Submitted to the Annals of the Department of Statistics and Operations Research

SCRAPING THE SYNTHETIC RISK AND REWARD INDICATOR FROM FUNDS' KEY INFORMATION DOCUMENTS

By Fabian Blasch

University of Vienna

INSERT ABSTRACT

1. Introduction. According to BGBl. II Nr. 265/2011, effective 2011, every undertaking for collective investment in transferable securities (UCITS), has to publish a key information document (KID)(FMA, 2011). Said document is supposed to inform potential and current investors about various characteristics of the investments at hand. Besides a short description of the investments as well as performance measured against a benchmark, this document also contains a synthetic risk- and reward indicator (SRRI). As the name suggests, the purpose of this indicator is to measure the risk associated with an investment in the respective fund. How the SRRI is derived depends on the class of investment fund and will be described in greater detail in a separate subsection. Independent of the exact procedure of derivation the underlying interpretation of risk is measured via the volatility of returns.
The aim of this thesis is to firstly, give a short and precise introduction into the calculation of the SRRI.Then the data is briefly presented and the approach to measuring extraction performance is discussed. The following sections' focus shifts to extracting the SRRI which is displayed as a graph, usually on the first page of every KID. In order to keep the structure as simple as possible the proccess is split into a parent function which calls a variety of helper functions, all functions are implemented in R, besides the base library, the use of further libraries is pointed out in the descrription of the individual functions (R Core Team, 2021).

MSC 2010 subject classifications: Primary Key1, Key2; secondary Key3
Keywords and phrases: Scraping, LaTeX 2$_\varepsilon$

2. SRRI. The SRRI aims to measure risk via the volatility of weekly returns from the last 5 years, should weekly retuns be unobtainable, the calculation may be executed using montly returns.Accordingly, by ordinance the SRRI may be obtained as follows

$$\sigma_f = \sqrt{\frac{m}{T-1}\sum_{t=1}^{T}(r_{f,t} - \overline{r_f})^2},$$

where $r_{f,t}$ is the fund's return and $\overline{r_f}$ represents the mean of returns over T periods. Then scaling via m, the return frequency within a year, yields the standard deviation of yearly returns. For illustrative purposes, the calculation using weekly returns would result in $m = 52$, as there are 52 months in a year and $T = 5{*}52 = 260$ the number of months in 5 years. To obtain the SRRI which is measured on a scale from 1 to 7 the regulating autorithy provides a table.

TABLE 1
SRRI Classification

SRRI	σ_f
1	$0\% \leq \sigma_f < 0.5\%$
2	$0.5\% \leq \sigma_f < 2\%$
3	$2\% \leq \sigma_f < 5\%$
4	$5\% \leq \sigma_f < 10\%$
5	$10\% \leq \sigma_f < 15\%$
6	$15\% \leq \sigma_f < 25\%$
7	$25\% \geq \sigma_f$

3. Measuring Scraping Performance. The SRRI is measured on an increasing ordinal scale, with 7 being associated with the highest risk and simultaneously also with the highest return. This would in theory allow for a measurement of predictive performance utilizing the abolute or squared difference between prediction and actual value. However, in this case we want to measure whether the read-out was successful or not. Hence the performance evaluation will be based on predicting correctly. Formally this means, we will use a discrete metric, i.e

$$Accuracy = \frac{1}{n}\sum_{i=1}^{n}\mathbb{1}_{\{pred_i \,=\, act_i\}}.$$

Put differently, the accuracy measure is equivalent to the relative amount of correctly predicted cases.

4. Data. As previoulsy mentioned the KIDs are available to download from the respective fund managing firm's website. The sample contains 110 Key Information Documents, and can be found here. First ten Kapitalanlagegesellschaften (KAG) that manage funds registered in austria were chosen.Then the sample was generated via randomly choosing funds on the respective websites of the fund managing firm. The distribution of the SRRI as well as the amount of KIDs per KAG within the sample are displayed below. Further, to obtain a test set, a stratified sampling proccess was applied, for each KAG two KIDs were randomly chosen. Accordingly, the training set used to optimize function input parameters includes 90 files while the test set contains 20 files. The code to generate the test set may be found here. The rationale behind stratified sampling is that the sample is already quite small, random sampling across all files could thus poorly reflect the true document population. Moreover, for one of the functions applied, a color has to be passed as an argument, hence the stratified sampling allows to determine the color that is passsed via the file location which drastically reduces KAG identification effort for the document at hand.

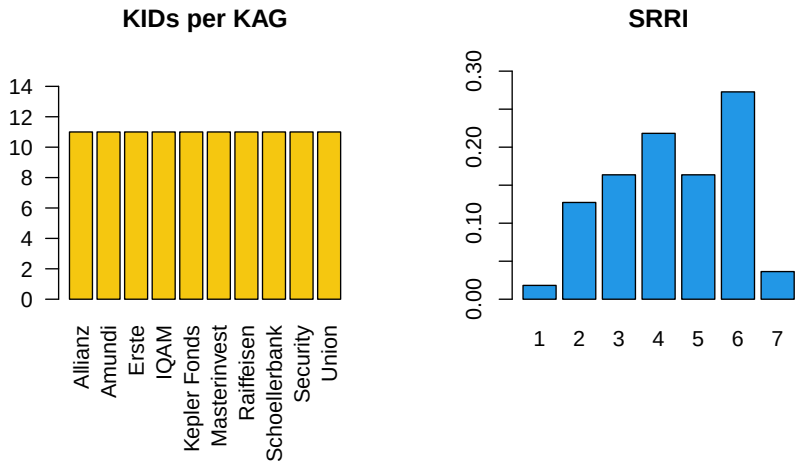

FIG 1. *Number of KIDs per KAG and distribution of the SRRI*

The two depicted plots give a good impression of the sample at hand. We can immediately observe that for all fund managing firms, 11 KIDs are available in the sample. Regarding the distribution of the SRRI, the graph depicted on the right enables us to notice that the sample seems to contain very little funds that are associated with an SRRI of 1 or 7. The low number of funds classified into the first class may be due to the way the standard deviation of returns is converted into the SRRI. The intervals which are used to determin the SRRI increase in length quite signigicantly. The first interval for which $0\% \leq \sigma_f < 0.5\%$ spans a range of half a percent, while the intervall for the classification into the 6th

class $15\% \leq \sigma_f < 25\%$ spans 10%. The other end, namely the SRRI of 7, could be underrepresented because of a lack in demand for such high risk products. Of course this is just mere speculation but interesting nonetheless. In regards to the extraction, to ease illustration of the coming chapters, an example document from Erste can be found here. Additionally, an example of the SRRI graph as it can be found in an Erste KID is provided below.

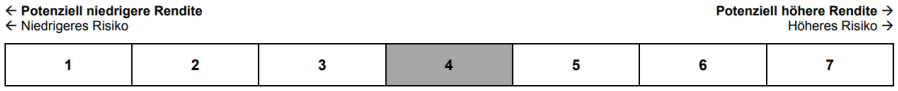

FIG 2. *Example of the SRRI graph from an Erste KID*

5. Extraction.

5.1. *Helper Functions.* Before splitting into the two approaches taken to obtain the SRRI, the helper functions that are utilized in both cases have to be explained. The naive approach relies on knowledge of the background color of the file at hand. In order to obtain this color, one can either assume that it is white, ie the color identification code being #ffffffff or extract the most common color in the pdf. Both options are implemented, since obtaining the background color is a computationally intensive operation. The function called bg_col() first converts the input pdf file into a bitmap utilizing the R package pdftools (Ooms, 2021). A bitmap is an array of dimension $n \times m \times 4$, where n and m can be interpreted as the vertical and horizontal coordinates respectively while 4 signifies that each color entry is split into four substrings, the so called Hex Code of the color. Hex which stands for Hexadecimal would usually imply a string of six characters. However, in the case of inclusion of an alpha value the string includes 8 characters. The alpha value is used to alter the transparency of the color. An example for illustrative purposes, should we be interested in the color of the pixel that lies the furthest to the left on the top of the page we would subset with $m = 1$ and $n = 1$ to obtain the four substrings containing two characters each. Those 4 substrings then represent the color of the pixel. In order to obtain the background color, i.e., the most common color in the document we thus first flatten the array to an $n \times m$ matrix by pasting together the substrings of the Hex code to then find the color with the most entries in the matrix.
Besides the backgroundcolor, it is also important to localize the SRRI scale on the page, this task is designated to coord_id(), which calls two additional helper functions. As each document contains many numbers, to identify which of those make up the SRRI graph, scale_cand_cord() first identifies which single digit numbers could possibly be part of the SRRI (Ooms, 2021). This is done by extracting all single digit numbers between 1 and 7 from the document, including the vertical and horizontal position on the page. Summarized in a matrix, this information is then passed to coord_id(), which first finds the absolute difference in vertical position of all candidates utilizing abs_dis_vec(), including some tolerance level, to then check if there exist 7 single digit entries with the same vertical position. If this is the case,

the function has succesfully identified the position of the SRRI. Accordingly, the function then returns a matrix containing the vertical and horizontal position of each scale entry as well as a string specifying wether the SRRI is displayed vertically or horizontally. Even though not a single occurance of a vertical SRRI depiction is included in the random sample of KIDs, this feature was included for the sake of completeness.

5.2. *Naive Approach.* To establish a benchmark for the extraction performance, the naive approach, as the name might suggest, does not make use of any statistical methods. The function SRRI_ext_rec() first localizes each of the 7 SRRI entries on the page via coord_id(). Then the rectangles as displayed in blue in the figure below are built using vertical (yellow) and horizontal (red) tolerances measured from the location of each SRRI entry. Those tolerances are an argument of the function SRRI_ext_rec(), however, default values are implemented that achieve an accuracy of 100% on all sample documents that are not scanned. Once the rectangles are built, the bitmap containing all color entries is subset to obtain the colors within the rectangle. Subsequently, the percentage of non-background colored pixels is calculated. Finally the rectangle with the least amount of background-colored pixels is choosen as the predicted SRRI.

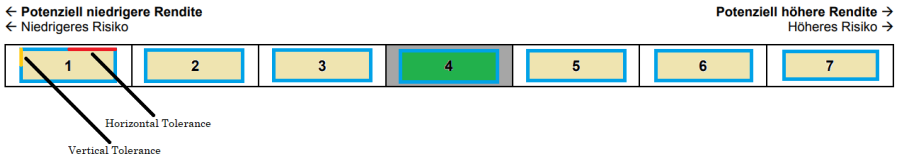

FIG 3. *Illustration of the Naive Approach*

5.3. *Clustering Approach.*

5.3.1. *Motivation.* The second approach is based on agglomerative hierarchical clustering. Before going into detail in regards to the technicalities of this clustering technique, firstly, the idea will be outlined using some graphical illustrations. Assume one knows the color that is used to shade the SRRI, in the case of figure 3 that would be grey (HEX: #a6a6a6ff). Then an intuitive approach would be to subset all pixels in the color of the SRRI shade to then produce some sort of representative value of the horizontal location of the SRRI shade.

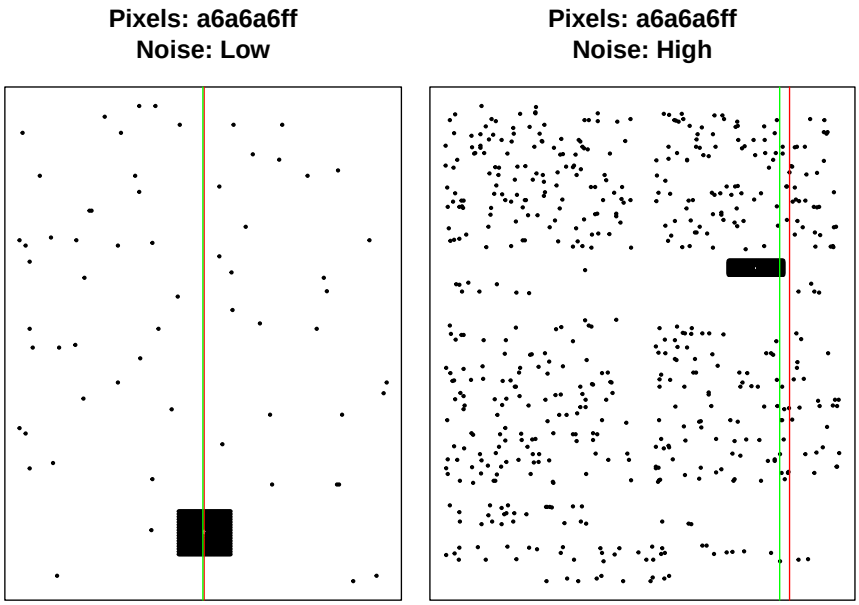

FIG 4. *Pixels in the color of the SRRI shade*

Above we observe the subset of pixels in the color of the SRRI shade for two different documents. Further, the mean of the horizontal coordinates of the pixels is represented by the green line while the median is represented by the red line. Unfortunately, this makes it painfully obvious that neither the median nor the mean of the horizontal coordinates will suffice as a representative of the cloud of pixels in the color of the SRRI shade. To be precise, the median is not a good representative, in a high noise enviroment as displayed on the right. Thus if we manage to reduce said noise to obtain a situation somewhat similar to the one on the left, we may be able to succesfully use the median as a good representative of the horizontal position of the SRRI pixel cloud. This noise reduction may be achieved via unsupervised classification into groups based on agglomerative hierarchical clustering.

5.3.2. *Agglomerative Hierarchical Clustering.* The goal of noise reduction via clustering is achieved by grouping the pixels. The idea is to cluster the set of all pixels S into subsets $G_1, G_2, ..., G_n$, such that the union of all groups $\cup_{i=1}^{n} G_i$ forms a partition on the set of all pixels, i.e., $S = \cup_{i=1}^{n} G_i$ where $G_i \cap G_j = \emptyset$ for $i \neq j$ (Rokach, 2009). Since the grouping is based on the simuilarity across pixel coordinates, we need a measure for similarity. In this thesis the euclidean distance is used to determine similarity.
The approach of agglomerative hierarchichal clustering starts by initially assigning each pixel its own cluster. Then in each iteration based on the minimal distance, clusters are merged until the desired number of clusters is obtained. In detail the algorithm can be described as follows:

Algorithm 1: *Agglomerative Hierarchical Clustering*

Data: N vectors of pixel coordinates $\{x_i\}_{i=1}^{N}$ and p the number of desired clusters.
$\mathbf{X} \leftarrow \emptyset$; `/* Initialize empty set as storage for clusters */`
for $k \leftarrow 1...N$ **do**
  $\mathbf{X} \leftarrow \mathbf{X} \cup \{\{x_k\}\}$; `/* Assign each pixels its own cluster */`
end
$\mathbf{T} \leftarrow \mathbf{X}$; `/* Storage of clusters, to create a dendrogramm at a later stage */`
while $|\mathbf{X}| > p$ **do**
  $G_1^*, G_2^* \leftarrow \underset{G_i, G_j \in \mathbf{X}}{\operatorname{argmin}} \; Dist(G_i, G_j) \; for \; i \neq j$; `/* Find minimal distance */`
  $\mathbf{X} \leftarrow (\mathbf{X} \setminus G_1^*) \setminus G_2^*$; `/* Remove individual occurances */`
  $\mathbf{X} \leftarrow \mathbf{X} \cup \{G_1^* \cup G_2^*\}$; `/* Add union as new cluster */`
  $\mathbf{T} \leftarrow \mathbf{X} \cup \{G_1^* \cup G_2^*\}$; `/* Add union to secondary storage for traceability */`
end

Given that euclidean distance is used to judge similarity the first iteration of the while loop is unambigous as only the distance between pixels is required for exection. However, starting from the second iteration we need to find the distance between pixels, other pixels and the first cluster. Hence, $Dist(G_i, G_j)$ needs to be defined for G_i and/or G_j as clusters. In this thesis three diffrernt approaches to determin the above described distance are considered.

1. **Average-link:** $D(G_i, G_j) = D(\{x_k\}_{k=1}^{K}, \{y_s\}_{s=1}^{S}) = \frac{1}{KS} \sum_{k=1}^{K} \sum_{s=1}^{S} ||x_k - y_s||$
2. **Single-link:** $D(\{x_k\}_{k=1}^{K}, \{y_s\}_{s=1}^{S}) = \underset{k,s}{min} ||x_k - y_s||$
3. **Complete-link:** $D(\{x_k\}_{k=1}^{K}, \{y_s\}_{s=1}^{S}) = \underset{k,s}{max} ||x_k - y_s||$

At a later stage the link will be chosen based on prediction accuracy. For illustrative purposes, the clustering for different links and varying numbers of final clusters is displayed.

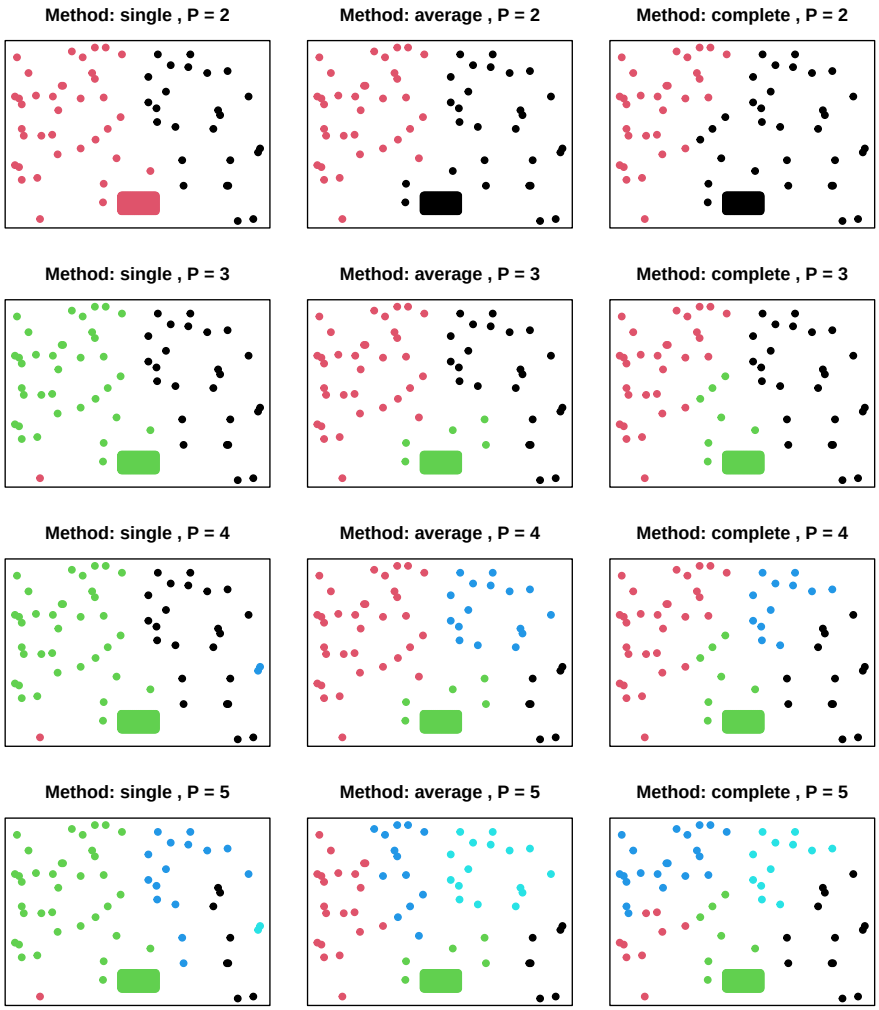

FIG 5. *Example: Different values for P and varying linking methods*

Coming back to the problem at hand, noise reduction, we would like to exclude as many pixels as possible from the coloured square, i.e., keep the cluster that contains the SRRI shade in tact while removing as many pixels that are obviously not part of the SRRI pixel cloud. Figure 5 gives the impression that complete and average linkage seem to be able to better exclude far off pixels from the SRRI pixel cloud.Further, one can observe that for values p between 2 and 5 the SRRI pixel cloud is not split into multiple clusters, thus a higher value for p is probably advisable. Accuracy comparisons accross different values for p and linking methods will show which combination will yield the best performance.

5.3.3. *Implementing the AH Clustering Approach.* The function that executes the clustering approach is called SRRI_ext_loc(), as inputs it requires the document and the HEX code of the color of the SRRI pixel shade. Additionally, one may choose the linking method as well as the number of final clusters as explained in the previous subsection. The function first slices the bitmap of the document horizontally above and below the scale that is located by coord_id(). Then using this subset of the original bitmap all pixels in the color of the SRRI shade are extracted and subsequently clustered into groups in order to reduce noise. Next, all groups containing less than a cutoff value of points are discarded to then choose the group with the lowest vertical variance in pixel position as the one being the SRRI shade. The idea is that the removal of all pixel groups below a number of points ensures that clusters with very little observations do not end up having the lowest horizontal variance by chance. Alternatively, one could also generate features that describe each of the clusters, such as number of pixels within a cluster, average distance of points within cluster, average variance of vertical and horizontal pixel coordinates to then use predictive methods like decision trees or elastic nets to determin which of the clusters most likely contains the SRRI shade.

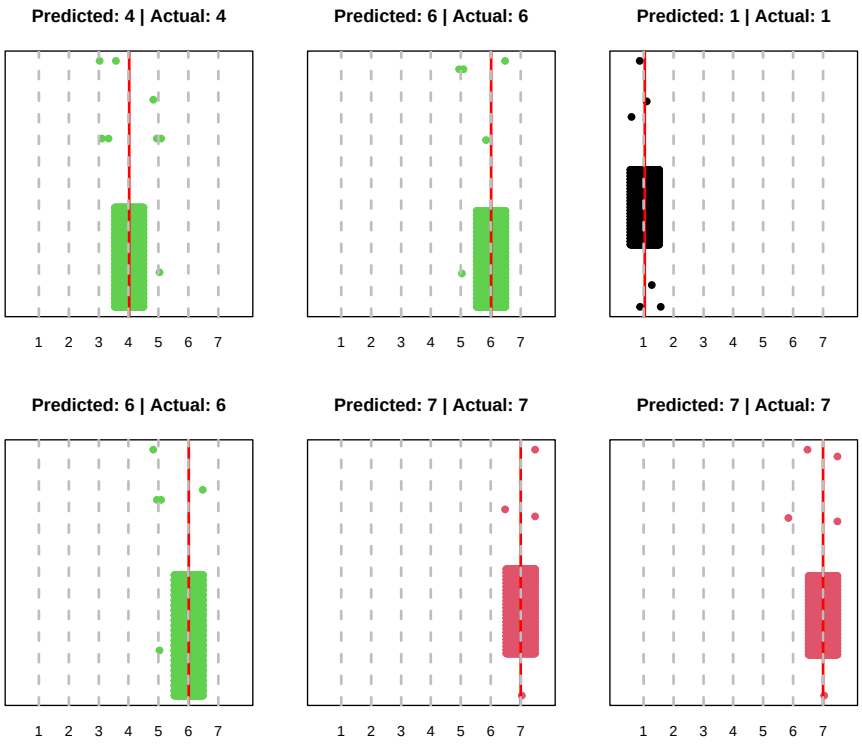

FIG 6. *SRRI extraction via SRRI_ext_loc() using average linkage and* $P = 5$

For the actual proccess of clustering the package fastcluster is used, as hierarchichal clustering is not particularly computationally efficient, i.e., depending on the implementation the computational complexity of the algorithm in time is either $\mathcal{O}(n^3)$ or $\mathcal{O}(n^2)$ (Müllner, 2013; Rokach, 2009). For ease of imagination a graphical depiction of Erste KID SRRI extraction via SRRI_ext_loc() is displayed in figure 6. We observe that the pixel clusters contain very little noise indicating that the function performed as intended.

6. Parameter Choice and Performance Comparison.

TABLE 2
SRRI Classification - In Sample Performance

Tolerance	P	Method	Per Gross	NA	Per Net
20	4	single	1	0.356	0.644
20	6	single	1	0.356	0.644
20	8	single	1	0.356	0.644
20	4	average	1	0.356	0.644
20	6	average	1	0.356	0.644
20	4	complete	1	0.356	0.644
20	6	complete	1	0.356	0.644
20	8	average	1	0.467	0.533
30	8	average	1	0.489	0.511
30	8	complete	1	0.500	0.500

APPENDIX A: CODE

Appendix for code and additional illustrations

A.1. Functions.

REFERENCES

FMA. (2011). Ordinance of the financial market authority. *KID-V, BGBI II 265/2011*.

Müllner, D. (2013). Fastcluster: Fast hierarchical, agglomerative clustering routines for r and python. *Journal of Statistical Software*, *53*(9), 1–18.

Ooms, J. (2021). *Pdftools: Text extraction, rendering and converting of pdf documents* [R package version 3.0.1].

R Core Team. (2021). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. Vienna, Austria.

Rokach, L. (2009). A survey of clustering algorithms, 269–298.