Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Problem statement

More than 2.5 quintillion bytes of data are produced daily as the field of data analysis continues to grow. "Statistical thinking and methodology" has become the framework for disciplines such as education, agriculture, economics, biology, medicine, astronomy, geology, and physics [3], but there is still a lack of accountability and consistency in the field. What is striking in the current practice of data analysis is the lack of progress on this particular subject beyond the development of numerical methods. The rapid increase in computing resources has led to the proliferation of high-dimensional datasets, which are more tedious to efficiently understand patterns in the data and verify numerical estimators. In fact, the "physical limitations of display devices and our [human's] visual system prevent the direct display and instantaneous recognition of structures with higher dimensions than **two or three**" [9] (emphasis mine). One solution is to manually plot each explanatory variable against the response variable, but this becomes computationally tedious and unfeasible to sort through when there are even a few hundred variables. This problem gets even more complicated when considering interaction terms (various transformations or combina-

tions of explanatory variables). Although methods for dimension reduction have been developed [9], it is still unclear how the analyst can easily check the resulting model to ensure that the variables which were culled in the dimension reduction process are actually undesirable. Thus, the problem with current high-dimensional visual analysis is two-fold: (1) there are too many potential plots to sort through manually, and (2) it is tedious to verify numerical results with visual results and vice versa.

In computer science, a framework for "clean code" has been extensively documented and is the accepted industry standard for writing, interacting with, and thinking about code. But in empirical data analysis with large datasets, analysts blindly depend on estimators and hypothesis tests to explore the data and have no justification of their analyses aside from asymptotic, mathematical guarantees. Furthermore, since each estimator inherently performs well or poorly under different settings, data analysts are unable to differentiate between the properties intrinsic to the dataset and the spurious properties the estimators added. Little, a Professor of Biostatics at the University of Michigan, notes that "developing good statistical solutions to real applied problems, based on good science rather than 'cookbookery,' is far from easy" [7]. This lack of agreement and "cookbook" mentality of data analyses has far-reaching consequences. It is simple to run the data through a list of many estimators and cherry-pick the most "interesting" result. Similarly, an analyst can remove undesirable data points without justification or unknowingly fit egregiously incorrect models. Regardless of whether all these situations are performed maliciously or with good intentions, the art of data analysis is unclear without standards. The lack of clear-cut guidelines makes it difficult for analysts to discern the "truth" from the data and avoid the aforementioned pitfalls while simultaneously making it difficult for consumers of the resulting analyses to evaluate how trustworthy it is. This mentality arises due to the difficulty in visualizing high-dimensional data; plotting is

one of the most consistent and universally interpretable "sanity checks" for numerical results.

Consider the following scenario with two different univariate datasets (Appendix A.1 and A.2). The problem is if x contains explanatory power of y. Common numerical analysis techniques yield the results summarized in Table 1.1.

Table 1.1: Numerical analysis in the univariate case. The results suggest that the data are uncorrelated. For Dataset 1, refer to Appendix A.1. For Dataset 2, refer to Appendix A.2

Dependency test	**Dataset 1**	**Dataset 2**
Linear regression	$y = 0.461 + 0.008x$	$y = -0.131 - 0.2699x$
p-values	(2e-16) (0.911)	(0.488) (0.190)
Conclusion	Insignificant	Insignificant
ANOVA p-value	0.9109	0.1896
Conclusion	Insignificant	Insignificant
Shapiro p-value	0.5795	0.1632
Conclusion	Normally-distributed residuals	Normally-distributed residuals
Pearson's correlation	-0.1886	0.0113
p-values	0.1896	0.9109
Conclusion	Uncorrelated	Uncorrelated

Supposing that an analyst must rely on numerical tests alone, the reasonable conclusion to reach would be that x and y are uncorrelated. Given the power to plot quickly and efficiently, however, an analyst would quickly discover that the data exhibits a strong dependency (Figure 1.1). There are certainly many more ways to numerically analyze the data, and in retrospect, it can be argued that an analyst might have tried an estimator that captured the dependency properly. Even then, without the ability to plot, the previous numerical results (which were strongly uncorrelated) cast doubt on the sole correlated estimator.

It is interesting to note that Dataset 2 (Figure 1.1, right) is clearly linear yet common tests of linear correlation (linear regression, Pearson's correlation. See Sec-

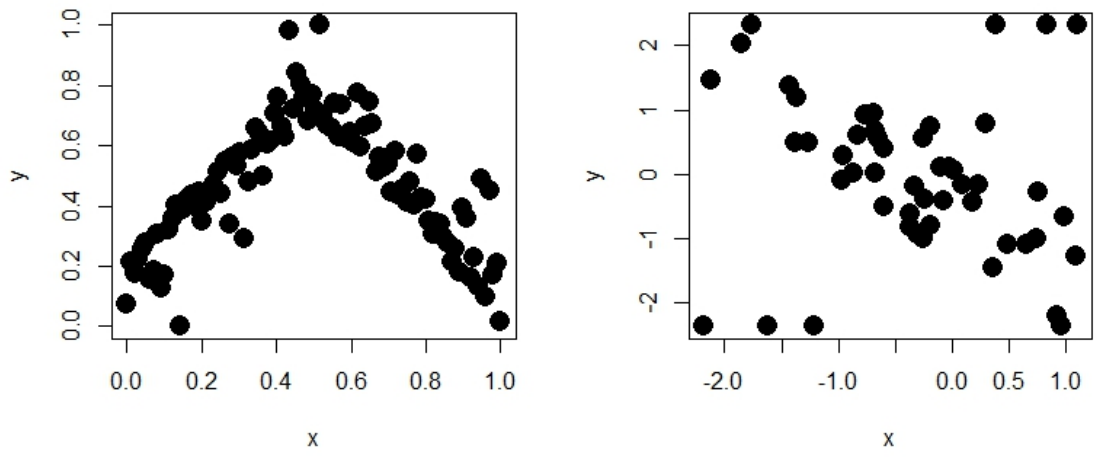


Figure 1.1: Visual analysis in the univariate case. The data exhibits a strong visual dependency but fails common numerical tests of dependence (Table 1.1). *Left*: Dataset 1 (Appendix A.1), *Right*: Dataset 2 (Appendix A.2)

tion 1.2) are not significantly different from 0 (Table 1.1). Indeed, the data used in these examples was purposefully constructed to be dependent but bypass common tests for dependency. However, if it is possible to construct datasets in one-dimension that evade commonly-used numerical methods, it is believable that it is even easier to construct analogous datasets in higher dimensions. Hence, no such standards of "clean analysis" currently exist in data science despite its importance in financial decisions, judicial evidence, government policy, and scientific discovery. Verification of numerical methods is especially important in finance as equity markets are large and involve billions of dollars; portfolio selection often involves determining the relationship among as many stocks as possible in order to be thorough and achieve the best possible portfolio.

1.2 Correlation graphs

Correlation graphs are one way to discover the dependency structure among different stocks whose returns may be represented as random variables following some distribution. Let $G = (V, E)$ be an undirected graph with vertices $V_1, ..., V_d$ (a d-dimensional distribution) and edges $E_{i,j} \in \{0, 1\}$. We set $E_{i,j} = 1$ when there is an edge between V_i and V_j, and 0 otherwise. An edge is drawn between V_i and V_j iff the two random variables are correlated. This graph can be drawn from a correlation matrix $\sum$ where $\sum_{i,j} = corr(V_1, V_2)$ with the following heuristic:

If $\sum_{i,j} > p$, draw edge $E_{i,j}$ where p is the p-value for the desired confidence level

---

We differentiate between "visual correlation" (which can be thought more of as "pairs of variables that marginally appear dependent") and the more common mathematical interpretation of "correlation". More specifically, we would like to visually understand what a correlation graph looks like and compare it to correlation graphs constructed with the traditional interpretation of correlation. What follows is an overview of common numerical methods to estimate the correlation between two random variables.

1.2.1 Pearson's correlation

Pearson's correlation measures the linear dependence among two random variables X and Y. In a population, the correlation is given by

$$\rho_{X,Y} = \frac{\mathrm{Cov}(X, Y)}{\sigma_X \sigma_Y} = \frac{\mathrm{E}(XY) - \mathrm{E}(X)\mathrm{E}(Y)}{\sqrt{\mathrm{E}(X^2) - \mathrm{E}(X)^2}\sqrt{\mathrm{E}(Y^2) - \mathrm{E}(Y)^2}}$$

The formulation above is not as useful in practice as datasets are regarded as samples of a population. Given n observations, the sample expectation is given by

the formula $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$. By substituting into the above and multiplying by n^2/n^2, we can estimate the Pearson's correlation with the following:

$$\rho_{x,y} = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sqrt{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}\sqrt{n\sum_{i=1}^{n} y_i^2 - \left(\sum_{i=1}^{n} y_i\right)^2}}$$

such that $-1 \leq \rho \leq 1$. With perfect positive and negative linear dependence respectively, $\rho = \pm 1$. It is important to note that $\rho = 0$ does not necessarily indicate independence, though it is an indication of **linear** independence.

1.2.2 Spearman's correlation

Spearman's correlation is more broad than Pearson's; it measures monotonic dependence among two random variables X and Y. Monotonic functions are either strictly increasing or decreasing; while linear functions are monotonic, monotonic functions are not necessarily linear. Subsequently, Spearman's correlations may also capture non-linear dependencies. Spearman's correlation is computed by computing the Pearson's correlation among "ranked variables". Each sample observation x_i of X is ranked from 1 to n based on its position relative to $x_j, j \in \{1, ..., n\} \backslash i$. The ranking is also computed for all observations of Y. We then define the difference of a sample (x_i, y_i) as $d_i = x_i - y_i$ and compute Spearman's correlation as

$$\rho_{x,y} = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2-1)}$$

such that $-1 \leq \rho \leq 1$. With perfect increasing and decreasing monotonic dependence respectively, $\rho = \pm 1$. Again, it is important to note that $\rho = 0$ does not necessarily indicate independence, though it is an indication of **monotonic** independence.

1.2.3 Kendall's tau

Similar to Spearman's correlation, Kendall's tau is another method of identifying monotonic dependence among two random X and Y as it also computes correlation among ranked variables. However, it does not utilize the difference among a single sample. Instead, it compares pairs of samples among each other. For $i \neq j$, (x_i, y_i) and (x_j, y_j) are defined as "concordant" if the ranks of both elements agree i.e. $x_i > x_j$ and $y_i > y_j$ or $x_i < x_j$ and $y_i < y_j$. Pairs are defined as "discordant" if the ranks of both elements disagree i.e. $x_i > x_j$ and $y_i < y_j$ or $x_i < x_j$ and $y_i > y_j$. In the case where ranks of either element are equal, the pair is ignored. Let $c =$ the number of concordant pairs and $d =$ the number of discordant pairs. Then Kendall's tau is computed as

$$\tau_{x,y} = \frac{c - d}{n(n-1)/2}$$

such that $-1 \leq \tau \leq 1$. Kendall's tau is less sensitive to errors in the data as its correlation is based on sample pairs rather than deviations within an observation, though the resulting values tend to result in the same interpretations. As with Spearman's correlation, $\tau = \pm 1$ with perfect increasing and decreasing monotonic dependence respectively. Furthermore, $\tau = 0$ does not necessarily indicate independence, though it is an indication of **monotonic** independence.

1.2.4 Distance correlation

While the aforementioned correlation metrics are well-known and commonly used, they are constrained to monotonic functions. Distance correlation was first proposed in 2007 as a way to further test for non-monotone dependence between X and Y [13]. The distance correlation is a function of the distance covariance and distance variance of X and Y. We define the $n \times n$ matrices a and b as the distance matrices of X and Y respectively. The elements $a_{k,l}$ and $b_{k,l}$ are respectively defined as $||X_k - X_l||$

and $||Y_k - Y_l||$ for all $k, l = 1, 2, ..., n$ where $||z||$ is the Euclidean norm $\sqrt{z_1^2 + ... + z_n^2}$. Then we define the distance covariance and distance variance as

- $\mathrm{dCov}(X, Y) = \sqrt{\frac{1}{n^2} \sum_{k=1}^{n} \sum_{l=1}^{n} A_{k,l} B_{k,l}}$
- $\mathrm{dVar}(X) = \mathrm{dCov}(X, Y)$
- $\mathrm{dVar}(Y) = \mathrm{dCov}(Y, Y)$

where $A_{k,l} = a_{k,l} - \bar{a}_{k,} - \bar{a}_{,l} + \bar{a}$ and $\bar{a}_{k,}$ is the kth row mean of a, $\bar{a}_{,l}$ is the lth column mean of a, and $\bar{a}$ is grand mean of a. Similarly, $B_{k,l} = b_{k,l} - \bar{b}_{k,} - \bar{b}_{,l} + \bar{b}$ and $\bar{b}_{k,}$ is the kth row mean of b, $\bar{b}_{,l}$ is the lth column mean of b, and $\bar{b}$ is grand mean of b. The distance correlation is then defined as

$$\mathcal{R}_{X,Y} = \frac{\mathrm{dCov}(X, Y)}{\sqrt{\mathrm{dVar}(X)\mathrm{dVar}(Y)}}$$

such that $0 \leq \mathcal{R} \leq 1$. As before, $\mathcal{R} = 1$ indicates perfect dependence. However, the interpretation of $\mathcal{R} = 0$ is different, which is one of its two important properties [13]:

1. X and Y may be of different dimensions.
2. $\mathcal{R} = 0$ if and only if X and Y are independent.

1.3 Portfolio management

An important application of correlation graphs is in modeling the dependencies among financial equities. Determining the relationship among various stocks is especially useful when managing portfolios. One such methodology is the "buy and hold" tactic where an investor selects a portfolio of stocks and never rebalances. The idea is to select diversified stocks with low correlation and positive drift such that losses are offset by gains, and the portfolio gains on average. This strategy is especially useful when transaction costs are high as fees are prohibitive to rebalancing gains.

In a world with low transaction costs, however, there is more to be gained (less to be lost, alternatively) by frequently rebalancing the portfolio rather than holding. The concept for stock picking is similar to that of "buy and hold", though there are some key difference. Another component for successful rebalancing gains is for the stock returns to be relatively independent [8]. Thus, when one stock goes down, the others do not fall with it; with perfect independence, rebalancing gains become a function of the volatility of the stocks rather than of stock returns. This is further important because frequent asset trading may move the market and lead to unexpected price swings. With independent stocks, the price risk associated with rebalancing is minimized. In summary, rebalancing gains are best suited to markets with low correlation, independent returns, and high volatility [8].

Another financial application is the creation of a predictive stock model. Predictive models are a form of "model selection" where dependence rather than independence is important. Model selection addresses the problem of determining which explanatory variables (commonly seen as columns of the matrix X) are informative to explain the variation in the response variable (commonly seen as the vector Y). This is also related to the concept of "sparsity," a statistical term referring to the fact that many coefficients of a fitted model should be 0. Model selection with sparsity aids in the interpretability of the model since there are fewer variables for data analysts to understand. This is another way to view rebalancing; the predictive model, if it is to be believed, can signal price swings to come which open up arbitrage opportunities. Furthermore, stocks which are uncorrelated are those with insignificant coefficients in the resulting model; this "negative space" point of view is partly utilized in the stock selection methodology for correlation graphs proposed in Section 5.2.

It must be noted that correlation alone cannot capture the full complexity of these financial applications. However, it is still an important component of portfolio management in theory and in practice. Regardless, correlation graphs and their financial

applications are not solutions to the two problems proposed in Section 1.1. Rather, correlation graphs and their financial applications are a concrete application of whatever the proposed solution is (A detailed application may be found in Chapter 5, and a stock selection methodology for correlation graphs is proposed in Section 5.2). What follows is a roadmap of the high-dimensional visualization solution that is developed further in this paper.

1.4 Summary

In this work, we tackle the problem by developing a sophisticated visualization system (abbreviated VS) to explore the data and numerical model in a different way. Specifically, we focus on two aspects of the VS which address the two problems raised in Section 1.1: (1) the procedure of sorting through plots is efficiently automated by learning the user's interests, and (2) the procedure of comparing the numerical and visual output is also automated. This allows the system to find visually interesting relationships that the numerical model may have missed and/or toss out relationships which turn out to be uninteresting. Furthermore, this allows future analysts to combine visual feedback with the numerical feedback from estimators to make better decisions during data analysis and provide clear justification of their decisions.

A broad overview of the VS and its framework may be found in Chapter 2. Chapter 3 then focuses on the active learning stage of the VS, which is part of the first solution to the problems raised in Section 1.1. The chapter details and simulates various active learning methods to be used in the financial application in Chapter 5. Chapter 4 focuses on the second solution to the problems raised in Section 1.1. The chapter is concerned with the VS output, which quantifies the differences between numerical and visual correlation graphs for the user. Subsequently, the chapter details various graph comparison methods and ultimately selects one for usage in the

financial application in Chapter 5. Finally, Chapter 6 recaps the work and presents future extensions of the VS.

Chapter 2

Visualization System

It is extremely important to consider the way in which the system presents plots to the user as that can change the way the user perceives the data. Furthermore, user interactivity is a critical component of high dimensional analysis as noted earlier [9]. (see paragrapah below)

Regardless of whether high dimensional data visualization methods are computationally heavy or interaction heavy, user interactivity is still a vital component for processing high dimensions for visualization; it is simply a question of what degree [9]. We develop a system that first learns what visual patterns the data analyst finds promising, querying the user where the decision tree is ambiguous. It then automatically iterates through thousands of possible plots. Finally, it suggests relationships to exclude or include, which it believes to match the data analyst's interests, and their corresponding plots. This allows users to compare and contrast visual feedback with numerical algorithms for improved model selection.

User provides numerical graphical model and data that they wish to analyze/confirm visually → program analyzes the plots among edges that do exist in the graphical model and extract relevant features (**??**) → initialize the decision tree given the resulting data (2.2.1) → perform a line-up test to determine how accurate the initialized

tree is → selectively pick plots corresponding to ambiguous parts of the tree (2.2.2) → periodically generate line-up tests from ambiguous parts of the tree to confirm the current iteration of the decision tree is "correct" (**??**) → if the user passes a number of line-up tests, display the final classifier and heat map (2.3.1) → allow user to tweak if desired → output (2.3.2)

2.1 Scatterplot characterization

2.1.1 Characteristics of a "good" plot

The simplest scatterplot is the response against the observed variables. This, however, may not be the best way to ascertain independence for the user. This notion is illustrated in Figure 2.1. The left plot appears to be independent as its a cluster of points near the origin, but it's not entirely clear due to the multitude of stray points around the concentrated section. By looking at the outliers, it could also be argued that there is some dependency. However, applying the CDF in both directions creates a plot distributed on (0,1). It should be noted that this transformation is non-destructive and preserves dependency in the data if it exists. The data is clearly independent as the points appear to be uniformly distributed within the plot.

Restricting the plot to a unit box allows analyst's visual systems to focus on locations where there is low spatial frequency, which is ideal for detecting dependence [5]. The effects of this can be progressively observed by looking from the left to the right in Figure 2.2 below.

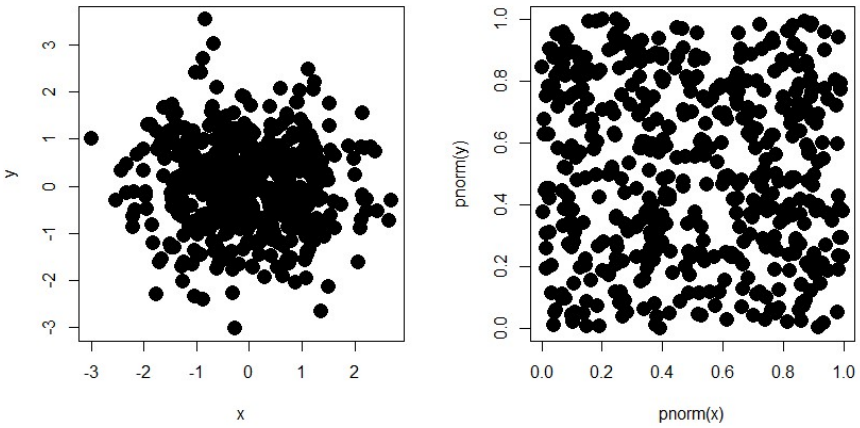
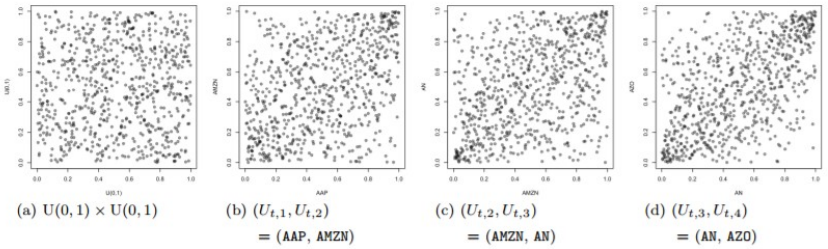


Figure 2.1: A plot of y against x with no transformation (left) and after the CDF is applied in both directions (right). The code for this example may be found in A.3



(a) $U(0,1) \times U(0,1)$ (b) $(U_{t,1}, U_{t,2})$ = (AAP, AMZN) (c) $(U_{t,2}, U_{t,3})$ = (AMZN, AN) (d) $(U_{t,3}, U_{t,4})$ = (AN, AZO)

Figure 2 Scatterplots of (a) independent U(0, 1) random variables and (b, c, d) the pseudo-observation pairs $(U_{t,j}, U_{t,j+1})$, $j \in \{1, 2, 3\}$. Ticker symbol abbreviations: `AAP` = Advanced Auto Parts, `AMZN` = Amazon.com Inc., `AN` = AutoNation Inc., and `AZO` = AutoZone Inc.

Figure 2.2: Scatterplots of (a) independent $U(0,1)$ random variables and (b,c,d) the pseudo-observation pairs $(U_{t,j}, U_{t,j+1}), j \in \{1,2,3\}$. Ticker abbreviations: AAP = Advanced Auto Parts, AMZN = Amazon.com Inc, AN = AutoNation Inc., AZO = AutoZone Inc. Images from Hofert and Oldford 2016 [5]

2.1.2 Feature extraction from plot

Numerical features

((Our goal is to quantify various features of a scatter plot. One category of features are numerical features. These include Pearson correlation, tests of independence, mutual information criterion, etc))

Visual features

((The other category of feature are visual. How many points are near the center of the plot? How many points lie above the linear regression line?))

2.2 Active learning (Stage 1)

2.2.1 Decision tree classification of user interests

A decision tree is a method of classifying and labeling plots. An "active learner" adapts as the process moves forward, choosing its points of query intelligently. Active learning increases efficiency when searching through the hypothesis space, which is any fitted tree that agrees with the labeled data as much as possible [2]. Every time new data (a new label) is received, the hypothesis space shrinks as the label removes certain classifiers from the running [2]. An active learner queries from ambiguous parts of the current hypothesis space so as to shrink it as quickly as possible. It is problematic to start from scratch, however; how does the system determine the best first point of ambiguity when it knows nothing (the hypothesis space is everything)? While this problem is difficult, we can exploit the fact that the user is already providing a numerical model that they believe to be a good representation of the data which they would like the visualization system to check visually. Given this data, the system builds a decision tree that utilizes the various properties of the plots to determine whether one is interesting or uninteresting. Doing so greatly narrows the hypothesis space and makes it easier to determine points of ambiguity. However, to reconcile with the fact that the user wishes to check the numerical model and may not necessarily believe it is a good representation of fit, the learner must then perform several line-up tests (Section **??**) to check whether the initial decision tree is a proper fit. As the user then proceeds to label various conditional plots as "interesting" or "not interesting,"

the classifier learns the users interest and continues to evolve. This models plot characteristics that the user found interesting to study.

2.2.2 Selective plot generation

((To build a better classifier, we want to have the user label plots that the system (at the time) is uncertain about. This is where active learning comes in since we want to build our system to cleverly give users vital plots to label so the system can best learn the users interests. The system will have to determine which features it is uncertain about classifying and return a plot matching those characteristics to the user.))

2.3 Automated plot generation (Stage 2)

Utilize the CLASSIFIER to fit the rest of the d choose 2 scatterplots

Classifier = an instance of a classsification model (i.e. LDA, QDA, random forest, logistic regression, etc.)

2.3.1 User interaction with active learning output

((Now that the system has learned the users interest, the user should be able to understand his/her own interests. We provide visualization tools of the resulting classifier itself such as a heat map, which there are also multiple ways to visualize))

2.3.2 Plot generation and feedback

((Equipped with the learned classifier of the users interest, the system can now automatically generate thousands of new plots and label them automatically. The most interesting plots are returned the user along with the explanatory variable (including possible interaction terms) corresponding to the plot))

3 options....list out the interesting vs not interesting plots, refine the decision tree via lineup tests, or return graph comparison feedback between the VISUAL and NUMERIC graph See FUTURE EXTENSIONS for the 2nd option

2.4 Specific focus: AL and GC

Specific focus of thesis: AL and GL.

Chapter 3

Active learning

Active learning is a subset of machine learning wherein an algorithm is "trained" on labeled data instances in order to learn from and make predictions on the data. Consider a large set of unlabeled data X with a hidden label from a finite set Y that can be queried from some human "oracle"; we would like to learn a good classifier of the data, some mapping $h : X \to Y$ from the set $\mathcal{H}$ without making too many queries [2]. Active learning is the process of intelligently selecting the queries (a constrained resource) to learn as much as possible. When a learning algorithm is allowed to choose its next query, it performs better with less training [12]. This property is especially desirable when labeled data are difficult, time-consuming, or computationally expensive to obtain. Stage one of the VS (Section 2.2) is one such classification task where $Y \in \{\text{interesting, non-interesting}\}$. Classification and filtering tasks are both tedious and redundant, which necessitates intelligent selection of queries [12]. What follows is an active learning literature review in Section 3.1, an overview of active learning methods and their algorithms in Section 3.2, and a simulation study in Section 3.3 to determine the method that is best-suited for usage with the VS in Chapter 5's equity application.

3.1 Literature review

It is important to consider the situation in which the learning algorithm receives the points in which it selects a query from. There are three different scenarios in which the learner may request queries as described by Settles [12]:

1. **Membership query synthesis**: The learner may select a query from an unlabeled samples in X.
2. **Stream-based selective sampling**: An unlabeled sample is randomly selected from X, and the learner decides whether to query or not.
3. **Pool-based sampling**: k unlabeled samples are randomly selected from X, and the learner picks one to query.

While the methods above may apply to many different active learning environments, the specific situation for the VS is pool-based selective sampling. Because it is computationally expensive to extract features from every single $\binom{d}{2}$ plot (Section 2.1.2), membership query synthesis is not an option. While stream-based selective sampling may work, it has a similar problem as membership query synthesis because there are no constrains on the number of samples that the algorithm may choose to discard.

The next problem, then, is to determine the informativeness of unlabeled instances that are presented by the methodologies above. This is the crux of active learning as it allows for intelligent selection of queries. Dasgupta identifies two different approaches to active learning that are fundamental drivers to the process of query selection [2]:

1. **Efficient search through hypothesis space $\mathcal{H}$**: The idea is to select a query that shrinks $\mathcal{H}_t$, the set of all possible classifiers at time t that explain the labeled data, as much as possible.
2. **Exploiting cluster structure in data**: The idea is to cluster the data and select queries based on cluster structure (i.e. query from each cluster). While clusters may be split over time if they are discovered to be non-homogenous, a learner may leverage situations where clusters are fairly homogeneous in order to classify points by propagating the labels to its neighbors.

Uncertainty sampling and query by committee are active learning algorithms that are a form of the aforementioned efficient search through the hypothesis space. These

algorithms may be found in Sections 3.2.1 and 3.2.2 respectively. We also present a clustering partitioning algorithm in Section 3.2.3 that seeks to exploit the cluster structure in the data.

3.2 Overview of active learning methods

3.2.1 Uncertainty sampling

In uncertainty sampling, the active learner selects the query q that it is most uncertain on how to label; in other words, the algorithm queries the label that has the highest posterior probability [6]. With binary classification labels such as the VS system (either "interesting" or "non-interesting"), this reduces to the case of querying the instance whose posterior probability of being "interesting" is closest to 0.5 [6]. While the algorithm presented later follows this methodology and may subsequently only be used with classification models that are able to encompass posterior probability computations, there has been much work done to expand uncertainty sampling to non-probabilistic classifiers such as decision trees and nearest-neighbor [12].

Uncertainty sampling is simple but not without problem. Given that the number of possible classification labels is $k > 2$, uncertainty sampling only considers the information about the most probable label i and ignores the other possible labels $j \in \{1, ..., k\} \backslash i$. "Margin sampling" and "entropy" are variants that try to solve for these problems, but both reduce to the scheme above (selecting q with a posterior probability closest to 0.5) when $k = 2$ [12].

We have developed an algorithm for uncertainty sampling based on the literature review (see Appendix A.4 for code):

Algorithm 1 Uncertainty sampling (as described by Settles [12])

procedure (x is a $n \times d$ matrix of d observations of all n variables, y is a $d-$length vector of labels for each variable in x (y_i=N/A when $x_{n,}$ has no label))
    $tout \leftarrow$ train($x^{labeled}, y^{labeled}$, classifier).
    $p \leftarrow$ predict($tout, x^{unlabeled}, y^{unlabeled}$, posterior prob.)
    loop from $i = 1$ **to** len(p):
        $p_i \leftarrow |p_i - 0.5|$
    return where($p == \min(p)$)

By searching for the most "uncertain" point, uncertainty sampling is able to further refine the classifier as the oracle must label the point either "interesting" or "non-interesting". This can be viewed as a search within the hypothesis space $\mathcal{H}$ that contains multiple classifiers, instances of a single classification model.

3.2.2 Query by committee

Query by committee is a clearer case of efficient search through the hypothesis space. In query by committee, a "committee" of classification models are trained on the current labeled instances. Each model represents a competing hypotheses, and its prediction for an unlabeled query candidate q is a "vote" of weight one on q; the most disagreeable candidate is then selected [12]. Settles reviews various methods for committee selection for both probabilistic and non-probabilistic models, though the implementation of the algorithm (Appendix A.5) allows the user to specify the committee members [12].

We may write the basic algorithm as follows:

Algorithm 2 Query by committee (as described by Settles [12])

1: **procedure** (x is a $n \times d$ matrix of d observations of all n variables, y is a $d-$length vector of labels for each variable in x (y_i=N/A when $x_{n,}$ has no label). Let C be the vector of all committee members)
2: **loop from** $i = 1$ **to** len(C):
3: $tout_i \leftarrow \text{train}(x^{labeled}, y^{labeled}, C_i)$.
4: $p_{i,} \leftarrow \text{predict}(tout_i, x^{unlabeled}, y^{unlabeled})$
5: $d \leftarrow \text{disagreement}(p)$
6: **return** where($d == \max(d)$)

While it is simpler to constantly maintain the same committee throughout, it would be more informative to prune the committee as the algorithm proceeds; it may simply be the case that a model is ill-suited for the problem at hand and consistently returns predictions that skew the voting procedure. Subsequently, a model is removed from the committee if it is consistently out-of-line with whatever the "true" label of q_t (the next query point decided and then queried at time t) turns out to be. It is important to note that the "true" label is not known until *after* the algorithm has return the index of q_t. Subsequently, the committee from t is pruned at time $t + 1$ at the start of the algorithm using the retrieved label from time t. Furthermore, the query by committee methodology described by Settles is only focused on selection of q independent of the final classification model once the rest of the training data is selected [12]. However, there is merit in maintaining the final pruned committee as its own classification model after the querying is complete and the training data is selected. This may be achieved by selecting labels on unlabeled instances via majority vote. Further details on implementation and performance may be seen in the simulation study (Section 3.3). The revised selection algorithm is as follows (see Appendix A.5 for code):

Algorithm 3 Query by committee (revised framework)

procedure (x is a $n \times d$ matrix of d observations of all n variables, y is a $d-$length vector of labels for each variable in x (y_i=N/A when x_n, has no label). Let C be the vector of all committee members, E be the error ratio of the respective committee members (initialized to 0), $0 < \epsilon < 1$ be some threshold for the error ratio, and t be the current iteration of QBC starting at $t = 1$)
  function QBC
    loop from $i = 1$ **to** len(C):
      $tout_i \leftarrow$ train($x^{labeled}, y^{labeled}, C_i$).
      $p_{i,} \leftarrow$ predict($tout_i, x^{unlabeled}, y^{unlabeled}$)
    $d \leftarrow$ disagreement(p)
    return $j =$ where($d ==$ max (d))
  function ORACLE
    return l, the label of x_j
  $y_j \leftarrow l$
  function PRUNE
    $prune =$ [ ] (empty vector)
    loop from $i = 1$ **to** len(C):
      If $p_{i,j} = y_j$ **then** $iv = 0$ **Else** $iv = 1$
      $E_i = E_i + \frac{iv - E_i}{t}$
      If $E_i > \epsilon$ **then** $prune$.append(i).
    $t++$
    return $prune$
  loop from $i = 1$ **to** len($prune$):
    Delete E_{prune_i}, C_{prune_i}, $p_{prune_i,}$, and $tout_{prune_i}$

The algorithm contains a generic disagreement function, so it is important to note that there are different measures of disagreement among the committee members. There are two main methods of disagreement measurement described by Settles [12]:

1. **Vote entropy**: The disagreement for variable d is computed

$$x^*_{VE} = \arg\max_x - \sum_{j \in \text{all possible labels}} \frac{V(y_j)}{\text{len}(C)} \log \frac{V(y_j)}{\text{len}(C)}$$

   where $V(y_j)$ is the number of votes each possible label y_i for d received. The interested reader may refer to [1] for further details on this methodology.

2. **Kullback-Leibler divergence**: The disagreement is computed

$$x^*_{KL} = \arg\max_x \frac{1}{\text{len}(C)} \sum_{i \in 1}^{\text{len}(C)} \sum_{j \in \text{all possible labels}} \text{P}_{C_i}(y_j|x) \log \frac{\text{P}_{C_i}(y_j|x)}{\text{P}_C(y_j|x)}$$

Since C was defined as the committee, we can interpret $\mathrm{P}_C(y_j|x) = \frac{1}{\mathrm{len}(C)} \sum_{k=1}^{\mathrm{len}(C)} \mathrm{P}_{C_k}(y_j|x)$ as the probability that the label with the most votes will be y_j where $\mathrm{P}_{C_k}(y_j|x)$ is the probability that committee member k votes y_j for variable d.The interested reader may refer to [10] for further details on this methodology.

These two disagreement measures have have been implemented in Appendix A.6 along with a variety of other disagreement measures .

3.2.3 Clustering partitioning

Thus, the algorithm is as follows (see Appendix A.7 for code):

3.3 Simulations

We utilize a simulation study in order to determine the method that is best-suited for usage with the VS in Chapter 5's equity application.

Chapter 4

Graph comparison

Introduction/how it relates to the VS (rehash end of chapter 2)

4.1 Literature review

Graphic summarization

4.2 Overview of methods

4.2.1 Histogram of edge density

4.2.2 Shortest-path matrix

4.2.3 Graph distance kernel

4.3 Examples

Table summarization of method outputs when applied to various examples we've thought of previously

Chapter 5

Application and results

5.1 Application of the visualization system

Let X be a $n \times d$ data matrix where there are n observations of d variables. The application of the visualization system is as follows:

1. Create four different numerical correlation graphs $G_i^{\text{num}}(V, E)$ for all $i \in \{1, ..., 4\}$, one for each correlation coefficient described in Section 1.2.

2. Run the VS on the same dataset. Stage 1 of the system utilizes the active learning algorithm selected in Chapter 3 to learn what is "correlated" and "not correlated." Stage 2 of the system then iterates through all possible unlabeled plot pairs and returns a graph $G(V, E)$ where

---

For $i, j \in \{1, ..., d\}$, $E_{i,j}$ exists if the plot of j against i is "correlated"

   As a reminder, correlation is used loosely to refer to a visual interpretation of the mathematical term.

3. Utilize the graph comparison algorithm selected in Chapter 4 for each pair (G, G_i^{num}) for all $i \in \{1, ..., 4\}$. Select N_i such that $GC(G, G_i^{\text{num}}) \leq GC(G, G_j^{\text{num}})$ for all $j \neq i$. The numerical correlation graph $G_i^{\text{num}}(V, E)$ is the graph which best matches the visual interpretation of the relationships among the dataset's variables.

5.2 Stock selection methodology

Given a correlation graph $G(V, E)$, we wish to select k stocks (represented as vertices) such that each stock is as independent as possible of the other stocks. Two random variables X and Y are independent if and only if their joint cumulative distribution function may be written as $F_{X,Y}(x, y) = F_X(x)F_Y(y)$. With this formal definition, Santos *et al.* notes that "we say two random variables X and Y are dependent if they are not independent" so the problem then becomes one of "how to measure and detect dependence from the observation of the two random variables"[11]. With the correlation graph $G(V, E)$, we now have a measure of dependence. However, it was made clear that, in general, it is incorrect to say that "uncorrelated" equates "independence". Regardless, non-correlation is still a useful tool to inform stock selection for portfolio management (Section 1.3). So, we select k stocks such that each stock is as *uncorrelated* as possible with the other stocks.

As V_i and V_j have no edge when $\text{Corr}(i, j)$ is below some threshold p, we might consider the following stock selection strategy:

Algorithm 4 Naive stock selection strategy

procedure (k is the number of stocks to select and A is the adjacency matrix of $G(V, E)$)
    $d \leftarrow \text{len}(A)$
    $z \leftarrow \text{Perm}(d, k)$*
    $min = \infty$
    $index = 0$
    loop from $i = 1$ **to** $\binom{d}{k}$:
        $c = 0$
        loop from $j = 1$ **to** k:
            loop from $l = 1$ **to** k:
                $c \leftarrow c + A_{z_{i,j}, z_{i,l}}$
        If $c < min$ **then**
            $min = c$
            $index = i$
    return $z_{index,}$

*Perm(d, k) is a function that returns all possible permutations of k values selected from d i.e. $z_{1,} =< 1, 5, 7 >$ is a permutation for $d = 10, k = 3$.

This becomes computationally difficult as d increases and does not account for average stock returns; as discussed in Section 1.3, stocks with positive drift and high volatility are shown to improve the performance of the portfolio over time in the "buy and hold" model, or improve gains in the case of rebalancing. These two criteria may be used to minimize the space of stocks to select from. Drift over time is simplified as positive return averaged over time. We let p_d and p_v denote the threshold values for drift and volatility respectively. These values will be dependent on the size of the dataset as we wish to preselect p_d and p_v such that $k < d < 2k$ to reduce the computational burden. The selection strategy then becomes

Algorithm 5 Pruned stock selection strategy

procedure (k is the number of stocks to select and A is the adjacency matrix of $G(V, E)$. Let D be the vector of sample average returns and V be the vector of sample standard deviation.)
    function DRIFT
        $d \leftarrow \text{len}(A)$
        $prune = [\,]$
        loop from $i = 1$ to d:
            If $D_i \leq p_d$ **then** $prune$.append(i)
        return prune
    loop from $i = 1$ to len($prune$):
        Delete row $prune_i$ and column $prune_i$ from A
    function VOLATILITY
        $d \leftarrow \text{len}(A)$
        $prune = [\,]$
        loop from $i = 1$ to d:
            If $V_i \leq p_v$ **then** $prune$.append(i)
        return prune
    loop from $i = 1$ to len($prune$):
        Delete row $prune_i$ and column $prune_i$ from A
    function SELECTION
        $d \leftarrow \text{len}(A)$
        $z \leftarrow \text{Perm}(d, k)$*
        $min = \infty$
        $index = 0$
        loop from $i = 1$ **to** $\binom{d}{k}$:
            $c = 0$
            loop from $j = 1$ **to** k:
                loop from $l = 1$ **to** k:
                    $c \leftarrow c + A_{z_{i,j}, z_{i,l}}$
            If $c < min$ **then**
                $min = c$
                $index = i$
        return $z_{index,}$

*Perm(d, k) is a function that returns all possible permutations of k values selected from d i.e. $z_{1,} =< 1, 5, 7 >$ is a permutation for $d = 10, k = 3$.

5.3 Healthcare stock data

We select 43 stocks from the healthcare industry to apply the system to. There are 4113 observations of the stocks from 1998 to 2014. We have cleaned the data to get

rid of the dependency on time so that each observation of a stock may better simulate an independent draw from some distribution. This is done by fitting a regression to the values and using the residuals as the new dataset.

For $i \in \{1, ..., 4\}$, we employ a "buy and hold" tactic to compare the performance of each portfolio P_i selected utilizing the methodology described in Section 5.2), which correspond to correlation graphs $G_i^{\text{num}}(V, E)$. While a rebalancing method performs better in practice [8], the "buy and hold" is better-suited for observing the portfolio performances over time, providing a more concrete basis of comparison for the selected correlation coefficient.

5.4 Results

Here, we have a comparative time plots of the "buy and hold" performance (yearly returns) of the S&P 500, stocks selected from the final numerical correlation graph $G_i^{\text{num}}(V, E)$, and other correlation graphs $G_j^{\text{num}}(V, E)$ where $j \neq i$. We hope to find that correlation graph i, which was selected from the usage of the VS, is a top performer.

We may also analyze the VS graph output $G(V, E)$ against the final selection $G_i^{\text{num}}(V, E)$ by asking the following questions: What new links were formed? Which links were deleted? Did we actually find the plots of those links interesting/not interesting? If it is possible (if the graphs are not too dense), then I would also include a visualization of $G(V, E)$ and $G_i^{\text{num}}(V, E)$.

Chapter 6

Conclusion

Although this paper has focused on its financial applications, visualization systems nevertheless enable better decisions during any sort of data analysis. In the univariate case (Section **??**, **??**), it is clear that plotting has augmented the data analysis process by allowing the user to double-check their fitted models. In higher dimensions with more complex data sets and an inability to plot everything, it is even more believable that numerical methods cannot completely replace the valuable information obtained from visual methods. The visualization system developed in this text provides a way to safeguard data analysts against their own biases.

This visualization tool is one important step in streamlining the future of clean analysis. It provides a systematic way for confirming and suggesting dependencies among variables that match the analyst's concepts of a dependent (or "interesting" plot) and produces an explicit decision tree that allow others to understand and replicate the data analysis process. This removes the tediousness associated with high-dimensional data and allows the user to quickly see ways in which the numerical model may have fallen short of the "true" relationship between variables. In other words, it systematically provides the user a way to validate their methodology and model selection. Furthermore, it improves accountability in the analysis as it allows

the decision-making process (in the form of a decision tree) to be clearer to those reviewing the results. Nevertheless, the graphical model that we develop is not fool proof, and further work can be done to refine the model and improve our concept of "clean analysis."

6.1 Further extensions

6.1.1 Estimator selection

((Estimator selection involves actively fitting the best model as opposed to "checking" a numerical model thats been given. This problem is more difficult to define, and the value that the visualization system adds is not as concrete))

6.1.2 Outlier removal

((Outliers are unavoidable in raw data and can skew results quite a bit. When can the system remove outliers? What criteria should it use?))

6.1.3 Edge-weighted graphs

((This is more of an extension on graphical models, but it is still related to estimator selection. In an edge-weighted graph, edges can be weighted depending on the type of conditional dependence negative or positive, assign 1 or -1. 0 (no edge) still implies conditional independence))

6.1.4 Regression and graphical models

A single correlation can be thought of as a regression of the response variable against only one observed variable; it is a "local" property because it compares the behavior of only two random variables. On the other hand, a single link in a graphical model

can be thought of as a regression of the response variable against all variables in the space. It is more nuanced than that, but the idea is that a graphical model has a global property because it takes all of the other variables into account. Although it may seem simple to numerically quantify the dependencies with correlation as conditional dependencies are less intuitive to compute, correlations tend to fall short of the desired result due to the property of transitivity.

As mentioned earlier, we would like to find stocks that are as uncorrelated as possible in order to achieve the most robust portfolio. A common methodology to do this is to forego plotting entirely and numerically compute the correlation matrix then create a graph from that. By construction, the correlation matrix applies the same correlation computation to each set of observed and response variables. Correlation coefficients are rather limited and have their own flaws, so no coefficient is a one size fits all solution. For example, some variables could share a linear relationship with the response (ideal for a Pearson correlation) while others may not. While on the subject of the Pearson correlation, it should be noted that correlation graphs are still interesting because two variables that have a correlation coefficient near 0 may not necessarily be uncorrelated. A visualization tool can reveal this by showing outliers or patterns in the data that the analyst wasn't expecting (Section **??**). Thus, the analyst needs to perform a visual check of the resulting correlation matrix rather than blinding accepting the results. This is important for improving accountability, as well, but it is a more difficult task than one might believe. In order to confirm that the coefficient used applies to each potential relationship, the analyst must plot all possible sets of data, which we have already established as computationally infeasible and tedious to sort through in high dimensions.

Furthermore, the result itself can be uninformative. Consider the property of transitivity, which states that if X is correlated to Y, and Y to Z, then X is also correlated to Z. Although correlation is not always transitive, situations where the

correlation is close to 1 or 0, then the transitivity of correlation can be recovered and observed in the relevant data [14] Furthermore, correlation is a "local" property because it compares the behavior of only two random variables and ignores the rest of the data space, which may lead to an increasing amount of "false positives". Let us assume that the universe consists of Apple, Google, and Silicone (a manufacturer providing chips to both Apple and Google) stock. Suppose an analyst wants to model Google stock. Apple stock moves with Silicone stock as they depend on them for their chips. Similarly, Google stock (unbeknownst to the analyst) also moves with Silicone stock but not with Apple stock. The correlation between observed prices of Google and Apple stock will clearly and erroneously be positive without considering the way the stocks are connected to the other observed variable (Silicone stock). Given that correlation tends to be transitive, a correlation graph can have too many edges. This goes against the concept of "sparsity" and clutters the resulting space of observation variables that explain the response variable for the user. In the end, the analyst is left with an uninformative and unaccountable numerical solution.

Graphical models alleviate some of the problems associated with correlation graphs, but they have their own set of problems, as well. Let G=(V,E) be an undirected graph with vertices $V_1, ..., V_d$ ~P (a d-dimensional distribution) and edges $E_{i,j} \in \{0, 1\}$. We set $E_{i,j} = 1$ when there is an edge between V_i and V_j, and 0 otherwise. Do not draw an edge between V_i and V_j iff $V_i \perp V_j$ given V_k where $k \in \{1, ..., d\} \setminus \{i, j\}$. In other words, do not draw an edge if

$$P(V_i, V_j|V_k) = P(V_i|V_k)P(V_j|V_k)$$

This is known as a graphical model. The drawback is the difficulty in empirically computing conditional distributions and the problems associated with fitting distributions to real data. While there are simplifications that can be made for plotting

(Section **??**), the solution is not always so clear. However, the conditional independence of graphical models is more of a global property than correlation is because, for every pair of variables, it conditions on all the remaining variables. Returning to the universe of Google, Apple, and Silicone stocks, conditioning Google on Silicone and Apple on Silicone makes the relationship between the two clearly uncorrelated. Thus, although conditional independence tends to be more difficult to determine, it will tend to give a sparser network that is more interpretable for the analyst. The search through the rest of the data space is akin to the way regressions are fitted.

There are many ways to numerically perform model selection, and regression is the most common. In low-dimensional settings (where there are more samples than explanatory variables), the most common way is to fit a least-squares linear regression and perform a hypothesis test on each coefficient. The F-test is another useful tool for numerically informing the user if a regression model with more variables has significantly more explanatory power over a nested regression model. There are also ways to perform regression and model selection simultaneously. The most common estimator is the Lasso, the Least Absolute Shrinkage and Selection Operator (Tibshirani 1996). The drawback to all these methods is that their theoretical properties tend to either be asymptotic or reliant on assumptions. The former is unsatisfactory since datasets in practice are typically of a fixed size, far from the number of samples to achieve desirable properties analogous to when the number of samples goes to infinity. The latter is unsatisfactory because these assumptions are typically hard to verify or provide convincing justifications of.

Analysts may choose to use correlation over graphical models or vice versa as each has its own niche to fill. It has been shown that the rebalancing method, which leverages independent stocks and is more akin to graphical models, outperforms the "buy and hold" method, which leverages negatively correlated variables and naturally lends itself to the correlation approach [8]. Due to the nature of our financial

application, we choose to utilize graphical models in our analysis of the data, but it is important to understand that both methodologies have the same need for a program that makes high dimensional visualization simple. This allows the analyst to make an informed decision on whether to confirm or reject the numerical result and improves decision-making in the financial industry.

((There are many ways to produce this scatter plot. One way is to marginally plot one explanatory variable against the response variable. We experiment with another way where, similar to regression, we control for the behavior of all the other explanatory variables while making this scatter plot.))

6.1.5 Ordering

As people interact with graphs, they maintain a "mental map" of the graph. Federico and Miksch note that the importance of the mental map depends on various factors such as the user preferences and tasks that they must complete [4]. By "mental map," Federico and Miksch mean that when users label a new graph, they remember the previous plots that they labeled [4]. Supposing that the graphs that a program wants to show the user are already decided. Given a gradation of graphs (showing graphs that are most alike one after the other), users are less able to distinguish between differences than if they are shown graphs from different ends of the spectrum at different times [4]. While their work primarily concerns itself with ordering graphs for interpretability, their results can be applied to scatter plots. To put it concisely, the scatter plot display itself is not the only thing that matter. The ordering of the display matters, as well, and it is best to show the plots in an order that allows users to distinguish the differences among graphs that they have already seen. By improving their understanding of the plots, careful display ordering advances the accuracy of user responses. User responses can be thought of as observations of the users true

preferences, and the ordering of plots as a way to future tune the precision of the decision tree (Section 2.2).

6.1.6 Line-up tests

One of the pitfalls of data visualization is "apophenia," a phenomenon where the user sees patterns in random noise. Part of the reason for this is due to the vagueness of defining "independence" on a non-uniform domain and range (Section 2.1.1). Wickham et al. propose a line-up protocol that is similar to the Rorscach test where subjects are asked to interpret abstract blots of ink [15]. In the line-up test, users are asked to identify the real data from a set of n plots where n 1 plots are synthetically generated (Figure6.1). Analogously in the context of the classification problem, the learner generates 4 "uninteresting" plots (following the proposed decision tree) with 1 "interesting" plot and asks the user if he/she can identify the interesting plot. If the user is able to consistently identify the interesting plot, it is an indication that the current decision tree is a close fit of the users preferences.

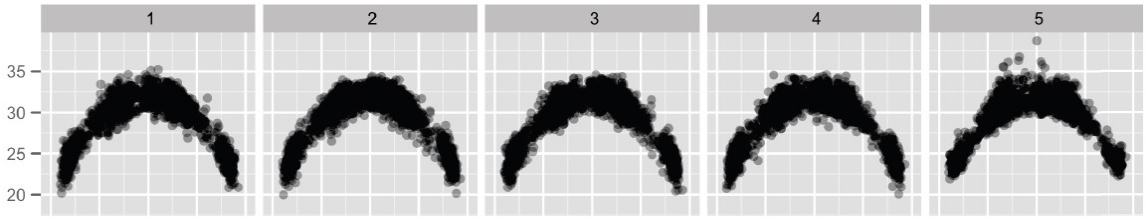


Figure 6.1: A line-up test for $n = 5$. Consistently identifying the raw data against the synthetic data indicates that the fitted model may not be good enough. Images from Wickham et al. 2010 [15]

6.1.7 Rejection classification

So far, we have been discussing classification as black and white, interesting versus non-interesting. While interacting with the data visually, the user's concept of what's

interesting in the specific dataset may evolve over time; at the beginning, they have no idea what the data looks like and where to set the bar for their own standards of dependence. There are several ways to take this into consideration.

First, the visualization system could assign a weight to the analyst's responses by trial number where the last few plots are more valuable than the first few. However, this may destabilize cases where the user's preferences dont end up changing, and it is difficult for the classifier to continuously rebalance each round given the weights (which, when applied to 2 black and white non-numerical responses, may also be vague). Secondly, the system can include an alternative option that allows the user to refuse to label a plot when it's too close to their decision boundary. This is welcome for the user who is not forced into making a decision he/she is uncertain about, but it is problematic for the learner as it causes the hypothesis space to remain unchanged rather than shrink. The point of the active learning segment is to have the user indicate to the learner which plots are interesting or not in order to let the computer better understand their preferences. By allowing for this option, the active learner may run for too long or return a poorly-defined tree. Finally, there is a way to consolidate the considerations within each methodology. The system can contain a third option that permits the user to "recycle" the plot. This allows to user to return to the plot later when he/she has learned more about what the data looks like and understand their own preferences better, and it ensures that the active learner will eventually receive data on the ambiguous plots that it has given the user to label. The main concern is that this could potentially de-balance an ordering procedure (Section **??**), but the system can strategically insert the recycled plot between two plots it differs from with the constraint that the insertion location is after the current plot.

Appendix A

Implementation Details

A.1 Code for figure 1.1, left

```
#generate a reproducible dataset and scale to [0,1]
set.seed(10)
x <- seq(0, 1, length.out = 100)
y <- rnorm(100)
y <- (y-min(y))/(max(y)-min(y))

#sort the noise
y <- sort(y)
y <- y[c(seq(1,99,length.out=50), seq(100,2,length.out
   =50))]

#local swapping
for(i in 4:96){
y[(i-3):(i+3)] <- y[sample((i-3):(i+3))]
}

idx <- sample(1:100)
x <- x[idx]; y <- y[idx]

######################
#numerical feedback

## fit linear regression
fitlm <- lm(y ~ x)
anova(fitlm)
```

```
## see if any coefficients are significant
summary(fitlm)

## see if residuals are normally-distributed
shapiro.test(fitlm$residuals)

## correlation is not significantly different from zero
cor.test(x, y)

####################
#visual feedback
plot(x, y, pch = 16, cex = 2)
```

A.2 Code for figure 1.1, right

```
## generate a reproducible dataset
set.seed(10)
n <- 50
x <- sort(rnorm(n))
sd.vec <- c(seq(1, 1.5, length.out = 50), seq(1.5, 1,
   length.out = 50))
y <- -x + 0.5*rnorm(n, sd = sd.vec)
y <- scale(y)

y[c(1,5,10)] <- min(y)
y[c(n-10, n-5, n)] <- max(y)

####################
#numerical feedback

## fit linear regression
fitlm <- lm(y ~ x)
anova(fitlm)

## see if any coefficients are significant
summary(fitlm)

## see if residuals are normally-distributed
shapiro.test(fitlm$residuals)

## correlation is not significantly different from zero
cor.test(x, y)
```

```
####################
#visual feedback
plot(x,y, pch = 16, cex = 2)
```

A.3 Code for figure 2.1

```
## generate the dataset
set.seed(10)
n <- 500
x <- rnorm(n)
y <- rnorm(n)

par(mfrow=c(1,2))
plot(x,y, pch = 16, cex = 2)
## apply the cdf
plot(pnorm(x),pnorm(y),pch = 16, cex = 2)
```

A.4 Uncertainty sampling implementation

A.5 Query by committee implementation

A.6 Query by committee disagreement implementation

A.7 Clustering partitioning implementation

Bibliography

[1] I. Dagan and S. Engelson. Committee-based sampling for training probabilistic classifiers. *Proceedings of the International Conference on Machine Learning*, pages 150–157, 1995.

[2] Dasgupta, S. Two faces of active learning. `http://cseweb.ucsd.edu/~dasgupta/papers/twoface.pdf`, 2011.

[3] B. Efron. Why Isn't' Everyone a Bayesian? *The American Statistician*, pages 1–5, February 1986.

[4] P. Federico and W. Oldford. Evaluation of two interaction techniques for visualization of dynamic graphs. *arXiv preprint arXiv:1608.08936*, pages 1–15, August 2016.

[5] M. Hofert and W. Oldford. Visualizing Dependence in High-Dimensional Data: An Application to S&P 500 Constituent Data. *arXiv preprint arXiv:1609.09429*, pages 1–33, September 2016.

[6] D. Lewis and W. Gale. A sequential algorithm for training text classifiers. *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12, 1994.

[7] R. J. Little. In Praise of Simplicity not Mathematistry! Ten Simple Powerful Ideas for the Statistical Scientist. *Journal of the American Statistical Association*, pages 359–369, July 2013.

[8] H. Liu, J. Mulvey, and T. Zhao. A semiparametric graphical modelling approach for large-scale equity selection. *Quantitative Finance*, pages 1053–1067, 2016.

[9] S. Liu, D. Maljovec, B. Wang, P. T. Bremer, and V. Pascucci. Visualizing high-dimensional data: Advances in the past decade. *IEEE Transactions on Visualization and Computer Graphics*, pages 1249–1268, December 2016.

[10] A. McCallum and K. Nigam. Employing EM in pool-based active learning for text classification. *Proceedings of the International Conference on Machine Learning*, pages 359–367, 1998.

[11] S. Santos, D. Y. Takahashi, A. Nakata, and A. Fujita. A comparative study of statistical methods used to identify dependencies between gene expression signals. *Briefings in Bioinformatics*, pages 1–13, August 2013.

[12] Settles, B. Active Learning Literature Survey. `http://burrsettles.com/pub/settles.activelearning.pdf`, 2010.

[13] G. Szekely, M. Rizzo, and N. Bakirov. Measuring and testing independence by correlation of distances. *The Annals of Statistics*, pages 2769–2794, March 2007.

[14] Tao, T. When is correlation transitive? `https://terrytao.wordpress.com/2014/06/05/when-is-correlation-transitive/`, 2014.

[15] H. Wickham, D. Cook, H. Hofmann, and A. Buja. Graphical inference for infovis. *IEEE Transactions on Visualization and Computer Graphics*, pages 973–979, December 2010.