types but the main two classes are *supervised* and *unsupervised* methods. These two settings are quite dissimilar even though there are algorithms that can work in both settings.

7.1.1 Supervised machine learning

Supervised machine learning indicates that we have some information about parts of the data we are examining. One common example is the distinction between pictures of cats and dogs. In such an example the AI must learn what a dog or a cat looks like and thus needs to be fed training pictures where it is also given the answer. This is achieved by labeling the data. This setting is however not that common when trying to cluster data but rather when the goal is to classify data.

7.1.2 Unsupervised machine learning

This setting is often used when we know nothing or very little about the connection between data points. Since we don't have all the information we cannot set a label on certain data points and it is instead entirely up to the algorithm to distinguish common features among the data points. This is why clustering is often set in an unsupervised environment where it is up to the algorithm to solve the problem on its own. This also means that this is a conceptually harder problem to solve.

7.2 Clustering algorithms

A clustering algorithm is used to group data points together, hence the name "clustering". There are several clustering methods available and they can potentially

work very differently, but the two methods chosen for this thesis are Kmeans-clustering and Gaussian Mixture Models (using expectation maximization). The Kmeans algorithm had to be slightly altered however in order to accommodate the restrictions of this thesis (see Appendix A).

Both Kmeans and GMM:s need to have a hyperparameter pre-determined before initializing the clustering. That hyperparameter is the *number of clusters*. A crucial and often occurring problem using clustering algorithms, especially in an unsupervised setting is that we typically don't know the number of clusters. This is why we need to test a range of numbers of clusters using certain metrics in order to determine which is the optimal number of clusters (more about this in Section 8).

7.2.1 Kmeans clustering

The arguably most common version of the Kmeans is a method originally used in signal processing that is attributed to Stuart P. Lloyd who proposed the method in 1957 but didn't publish his findings until 1982 [11].

Kmeans is what is known as a *hard* clustering method meaning that it doesn't categorise each data point with a probability of belonging to a certain cluster but rather that it strictly belongs to the cluster. This is a stricter way of categorising data than soft clustering methods. The methodology proposed by Lloyd is described in Algorithm 1.

Algorithm 1: Lloyd's algorithm

Data: Dataset $\mathbb{D}$, number of predetermined clusters k.

Result: Vector of cluster label for each data point.

Start by randomly initialize a set of k means $\{\mu_1^1,...,\mu_k^1\}$.

while Assignment $S_i^t \neq S_i^{t-1}$ **do**

  for each point $x_i \in \mathbb{D}$ **do**

    $S_i^t = \left\{x_p : \|x_p - \mu_i^t\|^2 \leq \|x_p - \mu_j^t\|^2 \quad \forall j, 1 \leq j \leq k\right\}$

    $\mu_i^t = |S_i^t|^{-1} \sum_{x_j \in S_i^t} x_j$

  end

end

Which in less mathematical terms can be boiled down to three steps where the two last steps are repeated until convergence:

1. Initialize k number of agents called *centroids* randomly scattered among the data points.

2. Assign each data point to its nearest centroid (i.e give all data points a cluster label).

3. Update the centroids position by calculating the mean of each data point assigned to the centroid.

As mentioned the Lloyds algorithm isn't entirely suitable for our problem setting (an example shown in Section 10 will make it clear as to why this method proves unsuitable) and a slightly altered version is instead used which is described in Appendix A.

7.2.2 GMM (Gaussian Mixed Models) and Expectation Maximization

The GMM and EM algorithms has an underlying assumption that the data that is clustered comes from a certain assumed distribution. When using GMM one assumes, as the name suggests, a Gaussian distribution for the data points x,

$$f\left(x_i\right) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x_i-\mu)^2/2\sigma^2}. \tag{4}$$

However the mean μ and variance σ^2 are generally unknown.

This problem of unknown variables could be solved quite easily if one knew the labels of each data point, i.e from which distribution the points were drawn from. But if we work in an unsupervised setting we don't know which data point belongs to which class, this is the whole point of our clustering algorithms.

Using the same reasoning but reversed it would be easy to figure out which data point belonged to which class (i.e which distribution) if one would know the mean and variance of the distributions.

This chicken-and-egg problem is what the EM-algorithm is designed to solve.

The algorithm used in this thesis for Expectation Maximization is often attributed to A. P. Dempster, N. M. Laird and D. B. Rubin in a paper published in 1977 [12] and the basic outline of the algorithm is described in Algorithm 2.

Algorithm 2: Expectation maximization

Data: Dataset $\mathbb{D}$, number of clusters k

Result: Vector of cluster labels for each data point.

Start by randomly initializing a set of parameters θ_k for each class K

while $|\theta_k^{t+1} - \theta_k^t| > \varepsilon$ **do**

  for each point $x_i \in \mathbb{D}$ **do**

    $Q(\theta_k|\theta_k^t) = \mathbb{E}_{x_i|X,\theta_k^t}\left[logL(\theta; X, x_i)\right]$

    $\theta_k^{t+1} = argmax_{\theta_k^t} Q(\theta_k|\theta_k^t)$

  end

end

So the combination of GMM and EM means that the K different sets of model parameters consists of the mean μ and the standard deviation σ used to define

the PDF described in Equation (4), we denote these model parameters: $\theta_k = \{\mu_k,\sigma_k\}$ produced by Algorithm 2.

In simpler terms one can say that the GMM-EM algorithm initially guesses the mean values $\mu_1,...,\mu_k$ and variances $\sigma_1^2,...,\sigma_k^2$. Then the following two steps are repeated until convergence:

1. The data points are each assigned a distribution that fits them best based on the log-likelihood that the points are drawn from the specific distribution.
2. The mean values $\mu_1,...,\mu_k$ and variances $\sigma_1^2,...,\sigma_k^2$ are then changed according to the points that where assigned to each cluster/distribution.

8 Metrics for performance of clustering algorithms

We aim to use several different performance metrics in order to establish the correct number of clusters (k). As described in Section 7 this is one of the immediate problems when working with clustering algorithms in an unsupervised environment and hopefully using different sorts of performance methods could help distinguishing how many clusters to use for each data set and method.

8.1 General performance measures

The measures described in this subsection can be performed on both the GMM method and the Kmeans method.

8.1.1 Silhouette score

Silhouette score is a score based on likeness between data points within the cluster (cohesion) and the difference between data points from different clusters (separation) and produces a measurement between -1 and $+1$ where a *high positive score* is desired [13]. Cohesion is defined as the mean distance between data points within a cluster according to,

$$a(i) = \frac{1}{n_{C_i} - 1} \sum_{j \in C_i, i \neq j} d(i,j). \tag{5}$$

Where C_i is the cluster in which the data point i resides and j is another data point within the same cluster as i, $d(i,j)$ is the Euclidean distance between i and j and n_{C_i} is the total number of data points in cluster C_i.

Note here that we divide by $n_{C_i} - 1$ since we do not count the distance d(i,i).

The separation b is defined as,

$$b(i) = \min_{C_l \neq C_i} \frac{1}{n_{C_l}} \sum_{j \in C_l} d(i,j). \tag{6}$$

Here the notation $min_{l \neq i}$ means that we are looking at the distance between data points in the nearest cluster measured from the cluster center of cluster C_i.

The Silhouette score for data point i is then calculated using the following calculations,

$$s(i) = \begin{cases} 1 - a(i)/b(i), & \text{if } a(i) < b(i), \\ 0, & \text{if } a(i) = b(i), \\ b(i)/a(i) - 1, & \text{if } a(i) > b(i). \end{cases} \tag{7}$$

To get the final Silhouette score we then calculate the average score $\overline{s(i)}$ for all data points.

8.1.2 Calinski-Harabasz

Calinski-Harabasz (often denoted CH) is a method that is using the intra-cluster variation (W) and inter-cluster variation (B) with respect to number of clusters (k) to estimate the performance of a clustering method [14]. The formula used to calculate this metric is shown below where a *high* CH value indicates a good clustering,

$$CH(k) = \frac{B_c(k)}{(k-1)} / \frac{W_c(k)}{(n-k)}. \tag{8}$$

Here n is the total number of data points and B_c (the between-cluster sum of squares) and W_c (the within-cluster sum of squares) are calculated as described in Equation (9) and (10) respectively,

$$B_c = \sum_{k=1}^{K} n_{C_k} \|\overline{C_k} - \bar{x}\|^2 \tag{9}$$

$$W_c = \sum_{k=1}^{K} \sum_{i=1}^{n} w_{k,i} \left\|x_i - \overline{C_k}\right\|^2. \tag{10}$$

Here K denotes the total number of classes (or clusters), n_{C_k} is the total number of data points in cluster C_k, x_i denotes data points, $\overline{C_k}$ is the centroid of cluster k and $w_{k,i}$ is an indicator function defined below,

$$w_{k,i} = \begin{cases} 1, \text{ if } x_i \in \text{ cluster } C_k \\ 0, \text{ Otherwise} \end{cases} . \tag{11}$$

8.1.3 Davies-Bouldin

The concept behind Davies-Bouldin score is that for a clustering to be considered good the separation between clusters (dispersion) as well as the the homogenity and compactness within the clusters should be high [15]. The dispersion (S) of cluster C_i is defined in the following way,

$$S_i = \left(\frac{1}{n_{C_i}} \sum_{x \in C_i} |x - \overline{C_i}|^p \right)^{\frac{1}{p}} for\, p > 0. \tag{12}$$

Where n_{C_i} is the total number of data points that lie within cluster C_i and $\overline{C_i}$ is the position of cluster C_i:s center. The distance measure that was used in our case was Euclidean distance, hence p = 2. The separation between cluster C_i and C_j is defined as,

$$D_{ij} = \left(\sum_{l=1}^{d} \left| \overline{C_{il}} - \overline{C_{jl}} \right|^t \right)^{\frac{1}{t}} for\, t > 1. \tag{13}$$

Where cluster centers i and j are denoted $\overline{C_{il}}$ and $\overline{C_{jl}}$ respectively. As was the case for the dispersion we choose the Euclicdean distance between cluster centers, hence t = 2 in our case.

The Davies-Bouldin score is then calculated using the following calculation,

$$V_{DB} = \frac{1}{k} \sum_{i=1}^{k} R_i, \tag{14}$$

where k is the number of clusters and R_i is defined as,

$$R_i = \max_{i \neq j} R_{ij}. \tag{15}$$

Here R_{ij} is calculated according to,

$$R_{ij} = \frac{S_i + S_j}{D_{ij}}. \tag{16}$$

The resulting metric is therefore considered better the *lower* it is since we want a low within-cluster-variance and a high cluster separation.

8.2 Kmeans-specific measures

The metrics described in this subsection were only used to establish the hyperparameter number of clusters for the Kmeans method.

8.2.1 V-Measure

V-measure is based on the metric V-score which weighs homogenity and completeness of the input data [16].

8.2.2 Homogenity h

Homogenity describes whether or not a cluster contains data points with different labels. A cluster containing data points of solely one sort (one label) would be considered completely homogeneous.

The homogenity (h) is calculated using the following formula,

$$h = 1 - \frac{H(C,K)}{H(C)}. \tag{17}$$

Here H(C, K),

$$H(C,K) = -\sum_{k=1}^{K}\sum_{c=1}^{C}\frac{a_{ck}}{N}\log\left(\frac{a_{ck}}{\sum_{c=1}^{C}a_{ck}}\right). \tag{18}$$

Where N is the total number of data points and a_{CK} is the total number of data points belonging to cluster K and class C. H(C) is defined as,

$$H(C) = -\sum_{c=1}^{C}\frac{\sum_{k=1}^{K}a_{ck}}{C}\log\left(\frac{\sum_{k=1}^{K}a_{ck}}{C}\right). \tag{19}$$

8.2.3 Completeness c

Completeness on the other hand describes the methods ability to capture all the data points of the same label. Meaning that if all the identically labelled data points are within the same clusters the clustering has as high completeness as possible.

The completeness c is calculated using the following formula and once again a_{CK} is the number of data points belonging to cluster K and class C,

$$c = 1 - \frac{H(K,C)}{H(K)}. \tag{20}$$

Here:

$$H(K,C) = -\sum_{c=1}^{C}\sum_{k=1}^{K}\frac{a_{ck}}{N}\log\left(\frac{a_{ck}}{\sum_{k=1}^{K}, a_{ck}}\right) \tag{21}$$

and:

$$H(K) = -\sum_{k=1}^{K}\frac{\sum_{c=1}^{C} a_{ck}}{C}\log\left(\frac{\sum_{c=1}^{C} a_{ck}}{C}\right). \tag{22}$$

8.2.4 V-score

The V-score is then given by the following equation:

$$V_\beta = \frac{(1+\beta)hc}{\beta h + c}. \tag{23}$$

β is a factor that can be altered in order to favour completeness or homogenity (in our case we don't favour either metric so $\beta = 1$).

The problem of using this method in our setting is that in order for the V-measure to work we need to have an established ground truth (a set of correct labels, this is what "C" represents in (20)). Since we are working in an unsupervised setting we don't have a ground truth. This problem can be circumvented using a small trick where we take the clustering partitioning done by the algorithm and then "scramble the edges" to get an artificial ground truth (see Appendix B).

This means that we take the partitioning made by the Kmeans and look in small intervals at the borders between two clusters. In this small interval we scramble the data points so that some data points that were assigned to cluster x is now assigned to cluster y and vice versa. This new partitioning of the data is what we then use as our ground truth, meaning that we assume that this is in fact the correct labelling of the data.

The interpretation of the V-score is that when the V-score levels out, i.e when there is no clear advantage in increasing the number of clusters, we have found the optimal number of clusters.

The V-score will generally increase when increasing the number of clusters so the argument is that if the V-score starts to plateau we have found the optimal number of clusters [16]. This concept is discussed using an example in Section 10.

8.2.5 Elbow method

The elbow method is based either on a metric called *distortion* (d) or *inertia* [17, 18, 19]. Distortion is the metric chosen for this thesis and distortion measures the average squared distance (here we use Euclidean distance) between each data point and its respective cluster center (see the following Equation (24)) where N is total number of data points and μ_k is the cluster center closest to data point x_i,

$$d = \frac{1}{N}\sum_{i=0}^{N} ||x_i - \mu_k||^2, \quad \mu_k = argmin_{\mu}||x_i - \mu||. \tag{24}$$

By calculating the sum of distortions for all the clusters we get a value that shows how well the clusters describe the data. However adding another cluster would presumably always decrease the distortion and therefore we are interested in the point where adding another cluster does not give any *significant* decrease in distortion.

The goal is therefore to test different numbers of clusters and extract the distortion. Looking at a graph of the plotted distortions as a function of number of clusters there should then be a point where adding another cluster does not decrease the score by any significant amount. There will be what looks like an "elbow" in the graph and it is at this elbow where one assumes the optimal number of clusters is (see Figure 24). Note that this is a heuristic approach, which is why most people agree that there are more unambiguous methods that are better to use when evaluating number of clusters.

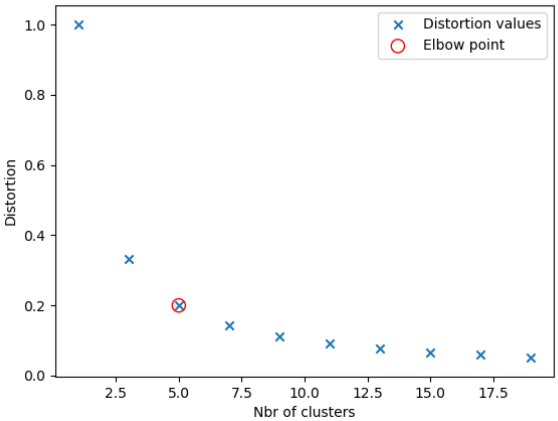

Figure 2: *Shows an illustration of how choosing number of clusters using the elbow method might look like.*

8.3 GMM-specific measures

The metrics described in this subsection were only used to establish the hyperparameter number of clusters for the GMM method.

8.3.1 Shapiro Wilks

The Shapiro-Wilks test was introduced by S.S. Shapiro and M.B. Wilks at General Electric CO and Bell telephone Laboratories in order to test normality in complete samples [20]. The metric is based on what they called the *W statistic* which is defined as,

$$W = \frac{\left(\sum_{i=1}^{n} a_i x_{(i)}\right)^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2}. \tag{25}$$

Here $x_{(i)}$ is called the order statistic and consists of the i:th smallest value in the sample (if we have a sample set $S = \{2,3,6\}$ the $x_{(i)}$:s would be $x_{(1)} = 2$, $x_{(2)} = 3$ and $x_{(3)} = 6$), not to be confused with x_i which denotes each data point. $\overline{x}$ is the sample mean and a_i is the coefficients given as a row vector below,

$$(a_1,...,a_n) = \frac{m^T V^{-1}}{(m^T V^{-1} V^{-1} m)^{1/2}}. \tag{26}$$

Where V is the covariance matrix between all order statistics and $m = (m_1,...,m_n)^T$ where m_n is the expected value of each order statistic sample independently drawn from a standard normal distribution (4).

The basic idea behind the metric is that the sample is compared to a normal distribution where a normal distribution and the sampling histogram are superimposed and the overlapping percentage is calculated [21]. Here a *high* score is desirable since it indicates similarity between the standard normal distribution and the sample distribution.

8.3.2 Kolmogorov-Smirnov

The one sample Kolmogorov-Smirnov performance metric (which is the type of K-S metric we are interested in this thesis) is based on the distance between the samples empirical distribution function and the cumulative distribution function of a reference distribution, in our case the normal distribution [22]. For the Kolmogorov-Smirnov to produce a good score we need the maximal distance (D) between the sample distribution and the theoretical distribution to be as small as possible. Based on this distance metric and under the null hypothesis

that the sample distribution and the theoretical distribution are the same the K-S method compares the test statistic to a table [23]. Based on this table a metric given which is considered better the *higher* it is. In our case the kstest metric from the sklearn package computes this statistic automatically.

The test statistic D is defined according to,

$$D_n = \sup_x |F_n(x) - F(x)| . \tag{27}$$

Here $F(x)$ in our case is the cumulative distribution function for the normal distribution and $F_n(x)$ is the empirical distribution function based on an observation X_i, see below,

$$F_n(x) = \frac{1}{n} \sum_{i=1}^{n} I_{[-\infty,x]} (X_i) . \tag{28}$$

Here $I_{[-\infty,x]}$ is the indicator function defined as,

$$I_{[-\infty,x]} = \begin{cases} 1, \text{ if } X_i \leq x, \\ 0, \text{ otherwise.} \end{cases} \tag{29}$$

In general Shapiro-Wilk seems to be the more powerful test compared to Kolmogorov-Smirnov [24]. However when handling big sample sets as those we have in this project, some argue that the Kolmogorov-Smirnov is the better choice [25]. Therefore we chose to examine both metrics in this thesis.

9 Method to determine statistical significance

This statistical method is introduced in this thesis in order to evaluate the second part of the thesis where we want to determine if the clustering methods implemented actually can be used to find the support and resistance levels which we are interested in. By using this test we can get an interpretation of how likely it is that we actually have found what we are looking for.

9.1 P-value

In simple terms the P-value can be described as the probability of observing a certain observation **or** an even more extreme one [26]. This explanation is probably easier to understand using an illustration, see Figure 3.

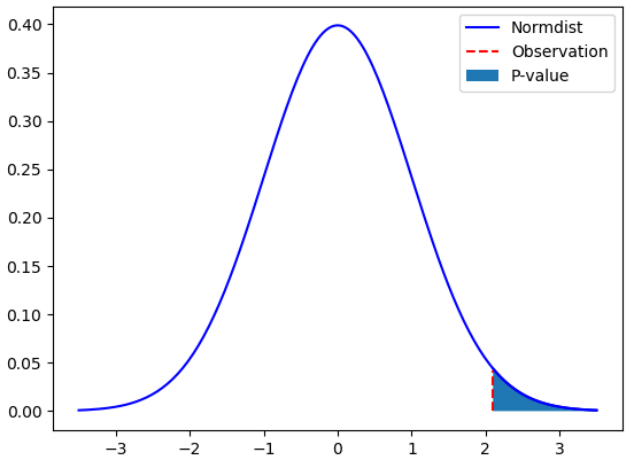

Figure 3: *Shows an illustration of how P-value should be interpreted. The dotted line represents the observation, for example this could be the probability of getting 85 "tails" when performing 100 coin flips, something that is quite rare. The P-value would then the summation of probabilities of getting 85 tails + 86 tails ... + 100 tails.*

Note here that we are using a so called one-tailed test which only examines the probability of getting more extreme observations towards one end of the spectrum (in the coin-tossing example we only examine the probability to get more tails, not the probability to get more heads).

P-value can be used as a way of evaluating statistical significance of our results based on the null hypothesis. The null hypothesis can in our case be defined as in definition 9.1. One can draw parallels here to the coin-tossing example.

Definition 9.1 ***Our Null hypothesis:*** *The probability of reaching a target and a SL is completely random and the likelihood of either case is equal.*

This means that if we can reject our null hypothesis we can claim that there is a statistical significance to our results. In order to simplify the investigation we do in this instance assume that the distribution of the currency exchange alteration is symmetric around the support/resistance levels. It is possible that this isn't the case but in order to keep the complexity down we chose to assume a Gaussian distribution.

10 The data and the process

The following section is meant to describe how we need to summarize our tick data in order to make it cluster-friendly, to show the reasoning behind the cluster selection process for both GMM and Kmeans and to show how we validate the models using mock trades (see Appendix C). In order for the reader to get a clearer picture of the process we will go through an example based on data gathered during one week in January 2019.

As we mentioned before there is also a problem regarding the V-measure in our particular setting and this problem will be addressed in this section as well.

10.1 Creating cluster friendly data

The data used for this thesis is as mentioned tick data, but tick data in and of itself is not suitable to perform clustering methods on so the first step in the process (see Figure 1) is to summarize this data in a way suitable for the algorithms. This can be seen in Figure 4 where the ticks for one weeks worth of data is depicted as a singular value for each price level.

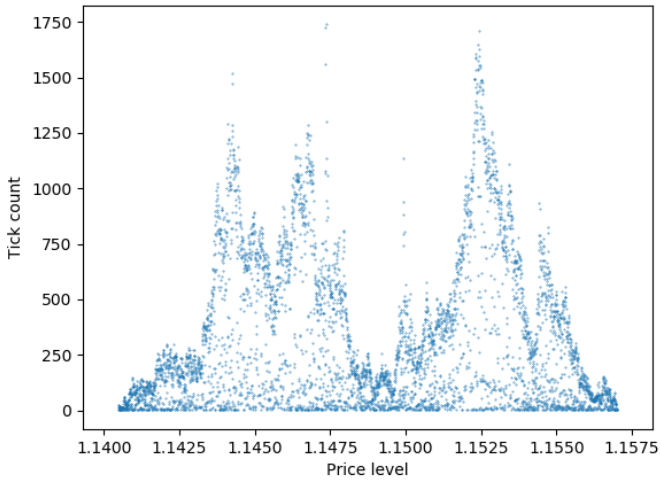

Figure 4: *Shows the number of ticks at each price level for the EUR-USD exchange rates during one week (2019-01-06 to 2019-01-11).*

By performing the clustering algorithms on the data set illustrated in Figure 4 we aim to partition the data *with respect to price*. This is needed since later in the process there need to be clear distinctions between price levels as these borders will determine the support and resistance levels.

10.2 The problem with Lloyds Kmeans and how the different Kmeans-versions partition our data

At a first stage the Lloyds Kmeans model from the scikit-learn package was used to partition the data set shown in Figure 4 and this partition is shown in Figure 5

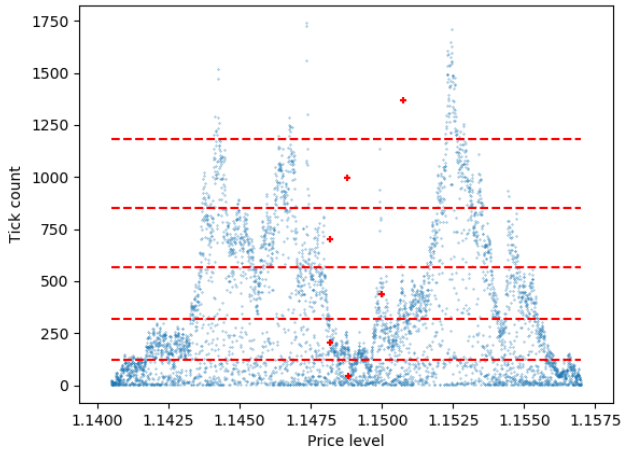

Figure 5: *Shows how the sklearns Kmeans model (i.e Lloyds algorithm) partition the tick data introduced in Figure 4. The red crosses represent cluster centers.*

It is clear here that according to this method the optimal way to partition the data is into horizontally aligned segments. However this is not desirable when examining support-and resistance levels since there is no clear distinction between *price levels*, as is evident in Figure 5 where all price levels contain all classes specified by the method.

Instead another slight variation of the Kmeans model was designed in order to only partition the data along the price dimension (see appendix A). The results of the partition produced by this model is shown in Figure 6.

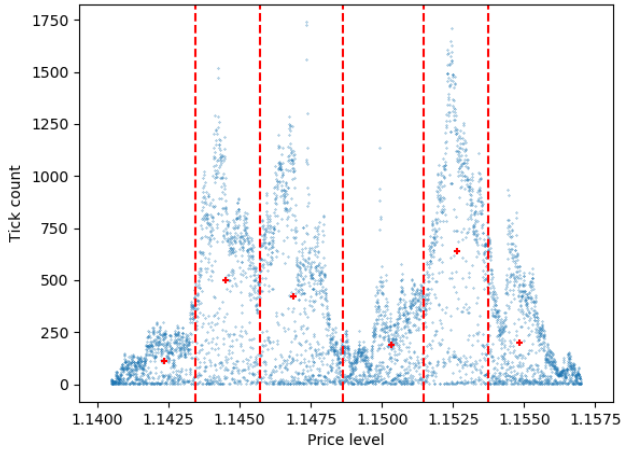

Figure 6: *Shows how the specialised Kmeans model described in appendix A partition the tick data introduced in Figure 4. The red crosses represent cluster centers.*

Here we can see that the data is clearly grouped in vertical segments with a distinct border in between price intervals which is what we need in order to establish support and resistance levels.

We have in this example not motivated why we choose to divide our data into exactly 6 clusters so we will therefore go through the metrics for this example as well.

10.3 Choosing numbers of clusters for Kmeans

For Kmeans we have chosen five metrics to look at, the three general performance metrics *Silhouette score*, *Calinski-Harabasz* and *Davies-Bouldin* and the two Kmeans-specific metrics; *V-Measure* and *Elbow method*.

The performance metric for each number of clusters for each general performance metric is shown in Figure 7 and the Kmeans specific metrics are shown in Figure 8.

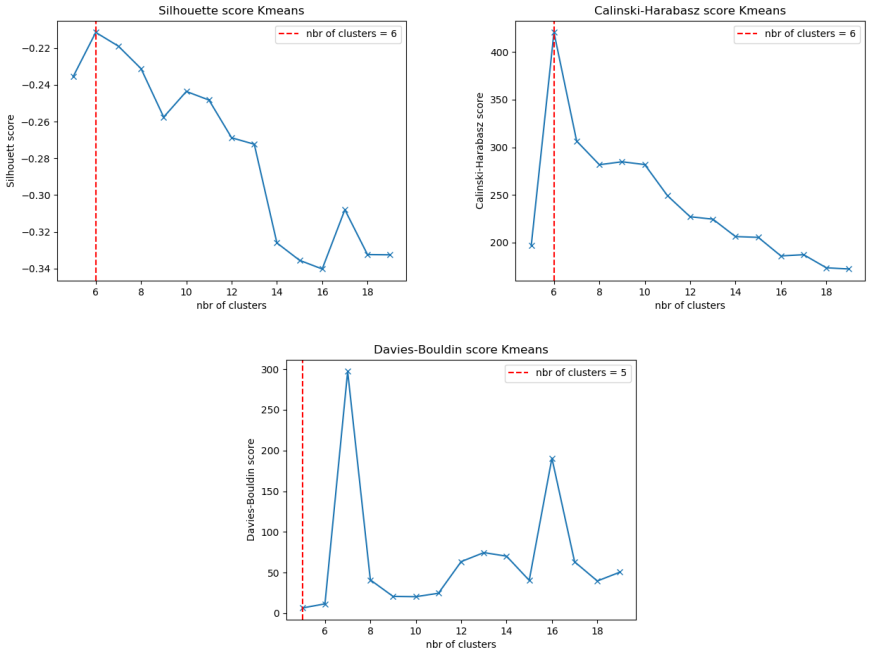

Figure 7: *Shows the general performance metrics used on Kmeans partitioning of the data set introduced in Figure 4 for different number of clusters. Top left shows the Silhouette score for the partitioning, top right shows the Calinski-Harabasz and bottom shows Davies Bouldin-score with all graphs having an indication of what the optimal number of clusters appear to be (6,6 and 5).*

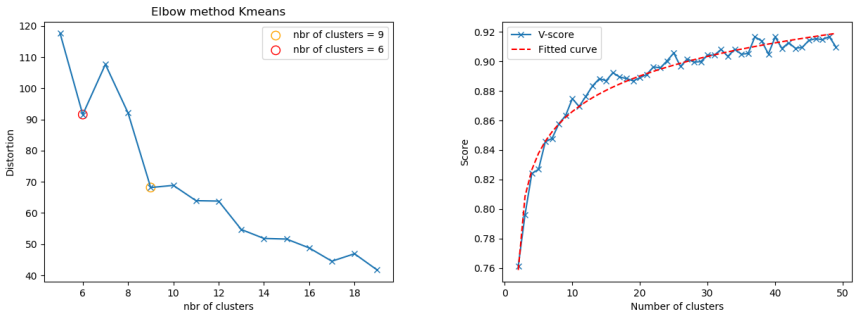

Figure 8: *The left graph illustrates the distortion scores for different number of clusters when Kmeans was used to partition the data introduced in Figure 4, here we can see two suggestions of what the optimal number of clusters might be (6 and 9). The right graph shows the V-scores as a function of number of clusters for the same setting. Here we can see that up to 50 clusters the performance gets better. We chose to not test for higher number of clusters than 50 since the convention is to have at most around half that many support and resistance levels [3].*

The interpretation of the general performance metrics is that 6 number of clusters is probably a good choice. Looking at the Kmeans-specific methods the elbow method appears to suggest either 6 our 9 clusters with subjectively 9 clusters being a slightly likelier choice.

What the V-measure shows us is that the cluster performance keeps increasing with an increase in clusters at least up to 50 clusters. In order to get a definitive result we would like to see some form of plateauing which isn't happening in this interval. What we can say is that the majority of the increase in performance comes between 2-20 clusters, which to some extent motivates us to focus our investigation on that interval. This is also what tends to happen in the financial industry where the number of support and resistance levels that are specified tend to be around 10 per day within a given time period [3].

10.4 How GMM partition our data

In order to better visualise the GMM method we create a histogram for the data points during our chosen week where tick counts and price is normalized (this is needed for the GMM method to work properly). This histogram is shown in Figure 9.

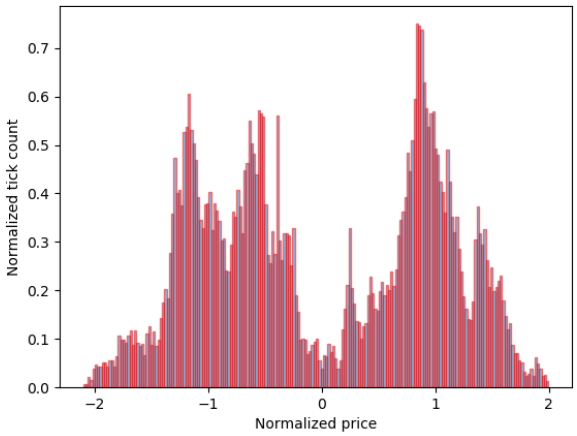

Figure 9: *Shows a histogram of the tick data introduced in Figure 4 with normalised price and tick counts.*

Using sklearns GMM method on the same data as is depicted in Figure 9 we get the distribution fitting shown in Figure 10. Here the label of each price level corresponds to the distribution with the highest value at that price level.

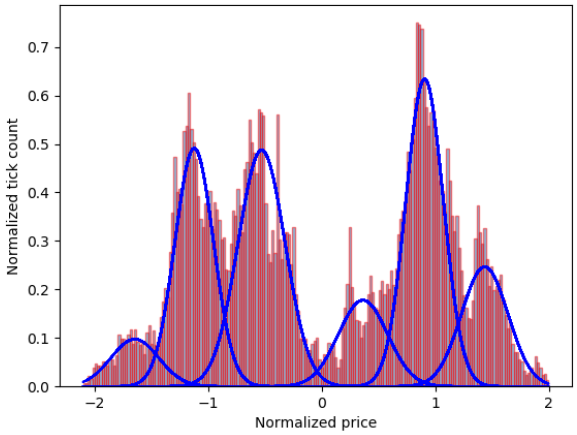

Figure 10: *Shows how the GMM model partitions the tick data introduced in Figure 4. Here each blue curve represents a Gaussian distribution designed to fit over the histogram from Figure 9 using EM and 6 separate distributions.*

Again we haven't shown why 6 clusters appears to be a good choice as number of clusters so therefore we go through the metrics in the same manner as we did with the Kmeans.

10.5 Choosing numbers of clusters for GMM

The general performance metrics are the same as they were for Kmeans; *Silhouette score*, *Calinski-Harabasz* and *Davies-Bouldin*, and the two GMM-specific metrics are *Shapiro-Wilks* and *Kolmogorov-Smirnov*.

The performance metric for each number of clusters for each general performance metric is shown in Figure 11 and the GMM specific metrics are shown in Figure 12.

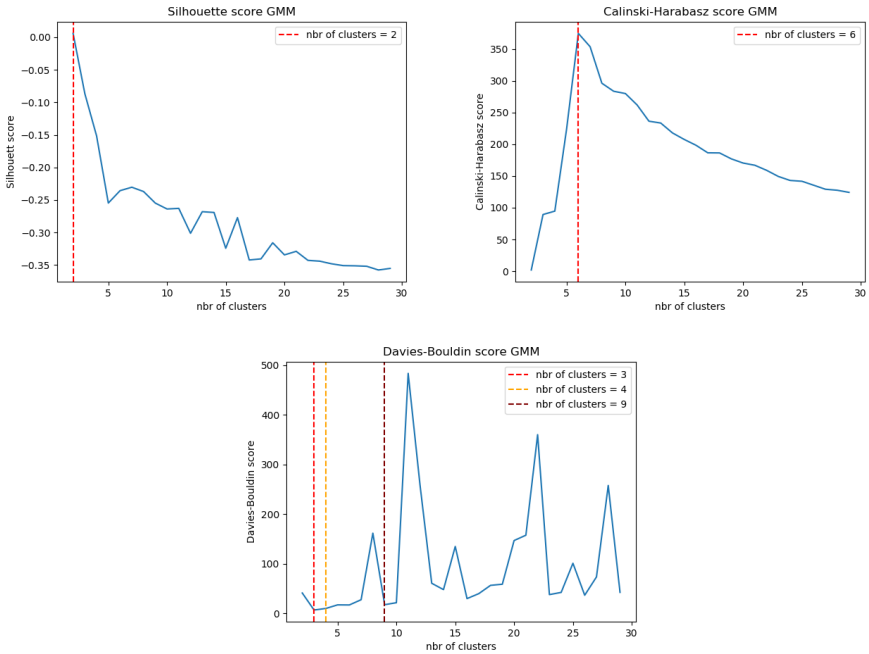

Figure 11: *Shows the general performance metrics used on GMM partitioning of the data set introduced in Figure 4 for different number of clusters. Top left shows the Silhouette score for the partitioning, top right shows the Calinski-Harabasz and bottom shows Davies Bouldin-score with all graphs having an indication of what the optimal number of clusters appear to be.*

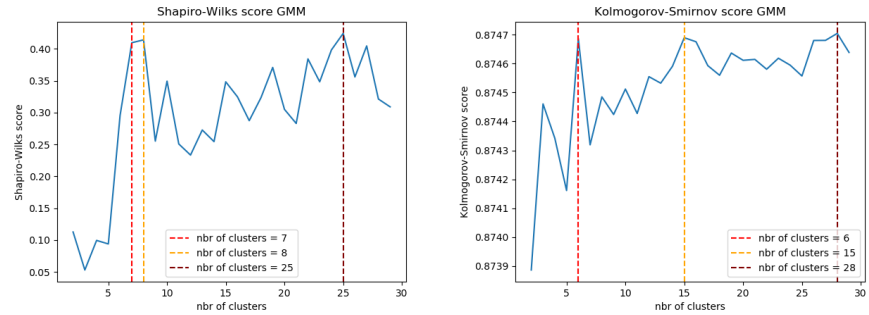

Figure 12: *The left graph illustrates the Shapiro-Wilk scores for different number of clusters when GMM was used to partition the data introduced in Figure 4, here we can see three suggestions of what the optimal number of clusters might be (7 ,8 and 25). The right graph shows the Kolmogorov-Smirnov scores as a function of number of clusters for the same setting. Here we can see that 6,15 and 20 clusters seems to be a good choice.*

In contrast to the Kmeans model the general performance metrics can't seem to agree on a good choice for number of clusters. Looking at the GMM-specific

metric they seem to point towards 5-6 clusters being a good choice of cluster (which also agrees with the Calinski-Harabasz score in Figure 11). From theses results we therefore choose 6 clusters for the GMM-method.

Now that we have chosen an optimal number of clusters to use for the two different methods we want to investigate if the methods have actually found support and resistance levels. We therefore initiate mock trades (see Appendix C) where we pretend to initialise trades and then observe if we hit a support and resistance level, and if so see if we bounce at that particular level. A bounce would then count as a *target* and a breakthrough would count as a *SL*.

In order to have a comparing element we also tried to use a heuristic approach to the problem. Meaning that we looked at the price curve and tried to determine support and resistance levels by identifying price-levels for which the curve bounces by eye. This is what is referred to as the *Trendline*-mehtod which we described briefly in Section 4.

11 Results

In this section we will go through the results from three different parts of the project; *Price-activity correlation, Support and resistance levels* and *target vs stop loss.*

11.1 Correlation

The results received when examining correlation is a good indication whether or not activity has any effect on price, or at least if the two are linked in some way. If there proves to be a high correlation between activity and price the chance is higher to find useful information in the tick data. An example of one days worth of data is depicted in Figure 13 where three sub graphs are shown illustrating the variance of the price, activity and price over the same time period.

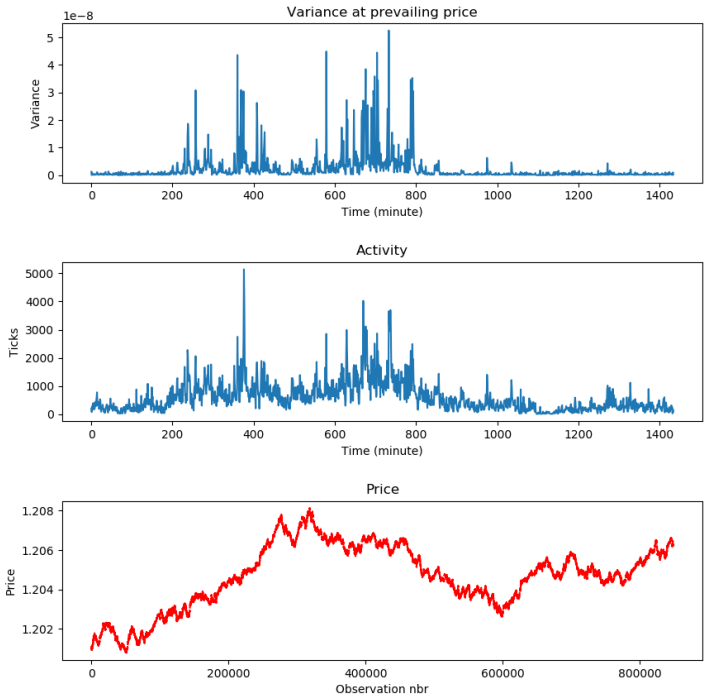

Figure 13: *Shows the correlation between price-variance and activity in the form of number of ticks over a given time period (entire day of 2019-01-02) as well as the price curve during the same time period. One can observe here that the peaks for activity and variance seem to match pretty well which indicates some form of correlation between the variables. This is both expected and desirable in order to extract information about price changes using tick data. Variance and activity are both measured per minute.*

A more quantifiable result is perhaps the correlation metrics described in Section 6, the results of which are shown in Table 1.

Correlation measure	Result
Pearson	0.75126
Spearman	0.82789

Table 1: *Shows the Pearson and Spearman correlations between activity (in the form of ticks) and price-variance for the EUR-USD exchange rate-curve during 2019-01-02. We can see that there is a positive correlation between the variables which means that if the number of ticks per minute increases the variance of the price per minute increases.*

We can observe that there does indeed appear to be some form of correlation between the two variables.

11.2 Support and resistance levels

The results in this section shows the computed support and resistance levels for the data set introduced in Figure 4 using both our altered Kmeans (Figure 14) and GMM (Figure 15) and they will be compared to each other as well as to the heuristic approach (Figure 17).

11.2.1 Kmeans

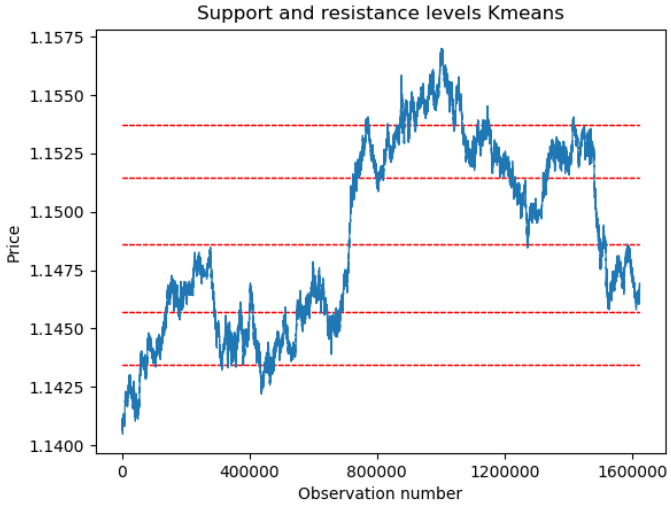

Figure 14: *Shows the support and resistance levels superimposed over price data from 2019-01-06 to 2019-01-11. The support and resistance levels are based on the partitioning done by Kmeans as depicted in Figure 6.*

11.2.2 GMM

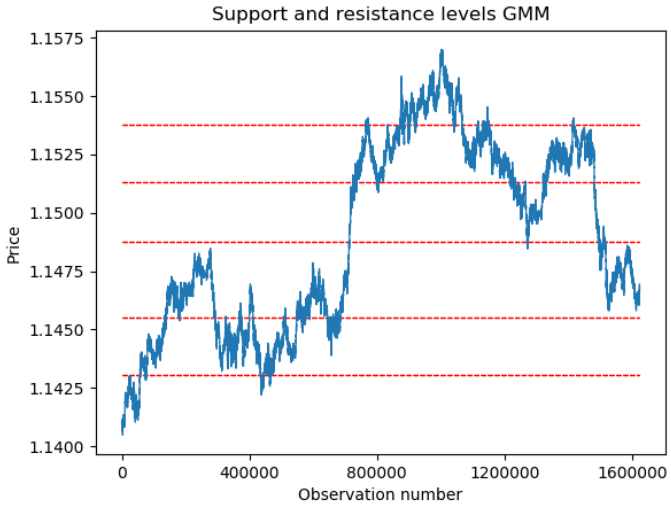

Figure 15: *Shows the support and resistance levels superimposed over price data from 2019-01-06 to 2019-01-11. The support and resistance levels are based on the partitioning done by GMM as depicted in Figure 10.*

11.2.3 Side-by-side comparison

In order to check if there are any significant differences between the support and resistance levels determined by the different methods we look at a side by side comparison between the results shown in Figure 14 and 15. This comparison is shown in Figure 16.

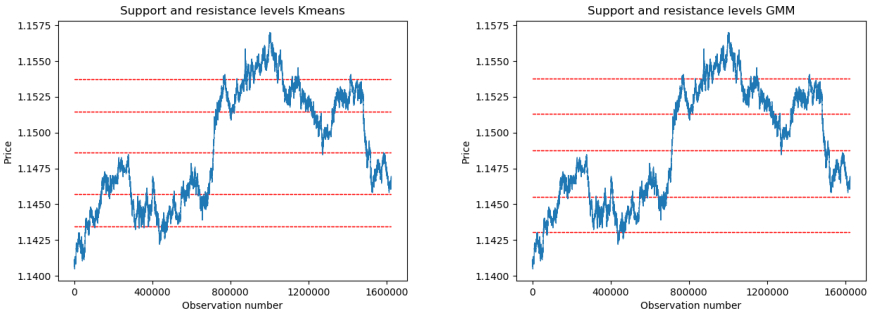

Figure 16: *Shows a comparison of the support and resistance levels established by the altered Kmeans and the GMM method respectively based on tick data from 2019-01-06 to 2019-01-11.*

We can see that there are some marginal differences in the support and resistance levels between the two models.

11.2.4 Heuristic approach to support and resistance levels

We also did a heuristic approach using the *Trendline*-method to determine support and resistance levels by looking at the price data and drawing support and resistance levels by hand. The results for this approach are presented in Figure 17.

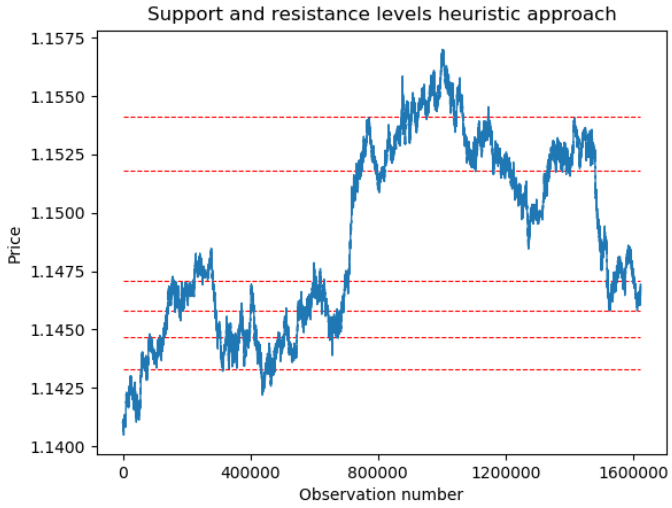

Figure 17: *Shows the support and resistance levels superimposed over price data from 2019-01-06 to 2019-01-11. The support and resistance levels are based on an heuristic approach called Trendlines where the support and resistance levels have been determined by hand.*

11.3 Target vs stop loss

The results in this section comes from initiating mock trades within certain time windows specified by Century Analytics. Since these windows constitute possibly lucrative information we are not allowed to show or discuss the actual windows in this thesis. We can however show how many of these trades that were initiated during the first 5 months of 2019 and whether or not these trades resulted in a target-score, a SL-score or a noscore (see Appendix C how this investigation was made) and the results are shown in Table 2.

Month	GMM			Kmeans		
Jan	target	SL	noscore	target	SL	noscore
	47	50	6	48	45	10
Feb	target	SL	noscore	target	SL	noscore
	42	40	7	42	44	3
Mar	target	SL	noscore	target	SL	noscore
	43	31	6	42	33	5
Apr	target	SL	noscore	target	SL	noscore
	29	18	10	23	26	8
May	target	SL	noscore	target	SL	noscore
	32	39	10	34	39	8
Total	target	SL	noscore	target	SL	noscore
	193	178	39	189	187	34
Total (%)	target	SL	noscore	target	SL	noscore
	47.1	43.4	9.5	46.1	45.6	8.3

Table 2: *Shows the results of the mock trades initiated during the first five months of 2019. The methodology behind this investigation is exemplified in Appendix C.*

11.4 Statistical significance

In order to evaluate the statistical significance for the "target vs stop loss"-results shown in Table 2 a P-value test was performed. The P-value was examined both for the entire 5 months investigated as well as for each month separately. The results for P-values are shown in Table 3.

Month	P-val. GMM	P-val. Kmeans
Jan	0.6576	0.4179
Feb	0.4561	0.6267
Mar	0.1003	0.1778
Apr	0.0719	0.7159
May	0.8288	0.7586
Total	0.2337	0.4794

Table 3: *Shows the results of the P-value investigation for each method and month during the first five months in 2019 as well as a combined P-value score for the whole five months.*

12 Discussion

In this section we will go through the significance of each result from Section 11, discuss which model between the GMM and the Kmeans that seemed to perform the best and try to answer the primary questions put fourth in the problem setting (Section 2). Some proposals for further studies based on the results in this thesis will also be discussed.

12.1 Significance of the results

In this section we will go through the significance of the three main investigations that were made in this thesis; Correlation, Support/resistance levels and the target vs stop loss investigation.

12.1.1 Correlation

Looking at the correlation plot in Figure 13 it does appear that activity and variance correlates to each other. This is pretty much what we expected but it is a reassurance that there might be valuable information to gather from tick data. The results depicted in Table 1 reinforces what we see in Figure 13, maybe with the additional indication that the relation between activity and variance isn't strictly linear, which is shown by the Spearman correlation being slightly higher than the Pearson correlation. Note however that the difference is so small that we can probably not draw any definitive conclusions whether or not the correlation is linear.

12.1.2 Support and resistance levels

Looking at Figures 14-16 it does appear that the partitioning finds some form of support and resistance levels that, at least to the eye, looks reasonable. Particularly interesting is the fact that both methods produced quite similar results (see Figure 16) which is some form of indication that using activity as a marker for support and resistance levels is a reproducible technique.

What is not shown in these results however but that are important to address is that since the cluster selection process for GMM versus Kmeans are different it means that they may not agree on how many clusters to use for a given data set. This is most likely the reason behind why we see a discrepancy between the results in target vs stop loss in Table 2. One clear example of this is the results for April where the difference in P-value is very big between GMM and Kmeans.

Looking at the differences between the both clustering methods using tick data compared to the heuristic approach based on *Trendlines* we can see that the results are markedly different. This does in some aspect support the idea of using clustering methods instead of relying on a human since the clustering methods seem to find information that a human would miss.

Preferably one would need to to the same kind of experiment as we did in this thesis where the results were tested using mock trades, however this would mean going through around 400 price curves and propose support and resistance levels for each one, this might be feasible but certainly not desirable.

12.1.3 Target vs stop loss and statistical significance

If we start by looking at Table 2 one could argue that since the percentage of times we hit target is higher than the percentage of times we hit stop loss we have succeeded in making a profitable model in both cases (GMM and Kmeans). However if we look at the results in Table 3 we get the notion that this might not be the case. What Table 3 says is that, especially in the case of the Kmeans model, the results might as well be random. In the case of the GMM the results are a bit more intriguing. When using P-value to determine statistical significance one usually assigns a cut-off threshold of for example 5 or 10%, meaning that if the P-value is smaller than say 10% we assume that we can reject the null hypothesis (in our case we want to reject the null hypothesis that the outcome of the target vs stop loss-results are random). The GMM method produces a P-value for the entire time period of approximately 20%, meaning that we probably cannot reject the null hypothesis, but we are in a bit of a Gray area. Especially interesting is that some months during 2019 the P-value is as low as 7%, however these values are based on quite a small number of data points so it could potentially be hazardous to draw any strong conclusions from these results.

12.2 GMM or Kmeans as a method to establish support and resistance levels

Based on the results from this thesis we would probably pick GMM as the potentially more viable clustering method for this problem. This is mostly backed up by the statistical significance shown by the GMM model in Table 3 where the results point towards the target versus stop loss being less likely to be a random outcome compared to the Kmeans model.

The reason to why GMM seems to perform better might be difficult to answer, but one reason could be that the data that the model is fed (see Figure 4) is

more suitable to be modelled using a Gaussian distribution rather than using centroids. Looking at Figure 4 one could argue that the peaks look suitable to model a Gaussian distribution after. This conclusion might be further motivated by the fact that we needed to introduce constraints to the Lloyds kmeans (i.e create our own Kmeans) in order to partition the data correctly. The original Kmeans would rather partition the data introduced in Figure 4 in vertical segments (as in Figure 5) rather than horizontal (see Figure 6), leading us to believe that the data in reality may not be entirely suitable to perform Kmeans clustering on.

So to answer the two questions stated in the beginning of this thesis:

1. **Can clustering methods such as Kmeans and GMM be used to partition tick data from EUR-USD exchanges efficiently?**

   - ***Yes***, *the clustering part of this thesis is quite straight forward and it is possible to divide tick data using an altered form of Kmeans as well as GMM. The word efficiently is subjective, but looking at Figures 10 and 6 one could argue that the results of the clustering of both GMM and Kmeans look reasonable.*

2. **Can this partitioning of tick data be used to establish support and resistance levels for EUR-USD exchanges with any statistical significance?**

   - *The answer to the first part of this question is* ***yes****, it is possible to create support and resistance levels that looks reasonable, such as the ones produced in Figure 14 and 15. The second part of the question is more of a Gray area but the answer is probably:* ***no****, not quite. However there are indications that show that with a bit more investigation and further studies there might be ways to make the model more viable, some of these are discussed in the "proposed further study"-section.*

12.3 Proposed further study

During this thesis we have realised that there are many areas whitin this thesis that could benefit from further analysis. Most of the improvements and new areas of interest are connected to the mock trades and how these are specified.

12.3.1 Pip-limits

One such example is the pip limits used to define target and stop loss scores (see Appendix C). We opted to use 10 pips to define the target and SL-limits after

discussions with the team at Century Analytics. One could however choose to have pip limit of 20 or 30 pips and then the results could be significantly different due to the data being so volatile.

12.3.2 Intra week vs intra day

Another example is the data windows that were used. In this thesis we only looked at trading windows that could be as short as one hour and this means that in some cases when initiating mock trades the price would not have time to move into a support and resistance level. This is why there are quite a few noscores in Table 2. Having fewer noscores could potentially give a better insight in whether or not the results we got are statistically significant.

12.3.3 Having 'soft' support and resistance levels

One interesting investigation would be if one could use soft support and resistance levels. In some cases we found that the price would approach a support an resistance level but bounce back just before it hit. In our setting this would count as the price not reaching the support and resistance level and thus would not initiate a target vs stop loss investigation. If one instead would use a setting where coming near a support and resistance level would count then we could potentially get fewer noscores.

An interesting addition to this investigation would also be to have a gliding trading scale, meaning that the closer the price came to the support and resistance level the more a trader would invest. This would create a more dynamic system than having a hard limit where we do all the investment as is the case in our setting.

13 References

[1] *Murphy, C. "Support and Resistance Basics", Investopedia, March 2020, investopedia.com. Accessed 8 Apr. 2020.*

[2] *Ahmad, F., "Technical Analysis Series — Article 1: Support and Resistance", Medium, September 2019, medium.com, Accessed: 2020-04-08*

[3] *Osler, C. L.,"Support for Resistance: Technical Analysis and Intraday Exchange Rates", Research Gate, February 2000, researchgate.net, Accessed: 2020-04-08*

[4] *Hayes, A.,"Moving Average (MA)", Investopedia, March 2020, investopedia.com, Accessed: 2020-04-08*

[5] *Kuepper, J., "Fibonacci and the Golden Ratio", Investopedia, March 2020, investopedia.com, Accessed: 2020-04-08*

[6] *saturdayquant, "ml_strat", GitHub, April 2017, github.com, date: 2020-04-08*

[7] *judopro, "Using Machine Learning to programmatically determine Stock Support and Resistance Levels", Medium, December 2019, medium.com, Accessed: 2020-04-08*

[8] *Kenton, W., "Order book", Investopedia, Jan 2020, investopedia.com, Accessed: 2020-04-10*

[9] *Spearman, C. "'General Intelligence," Objectively Determined and Measured." The American Journal of Psychology, vol. 15, no. 2, 1904, pp. 201–292. JSTOR, www.jstor.org/stable/1412107. Accessed 10 Apr. 2020. Source*

[10] *Laerd statistics, "Spearman's Rank-Order Correlation", statistics.laerd.com, Accessed: 2020-04-10*

[11] *Lloyd, S., "Least Squares Quantization in PCM.", IEEE Transactions on Information Theory 28, no. 2, March 1982: 129–137. doi:10.1109/tit.1982.1056489. Source*

[12] *Dempster, A.P; Laird, N. M.; Rubin, D. B., "Maximum Likelihood from Incomplete Data via the EM Algorithm", 1977, Source*

[13] *Rousseeuw, P., "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis", 1987 Source*

[14] *Caliński, Tadeusz JA, Harabasz. "A Dendrite Method for Cluster Analysis". Communications in Statistics - Theory and Methods. 1974. 3. 1-27. 10.1080/03610927408827101, Source*

[15] *Thomas, R., Carlos, J., & Peñas, S., Matilde & Mora, Marco. (2013). "New Version of Davies-Bouldin Index for Clustering Validation Based on Cylindrical Distance". 49-53. 10.1109/SCCC.2013.29. Source*

[16] *Rosenberg, A., & Hirschberg, J., "V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure", Association for Computational Linguistics. 2007, 410-420, Source*

[17] *AlindGupta, "Elbow Method for optimal value of k in KMeans", GeeksforGeeks, geeksforgeeks.com, Accessed: 2020-04-08*

[18] *The scikit-yb developers, "Elbow Method", SciKit-Yellowbrick, scikit-yb.org, Accessed: 2020-04-08*

[19] *Li, L., "K-Means Clustering with scikit-learn", Towards data science, May 2019, towardsdatascience.com, Accessed: 2020-04-08*

[20] *SHAPIRO, S. S., Wilk, M. B., "An analysis of variance test for normality (complete samples)", Biometrika, Volume 52, Issue 3-4, December 1965, Pages 591–611, https://doi.org/10.1093/biomet/52.3-4.591, Source*

[21] *SPSS tutorials, "SPSS Shapiro-Wilk Test – Quick Tutorial with Example", SPSS, spss-tutorials.com, Accessed: 2020-04-11*

[22] *Arsenault, M. O., "KOLMOGOROV–SMIRNOV TEST" , Towards data science, Nov 2017, towardsdatascience.com, Accessed: 2020-04-10*

[23] *Zaiontz, C., "One-Sample Kolmogorov-Smirnov Table" , Real Statistics, real-statistics.com, Accessed: 2020-04-10*

[24] *Razali, N. M., & Wah, Y. B., "Power comparisons of Shapiro-Wilk, Komogorov-Smirnov,Lilliefors and Anderson-Darling tests", Faculty of Computer and mathematical Sciences, Universiti Teknologi MARA, Journal of Statistical Modelling and Analytics Vol. 2 No.1 21-33,2011, Source*

[25] *Samuels, P., "How do we know which test to apply for testing normality?", Research Gate, Sep 2015, researchgate.com*

[26] *Lee, A., "P-values Explained By Data Scientist", Towards Data Science, 2019, towardsdatascience.com, Accessed: 2020-04-03*

A Altered Kmeans

The initial parts of the altered Kmeans are the same as for Lloyd's algorithm but the way that the cluster centers (centroids) positions are updated differs. Lloyd's algorithm uses euclidean distance to identify the middle point of a cluster, but the algorithm we used for this thesis is based on the calculations of center of mass using distances and point masses shown in 30, with x_{cm} being the center of mass in the x-dimension, M is the total mass and x_i and m_i is the x-coordinate and mass of data point i.

$$x_{cm} = \frac{\sum_{i=0}^{N} x_i m_i}{M}, \quad M = \sum_{i=0}^{N} m_i \tag{30}$$

Where in our case mass m is instead activity and length x is instead price.

Algorithm 3: Altered Lloyds

Data: Dataset $\mathbb{D}$ containing price x and activity a, number of clusters k.

Result: Vector of cluster label for each data point.

Start by randomly initialize a set of k means $\{\mu_1^1,...,\mu_k^1\}$ *along the price-axis.*

while Assignment $S_i^t \neq S_i^{t-1}$ **do**

  for each point $x_i \in \mathbb{D}$ **do**

    $S_i^t = \left\{ x_p : \|x_p - \mu_i^t\|^2 \leq \|x_p - \mu_j^t\|^2 \quad \forall j, 1 \leq j \leq k \right\}$

    $\mu_i^t = x_{i_0} + |A_i^t|^{-1} \sum_{x_j \in S_i^t} x_j a_j \quad A_i^t = \sum_{a_j \in S_i^t} a_j, \; x_{i_0} = min_x\{S_i^t\}$

  end

end

In layman's terms we could say that we are looking for the "centers of mass" along one dimension.

B Scrambling edges for V-measure

Figure 18 shows an example using data taken from an arbitrarily chosen day from 2019 where a Kmeans method has been used to initially partition the data

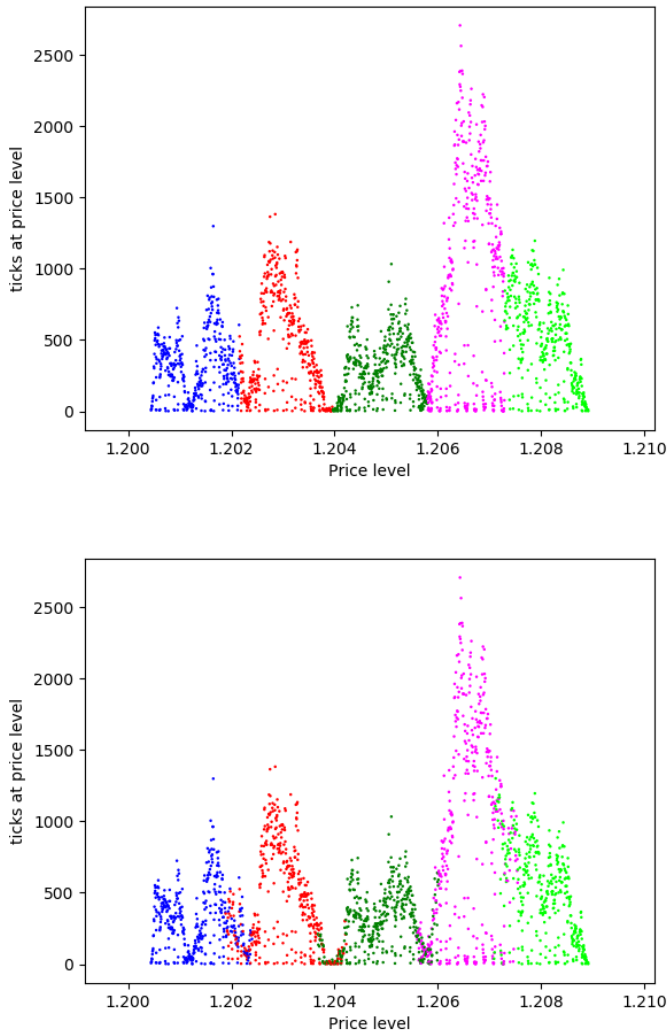

Figure 18: *Shows how the Kmeans partitioning looks before and after scrambling of edges when looking at a data set extracted from 2019-02-04 to 2019-02-05. One can see in the borders in between neighbouring clusters in the bottom figure that they overlap to some extent.*

Here the bottom illustration in Figure 18 is what is considered the ”ground truth”. The reasoning behind this method is that we assume that the Kmeans partition the data almost entirely ”correct” but that it miss classifies some data

points in the border in between clusters.

Using this scrambled partition of the data as the ”truth” and the originally partition as the ”guess” we can then produce a V-score using the equations described in Section 8. By repeating this process for many different number of clusters we can get a graph such as the one in Figure 8.

C Mock trades

In order to simplify for the reader we will only go through two examples produced by initiating mock trades when having used GMM to partition data. One where we reach a target result and one where we reach a stop loss result. To further facilitate for the reader we do not add any additional data and instead use the same data set as in Section 10.

The target and SL scores are here defined by a range of 10 pips. So if we initiate a trade which turns out to reach a support and resistance level we then set two separate levels above and below this "collision" point. One target level and one SL level. Depending on if we move *down* in order to reach a support and resistance level the target is set at a price 10 pips *above* the current price and the SL is defined at a price 10 pips *below* the current price. This is then reversed if we move *up* into a support and resistance level.

When using real data we would use a training set which consists of data before the specified time window to establish support and resistance levels and then use a prediction set (which consists of the window specified by Century Analytics) wherein the mock trades are initiated.

If we have a case where a support and resistance level is not reached within the time window we count that as a *noscore*.

C.1 Example of target score

Figure 19 shows an example where a trade is initiated and a target score is achieved.

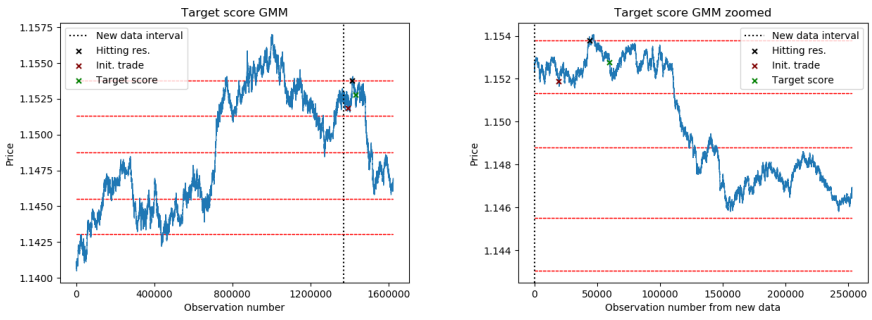

Figure 19: *Shows an example of a mock trade which resulted in a target score initiated in a setting defined by support and resistance levels computed using a GMM method. The left graph shows the whole data interval which is used in the investigation with all data before the dotted line being defined as the training set (on which the clustering is made) and the data after the dotted line represents the prediction set. The right graph shows snippet of the right graph where we have zoomed in on the prediction set*

C.2 Example of SL score

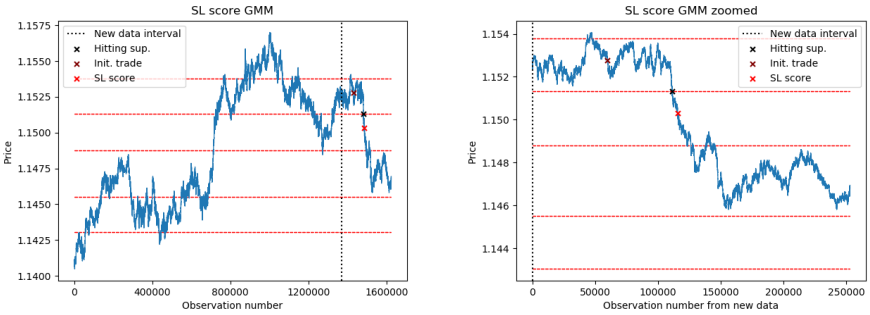

Figure 20: *Shows an example of a mock trade which resulted in a SL score initiated in a setting defined by support and resistance levels computed using a GMM method. The left graph shows the whole data interval which is used in the investigation with all data before the dotted line being defined as the training set and the data after the dotted line represents the prediction set. The right graph shows snippet of the right graph where we have zoomed in on the prediction set*