Forecasting stock excess returns with SEC 8-K filings

Henry Han*[1], Yi Wu[2], Diane Li[3], Jie Ren[1]

Henry Han, [1]Department of Computer Science, Baylor University, Waco TX 76798, USA, Corresponding author Email: Henry_Han@baylor.edu

Yi Wu, [2]Center for Music Technology, Georgia Institute of Technology, Atlanta GA 30332, USA, Email: ywu663@gatech.edu

Diane Li, [3]School of Business & Technology, University of Maryland, Eastern shore, MD 21853, USA Email: dli@umes.edu

Jie Ren, [1]Department of Computer Science, Baylor University, Waco TX 76798, USA Email: Jie_Ren1@baylor.edu

Abstract

The stock excess return forecast with SEC 8-K filings via machine learning presents a challenge in business and AI. In this study, we model it as an imbalanced learning problem by proposing a multi-class SVM forecast with tuned Gaussian kernels to handle it. The proposed model performs better than peers from state-of-the-art deep and machine learning. We also show that the TF-IDF vectorization would demonstrate advantages over the BERT vectorization in the forecast. Unlike general assumptions, we find that dimension reduction generally lowers forecasting effectiveness compared to using the original high-dimensional vectorized data. Furthermore, inappropriate dimension reduction may increase the overfitting risk in the forecast or cause the machine learning model to lose its learning capabilities. We also find that resampling techniques cannot enhance forecasting effectiveness for high-dimensional imbalanced data. In addition, we propose a novel dimension reduction stacking method to retrieve both global and local data characteristics for high-dimensional vectorized data that outperforms other peer methods in forecasting and decreases learning complexities. The algorithms and techniques proposed in this work can help stakeholders optimize their investment decisions by exploiting the 8-K filings besides shedding light on AI innovations in accounting and finance.

Keywords: Form 8-K, excess return, dimension reduction stacking, NLP.

1 Introduction

With the surge of AI and social data science, more and more Securities Exchange Commission (SEC) filing data and relevant information are employed in AI-based decision-making in accounting and finance [1,2]. The SEC filing data mostly refers to those periodic (SEC) forms (e.g., 8-K, 10-K, 13-D), financial statements, and other required disclosures submitted by public companies. They disclose different important corporate actions and activities, the change of ownerships, or other significant information to investors. Since almost all SEC filings are textual data containing important sentimental information about companies, different natural language processing (NLP) and machine learning (ML) techniques are employed to predict the security tendency of the firms, identify possible artifacts in corporate management, or even provide information for strategic decision making [1,2,3]. For example, since an important acquisition reported in the SEC 8-K report may signal future corporate stock prices or other corporate actions, ML or deep learning method are used to query future stock price movements or predict the coming firm events or activities [3-5].

Form 8-K, which is known as the "current report", provides timely notification to shareholders about the occurrence of significant events. It is generally not filed at a specific time interval. When certain types of important events (e.g., acquisition) that should be aware by shareholders happen, an 8-K report

should be filed in four business days. Unlike other information that can be modified or even interpreted by a third party, the 8-K information is valuable and reliable because it is direct communication between the companies and investors. Also compared to its similar peers such as Forms 10-K and 10-Q, Form 8-K is more concise, readable, and informative, especially because the 10-K/Q filings cannot timely notify investors about the important corporate events due to their long release time intervals [5-6].

Many recent efforts were invested in accounting and finance fields by employing NLP and AI-based textual analytics to study the SEC filings to decipher market trends, discover future corporate events, or predict security returns. Ke *et al.* introduced novel text-mining methods to extract sentiment information from news articles to predict stock returns. Loughran revealed meaningful liability signals by analyzing 10-K reports [4]. Zhai applied deep learning methods to predict firm future event sequences using the 8-K reports [5]. Kogan *et al.* predicted stock return volatility from 10-K forms [7]. Lee *et al.* used the 8-K filings to predict stock returns [8]. Zhao categorized different 8-K reports into different categories to seek signals for stock returns [8]. Furthermore, Engelberg found that textual information has better long-term predictability for asset returns [10]. Li employed a Naïve Bayes learning approach to analyze corporate filings [11]. Aydogdu *et al.* used long short-term memory (LSTM) deep learning models to analyze SEC 13D filings [12].

The studies bring new insights into finance and accounting fields besides extending AI application domains. However, they have the following weakness and limitations. First, the results from the proposed methods are not good enough for real business practice because of the high nonlinearity of data and possible artifacts of the methods. Few improvements were shown that stock movement prediction based on the textual data was better than using classic quantitative features [8,13]. This is especially true for those stock return forecasting or relevant studies using the 8-K filings [1,3].

Second, it remains unknown how to handle the high dimensional vectorized data of the SEC filings well for the sake of the following ML prediction. The vectorized data obtained from the text vectorization process, which is a procedure to convert the original textual data (e.g., 8-K reports) into its numerical representation, can demonstrate very high dimensionality. For example, an 8-K report textual dataset with 10000 sentences can be converted to a corresponding numerical matrix with 10000 observations across 20000+ features after vectorization.

Interestingly, such vectorized data is a new type of high-dimensional data in which the numbers of variables and observations can both stay in the same high order level (e.g., $O(10^2)$). Traditional high-dimensional data is characterized by a large number of variables (e.g., $O(10^4)$) but a small number of observations (e.g., $O(10^2)$). In addition, the high-dimensional vectorized data may contain different noise from the original text and vectorization procedures. The traditional viewpoint believes dimension reduction is necessary or even a must before any serious downstream analysis such as machine learning [14]. However, it remains unknown how to conduct meaningful dimension reduction for high-dimensional vectorized data to gain effective feature extraction to balance the tradeoff between preserving useful information, besides removing redundancy and de-noising.

The widely used latent semantic indexing (LSI), an SVD-based dimensional reduction algorithm, or its variants (e.g., PCA) may not be able to represent the original data well in the low-dimensional space [14,15]. This is because they are holistic methods only good at capturing global data characteristics rather than local data characteristics [14]. The global and local data characteristics reflect those holistic (global) and local data behaviors respectively. To achieve high-performance learning, it is needed to

retrieve both global and local data characteristics from the vectorized data. However, it remains unclear how to extract both global and local data characteristics in dimension reduction in NLP analysis not to mention for the SEC filing data.

Third, it remains unknown which text vectorization models match input textual data (e.g., 8-K data) well though different models are available. They include Bidirectional Encoder Representations from Transformers (BERT), term frequency-inverse document frequency (TF-IDF), and Bag-of-words (BoW), etc. [16-17]. Since most studies only select one vectorization model to process textual data, the following ML results may not be optimal in business problem solving. Therefore, more comparisons and analyses are needed to examine the match between the ML methods and the vectorization models for the sake of decision making.

In this study, we address the challenges by forecasting the excess returns of SP500 companies using their 8-K filings. The excess return evaluates how a stock or portfolio perform in comparison to the overall market average return. It is generally represented as an Alpha metric computed by comparing with a benchmark index (e.g., SP500 index). The alpha value provides traders and portfolio managers a robust index to select securities or diversify portfolios in investment. Since the 8-K reports include the significant events of a company that would impact its stock performance, it will be interesting to predict the excess returns by exploiting the semantics of the 8-K filings. Therefore, predicting the excess return using the 8-K data not only explores more hidden relationships between the important corporate events and company stock performance, but also provides valuable insights for trading, portfolio management, and investment decision-making [18-19].

The stock excess return forecasting is modeled as an imbalanced multiclass classification problem in this study. We employ the data used in Ke *et al'*s study which includes the 8-K filings from the SP500 companies during the period 2015-2019 [3]. The daily excess return of each stock, i.e., daily alpha values, is used as a target variable in forecasting. The stock excess return prediction is rendered as an imbalanced multi-classification problem by discretizing the target variables as three classes, because the number of observations in the majority class counts >50% among all observations. Unlike other classification problems in textual analytics with roughly balanced label distributions, the imbalanced learning problem itself not only challenges the excess return forecasting but also ML itself for its imbalanced nature.

We propose a multiclass support vector machines (multi-SVM) model for the excess return forecasting for its good efficiency and reproducibility [20,21]. We compare it with different peers under various dimension reduction and vectorization methods. Unlike the general assumptions, we find that dimension reduction may not contribute to enhancing the excess return prediction well. They generally bring slightly lower or at most equivalent forecasting performance though they lower the complexity of the SVM learning. Moreover, inappropriate dimension reduction methods (e.g., PCA+tSNE) may generate an 'imbalanced point' in the forecast. The imbalanced point is a special overfitting state in which the ML model misclassifies all minority samples as the majority type. Furthermore, we find more complicated deep learning models (e.g., transformer) are more likely to generate the imbalanced point than those with simple learning topologies (e.g., *k*-NN), besides resampling techniques may not be able to enhance the forecast performance.

In addition, we propose a novel dimension reduction stacking technique in this study to capture both global and local data characteristics of the vectorized data. The data under the new dimension reduction technique achieves an equivalent performance in comparison with using the original data besides

decreasing the learning complexity of the stock excess return forecast. To the best of our knowledge, it is the first time that such a technique has been proposed and will bring positive impacts on meaningful NLP dimension reduction. Moreover, in contrast to the general viewpoint that BERT is more suitable for large-scale text data vectorization than TF-IDF for its more semantics-oriented vectorization, we find that TF-IDF leads over BERT in the stock excess return forecast [17].

This paper is structured as follows. Section 2 introduces the 8-K data preprocessing along with text vectorization. It also models the excess return forecast as an imbalanced learning problem and introduces imbalanced point generation and an explainable metric: d-index for the stock excess return forecast evaluation. Section 3 introduces the dimension reduction stacking technique for the global and local data characteristics retrieval for the vectorized data. Section 4 demonstrates the advantages of the proposed SVM model by comparing it with different peers under different dimension reduction methods and vectorizations. It also investigates how resampling techniques impact the stock excess return forecast. Section 5 discusses the improvements of the proposed techniques and concludes this study.

2 Data preprocessing, text vectorization, and imbalanced learning forecast

The goal of this study is to predict daily excess returns (alpha values) for SP 500 stocks with the 8-K data. We employ the 8-K filings from all the SP500 companies during the period 2015-2019 used in Ke *et al's* study [3]. The original textual data includes a total of 119762 sentences from collected 8-K documents [22]. The stock daily excess return, i.e., daily alpha, is used as a target variable in learning. We have valid 17648 daily stock excess returns by removing all missing data in preprocessing.

We model the stock excess return prediction with the 8-K data as an imbalanced multi-classification problem by discretizing the daily alpha values into three classes. The three classes demonstrate imbalanced distributions: the majority class counts >50% of the total observations. This is because we partition the daily alpha values into three classes: class 0: alpha <-0.01, class 1: alpha in (-0.01, 0.01), and class 2: alpha> 0.01. We simply call them 'buy', 'hold', and 'sell' though they do not represent real trading recommendations. The 'hold' class is the majority class with 9109 samples counting 51.61% of all observations. The 'buy' and 'sell' classes have 4138 and 4401 samples counting 23.45% and 24.94% of all observations, respectively. The preprocessed 8-K data is further partitioned as training data with 13655 samples and test data with 3993 observations. The three class distribution ratios among the test data are 25.44%, 48.01%, and 26.55%.

The stock excess return forecast using the 8-K data is challenging ML for its imbalanced nature. Imbalanced learning would bring biases to the classification accuracy metric so that it cannot reflect the true forecasting correctly [23]. This is because the forecast can be 'hijacked' by the majority class in learning, i.e., the ML model misclassifies most or even all minority observations as the majority type. In other words, the whole forecast is 'overfitted' to the majority type. As a result, the ML model loses its learning capability, and the accuracy will approach the majority ratio of the test data. If the ratio is high enough, the ML model will achieve deceptively decent performance even if it loses its learning capability. We say the ML model will generate an imbalanced point in this situation.

To avoid the accuracy bias from imbalanced learning, we employ an interpretable ML assessment metric diagnostic index (d-index) to evaluate the stock excess return forecast [23]. The d-index falls in

(0,2] and a large d-index indicates better ML performance. The imbalanced point and d-index are described as follows. More detailed information can be found in the authors' recent work [23].

Imbalanced point. Given training data $X_r = \{x_i, y_i\}_{i=1}^m$, in which $y_i \in \{1,2,..k\}$ is the label of the observation x_i, under an ML model Θ, we assume the class k is the majority type class and the majority ratio is defined as $\gamma = \frac{|\{x_i: y_i = k\}|}{\sum_{i=1}^k |\{x_j: y_j = i\}|} \gg \frac{|\{x_j: y_j = i\}|}{\sum_{i=1}^k |\{x_j: y_j = i\}|}, i \neq k$. Then the ML model is said to reach an imbalanced point, provided $\hat{f}(x|\Theta, X_r) = k$ for $\forall x$ whose label is unknown, in which $\hat{f}(x|\Theta, X_r)$ is the prediction function built under the model Θ with the training data X_r. The accuracy of the ML model will be the majority ratio γ at the imbalanced point if the training and test data share the same majority ratio. Otherwise, the accuracy will be the majority ratio of the test data.

d-index. Given an implicit prediction function $\hat{f}(x|\Theta): x \to \{-1, +1\}$ constructed from training data $X_r = \{x_i, y_i\}_{i=1}^m$ under an ML model Θ, where each sample $x_i \in R^p$ and its label $y_i \in \{-1,1\}$, $i = 1,2,\cdots m$, d-index evaluates the effectiveness of $\hat{f}(x|\Theta)$ to predict the labels of test data $X_s = \{x_j', y_j'\}_j^l$, where x_j' is a test sample and its label $y_j' \in \{-1,1\}$. The d-index is defined as:

$$d = \log_2(1 + a) + \log_2\left(1 + \frac{s+p}{2}\right) \quad (1)$$

where a, s, and p represent the corresponding accuracy, sensitivity, and specificity in diagnosing test data respectively. The d-index under multiclass classification can be extended from the binary version definition. More details about the d-index can be found in [23].

The d-index provides a good metric to compare the performance of different ML models for the sake of more robust model selection, especially under imbalanced learning. It is more explainable and comprehensive than other measures used for imbalanced learning evaluation such as MCC (Matthews correlation coefficient) and weighted F1 score [23]. More importantly, it can model imbalanced learning behavior sensitively by reporting anomaly learning states. The following theorem states the special d-index value at the imbalanced point. The detailed proof is omitted for the sake of brevity.

Theorem 1. Imbalanced point theorem. The imbalanced point under an ML model Θ will have an accuracy of γ and a d-index of $\log_2\left(\frac{3(1+\gamma)}{2}\right)$, where γ is the majority ratio of the test data.

In practice, we also count an approximately imbalanced point (AIP), in which the d-index of the ML model approximates $\log_2\left(\frac{3(1+\gamma)}{2}\right)$, as an imbalanced point. This is mainly because it may misclassify only a small portion of majority observations or not classify a few minority observations into the majority type. The AIP also indicates that the ML model loses its learning capabilities.

2.1 Text vectorization

Text vectorization is an important feature extraction procedure to translate input text documents to their numerical matrix representations. There are several text vectorization methods such as BERT, TF-IDF, BoW, etc. to convert input 8-K text data into its corresponding numerical representations. Although it is generally unknown which text vectorization models will vectorize the 8-K data better for the following forecast, we have the following reasons to choose BERT and TF-IDF rather than BoW [14].

BERT is a bidirectional encoder that captures more context semantics in text vectorization. BERT employs the bidirectional training of Transformer neural networks in vectorization and the other peers only conduct one direction training or scan. Therefore, it captures more context information and

produces more informative feature extractions. BERT is good at vectorizing large-scale text data (e.g., 8-K) because it is powered by a deep learning transformer model. The fine-tunning process allows BERT to learn more semantics from text data.

Both BERT and TF-IDF can avoid the negative impacts of the high-frequency words from the 8-K reports. The 8-K documents generally have similar words with very high frequencies to appear due to the filing syntax formats, most of which can be less significant or even meaningless. TF-IDF calculates the inverse document frequency to emphasize the significance of each word to avoid this issue. The semantics-oriented BERT avoids this by the bidirectional scan in vectorization. On the other hand, BoW relies on counting the frequency of each word in the document to build feature vectors to represent the original text data. Thus, the high-frequency words would bring noise in the BoW vectorization and impact the following forecasting negatively. Furthermore, since grammar information is completely lost in BoW, it can be hard to capture those features representing context semantics.

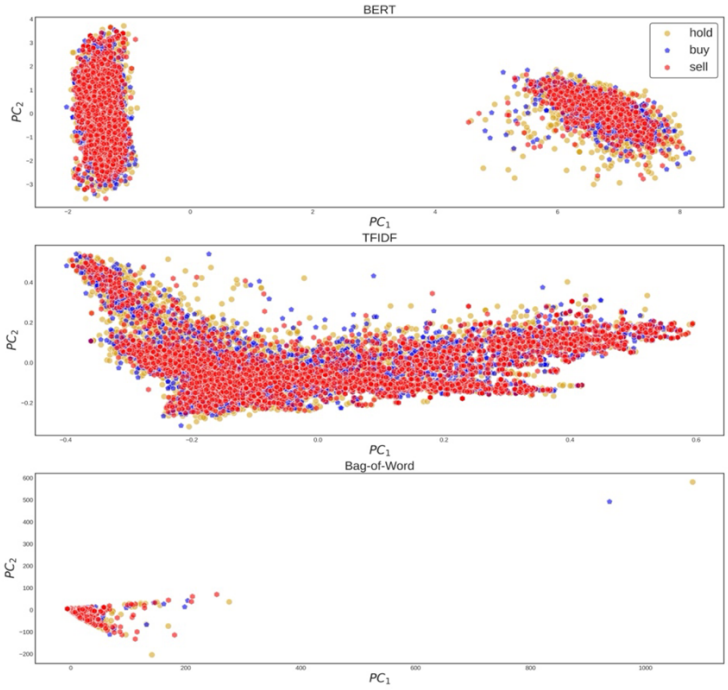

Fig. 1. The comparisons of the PCA visualizations of the BERT, TF-IDF, and BoW vectorization data, in which the daily stock excess returns are partitioned into three groups: 'buy', 'hold', and 'sell'. The BERT and TF-IDF vectorizations show a reasonable spread of three types of excess returns, but the BoW vectorization shows all data points are condensed in a small region.

Figure 1 compares the visualizations of the BERT, TF-IDF, and BoW vectorized data with PCA. They can be viewed as the vectorization signatures of the three methods. It shows that all data points under BoW are jammed in a small region. This suggests that most of the data points of the vectorized data have close or similar variances. As such, it is almost impossible to distinguish different types of stock excess returns from the BoW data. This indicates that BoW may not be a suitable vectorization model

for the 8-K data. On the other hand, both TF-IDF and BERT show a good spread of the data points and three types of returns are detectable though the majority class ('hold') still distinguishes itself. It is reasonable to expect that they will impact following forecasting more positively than BoW.

BERT produces a relatively lower number of features than TF-IDF and BoW for its semantics-oriented vectorization. The BERT vectorized data has only 768 features, but the TF-IDF vectorized data is high-dimensional data which has 27,147 features and 17,648 observations. To some degree, the TF-IDF vectorized data can be much more challenging in ML compared to the BERT vectorized data at least in computing complexities.

3 Dimension reduction stacking

Since the vectorized data of the 8-K textual data is characterized by their high dimensionality and nonlinearity, dimension reduction is assumed as an essential to remove redundancy, decrease noise, and reduce forecasting complexities. The widely used dimension reduction methods: LSI, PCA, LDA, and manifold learning methods (e.g., t-SNE) are generally employed to accomplish it [24]. However, it remains unknown how to extract both global and local data characteristics from the dimension reduction procedures because they are either holistic or local dimension reduction methods. High-performance forecasting needs both global and local data characteristics to achieve good prediction.

We address the challenge by proposing a novel dimension reduction stacking to capture both global and local data characteristics for the sake of stock excess return forecasting. The dimension reduction stacking is to map input data $\mathrm{X} = \{x_i\}_{i=1}^{n}, x_i \in R^p$ to its low-dimensional embedding $\mathrm{Y} = \{e_i\}_{i=1}^{n}, e_i \in R^l, l \ll p$ by stacking two different dimension reduction algorithms: $f_g(f_l(X)) \to Y$. The algorithm f_l is a local dimension reduction to capture local data characteristics and f_g is a holistic dimension reduction to gain global data characteristics.

We select f_l and f_g as uniform manifold approximation and projection (UMAP) and principal component analysis (PCA) respectively in the dimension reduction stacking though other options are available theoretically [25]. UMAP is a manifold learning technique producing a low-dimensional embedding by minimizing the fuzzy set cross-entropy between two distributions modeling the similarities of data points in input and embedding spaces [25]. Unlike PCA, it captures local data characteristics well especially when the neighborhood size is set in a relatively small range (e.g.,10). Different from t-distributed stochastic neighbor embedding (t-SNE), it is less random and more explainable by using a 'transparent' neighborhood size rather than a vague perplexity parameter to define a neighborhood and estimate the Gaussian distribution variance for input data [14].

The proposed dimension reduction stacking: UMAP+PCA calculates the UMAP embedding for input data to capture local data characteristics before applying PCA to the embedding to retrieve global data characteristics. It maps the daily excess observations in $\mathbb{R}^{n\times p}$ to the UMAP embedding space $\mathbb{R}^{n\times k}$ before obtaining the PCA embedding in $\mathbb{R}^{n\times l}$: $p \gg k \geq l$, i.e., $f_{pca}\left(f_{umap}(X \in \mathbb{R}^{n\times p})\right) \to Y \in \mathbb{R}^{n\times l}$. The complexity of the UMAP+PCA stacking is $O(nk^2 + k^3 + nlogn)$. In the implementation, we choose the dimensionality of the UMAP space $k = 10$ and project the UMAP embedding to the subspace spanned by l principal components (PCs) such that they have the explained variance ratio $\eta \geq 90\%$. The input vectorized data is normalized through the minmax normalization for its good performance. Algorithm 1 describes the proposed algorithm.

Algorithm 1: Dimension reduction stacking algorithm (X, nb_{size}, k, η)

Input:

The input data $X \in \mathbb{R}^{n \times p}$

The neighborhood size in UMAP: nb_{size}

The dimensionality of the UMAP embedding: k

The explained variance ratio: η

Output:

The dimension reduction stacking embedding: $X_{embedding}$

1. *// UMAP embedding*
2. $X_{umap} \leftarrow umap(X, nb_{size}, k)$
3. *//PCA for the UMAP embedding*
4. $newData, pcVariance \leftarrow pca(X_{umap})$
5. *//Retrieve the final embedding*
6. **if** $\frac{\sum_{i=1}^{l} pcVariance_i}{\sum_{i=1}^{k} pcVariance_i} \geq \eta$
7. $X_{embedding} \leftarrow newData[:, 1:l]$
8. **Return** $X_{embedding}$

Different orders of f_{pca} and f_{umap} in stacking can bring totally different dimension reduction results. For example, the proposed UMAP+PCA stacking captures local data characteristics before seeking global ones from the UMAP embedding. The order guarantees that UMAP+PCA can extract local and global data characteristics well in dimension reduction. If PCA goes before UMAP, it will be hard to retrieve local data characteristics because of the dominance of global data characteristics in PCA. The embedding of the PCA+UAMP stacking may lead to overfitting because of the overrepresented global data characteristics in the embedding. Theoretically, f_{pca} or f_{umap} can be substituted by any dimension reduction methods (e.g., $f_{umap} \leftarrow f_{tsne}$), but they may not guarantee to achieve the same stacking results and some inappropriate stacking (e.g., PCA+tSNE) can fail the following forecasting by generating the imbalanced point.

4 Results

The stock excess return forecasting with the 8-K reports is challenging because of the nature of imbalanced learning and the high nonlinearity of data. The mixture of the three types of observations in Figure 1 reveals it via a visualization approach. Therefore, it can be difficult to achieve high-performance forecasting on such nonlinear data under imbalanced learning and an efficient and robust ML model is needed to tackle the challenge.

We propose a multi-class SVM model to forecast the stock excess returns for its learning efficiency, reproducibility, and interpretability. The one-versus-one ('*ovo*') scheme is employed to extend relevant

binary SVM forecasting to corresponding multiclass forecasting to mitigate the impacts of imbalanced data. Unlike deep learning models, SVM owns good interpretability for its simple learning structure and transparent parameter setting. Compared to other ML models such as random forests (RF), it is more reproducible to repeat decent performance because of its rigorous quadratic programming solving [20]. We briefly describe the binary SVM model as follows.

A binary SVM model constructs an optimal hyperplane $y = w^T x + b$ to separate two groups of data points of the training data $X = \{x_i, y_i\}_{i=1}^n, x_i \in R^p, y_i \in \{-1, +1\}$ to by solving a quadratic programming problem:

$$min_w \ \frac{1}{2} w^T w + \mathrm{C} \sum_{i=1}^n \xi_i \ , w \in R^d \ , \xi_i \in R, b \in R \tag{2}$$

$$s.t. \ y_i(w^T \varphi(x_i) + b) \geq 1 - \xi_i \ , \xi_i \geq 0, i = 1,2 \cdots n,$$

where $w \in R^p$ is the weight vector, $\mathrm{C} \in R^+$ is the regularization parameter, and $\varphi(\cdot)$ is an implicit feature function mapping input data to the high-dimensional feature space for evaluation using kernel tricks. Once the optimization problem is solved, for given test entry x, the binary decision function is constructed as $f_{binary}(x) = sign(\sum_{i \in SV} y_i \alpha_i K(x_i x) + b)$. The set SV is the index set of all the support vectors corresponding to Lagrange parameters $\alpha_i > 0$.

For a given test entry x, multi-class SVM (*'ovo'*) seeks totally $k(k-1)/2$ binary prediction functions $f_\tau(x), \tau = 1,2 \ldots \mathrm{k}(\mathrm{k}-1)/2$ from the corresponding binary SVM. It then applies $f_\tau(x)$ to the test entry and seeks the maximum votes from the binary classifications. The label winning the maximum vote will be the final label of the test entry. The final prediction function $F(x)$will be formulated as

$$F(x) = argmax_l \sum_{\tau=1}^{\mathrm{k}(\mathrm{k}-1)/2} [f_\tau(x) = l], \ l = 0,1,2 \ldots, k\text{-}1 \tag{3}$$

The Gaussian kernel $K(x_i x_j) = e^{-\Upsilon \|x_i - x_j\|}$, $\Upsilon > 0$ is employed in SVM to model nonlinear relationships. The kernel parameter Υ is tuned under the grid search technique for the sake of effective forecasting.

Peer methods. We employ ML models from shallow learning, midlevel learning, and deep learning as the peer methods of SVM forecasting. Shallow learning refers to those ML models with neither a rigorous mathematical model nor a formal learning topology. For example, *k*-NN and decision trees (DT) are typical shallow learning methods. Deep learning refers to those with both complex learning topologies and complicated mathematical models. The long short-term memory (LSTM), convolution neural networks (CNN), and transformer models all are typical deep learning methods [26-29]. Midlevel learning refers to those with complicated mathematical models but less complex learning topologies. SVM, random forests (RF), and other ensemble learning models are typical midlevel learning methods [30].

We find that SVM generally outperforms its peers in forecasting under the TF-IDF vectorization. SVM achieves the best performance compared to *k*-NN, RF, and three deep learning models: CNN, LSTM, and Transformer for its highest d-index (1.206). LSTM has the second-best performance with a d-index: 1.187 with an accuracy: 0.48, sensitivity: 0.380, and specificity: 0.697. The *k*-NN and RF report equivalent performance with d-index:1.178, but Transformer misclassifies all minority observations as

the majority type with a d-index of 1.151 by generating an imbalanced point. It echoes the result of Theorem 1 because of $d = \log_2\left(\frac{3(1+\gamma)}{2}\right)_{\gamma=0.4801} = 1.1506$.

Table 1. The SVM excess return forecast under TF-IDF

Measures	SVM	SVM_{pca}	$SVM_{pca+tsne}$	$SVM_{pca+umap}$	$SVM_{umap+pca}$
D-index	1.206	1.162	1.150	1.161	1.186
Accuracy	0.496	0.484	0.479	0.483	0.482
Sensitivity	0.384	0.342	0.333	0.343	0.376
Specificity	0.699	0.673	0.667	0.672	0.694
Precision	0.425	0.406	Nan	0.440	0.398
NPR	0.740	0.756	Nan	0.741	0.724

In contrast to the traditional belief that dimension-reduction is 'a must' for vectorized data in NLP, we find that the original data will show advantages over the reduced data from dimension reduction in forecasting under the TF-IDF vectorization. Table 1 compares the SVM forecasting on the original data and different reduced data in terms of the d-index and classic measures such as accuracy, sensitivity, specificity, precision, and NPR (negative prediction ratio). It shows that SVM can forecast the stock excess returns better using the original data than the reduced data. For example, SVM forecasting achieves a 1.206 d-index on the original data but only a d-index of 1.162 on the reduced data from PCA (SVM_{pca}). This suggests that dimension reduction may cause some information missing so that the signal-to-noise (SNR) ratio of the reduced data decreases. Moreover, SVM forecasting with PCA+UMAP stacking ($SVM_{pca+umap}$) has almost the same performance as SVM_{pca}. It implies the holistic dimension reduction algorithm PCA may dominate the feature extraction in the PCA+UMAP stacking.

However, the proposed UMAP+PCA outperforms PCA, PCA+UMAP, and PCA+tSNE in SVM forecasting because of the 1.186 d-index value of $SVM_{umap+pca}$. The SVM_{pca}, $SVM_{pca+umap}$ and $SVM_{pca+tsne}$ forecasting all generate the imbalanced points or AIPs because their d-index values are close to 1.1508, the d-index value at the imbalanced point. It seems that the inappropriate stacking: PCA+tSNE forces SVM to lose its learning capability by generating the imbalanced point in forecasting for its d-index of 1.150. It is possible that the global data characteristics are over-retrieved in the PCA+tSNE stacking which may cause the Gaussian kernel in SVM to lose its discriminability. On the other hand, the better performance of $SVM_{umap+pca}$ may lie in the built-in advantage of UMAP+PCA in retrieving both global and local data characteristics to avoid the dominance of global data characteristics in dimension reduction. The UMAP+PCA stacking can bring almost equivalent forecasting besides greatly decreasing the complexity of foresting, by producing well-defined reduced data for high-dimensional vectorized data.

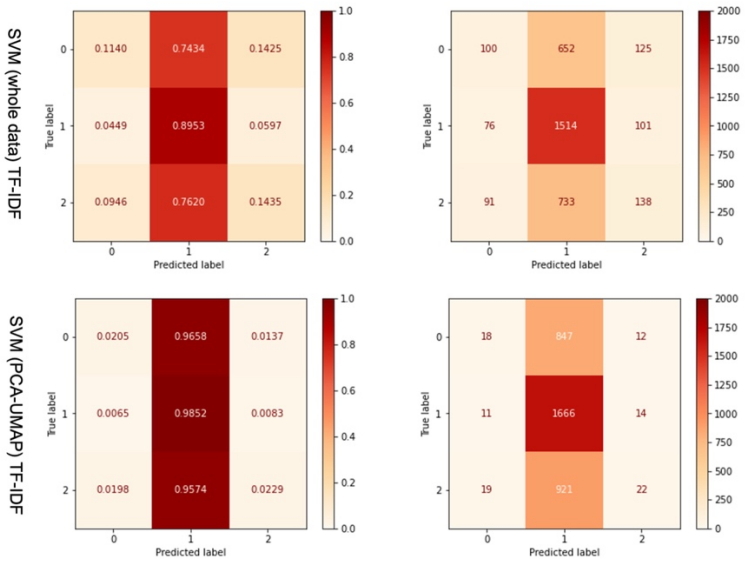

Fig. 2. The normalized and general confusion matrices of the SVM and $SVM_{pca+umap}$ forecasts under the TF-IDF vectorization. The $SVM_{pca+umap}$ forecast is affected more by the majority type ('hold') because 96.58% of 'buy' and 95.74% of 'sell' observations are misclassified as the majority type respectively. However, the corresponding ratios of the SVM forecasting on the original data are 74.34% and 76.20%

Figure 2 compares the normalized and general confusion matrices of the SVM and $SVM_{pca+umap}$ forecasts under the TF-IDF vectorization. It shows that SVM without dimension reduction demonstrates a better excess return forecast than $SVM_{pca+umap}$. The $SVM_{pca+umap}$ forecast is hijacked more by the majority type ('hold') because 96.58% of 'buy' and 95.74% of 'sell' observations are misclassified as the majority type respectively. However, the corresponding ratios of the SVM forecast without dimension reduction are 74.34% and 76.20%.

4.1 Dimension reduction stacking under the BERT vectorization

Unlike the general assumption that BERT is superior to TF-IDF in text vectorization, we find the SVM forecast shows advantages under the TF-IDF vectorization in comparison to the BERT vectorization for the 8-K data [14]. For example, the best d-index achieved by SVM without dimension reduction is 1.189 with an accuracy of 0.480, a sensitivity of 0.384, and a specificity of 0.699 under the BERT vectorization. It is slightly lower than the forecast performance with a d-index of 1.206 under TF-IDF. Similarly, The SVM forecast under UMAP+PCA stacking has a lower-level performance under the BERT vectorization than TF-IDF.

Furthermore, we compare the peer methods under UMAP+PCA stacking under the BERT vectorization with SVM without dimension reduction. The peer methods include *k*-NN from shallow learning, RF and SVM from midlevel learning, and CNN, LSTM, and Transformer from deep learning. We describe the deep learning models briefly for the sake of understanding as follows.

CNN is characterized by a partially connected layer structure and different layers have different functionalities such as convolutional, and max/average pooling. Topologically, it consists of a sequence

of locally connected convolutional layers, flatten, and fully connected layers. Each convolutional layer exacts input data features via convolution, activation function mapping, and pooling operations. The convolution layers extract input data features and conduct some sort of denoising. The flatten layer acts as an intermediate layer between the convolution layers and fully connected layers. It generally flattens 2D input data to a corresponding 1D vector. CNN and its variants demonstrate good learning capabilities for many types of data, especially for image data besides decent feature extraction [28].

LSTM overcomes the weakness of general recurrent neural networks (RNN) in handling long-time information dependence by employing LSTM cells consisting of three functional gates [26,26]. It evolves from traditional recurrent neural networks (RNN) to fix the gradient vanishing or exploding problems of RNN when capturing a long-distance information [28]. LSTM sees good applications in time series processing such as cryptocurrency price discovery and natural language processing (NLP) fields.

Unlike RNN, each LSTM unit evolves to a memory cell plus input, forget, and output gates, each of which is actually a neuron. The LSTM unit output is a weighted sum of the outputs from all three gates $h_t = (o_t * \tanh(i_t * c'_t + f_t c_{t-1})$, where o_t, i_t, f_t are outputs from the output, input and forget gates at time t, $c'_t(.)$ and $c_t(.)$ represent the candidate and final states of the memory cell at time t, and h_t is the output of the LSTM at time t. LSTM still employs gradient descent learning (e.g. ADAM) in training as RNN, but gradient vanishing will not happen for LSTM because the small loss value that causes the problem remains in the memory cell and the three gates have a built-in mechanism to adjust it.

Transformer can be viewed as a special feedforward neural network taking advantage of the self-attention mechanism in topology and learning, allowing the model to be trained in parallel and with more global information [29]. It is widely used in sequential data analysis, NLP, and various data science fields. A general transformer has an encoder-decoder architecture to map input data $X = (x_1, \ldots, x_n)$ into a sequence of consecutive representations $Z = (z_1, \ldots, z_n)$., The encoder makes input data each position passed through a self-attention layer by creating three meta-vectors, query, key, and value vector. The decoder calculates output as $z \leftarrow softmax\left(\frac{XW_q \times K^T}{\sqrt{d_k}}\right)V$, in which K and V are corresponding matrices by grouping the key and value meta-vectors, W_q is the weight matrix for input data, and d_k is the dimension of the key vectors.

We implement a 7-layer CNN model with two convolution layers, each of which has 64 neurons, two max-pooling layers, a flatten layer, and two dense layers with the '*relu*' activation function. The loss function is chosen as the sparse categorical cross-entropy for integer labels: $L(w) = \frac{1}{n}\sum_{i=1}^{n}[y_i log\ \hat{y}_i + (1 - y_i)\ log\ (1 - \hat{y}_i)]$, where w represents the weights of the network, y_i and $\hat{y}_i$, represent the true label and predicted label respectively.

We also implement a 6-layer LSTM model with 3 LSTM layers and 2 dense layers. The LSTM layers have [128, 64, 64] neurons with 'Tanh' activation functions and the dense layers have [128, 3] neurons with the *'relu'* and '*softmax*' activation functions. The loss function is selected as the sparse categorical cross-entropy. The drop rates increase gradually from 0.1 to 0.3 with the depth of the layers. In addition, we implement a transformer model with 4 dense layers and an embedding layer with the sparse categorical cross-entropy loss function. The drop rates are chosen as 0.2 generally.

Figure 3 compares the SVM forecast without dimension reduction with the peer methods with the UAMP+PCA stacking under the BERT vectorization in terms of classic classification metrics (left plot) and d-index values (right plot). The shallow learning model *k*-NN and the midlevel learning model RF

under UMAP+PCA achieve the best and second-best performance, but the deep learning models only achieve mediocre performance. In particular, the Transformer model generates an imbalanced point by misclassifying all minority observations in the test data as the majority type. This suggests that the attention mechanism of the transformer model may not work well under imbalanced data because it may amplify the impact of the majority type in learning [29]. Therefore, the sophisticated deep learning models may suffer from their built-in weakness in encountering overfitting for their complicated topologies compared to the shallow and midlevel learning models.

However, the best peer (e.g., RF) under the UMAP+PCA stacking may still fall behind the SVM model without dimension reduction. It echoes the previous result that dimension reduction may not contribute to enhancing forecasting even though vectorized data with high dimensionality.

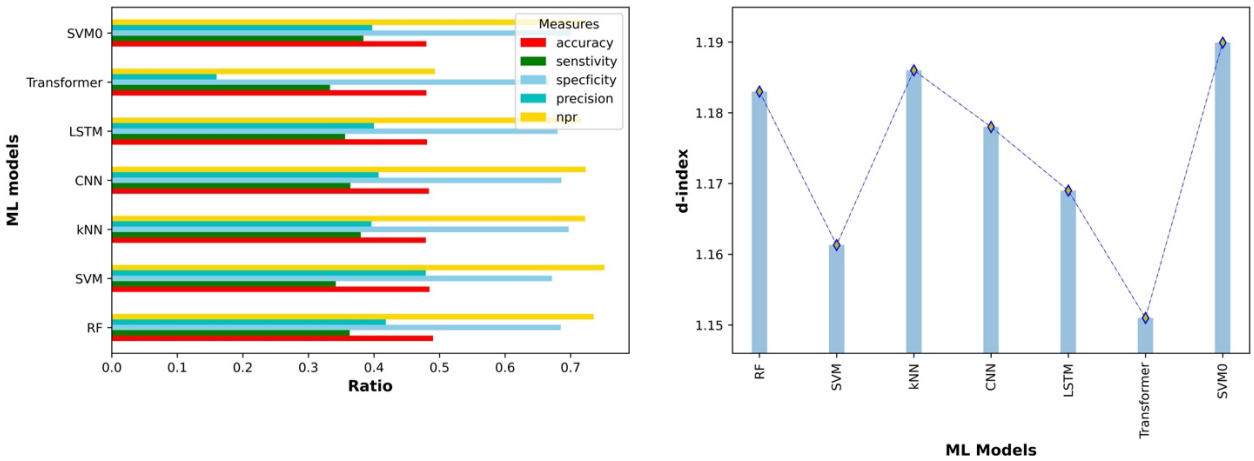

Fig. 3. The comparisons of peer methods under the UMAP+PCA stacking with the SVM forecast without dimension reduction (SVM0) under the BERT vectorization. The SVM forecast outperforms the peers with the stacking. The deep learning models only have mediocre or even much poor performance compared to *k*-NN and RF under the stacking for its complicated topologies.

4.2 Stock excess return forecast under imbalance resampling

Since the stock excess return prediction is essentially an imbalanced learning problem, we employ resampling techniques: random oversampling (ROS) and Tomek links (TL) to handle data imbalance [31-32]. ROS is an oversampling method to increase the number of observations of the minority type(s) and TL is an under-sampling method to decrease the number of majority-type observations to achieve data balance. We choose TF-IDF vectorized data for its advantage in forecasting.

We find the resampling techniques generally cannot enhance the SVM forecast. Instead, ROS mitigates the risk of overfitting by increasing misclassification ratios. TL tends to increase the risk of overfitting and generate the imbalanced point. Table 2 compares the performance of the SVM and $SVM_{pca+tsne}$ under TL and ROS. SVM has obvious advantages over $SVM_{pca+tsne}$, which generates the imbalanced point, under the two resampling methods in terms of the d-index values. The SVM forecast under TL stands at the same level as the original SVM forecast for their close d-index values: 1.210 and 1.206, but the SVM forecast under ROS only achieves a d-index of 1.175. It suggests that the ROS

resampling would bring more noise into forecasting through the oversampling procedure and decrease the effectiveness of prediction.

Table 2. The SVM excess return forecast with resampling under TF-IDF

Measures	SVM (TL)	SVM (ROS)	$SVM_{pca+tsne}$ (TL)	$SVM_{pca+tsne}$ (ROS)
D-index	1.210	1.175	1.160	1.149
Accuracy	0.492	0.454	0.478	0.438
Sensitivity	0.395	0.400	0.347	0.384
Specificity	0.705	0.708	0.676	0.700
Precision	0.426	0.402	Nan	0.382
NPR	0.735	0.712	0.721	0.703

Figure 4 compares the confusion matrices of the SVM and $SVM_{pca+tsne}$ forecasts under TL and ROS. It shows that the SVM forecast under TL tends to increase the percentages of the observations predicted as the majority type than ROS. The NW plot of Figure 4 illustrates 67.05% class 0, 84.86% class 2, and 68.30% class 3 observations in the test data under TL are predicted as the majority type (class 1). Similarly, the $SVM_{pca+tsne}$ forecast generates the imbalanced point under TL because 91.33% class 0, 95.39% class 1, and 90.85% class 2 observations are misclassified as the majority type illustrated by the SW plot of Figure 4.

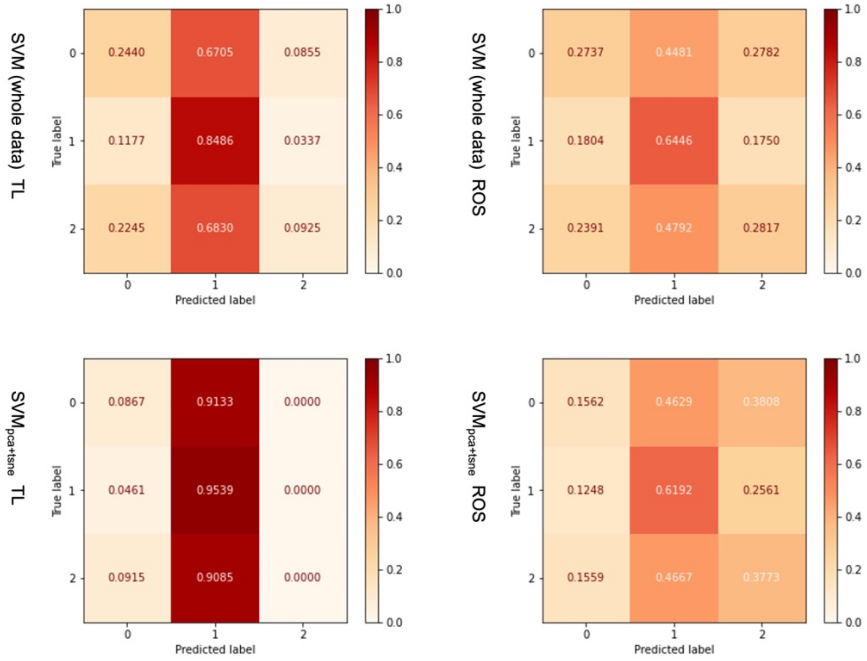

Fig. 4. The normalized confusion matrix comparisons of the SVM and $SVM_{pca+tsne}$ forecasts under the random oversampling (ROS) and Tomek links (TL) resampling techniques. TL tends to increase the percentages of the observations in all classes predicted as the majority type. ROS mitigates the overfitting risk of generating the imbalanced point by increasing more misclassifications.

On the other hand, ROS mitigates the risk of generating the imbalanced point by increasing the misclassification ratios. The NE plot of Figure 4 illustrates that only 44.81% of class 0, 64.46% of class 1, and 47.92% of class 2 observations in the SVM forecast are classified as the majority type under ROS. The SE plot of Figure 4 reveals that the overfitting of the $SVM_{pca+tsne}$ forecast is decreased under ROS.

5 Discussion and conclusion

We model the stock excess forecast using the SEC 8-K filings as an imbalanced learning problem and propose an multi-class SVM model with tuned Gaussian kernels to achieve applicable forecasting in business practice. Besides good reproducibility, the proposed model demonstrates favorable performance compared to the peers. We also find that dimension reduction may not contribute to improving forecast effectiveness as expected compared to using the original data. Instead, they may lower the forecast performance slightly though they decrease the learning complexities in the forecast. In addition, Inappropriate dimension reduction (e.g., PCA+tSNE) can increase the risk of overfitting in the forecast or even cause the ML model to lose learning capabilities. It also shows that resampling techniques cannot enhance the forecasting effectiveness. The TL resampling tends to increase the risk of overfitting in forecasting, but the ROS resampling can mitigate the risk of overfitting by increasing misclassifications.

We also propose a novel dimension reduction stacking method: UMAP+PCA to retrieve both global and local data characteristics for vectorized data. It outperforms other dimension reduction methods in SVM forecasting and can even substitute the SVM forecast to avoid the huge forecasting complexity from high-dimensional vectorized data. Besides, we find that the TF-IDF vectorization would demonstrate some advantage over the BERT vectorization in the stock excess return forecast.

Some questions remain to be answered on this topic. For example, how to enhance forecasting by minimizing the impact from the majority type? We employ sparse coding techniques to conduct post-processing for input vectorized data under TF-IDF, but the excess return forecasting results are not improved as expected. It may suggest that sparse coding may not overcome the negative impact from data imbalance. On the other hand, since the classic resampling methods fail on this imbalanced learning problem, it is possible to employ GAN or more complicated models to generate minority samples to balance data to seek an alternative solution [33-34]. We are investigating the integration of classic Fourier analysis with the novel dimension stacking technique to extract more meaningful features to attain high-performance forecasting [35-37].

In addition, it can be promising to integrate the stacking technique with interpretable deep learning models to address the high nonlinearity of the 8-K data [38]. An important reason why deep learning models can't work well for such excess return forecasting is because they all lack interpretable topologies to handle highly nonlinear imbalanced vectorized data. It can be desirable to design more transparent and explainable layers or even more customized activation functions in the deep learning

models to conduct data argumentation, de-noising, feature extraction, and data-oriented forecasting. Another important aspect can be the normalization methods for high-dimensional vectorized data, we employ the minmax normalization to normalize input data that demonstrates good advantages in the following forecasting than using raw vectorized data or normalized data after the standardized scaling normalization that generally assumes data is normally distributed. This implies that normalization may play an important role in this excess return prediction procedure. We are interested in exploring or designing more customized normalization methods to prepare data better for downstream ML.

To the best of our knowledge, the algorithms and techniques developed in this study will not only help stakeholders to optimize their investment decisions by exploiting the SEC 8-K filings but also shed light on AI innovations in digital business. The data from accounting and finance filings may contain more noise and demonstrate more different distributions in comparison with other general business data. Our study suggests that TF-IDF vectorization demonstrates advantages in forecasting over the more complicated BERT vectorization. It is probably due to TF-IDF can extract more features than BERT and more features may bring more information and less noise. It may explain well why the extra excess return forecasting would achieve better performance without dimension reduction.

Acknowledgements

This work is partially supported by McCollum endowed chair startup fund.

References

1. Loughran, T., & McDonald, B. Textual analysis in accounting and finance: A survey. *Journal of Accounting Research*, *54*(4), 1187-1230 (2016).
2. Xie et al. Semantic frames to predict stock price movement. In *Proceedings of the 51st annual meeting of the association for computational linguistics,* pp. 873-883 (2013).
3. Ke, Z., Kelly, B., & Xiu, D. *Predicting returns with text data* (No. w26186). National Bureau of Economic Research (2019).
4. Loughran, T., McDonald, B. When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks. *The Journal of finance*, *66*(1), 35-65 (2011).
5. Zhai, S., Zhang, Z. Forecasting firm material events from 8-K reports. *Proceedings of the Second Workshop on Economics and Natural Language Processing,* 22-30 (2019).
6. Liu J., Chen Y., Liu K., Zhao J. (2017) Attention-Based Event Relevance Model for Stock Price Movement Prediction. Knowledge Graph and Semantic Computing. Language, Knowledge, and Intelligence. CCKS 2017. Communications in Computer and Information Science, vol 784. Springer, Singapore
7. Kogan et al. Predicting risk from financial reports with regression. *Proceedings of human language technologies: the 2009 annual conference of the North American Chapter of the Association for Computational Linguistics,* pp. 272-280 (2009).
8. Lee et al. On the Importance of Text Analysis for Stock Price Prediction. *LREC,* 2014:1170-1175 (2014).

9. Zhao, X. Does information intensity matter for stock returns? Evidence from Form 8-K filings. Management Science, 63(5), 1382-1404 (2017).
10. Engelberg, J. Costly information processing: Evidence from earnings announcements. *AFA 2009 San Francisco meetings paper* (2008).
11. Li, The information content of forward-looking statements in corporate filings, a naïve Bayesian machine learning approach. *Journal of Accounting Research*, *48*(5), 1049-1102 (2010)
12. Aydogdu *et al.* Using long short-term memory neural networks to analyze SEC 13D filings: A recipe for human and machine interaction, *Intelligent Systems in Accounting, Finance and Management*, doi/10.1002/isaf.1464, (2020)
13. Shah, D.; Isah, H.; Zulkernine, F. Stock Market Analysis: A Review and Taxonomy of Prediction Techniques. *Int. J. Financial Stud.* **2019**, *7*, 26. https://doi.org/10.3390/ijfs7020026
14. Han, H, *et al.* Enhance Explainability of Manifold Learning, *Neurocomputing* 500:877-895 (2022)
15. Lee S, Document vectorization method using network information of words. *PLoS ONE* 14(7): e0219389. https://doi.org/10.1371/journal.pone.0219389, (2019)
16. Devlin *et al.* BERT*:* Pre-training of Deep Bidirectional Transformers for Language Understanding, arXiv:1810.04805. (2019).
17. Nothman, J, Qin, H and Yurchak, R (2018). "Stop Word Lists in Free Open-source Software Packages". In Proc. Workshop for NLP Open Source Software.
18. Lansing, K., LeRoy, S, Ma. J. Examining the Sources of Excess Return Predictability: Stochastic Volatility or Market Inefficiency? Federal Reserve Bank of San Francisco Working Paper (2018)
19. Faria and Verona, The yield curve and the stock market: Mind the long run,Journal of Financial Markets,Volume 50,2020,100508
20. Cristianini, N., Shawe-Taylor, J., An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press (2000).
21. Han, H. Hierarchical learning for option implied volatility pricing. Hawaii International Conference on System Sciences (2021)
22. NLP-for-8K-documents https://github.com/hatemr/NLP-for-8K-documents
23. Han et al. Interpretable Machine Learning Assessment (June 25, 2022). Available at SSRN: https://ssrn.com/abstract=4146556, (2022)
24. Van der Maaten, L., & Hinton, G. Visualizing data using t-SNE. *Journal of machine learning research*, *9*(11) (2008).
25. McInnes, L., Healy, J., & Melville, J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction (arXiv:1802.03426). arXiv, (2020)
26. Gu *et al.* Recent advances in convolutional neural networks, *Pattern Recognition* 77 (2018): 354-377.
27. Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural networks, 18(5-6), 602-610.
28. Zhang, H, Huang, H, Han, H (2021): A Novel Heterogeneous Parallel Convolution Bi-LSTM for Speech Emotion Recognition, Appl. Sci. 11(21), 9897
29. Han et al. Transformer in transformer. *Advances in Neural Information Processing Systems* 34 (2021).
30. Geurts, P, Ernst, D and Wehenkel, L: Extremely randomized trees. *Machine learning* 63.1 (2006): 3-42.
31. More, A. (2016). Survey of resampling techniques for improving classification performance in unbalanced datasets. arXiv preprint arXiv:1608.06048.

32. Chawla, N. V. (2009). Data mining for imbalanced datasets: An overview. Data mining and knowledge discovery handbook, 875-886.
33. Goodfellow *et al*. Generative adversarial nets. NIPS, pp 2672–2680, (2014)
34. Sampath, V., Maurtua, I., Aguilar Martín, J. J., & Gutierrez, A. (2021). A survey on generative adversarial networks for imbalance problems in computer vision tasks. Journal of big Data, 8(1), 1-59.
35. Han, H, Jiang X. Overcome Support Vector Machine Diagnosis Overfitting, *Cancer Informatics*, 13(1): 145–158, (2014)
36. Buda, M., Maki, A., & Mazurowski, M. A. (2018). A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106, 249-259.
37. Han et al (2021), Predict high-frequency trading marker via manifold learning, Knowledge-based system,213(5):106662
38. Gas et al Explainable Deep Learning: A Field Guide for the Uninitiated https://doi.org/10.48550/arXiv.2004.14545, (2021)