electronics

MDPI

Article

From Ranking Search Results to Managing Investment Portfolios: Exploring Rank-Based Approaches for Portfolio Stock Selection

Mohammad Alsulmi

Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh 11451, Saudi Arabia; malsulmi@ksu.edu.sa

Abstract: The task of investing in financial markets to make profits and grow one's wealth is not a straightforward task. Typically, financial domain experts, such as investment advisers and financial analysts, conduct extensive research on a target financial market to decide which stock symbols are worthy of investment. The research process used by those experts generally involves collecting a large volume of data (e.g., financial reports, announcements, news, etc.), performing several analytics tasks, and making inferences to reach investment decisions. The rapid increase in the volume of data generated for stock market companies makes performing thorough analytics tasks impractical given the limited time available. Fortunately, recent advancements in computational intelligence methods have been adopted in various sectors, providing opportunities to exploit such methods to address investment tasks efficiently and effectively. This paper aims to explore rank-based approaches, mainly machine-learning based, to address the task of selecting stock symbols to construct long-term investment portfolios. Relying on these approaches, we propose a feature set that contains various statistics indicating the performance of stock market companies that can be used to train several ranking models. For evaluation purposes, we selected four years of Saudi Stock Exchange data and applied our proposed framework to them in a simulated investment setting. Our results show that rank-based approaches have the potential to be adopted to construct investment portfolios, generating substantial returns and outperforming the gains produced by the Saudi Stock Market index for the tested period.

Keywords: rank-based systems; machine learning; stock selection and recommendation; financial analytics; learning to rank

Citation: Alsulmi, M. From Ranking Search Results to Managing Investment Portfolios: Exploring Rank-Based Approaches for Portfolio Stock Selection. *Electronics* **2022**, *11*, 4019. https://doi.org/10.3390/electronics11234019

Academic Editors: Phivos Mylonas, Katia Lida Kermanidis and Manolis Maragoudakis

Received: 24 October 2022
Accepted: 30 November 2022
Published: 4 December 2022

1. Introduction

Nowadays, financial markets (e.g., stock exchanges, currency markets, and commodity exchanges) play a major role in the global economy by reflecting countries' economic growth and stability [1,2]. The stock market is a type of financial market that provides an effective platform for listed companies and investment institutions to trade and exchange various types of securities (e.g., stocks, derivatives, and options). For listed companies in particular, stock markets can provide a way to realize fair share value, increase the potential of growing a company's capital, and provide liquidity for shareholders. For investors (both individuals and investment firms), stock markets provide a set of tangible opportunities to diversify investment portfolios and produce financial gains, while keeping a transparent environment [3].

However, making investments in financial markets is not an easy or straightforward task; it requires tremendous effort from financial analysts and investment advisers to study a target stock exchange in search of investment opportunities. Generally, domain experts perform extensive research on stock markets for companies, which involves

collecting a large volume of data (e.g., periodic financial reports, announcements, news, etc.), performing several data analytics tasks, and making inferences to reach investment decisions. With the rapid increase in data generated per company listed in those markets, the task of manually analyzing companies' financials gradually becomes much harder, especially when timely investment decisions are needed.

Fortunately, with the recent successes of computational intelligence methods that were adopted in a wide range of data analytics applications (particularly in the financial analytics services' domain [4–7]), these methods can be well exploited to assist financial analysts and stock market investors when analyzing companies and making informed investment decisions. This paper is an attempt to explore one category of these methods—rank-based approaches—and use them to select stock symbols from a target financial market and rank them according to their relevance to an investment plan. The rank-based approaches used in this work, known as learning to rank (LtR) algorithms [8], were originally proposed to search and retrieve textual content to be used in a variety of search applications (e.g., search engines, recommender systems, etc.).

The stock selection task, considered in this work, can be seen as a ranking task by nature. Therefore, in this paper, we advocate for adopting these methods for our task and formulate the investment task as a ranking problem. We also propose a set of features suitable for representing stock symbols in LtR methods. To examine the usefulness of our methods, we selected the Saudi Stock Exchange, one of the fastest-growing exchanges around the world, as a case study and performed our evaluations by creating several simulated long-term investment portfolios with an investment period of four years. The findings from our evaluations suggest that the rank-based approaches are very useful for long-term investments and for achieving substantial performance compared to the gains produced by the Saudi Stock Market's index.

To summarize, our work makes the following contributions. First, we created a new dataset consisting of many instances such that each instance is represented using a diverse set of features. This dataset allows us to experiment with LtR frameworks, specifically for financial analytics tasks. Secondly, we reformulated our investment and stock selection task as a ranking problem and conducted a comprehensive exploration of several rank-based methods (both learning-based and fusion-based). Lastly, we provided an evaluation framework and examined the usefulness of two performance measures used to evaluate the effectiveness of rank-based methods. Our examination shows which of these measures positively correlate with investment returns and indicate the real performance of rank-based methods.

The remainder of this paper is organized as follows. Section 2 provides a brief introduction to the topic of the paper by defining the research problem and examining prior work for stock market investment. Section 3 proposes our framework that applies rank-based approaches to stock investments and describes our dataset in a detailed way. In Section 4, we show an empirical evaluation of the proposed framework and provide analysis and discussions of the evaluation results. Finally, Section 5 summarizes the contributions of this paper and provides concluding remarks.

2. Background

Rank-based search systems, formally known as information retrieval systems, have been widely used in many applications, including web and multimedia searches [9,10], information filtering, task suggestions [11,12], question answering [13,14], and clinical decision support [15–17]. A typical search system works by accepting a search request provided by a user as an explicit query consisting of several keywords that define the user's information need. The search system then processes the search request and produces search results, usually as a list of retrievable items (e.g., webpages) that are ranked according to their estimated relevance to the user's query [18]. Consequently, the user will scroll down through the produced results list and consume some items, which may satisfy his or her needs.

Since the early adoption of search and retrieval systems, several rank-based approaches (i.e., retrieval models) have been proposed to rank search results. These approaches include some conventional models, such as the vector space model (VSM) [19], which ranks search results based on the cosine similarity scores between a query and a textual item, the language model (LM) [20], which ranks search results based on the similarities between a pair of language models for queries and items, and other probabilistic models such as BM25, which ranks results in decreasing order of their estimated relevance to queries [21]. However, due to the complexity of search and ranking tasks, relying solely on conventional ranking models may not be sufficient when building very effective systems to address users' needs (i.e., systems that can produce high-quality results). Incorporating other approaches, particularly learning-based methods such as learning to rank (LtR) [8], to combine various data sources (e.g., ranking models, user clicks, previous user interactions, and results of related queries) during retrieval time has been shown to be beneficial because it can leverage the performance of these systems and improve the quality of search results [22–24]. This will ultimately increase user satisfaction by servicing their needs.

2.1. Learning to Rank (LtR)

LtR is one category of machine learning (ML) methods that has been adapted for search and ranking problems. LtR methods learn by incorporating many parameters (i.e., features) that are extracted from each query and result-item pair (e.g., from a query and a potentially matching search result) [8]. The resulting models can then be used to rank search results for new search requests from users by predicting the relevance of items retrieved for a given query and ranking these items (i.e., results) accordingly. LtR methods are generally differentiated by the type of machine-learning methodology they rely on (e.g., regression trees, neural networks, or SVMs) and can also be differentiated by the type of loss function that they employ (i.e., pointwise, pairwise, or listwise function) [25–30]. LtR methods that use pointwise are trained to predict the relevance of an item to a query without taking into consideration the inter-dependency between the items in the search results list. In other words, predicting one item's relevance to a query is solely based on that item's feature values and without considering its relationship to other items in the search results list. In contrast, both pairwise and listwise methods consider the inter-dependency; pairwise methods consider the relative order between two items in the search results list, whereas listwise methods consider the predicted relevance of an item with respect to the other items in the list [8].

To learn an LtR model for a ranking task, a sample of training data that consists of the following sets is required:

- A set of queries, $Q = \{q_1, q_2, q_3, \ldots, q_m\}$, where each query q_i represents a potential search request by a user.
- A set of retrievable items (or documents), $D = \{d_1, d_2, d_3, \ldots, d_n\}$, where each item d_j can be a potential search result for query q_i.
- A set of relevance judgments, $R = \{r_{1,1}, r_{1,2}, r_{1,3}, \ldots, r_{2,1}, \ldots, r_{m,n}\}$, where each judgment $r_{i,j}$ is labeled by humans to indicate whether an item j is relevant to a query i.
- A matrix of query-item pair features, F, where each row consists of a vector $f_{i,j} = \{f_1, f_2, f_3, \ldots, f_k\}$ that captures certain properties related to a query i and item j.

Finally, each training instance $t_{i,j}$ (a pair of query i and item j) in the dataset can be represented as a vector of the feature values $f_{i,j}$ and a label $r_{i,j}$. Once a model is trained using this data, it can be used to rank search results for new (unseen) queries by extracting feature values from each query-item pair and using these values to predict whether an item is relevant to a given query. The final ranking of all items will be based on the estimated relevance of these items to query i [16].

2.2. Stock Selection as a Learning to Rank (LtR) Problem

The underlying problem that this work considers is concerned with assisting an investor or a financial analyst in building an investment portfolio that is represented by a set of stock symbols' shares selected from a target stock market (in our case, we will consider selecting stocks from the Saudi Stock Exchange). More specifically, we will focus on addressing how to select stock symbols from a given market in an effective way. Our main objective is to maximize the portfolio's returns at a specified time period. Assuming that we focus on long-term investments (i.e., no active management of a portfolio is needed), this period can be set to a single year.

Having discussed search and ranking problems, we propose reformulating our problem as a ranking task to potentially apply LtR methods. Ultimately, our goal is to build a ranking system that can rank a set of potential investment stock symbols (i.e., companies) based on their relevance to certain criteria (i.e., based on which stocks are expected to make positive returns at the end of a given period). Using this formulation, a user query can be defined as an implicit question (i.e., not in the textual form) of which stocks should be selected given a time frame (e.g., "I am at the start of the year 2020; which stocks will make high returns by the end of the year?"). The items in our formulation can be defined as the stock company symbols that are available for investment depending on the user query.

The features can also be defined as a vector of property values that captures certain aspects of a symbol for each query-stock symbol pair. For instance, if the user's query is to predict the most profitable stocks by the end of a given year (i.e., 2019 or 2020, where the prediction of each year is considered a distinct query), then the feature values will contain certain properties about a symbol (e.g., revenue, capital value, market value) to capture the period prior to and up to the time when the query is initiated (i.e., use the data collected up to the end of 2018 to predict stocks for 2019). Once the data are defined using the described formulation, LtR methods can be applied to train models, which can be applied to stock selection. In Section 3, we thoroughly describe our methodology for applying LtR to Saudi Stock data.

2.3. Related Work

Researchers and scientists have focused their attention on financial markets due to their importance in shaping the global economy and reflecting on countries' economic wealth. A wide range of computational intelligence approaches have been developed to analyze markets' movements and assist in enhancing the task of investing in markets. For machine-learning (ML) approaches, most of the proposed approaches are focused on developing models to assist investors and financial analysts with their long-term (i.e., passive) and short-term (i.e., active) stock market investments [4–7,31–34].

For instance, Chiang et al.'s [4] work is notably aimed at applying multi-layer perceptrons (MLPs) combined with particle swarm optimization to predict the movements of U.S. market indices (e.g., NASDAQ and SP500) for the next day in a trading period. Predicting the movements of these stock market indices has been shown to be useful in deciding the entry and exit points for trading actions and leads to investment returns. Alsubaie et al. [5] explored several ML models, such as support vector machines (SVMs), MLPs, and naïve Bayes, and considered various sets of technical indicators. The authors used these models to simulate active trading actions in the Saudi Stock Exchange and showed that different ML models resulted in different investment returns, with naïve Bayes resulting in a higher performance than the other models. Alsulmi and Al-Shahrani [6] also explored applying several ML models, including long short-term memory networks (LSTMs) and random forests, to the task of investment and trading in the Saudi Stock Market. Their study's findings suggest that combining ML-based trading with a portfolio's risk management techniques is very useful for the trading task and has the

potential to outperform the conventional hold-and-buy strategies adopted by many investors.

All of the approaches described above focus on exploiting various ML methods to invest and make financial returns by actively trading in the stock market (e.g., identifying entry and exit points of stock and actively buying/selling the stock within short periods). However, other types of approaches attempt to analyze stock market data by relying on various ML and computational intelligence techniques to select and recommend sets of stock symbols that are suitable for constructing long to medium-term investment portfolios. One example of this category is the active learning method introduced by Yan and Ling [7], which is also called prototype ranking and is based on clustering. The proposed approach learns a network model, mainly by utilizing two features (stock prices and the volume of traded shares) to select some of the potential stocks listed on NYSE and AMEX. The findings show that the approach is useful for stock selection and is comparable to other non-ML methods used for this task.

Yu et al. [31] introduced another stock selection method which relies on supervised ML with SVMs and principal component analysis (PCA). The method is used as a classifier rather than a ranker and is applied to predict the top stock symbols out of the 677 symbols listed in the Chinese A-share stock exchange; each symbol is represented by seven features that represent different ratios (e.g., earnings ability, cash ratios, and risk levels) and a target label. An analysis of this method indicates that it has the potential to identify top stocks from the target stock exchange. Yuan et al. [32] also explored several supervised ML models, such as SVMs, MLPs, and random forests, and used them for long-term investment and portfolio stock selection for the Chinese A-share stock market. The proposed method utilized a large number of features (mainly features related to the daily trading of stocks, including opening price, closing price, and volume) to predict which stock symbols are expected to perform the best. Similar to Yu et al. [31], the methods proposed in this study are used as classifiers, not rankers.

Other studies, such as Song et al. [33], explored using the LtR approach for stock selection, which is accomplished by defining the investment task as a ranking problem. Song et al. [33] used a set of statistics based on investor sentiment collected from news articles as features for training several LtR models. The aforementioned method is applied for stock selection in the U.S stock market by considering two investment strategies: long-only and long-short strategies. Findings from this work indicate the potential of LtR methods for this task due to it outperforming S&P 500 index's returns for the considered testing period. Saha et al. [34] also proposed formulating the stocks selection task as an LtR task by introducing an ML method that is based on relational graphs of market stocks. Although the method is applied in active daily trading and not long-term investments, empirical evidence indicated its usefulness for the task of stock selection by considering two U.S. markets (NASDAQ and NYSE).

Our work in this paper shares some similarities with prior work, such as representing our investment task as a ranking problem. Nevertheless, our work is distinguished because we reformulate the task using the LtR framework by clearly defining queries and items and explaining how they are linked using the pairwise feature values. This allowed us to consider a more comprehensive list of LtR learners and to explore a new set of features representing each query and item pair. Moreover, we applied our methods in the context of the Saudi Stock Market, and to our knowledge, this work is the first to adopt these methods for such a stock exchange.

3. Materials and Methods

This section describes our methodology for implementing the LtR framework into stock symbol selection. We first describe the proposed representation of our problem and then examine the data collection process used to gather the data for our approach. Afterwards, we discuss how to aggregate the collected data to generate learning features. Lastly, we discuss ways to apply model learning using several LtR algorithms.

3.1. Problem Representation

We represent our problem, which is concerned with selecting stock symbols to maximize a portfolio's returns, as a ranking problem. Therefore, as described in Section 2.2, we propose applying LtR methods to learn models for ranking stock symbols. We assume that users intend to build long-term investment portfolios and set the investment period to one year. Learning an LtR model for this task requires a set of queries Q (i.e., a set of implicit questions of which company stocks to select for each year), a set of items D (i.e., company stocks), a set of pairwise ground truth labels R (to indicate whether company stock is relevant to a given year's query), and a set of pairwise feature values $f_{i,j}$ (to indicate certain statistics about a company stock j for a given year's query i). Ultimately, each instance in our data will be a vector of pairwise feature values (for query i and item j) and a target label r_{ij} (e.g., $f_{i,j} = [f_1, f_2, f_3, \ldots, f_k] \rightarrow r_{ij}$). By applying an LtR algorithm to the provided data instances, we can train a model that predicts stock relevance and ranks them accordingly. Next, we discuss our process for collecting and generating the data to build our model.

3.2. Data Collection

The target market we consider in this work is the Saudi Stock Market (Tadawul), which has over 200 listed companies. Tadawul is one of the fast-growing stock exchanges worldwide and has a market capitalization of over US $ 2.22 trillion (ranked 9th among the 67 members of the World Federation of Exchanges) [35]. One limitation is that no publicly available dataset is suitable for applying LtR methods to our target stock exchange. Therefore, part of our methodology is concerned with collecting data from several sources and aggregating data to generate a dataset suitable for training LtR models. Consequently, we developed a bot for crawling our required data, which occurs through two main tasks: acquiring a company stock's profile information and gathering each stock's annual financial results. We describe these two tasks in Sections 3.2.1 and 3.2.2. We implement our bot using Java and by relying on jsoup parser [36] to fetch URLs, extract the required data properly, and further manipulate data.

In addition to the crawled data, our analysis will rely on the historical market data the Saudi Stock Market authority has released [37]. The data contain the daily trading information for all the listed stocks for the period we considered in this study. Section 3.2.3 provides more insights into this data, including the main parameters used.

3.2.1. Stocks' Profile Information

The market authority of Tadawul provides a profile for each company listed in the stock market. The profile presents detailed information about the company stock, including stock symbol code, listing name, sector, listing date, establishment data, and equity profile. Figure 1a–d show samples of company profile information provided on Tadawul's website.

Because we need the companies' profile information to generate some of our features, we run our bot on these profiles to extract a set of suitable HTML tags for the following attributes: symbol code, listing name, sector, listing date, paid-in capital, the number of issued shares, and paid-up value per share. In addition, we extract some statistics regarding the changes in a company's capital since its listing date in the market, as Figure 1d shows. This information will be processed later during the data aggregation stage to generate suitable features matched with a suitable query-item pair.

3.2.2. Stock Financial Results

In addition to companies' profiles, the market authority of Tadawul provides the financial results of the stock market's listed companies, which each company announces for several periods: three months, six months, nine months, and one year. The results include several attributes indicating the company's performance, such as revenues, net

profits, and profits per share for a given period. Figure 2 shows a sample of the financial results provided on Tadawul's website. From these results, we select the annual results for each stock (revenue per year, profit/loss per year, profit/loss per share, etc.), and we run our bot to crawl their data by extracting their suitable HTML tags. As with the company profile data, we will use the crawled data for this part later to produce features and match them with query-item pairs.

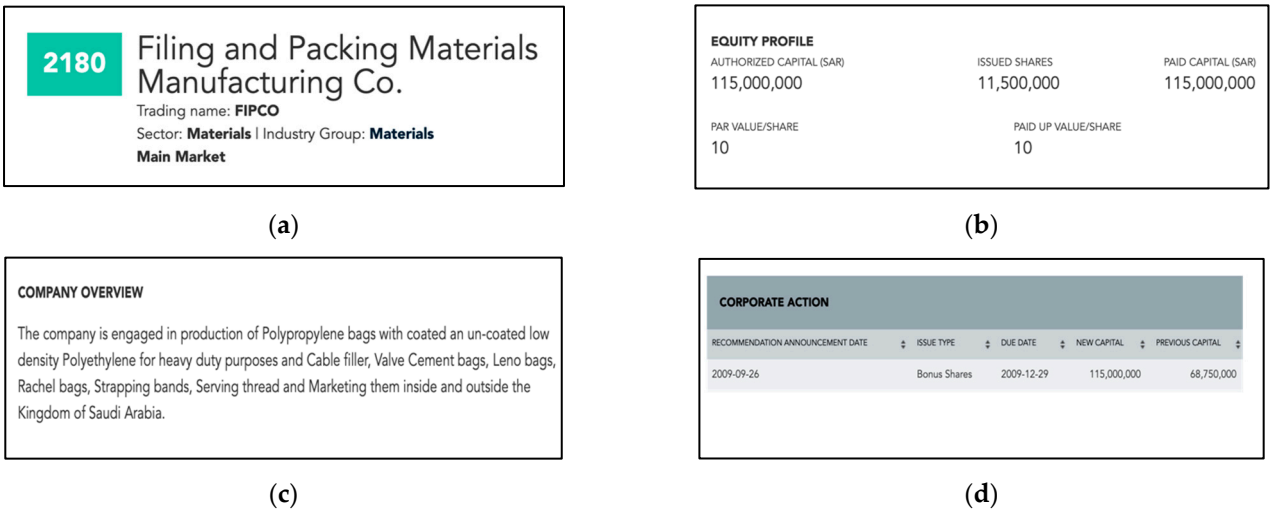

Figure 1. Samples of profile information that Tadawul has released for each stock symbol, including (**a**) company identification information, (**b**) equity profile, (**c**) company overview information, and (**d**) company capital-changes history.

ELEMENT LIST	CURRENT PERIOD	SIMILAR PERIOD FOR PREVIOUS YEAR	%CHANGE
Sales/Revenue	73,251,263	69,441,606	5.486
Gross Profit (Loss)	15,691,213	14,209,330	10.428
Operational Profit (Loss)	8,977,489	10,295,154	-12.798
Net Profit (Loss) after Zakat and Tax	7,996,628	9,404,310	-14.968
Total Comprehensive Income	8,888,280	9,404,310	-5.487
Total Share Holders Equity (after Deducting Minority Equity)	41,442,598	32,554,318	27.302
Profit (Loss) per Share	2.67	142.23	
All figures are in (Actual) Saudi Arabia, Riyals			

Figure 2. A sample of annual financial results companies release and Tadawul publishes.

3.2.3. Stocks' Historical Trading Data

The historical market data Tadawul's authority releases (through their EReference data service in [37]) include information about stocks' trading prices per day since their initial listing. Every instance of the data represents a trading day for a stock in the market. It includes several attribute values, such as stock company name, symbol code (each stock symbol's unique id), date, stock opening price, stock highest price, stock lowest price, stock closing price, and the volume of shares traded that day. Table 1 shows a sample of the historical trading data Tadawul provides. It is worth mentioning that these data are used to generate some feature values and to facilitate the process of producing the ground-truth labels for our training instances (which we will describe next).

Table 1. Sample of daily trading Saudi Exchange data releases by Tadawul's authority. Stock prices are reported in Saudi riyals (SAR).

Company Name	Symbol Code	Date	Open Price	Highest Price	Lowest Price	Close Price	Volume
Jarir	4190	2018-01-01	146.0	146.6	146.0	146.6	5669
Jarir	4190	2018-01-02	146.6	146.6	145.0	145.5	8050
Alrajhi	1120	2018-01-01	64.6	65.2	64.1	65.0	2,788,920
Alrajhi	1120	2018-01-02	65.1	65.3	64.6	64.6	2,605,433
STC	7010	2018-01-02	68.2	68.2	66.9	67.1	178,184
STC	7010	2018-01-03	67.5	67.5	66.9	67.4	164,584

3.3. Data Aggregation and Feature Generation

Having collected the data from several sources (i.e., companies' profiles, annual financial reports, and historical trading data), we now aggregate such data and use them to generate a dataset that is suitable for training LtR models. We produce feature values for each query-item pair such that for each query (i.e., each year included in our analysis), we produce a set of statistics for each company. These statistics are intended to indicate these companies' performance throughout a year (e.g., net profits, capital growth, and P/E ratios) [38] and differentiate companies. Additionally, these statistics can reflect the changes in companies' stocks from one year to another (increase in paid capital, change in market value, etc.). Overall, we generated a set of 15 features for each pair of query i and item j. Table 2 presents these features along with a description of each one. We extract some of the considered features directly from the aggregated data (e.g., symbol code, sector, paid capital, and total net profits/loss) whereas we estimate other features, such as market value, net profits to capital (as a percentage), price-earnings (P/E) ratio [38], and price-earnings (P/E) indicators [39], by performing simple calculations using the extracted data or applying a financial analyst rule of thumb.

Table 2. A set of 15 features is generated for each year i (query) and stock symbol j (item).

Feature	Description
Symbol code	Unique identifier of each company's symbol.
Sector	The company's main domain of activities (banking, telecommunication, insurance, etc.).
Paid-in capital	Total amount of capital investors paid.
Market value	Estimated by *number of shares* * *share market price*.
Stock price	Stock's closing price.
Total net profit/loss	Total annual revenues minus total expenses and operational costs.
Profit/loss per share	Total annual profit/loss divided by the number of shares.
Net profits to capital percentage	Estimated by (*net profit/paid-in capital)* * 100.
Market value to capital percentage	Estimated by (*market value/paid-in capital)* * 100.
Capital growth percentage	The difference (%) in paid-in capital between two consecutive years.
Capital growth frequency	The frequency of increases in a company's capitalization.
Market value growth (1 year)	Estimated by ($market\ value\ year_i — market\ value\ year_{i-1}$).
Market value growth (3 years)	Estimated by ($market\ value\ year_i — market\ value\ year_{i-3}$).
P/E ratio	Estimated by (*share market price/profit per share*).
P/E indicator	Indicator of whether a P/E ratio value is high, medium, or low, estimated by a financial analyst rule of thumb [39].

In addition to generating features' data, we produced the set of relevance judgments, the ground-truth label *R*, for each query *i* and item *j* (i.e., the year *i* and a stock *j*). Fortunately, rather than relying on human feedback, we can estimate those labels by examining the historical daily trading information and whether a stock symbol generates a positive return for a given year. For instance, to estimate whether a stock symbol, *j*, is relevant to invest in for year *i*, we generate the label $r_{i,j} \in \{0$: not relevant, 1: potentially relevant, 2: definitely relevant, 3: highly relevant$\}$ by measuring the difference in the price of *j* at the start of year *i* and its end. If the difference indicates a growth in the stock price, *j* is labeled with one of relevance labels for year *i*; otherwise, it will be labeled as not relevant. It is worth noting that to simplify our task and for illustration purposes, we only considered four labels (three levels of relevance and one for non-relevance). Additionally, the distinction among these labels is defined by setting the threshold values t_1, t_2, and t_3, as Equation (1) shows. Later, in our evaluation section, we discuss the suggested values for these parameters.

$$r_{i,j} = \begin{cases} 0, & if\ price\_growth(stock_j) \leq t_1 \\ 1, & if\ price\_growth(stock_j) > t_1\ \ and < t_2 \\ 2, & if\ price\_growth(stock_j) \geq t_2\ \ and <\ t_3 \\ 3, & if\ price\_growth(stock_j) \geq t_3 \end{cases} \quad (1)$$

3.4. LtR Model Learning

Once the features and labels are generated for each pair of *i* and *j*, we reformat the data to make the resulting dataset well-prepared for the LtR learning procedures. LtR frameworks, such as RankLib [40] and TF-Ranking [41], have a specific format for representing data instances such that each instance, a pair of query *i* and item *j*, is represented as ($r_{i,j}$ *qid:i 1:* $f_{i,j,1}$ *2:* $f_{i,j,2}$ *3:* $f_{i,j,3}$ …. *k:* $f_{i,j,k}$). $r_{i,j}$ is a label indicating the relevance of item *j* (a company stock in our case) for query *i* (i.e., a year), *qid* is the query id, and *1*, *2* through *k* represent feature values for that pair. Now, we can apply LtR learning procedures to train a model for the stock selection task such that for a new unseen query (i.e., a new year), it predicts the stocks with the most potential positive investment returns by the end of that year. Training an LtR model involves deriving a function that maps the input space (i.e., data instances) to the output space (i.e., predictions) relying on the feature values by the input data. In the derivation of such a function, a loss function is needed to guide the learning process and measure the correctness of produced predictions to the ground truth-labels. As described in Section 2.1, LtR algorithms are generally categorized according to their loss functions as pointwise, pairwise, or listwise (see [8] for a detailed review). Several algorithms have been proposed for LtR model learning, and in this work, we consider nine of these learners spanning various ML techniques (trees, boosting, neural networks, etc.) as well as various loss functions. We implement the considered algorithms using a recent version of the RankLib tool [40]. Table 3 lists these algorithms.

Table 3. The considered LtR algorithms with their corresponding ML models and loss function.

LtR Algorithm	ML Method	Loss Function
Linear regression [25]	simple regression	pointwise
MART [26]	trees	pairwise
LambdaMART [27]	trees	listwise
LambdaRank [28]	neural network	listwise
Coordinate ascent [29]	optimization search	pointwise
RankBoost [30]	boosting	pairwise
Random forests [42]	trees	pointwise
RankNet [43]	neural network	pairwise
ListNet [44]	neural network	listwise

In addition, we consider applying rank fusion methods that can produce ranked lists by combining the results from several LtR methods. Particularly, we examine two rank-based fusions, inverse square rank (ISR) [45] and reciprocal rank fusion (RRF) [46], which are defined by Equations (2) and (3) below.

$$ISRScore(j) = N(j) * \sum_{k=1}^{N(j)} \frac{1}{R_k(j)^2} \quad (2)$$

$$RRFScore(j) = \sum_{k=1}^{N(j)} \frac{1}{L + R_k(j)} \quad (3)$$

ISRScore(j) and *PRFScore(j)* in the above equations represent the scores of an item *j* after we apply the corresponding fusion method to combine the ranked lists from several LtR methods. *N(j)* represents the number of ranked lists that item *j* appears in, $R_k(j)$ represents the rank of item *j* in ranked list *k*, and *L* is a constant (it is usually set to 50).

Finally, to optimize LtR learners' learning process, we rely on the normalized discontinued cumulative gain (nDCG) [47]. It measures a ranked list's performance by utilizing items' graded relevance (i.e., it considers several levels of relevance, as in our case) rather than considering only binary relevance (e.g., relevant vs. not relevant), as in precision [48], recall [49], and F1 measures. nDCG works under the assumption that relevant items are more useful than marginally relevant items, which in turn are more useful than non-relevant items. Moreover, it favors highly relevant items appearing at the top of the ranked list and performs score penalization when they appear at the bottom. For query *i*, nDCG is measured at specific ranking position *k* (i.e., the top *k* results) according to the following equations,

$$nDCG@k = \frac{1}{iDCG} * \sum_{j=1}^{k} \frac{2^{r_{i,j}} - 1}{log_2(j+1)} \quad (4)$$

$$iDCG@k = \sum_{j=1}^{k} \frac{2^{ideal, r_j} - 1}{log_2(j+1)} \quad (5)$$

where *iDCG* is the ideal discontinued cumulative gain computed for a ranked list of ideal items as defined in Equation (5), $r_{i,j}$ is the degree of relevance of item *j* to query *i*, and $log_2(j + 1)$ is the discounting factor. Next, we describe our evaluation of the proposed approach relying on Saudi Stock Exchange data.

4. Results and Analysis

Having described our methods for applying LtR for the stock selection task, in this section, we evaluate these methods. We start by describing our setup for our experiments. Then, we report the results of evaluating LtR models' effectiveness and provide a case for applying these models when investing in the Saudi Stock Exchange. Finally, we analyze our results and provide further discussions.

4.1. Experimental Settings

The used dataset consists of the historical data for the Saudi Stock Market containing information about listed companies in the market (excluding REITs and ETFs). We accumulated the dataset using the procedures described in Section 3.3. The produced dataset covers the period from 2013 to the end of 2021 (nine years) and includes 1437 instances such that each instance is represented by 15 features and a target label (ranging from 0 to 3). We set the thresholds t_1, t_2, and t_3 for labeling data instances, defined in Equation (1), to 0%, 25%, and 50%, respectively (we selected these values because they

lead to an effective balancing of the data among the various labels and effective grouping of the stocks based on their returns).

We trained nine LtR models (described in Section 3.4) to select stock symbols for the last four years (2018, 2019, 2020, and 2021) in our dataset. For instance, to predict the top stock symbols for 2018 (i.e., rank the 167 stocks listed for that year), we trained our LtR models on the data for the period starting in 2013 and ending in 2017 (excluding any instances from 2019, 2020, and 2021). We did so to eliminate any potential learning bias and avoid overestimating these models' effectiveness. We did the same for 2019, 2020, and 2021 (e.g., to predict the top stocks for 2021, we trained with the instances for the period starting in 2013 and ending in 2020). Moreover, to fine-tune each learner, guide the learning process, and avoid overfitting, we randomly selected 10% of our training data and used it as a holdout validation set. Additionally, as described in Section 3.4, we used nDCG@10 as the main metric to optimize these learners on our dataset.

Finally, we performed two types of experiments, one to measure LtR models' effectiveness (i.e., the performance of these models) while they are used to rank stock symbols, and the second to examine these models' usefulness in constructing investment portfolios. For the first set of experiments, we report LtR models' effectiveness using two common measurements for search and ranking systems: precision@k [48], which relies on binary relevance and measures the proportion of items that are relevant in the top k results of a ranked list, and nDCG@k [47], which considers graded relevance and measures ranking effectiveness as defined in Equation (4).

For the second set of experiments, we created several simulated investment portfolios, each with a capitalization of 100 K Saudi riyals (SAR), and we simulated investment in the stock symbols each of the learned models selected. We measure these models' usefulness by estimating the returns (profits/losses) each portfolio made for the four years included in our testing data. We report the results for both experiments in Sections 4.2 and 4.3.

4.2. The Effectiveness of LtR Models for Stock Selection

We evaluated the considered LtR models' effectiveness in predicting top stock symbols for 2018, 2019, 2020, and 2021. Table 4 presents the results using precision (P) and nDCG. We report both metrics at two ranking cutoffs, 10 and 20 (i.e., top 10 and 20 stocks). We report these metrics' averages for the four years reported along with these results.

Table 4 shows that these learners resulted in a wide range of effectiveness values. More specifically, considering the precision measure, the performance is shown to be as high as 1.0, indicating that a model performed extremely well and that its selected stocks are relevant (e.g., RankNet with P@10 for 2019, 2020, and 2021), and it can be as low as 0.2, indicating that a learner performed poorly because only 20% of its selected stocks are relevant (e.g., RankBoost for 2020). However, because precision relies on binary relevance (i.e., all three labels indicating relevance are considered the same) and due to the nature of our task, these results may not accurately reflect the real model's performance and could be misleading. This shortcoming is apparent when a model has nearly all its selected stocks making only 1% of returns per year; then that learner will be deemed highly effective per its precision (because the labeling threshold, t_1, is set to 0%, precision will consider all the positive returns equally relevant).

Table 4. The ranking performance results for applying nine LtR models to predict the top stocks in Saudi Exchange for four years, 2018, 2019, 2020, and 2021. Underlined values represent the models with the highest effectiveness for a metric. Superscript numerals in parentheses represent the rank of a model among all models using nDCG.

2018				
Model	**P@10**	**P@20**	**nDCG@10**	**nDCG@20**
Linear regression (LR)	0.1000	0.2000	0.0122 (9)	0.0010 (9)
MART	0.5000	0.6000	0.1159 (7)	0.1228 (6)
LambdaMART (LMART)	0.6000	0.6000	0.5042 (1)	0.3268 (1)
LambdaRank (LRank)	0.2000	0.2000	0.3301 (4)	0.1313 (5)
Coordinate ascent (CA)	0.6000	0.5500	0.2221 (5)	0.1198 (7)
RankBoost (RB)	0.7000	0.7500	0.1483 (6)	0.1601 (4)
Random forests (RF)	0.6000	0.5000	0.3463 (3)	0.1870 (3)
RankNet (RNet)	0.6000	0.4500	0.3971 (2)	0.2186 (2)
ListNet (LNet)	0.3000	0.2000	0.0306 (8)	0.0024 (8)
2019				
Model	**P@10**	**P@20**	**nDCG@10**	**nDCG@20**
Linear regression (LR)	0.4000	0.3500	0.1967 (7)	0.2317 (5)
MART	0.5000	0.6000	0.2199 (5)	0.2285 (6)
LambdaMART (LMART)	1.0000	0.9000	0.3742 (3)	0.3360 (4)
LambdaRank (LRank)	0.7000	0.8000	0.2950 (4)	0.3735 (2)
Coordinate ascent (CA)	0.8000	0.8000	0.1786 (9)	0.1668 (8)
RankBoost (RB)	0.7000	0.6500	0.1871 (8)	0.1522 (9)
Random forests (RF)	0.5000	0.7500	0.5200 (1)	0.4369 (1)
RankNet (RNet)	1.0000	0.7500	0.4462 (2)	0.3658 (3)
ListNet (LNet)	0.6000	0.6500	0.1980 (6)	0.1946 (7)
2020				
Model	**P@10**	**P@20**	**nDCG@10**	**nDCG@20**
Linear regression (LR)	0.8000	0.8500	0.5086 (7)	0.6042 (5)
MART	0.8000	0.8000	0.7830 (3)	0.7327 (3)
LambdaMART (LMART)	1.0000	0.9500	0.6536 (5)	0.6200 (4)
LambdaRank (LRank)	1.0000	0.9500	0.9552 (1)	0.8582 (1)
Coordinate ascent (CA)	0.8000	0.8000	0.6009 (6)	0.5082 (7)
RankBoost (RB)	0.2000	0.4500	0.0571 (9)	0.0723 (9)
Random forests (RF)	0.8000	0.7000	0.7825 (4)	0.6007 (6)
RankNet (RNet)	1.0000	0.9500	0.4554 (8)	0.4446 (8)
ListNet (LNet)	0.9000	0.9000	0.9266 (2)	0.8502 (2)
2021				
Model	**P@10**	**P@20**	**nDCG@10**	**nDCG@20**
Linear regression (LR)	0.3000	0.4000	0.1535 (9)	0.1810 (9)
MART	0.8000	0.8000	0.4409 (4)	0.3658 (4)
LambdaMART (LMART)	0.7000	0.6500	0.5033 (2)	0.4504 (2)
LambdaRank (LRank)	0.9000	0.8500	0.4695 (3)	0.4443 (3)
Coordinate ascent (CA)	0.3000	0.4500	0.2199 (8)	0.2211 (8)
RankBoost (RB)	0.7000	0.6500	0.2612 (7)	0.2212 (7)
Random forests (RF)	0.8000	0.7000	0.4194 (5)	0.3330 (5)
RankNet (RNet)	1.0000	0.7500	0.2964 (6)	0.2997 (6)
ListNet (LNet)	0.5000	0.6000	0.5516 (1)	0.4725 (1)
Mean of four years				
Model	**P@10**	**P@20**	**nDCG@10**	**nDCG@20**
Linear regression (LR)	0.4000	0.4500	0.2265 (8)	0.2211 (8)

MART	0.6500	0.7000	0.3899 (6)	0.3625 (3)
LambdaMART (LMART)	0.8250	0.7750	0.5088 (2)	0.4333 (2)
LambdaRank (LRank)	0.7000	0.7000	0.5125 (1)	0.4518 (1)
Coordinate ascent (CA)	0.6250	0.6500	0.3054 (7)	0.2540 (7)
RankBoost (RB)	0.5750	0.6250	0.1634 (9)	0.1514 (9)
Random forests (RF)	0.6500	0.6500	0.4772 (3)	0.3608 (5)
RankNet (RNet)	0.9000	0.7250	0.3987 (5)	0.3322 (6)
ListNet (LNet)	0.5750	0.5875	0.4267 (4)	0.3612 (4)

Therefore, it would be more effective to consider nDCG the main indicator of the model's performance because it can accurately account for various relevance levels and rewards ranking models that have highly relevant stock symbols (i.e., generating returns of at least 50%) appearing at the top of a ranked list. nDCG is similar to precision because it shows high disparities in performance values among the LtR models, suggesting that these learners are not equivalent, considering our task. The ranking effectiveness, on average, is shown to range from 0.1634 (RankBoost) to 0.5125 (LambdaRank), considering nDCG@10, although such a difference can be much higher, as in 2020 for both models (RankBoost resulted in 0.0571 nDCG@10 whereas LambdaRank resulted in 0.9552). A multi-way ANOVA test shows that there is no statistically significant difference among those learners (as a group) considering our task (although the *p*-value of 0.06 is close to the significance threshold, 0.05). However, when we compare each pair of learners, a pairwise t-test would indicate that a significant difference remains among some of them in several models (e.g., LambdaRank vs. RankBoost). Table 5 summarizes the results for this part for all model pairs.

Table 5. A pairwise one-sided t-test is applied to each pair of the LtR model considering the nDCG metric. "1" indicates that a statistically significant difference among a pair was observed, whereas "-" indicates no statistical significance.

Models	LR	MART	LMART	LRank	CA	RB	RF	RNet	LNet
LR	-	1	1	1	-	-	1	-	1
MART	1	-	-	1	1	1	-	-	1
LMART	1	-	-	-	1	1	-	1	-
LRank	1	1	-	-	1	1	-	-	-
CA	-	1	1	1	-	-	1	-	1
RB	-	1	1	1	-	-	1	1	1
RF	1	-	-	-	1	1	-	1	-
RNet	-	-	1	-	-	1	1	-	-
LNet	1	1	-	-	1	1	-	-	-

Another observation from Table 4 is that the performance of LtR models degrades as one moves down in the ranked list. This is especially true for the nDCG measure (i.e., selecting the top 10 stocks would be more effective than selecting the top 20). This is often the case in various search and ranking tasks (e.g., as in [12,16,24]) because ranking models typically work by attempting to push more relevant items to the top of a ranked list as the user is expected to ignore the items further down in the list and only focus on the top (nDCG and other measures for evaluating ranking effectiveness are based on this assumption [47,48]).

To summarize our analysis for this part, our results in Tables 4 and 5 suggest that there is a noticeable difference among the various models used for this task. Thus, we can clearly see that four of our learners (LambdaRank, LambdaMART, Random forests, and ListNet) have achieved high effectiveness compared to other learners. The performance results of these learners are relatively high considering other search and ranking tasks (e.g., as in [16,50,51]), and our statistical analysis of these learners using the considered

testing period indicates that the four models are comparable. On the other hand, our analysis shows that two of the learners (RankBoost and Linear regression) performed poorly and resulted in the lowest effectiveness among all learners. This makes those learners less suitable for this task.

Besides our experiments for this part, we conducted further experimentation to examine whether combining the ranked lists produced by the different LtR models can lead to effectiveness that is higher than having a single model selecting a set of stock symbols. Table 6 presents the results of these experiments using the two rank fusion methods described in Section 3.4. As Table 6 shows, neither rank fusion method outperformed the top LtR models adopted for this task. ISR seems to be comparable with the top four LtR models described previously (statistical analysis confirms this observation). In contrast, one can see that the RRF fusion method performed poorly compared to a single LtR model. Our further analysis in the following section will provide more insights into the usefulness of these fusion methods.

Table 6. The ranking performance results for applying two rank fusion methods, ISR and RRF, to combine the ranked lists of the nine LtR models for four years, 2018, 2019, 2020, and 2021.

2018				
Model	**P@10**	**P@20**	**nDCG@10**	**nDCG@20**
ISR	0.7000	0.5500	0.3884	0.2883
RRF	0.5000	0.3500	0.0631	0.0500
2019				
Model	**P@10**	**P@20**	**nDCG@10**	**nDCG@20**
ISR	0.6000	0.7000	0.2559	0.3259
RRF	0.3000	0.5000	0.1849	0.2034
2020				
Model	**P@10**	**P@20**	**nDCG@10**	**nDCG@20**
ISR	0.8000	0.8500	0.7158	0.6802
RRF	0.9000	0.8500	0.6386	0.5641
2021				
Model	**P@10**	**P@20**	**nDCG@10**	**nDCG@20**
ISR	0.8000	0.6500	0.4268	0.3535
RRF	0.7000	0.7000	0.2021	0.1862
Mean of four years				
Model	**P@10**	**P@20**	**nDCG@10**	**nDCG@20**
ISR	0.7250	0.6875	0.4467	0.4120
RRF	0.6000	0.6000	0.2722	0.2509

4.3. The Usefulness of LtR Models for Investment Portfolios

We evaluated the usefulness of adopting LtR models to select stock symbols for investment portfolios. We did this by constructing several simulated investment portfolios and emulating the process of investing in our target stock market. We considered diversifying these portfolios by examining two scenarios: investing in the top 10 stock symbols selected by each model and investing in the top 20 selected stocks. For each scenario, we divided our investment capital of 100 K SAR equally among the selected stock symbols. The simulation was applied for the four years (2018, 2019, 2020, and 2021) in our testing such that the investment period is set to a single year (i.e., a set of stocks will be selected by a learner, and shares will be purchased at the start of a year and then sold by the end of that year). The performance of each learning model will be determined by the total returns (profits/losses) on its corresponding portfolio. Tables 7 and 8 show our results considering the two scenarios: investing in the top 10 selected stocks and investing

in the top 20. Both tables also compare the results to the returns of the Saudi Stock Market's main index, TASI. Moreover, Table 9 compares the results of our top portfolios to the returns produced by the best-performing hedge funds investing in the Saudi Stock Exchange [37] for the same testing period.

Table 7. The returns produced by each model's simulated portfolio (top 10 selected stocks) for the four years, 2018, 2019, 2020, and 2021. Underlined values represent the models with the highest earnings. Superscript numerals in parentheses represent the rank of a model among all models based on total and average returns.

Model	2018 (%)	2019 (%)	2020 (%)	2021 (%)	Total (%)	Average (%)
Linear regression	−20.65	8.67	57.73	−1.60	44.14 (8)	11.03 (8)
MART	1.22	23.63	116.91	38.43	180.19 (5)	45.05 (5)
LambdaMART	30.55	32.92	81.76	45.37	190.60 (4)	47.65 (4)
LambdaRank	28.33	18.19	157.70	43.71	247.93 (2)	61.98 (2)
Coordinate ascent	4.80	22.90	62.09	18.60	108.39 (6)	27.10 (6)
RankBoost	8.91	7.77	−6.54	20.43	30.75 (9)	7.65 (9)
Random forests	20.88	54.04	146.11	40.87	261.89 (1)	65.47 (1)
RankNet	16.34	30.03	32.13	29.73	108.23 (7)	27.06 (7)
ListNet	−8.15	9.59	158.61	46.70	206.74 (3)	51.69 (3)
ISR	30.01	24.87	117.74	33.06	205.68	51.33
RRF	−6.13	3.35	61.20	13.41	71.83	17.96
Market Index (TASI)	8.30	7.58	3.96	30.90	50.47	12.68

Table 8. The returns produced by each model's simulated portfolio (top 20 selected stocks) for the four years, 2018, 2019, 2020, and 2021. Underlined values represent the models with the highest earnings. Superscript numerals in parentheses represent the rank of a model among all models based on total and average returns.

Model	2018 (%)	2019 (%)	2020 (%)	2021 (%)	Total (%)	Average (%)
Linear regression	−17.10	4.99	70.52	6.11	64.52 (8)	16.13 (8)
MART	15.62	20.97	94.69	29.07	160.63 (2)	40.09 (2)
LambdaMART	16.18	25.59	74.49	35.75	152.01 (3)	38.01 (3)
LambdaRank	8.33	27.39	121.11	38.29	195.13 (1)	48.75 (1)
Coordinate ascent	2.99	15.47	55.37	12.54	86.37 (6)	21.59 (6)
RankBoost	12.62	5.62	−1.44	13.70	30.50 (9)	7.63 (9)
Random forests	7.10	38.41	78.73	23.35	147.49 (4)	36.87 (4)
RankNet	4.43	21.81	31.89	25.07	83.20 (7)	20.80 (7)
ListNet	−11.85	8.08	110.49	33.29	140.01 (5)	35.01 (5)
ISR	24.19	37.99	80.73	25.92	169.85	42.20
RRF	−7.81	6.94	47.62	10.41	57.16	14.29
Market Index (TASI)	8.30	7.58	3.96	30.90	50.47	12.68

Table 9. The performance of the best performing hedge funds managed by investment firms in Saudi Arabia. Returns are compared with the top two performing simulated LtR portfolios for the period from January 2018 to December 2021. Underlined values represent the portfolio with the highest earnings.

Portfolio	2018 (%)	2019 (%)	2020 (%)	2021 (%)	Total (%)
Alrajhi capital	11.44	10.46	19.41	<u>48.62</u>	89.93
Morgan Stanly-SA	17.8	15.25	9.09	45.21	87.35
Derayah capital	10.50	24.90	19.79	31.59	86.78
Alarabi national	10.51	14.34	15.03	36.06	75.94
Alriyadh capital	15.11	8.17	9.72	41.74	74.74
Aljazera capital	11.72	13.68	11.61	34.45	71.46
Albilad capital	8.81	18.5	10.7	29.94	67.95
Market Index (TASI)	8.30	7.58	3.96	30.90	50.47
LtR—Random forests	20.88	<u>54.04</u>	146.11	40.87	<u>261.89</u>
LtR—LambdaRank	<u>28.33</u>	18.19	<u>157.70</u>	43.71	247.93

Table 7 shows that the LtR models considered in this study can be categorized into two groups based on their returned earnings. On one hand, we see that five of our learners, namely Random forests, LambdaRank, ListNet, LambdaMART, and MART, resulted in high investment returns, having substantially outperformed the market index for almost all the years included in our testing (Table 10 shows a sample of the top stocks selected by the best two models reported for the last three years). The increase in performance (i.e., as a measure of returns) of these models is five times greater than the performance of the market index, TASI, or even more (e.g., in the case of Random forests). Additionally, comparing these models (in Table 9) to the best performing hedge funds investing in the Saudi stocks and managed by investment firms reveals the high potential of LtR models when considered for the investment task as it is shown that our top model, Random forests, resulted in returns that are three times higher than the best of these hedge funds.

Table 10. A sample of the ranked list of results showing the selected stock symbols (top 10) and their returns for two models, Random forests and LambdaRank. The table shows the results for the years 2019, 2020, and 2021.

Model	Ranked Lists and Returns					
	2019		2020		2021	
Random forests	Stock code	Returns (%)	Stock code	Returns (%)	Stock code	Returns (%)
	(1) 3008	169.26	(1) 4061	236.75	(1) 1832	183.73
	(2) 1832	233.50	(2) 2300	381.73	(2) 4141	14.38
	(3) 2110	−0.80	(3) 1213	100.17	(3) 2222	2.29
	(4) 4012	111.30	(4) 1832	174.66	(4) 8120	−18.92
	(5) 7030	41.92	(5) 2222	−0.71	(5) 2140	2.73
	(6) 8170	−14.78	(6) 4012	78.89	(6) 4051	24.83
	(7) 2310	−10.02	(7) 3008	21.69	(7) 6020	22.21
	(8) 8240	27.452	(8) 2060	0.01	(8) 4012	−17.85
	(9) 8230	−5.08	(9) 4191	178.94	(9) 8190	98.54
	(10) 2290	−12.38	(10) 7201	288.99	(10) 7200	96.73
LambdaRank	2019		2020		2021	
	Stock code	Returns (%)	Stock code	Returns (%)	Stock code	Returns (%)
	(1) 7030	41.92	(1) 1832	174.66	(1) 1832	183.73
	(2) 5110	33.56	(2) 8120	58.14	(2) 4280	27.04
	(3) 2310	−10.02	(3) 2300	381.73	(3) 2350	19.02
	(4) 4040	28.80	(4) 6060	84.24	(4) 1020	41.29

(5) 4007	−16.17	(5) 6020	73.79	(5) 4220	29.64
(6) 4290	6.62	(6) 8240	39.85	(6) 7030	−11.47
(7) 4031	13.59	(7) 7201	288.99	(7) 4310	36.03
(8) 4050	74.47	(8) 4061	236.75	(8) 4300	16.17
(9) 1010	21.09	(9) 6012	87.46	(9) 2060	45.91
(10) 4230	−11.95	(10) 8300	151.37	(10) 2380	49.78

On the other hand, we see that the remaining models resulted in lower investment returns compared to the first group such that most of these models did not outperform the market for all years. Particularly, we see that two models, Linear regression and RankBoost, performed very poorly compared to the market index, resulting in cases where their portfolios are taking losses (e.g., in the years 2018 and 2021 for Linear regression and 2020 for RankBoost). Additionally, compared to the top hedge funds investing in the target stock market, they did not outperform them (in fact, Linear regression resulted in very low returns compared to these hedge funds). It is worth mentioning that the (relatively) high returns of almost all models for the year 2020 are due to a significant market recovery after a market crash at the start of the year caused by the announcement of COVID-19 as a major health issue worldwide. Consequently, some small to medium-sized companies substantially recovered to share prices that are even higher than the prices for the period prior to this announcement.

Interestingly, as Table 7 also indicates, the two rank fusion methods, which combine the stocks selected by all models into single ranked lists, acted differently. More specifically, we see that ISR resulted in a total performance (i.e., total returns) that is comparable with those of our best three LtR models. Although ISR did not result in the highest returns per year, one can see that its performance is consistent across all four years. In contrast, we observe that RRF, which is reciprocal-based, resulted in performance comparable to that of the lowest three LtR models (consistent across four years) and only outperformed the TASI index for a single year, making such a combiner unsuitable for aggregating the results of several models.

It is worth noting that the results in Tables 7 and 8 correlate very well with our effectiveness results reported in Section 4.2. They show that the performance of a model's portfolio decreases noticeably as one moves down in the ranked list of stocks (i.e., considering 20 stocks rather than only 10). Although selecting more stock symbols can provide more diversification of an investment portfolio and minimize risk, it seems that having a high-ranking cutoff (e.g., 20 or more) will affect the portfolio returns negatively. This can be addressed with several reasons. One is the tendency of ranking models to move potentially more relevant items to the top of a ranked list while keeping what might be less relevant further down on the list (i.e., the stocks selected by a model when considering a deeper ranking cutoff will have more partially relevant and irrelevant stocks than when considering a shallow cutoff). Another reason is that we explicitly set the optimization parameter during the training stage of all learners to optimize for ranking cutoff 10, making these learners focus on enhancing the quality of the results that are at the top of the list.

It is also worth noting that our performance results, as a function of investment returns per model, show high correlations with the effectiveness results reported using nDCG in Table 4 and Table 6 of Section 4.2. The top models with the highest returns for our investment task are also among the top models evaluated by nDCG, as reported in the previous section. Likewise, we see that the lowest-performing models by one measure are also among the lowest by the other (e.g., RankBoost and Linear regression are the lowest by both nDCG and returns). We verified this observation by estimating Pearson's correlation coefficients among model returns and model effectiveness using both nDCG and precision. Table 11 shows the results.

It is clear from Table 11 that the nDCG metric, considering both ranking cutoffs, has achieved an almost perfect correlation with the investment returns made by the models'

portfolios, reaching 0.93 for both nDCG@10 and nDCG@20. This contrasts with the precision measure, which shows some degree of correlation with the models' returns; however, it is still low compared to nDCG. This result suggests that nDCG is, in fact, more suitable for indirectly inferring the models' performance values and estimating which model will produce higher returns than the other models. Therefore, using nDCG during the training stage of an LtR model (to optimize the learning process and evaluate loss function, as in our case) would be very effective in capturing the model's true performance (i.e., it is assumed to be as effective as training with the model's actual returns, except the latter may not be straightforward for our ranking task).

Table 11. Pearson's correlation coefficients (p) between models' returns and models' performances (using both nDCG and P) are estimated for the years 2018, 2019, 2020, and 2021.

Metric	p(2018 Returns)	p(2019 Returns)	p(2020 Returns)	p(2021 Returns)	Average
nDCG@10	0.92	0.86	0.96	0.96	0.93
nDCG@20	0.87	0.90	0.98	0.97	0.93
P@10	0.46	0.17	0.56	0.58	0.44
P@20	0.75	0.62	0.62	0.74	0.68

Finally, to recap our analysis for this part, we show a successful adaptation of several LtR models for selecting stock symbols for investment portfolios. Our results show that more than half of the considered models (including a rank fusion method, ISR) can produce relatively high investment returns and outperform the market for the specified period. From that, we can conclude that learning to rank, as a framework, can indeed be very useful (when using suitable learners) for providing investors and financial analysts with recommendations on which of the listed companies in the market to consider for an investment plan. Perhaps combing recommendations from several tools and sources would be even more useful than relying on a single tool or a model.

4.4. Further Analysis and Discussions

Having presented a study for implementing and adapting LtR models for stock selection in financial markets, we now provide further discussions and include some remarks about our work. One might note that, although our study indicates the potential usefulness of LtR models for the investment task, it is not clear what the impact of the considered features for training these models is (i.e., whether the included features are suitable for distinguishing between instances and learning to discriminate among different labels).

We addressed this question by measuring the importance of each feature as it is being used by itself for training an LtR model. Note that it is expected that the measured importance of features will vary from one model to another, as different models make different assumptions about these features. However, for simplicity and to reduce the dimensionality of our problem, we consider a single model, LambdaMART, one of the top LtR learners as shown in Section 4.2. We also use nDCG@10 as a measure of the feature's importance. Table 12 summarizes our results, showing the feature's importance as an average of nDCG@10 value (the feature "symbol code" is added to every single feature to distinguish between data instances).

Table 12. Feature importance is estimated by training a model for each feature and reporting average nDCG@10 values.

Feature	Importance (nDCG@10)
Capital growth percentage	0.1713
P/E ratio	0.1734
Paid-in capital	0.1978
Market value growth (1 year)	0.1996
Market value	0.2153
Sector	0.2630
Market value to capital percentage	0.2733
Stock price	0.2929
Total net profit/loss	0.3082
Profit/loss per share	0.3293
Capital growth percentage	0.3318
P/E indicator	0.3332
Capital growth frequency	0.3670
Market value growth (3 years)	0.4266
All features	0.5088

Table 12 indicates that the chosen set of features can indeed be used for our task, as they provide good discrimination among the different labels and predict the most relevant stock symbols for a given year. Those features, however, vary in their impact and importance, as we see that a statistic such as "the growth in the market value of a company within the last three years" is more significant than "a company's capital or its market value" (as indicated above, these observations can only be generalized for LambdaMART, but not all LtR learners). Combining those features in a single model is expected to result in high prediction effectiveness, as shown in Table 12.

Finally, we conclude our discussion by drawing the reader's attention to a few issues related to our work. First, in this work, we showed a case study for applying a set of machine learning and computational intelligence methods for the task of passive investment portfolio management. Nevertheless, our work does not aim to advocate for adopting passive management over active management (or vice versa). This is beyond the scope of our analysis, as our thought is that both have some benefits and some drawbacks (e.g., passive management comes at lower computational and managerial costs; however, it may lead to a major drawdown of an investment portfolio). Second, it should be noted the proposed framework in this paper is aimed at providing financial analysts and investors with a set of tools for assisting them in decision-making when considering the task of stock selection. It is not aimed at advocating the full automation of the investment task or replacing domain experts with machines. We believe that, due to the high risks associated with investing in financial markets, such tasks require human experts' supervision and intervention (if needed).

Lastly, although our framework has been shown to lead to high effectiveness and to have the potential to produce high investment returns, one might argue that our empirical study is limited in that it considered a period in which the markets were growing and trending upward. This is a viable concern and a limiting factor of this study, as indeed the period considered does not exemplify a recession period for financial markets. Moreover, there might be a potential bias in the used data as historical data for financial markets are generally known to be biased by their nature (which could be addressed due to a variety of macroeconomic-related factors). However, we argue that our analysis shows that, for some years in our testing period, the overall growth of the market is relatively low and is not comparable with the returns produced by our top-performing learners. This may suggest that these models are important even during recession periods, as they are

expected to detect stock symbols with high potential returns from a large pool of underperforming symbols.

5. Conclusions

This paper examines the application of rank-based approaches imported from the search and retrieval domain to facilitate the tasks of performing long-term investments in stock markets. Our work introduced a new dataset along with a set of features for exploring these methods for stock selection and ranking stock market companies according to their relevance to certain investment criteria. Moreover, we examined a variety of ranking algorithms and showed the feasibility and high potential of learning to rank (LtR) models for addressing the shortcomings of manual analysis of market data and selecting stocks for investment. Future research exploration will expand this work by considering the application of a variety of learning-based approaches (including LtR methods) for identifying and selecting stocks that are suitable for active daily trading (i.e., short-term investments). We will also examine the usefulness of other types of features related to stock price movements that can capture stock price trends and volatility, assisting in the process of selecting stocks for this task.

Funding: This research was funded by the Researchers Supporting Project (No. RSP2022R449), King Saud University, Riyadh, Saudi Arabia.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Acknowledgments: The author would like to thank the Researchers Supporting Project (No. RSP2022R449), King Saud University, Riyadh, Saudi Arabia, for supporting this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bahadur, S.; Neupane, S. Stock market and economic development: A causality test. *J. Nepal. Bus. Stud.* **2006**, *3*, 647–656.
2. Masoud, N.M. The impact of stock market performance upon economic growth. *Int. J. Econ. Financ. Issues* **2013**, *3*, 546–565.
3. Lykkesfeldt, P.; Kjaergaard, L.L. The Benefits and Drawbacks of a Stock Market Listing. *Invest. Relat. ESG Report. A Regul. Perspect.* **2022**, *1*, 3–9.
4. Chiang, W.C.; Enke, D.; Wu, T. An adaptive stock index trading decision support system. *Expert Syst. Appl.* **2016**, *59*, 195–207.
5. Alsubaie, Y.; Hindi, K.E.; Alsalman, H. Cost-sensitive prediction of stock price direction: Selection of technical indicators. *IEEE Access* **2019**, *7*, 146876–146892.
6. Alsulmi, M.; Al-Shahrani, N. Machine Learning-Based Decision-Making for Stock Trading: Case Study for Automated Trading in Saudi Stock Exchange. *Sci. Program.* **2022**, *2022*, 6542862.
7. Yan, R.J.; Ling, C.X. Machine learning for stock selection. In Proceedings of the ACM SIGKDD, San Jose, CA, USA, 12 August 2007.
8. Liu, T.Y. Learning to rank for information retrieval. *Found. Trends Inf. Retr.* **2009**, *3*, 225–331.
9. Sun, J.T.; Zeng, H.J.; Liu, H.; Huan, L.; Lu, Y.; Chen, Z. CubeSVD: A Novel Approach to Personalized Web Search. In Proceedings of the ACM WWW, Chiba, Japan, 10 May 2005.
10. Mei, T.; Rui, Y.; Li, S.; Tian, Q. Multimedia search reranking: A literature survey. *ACM Comput. Surv.* **2014**, *46*, 1–38.
11. Hanani, U.; Shapira, B.; Shoval, P. Information filtering: Overview of issues, research and systems. *User Model. User-Adapt. Interact.* **2001**, *3*, 203–259.
12. Alsulmi, M.; Alshamrani, R. Framework for tasks suggestion on web search based on unsupervised learning techniques. *J. King Saud Univ. CCIS* **2022**, *34*, 5525–5532.
13. Allam, A.; Haggag, M. The question answering systems: A survey. *J. Res. Rev. Inf. Sci.* **2012**, *2*, 1-12.
14. Soares, M.; Parreiras, F. A literature review on question answering techniques, paradigms and systems. *J. King Saud Univ. CCIS* **2020**, *32*, 635–646.
15. Roberts, K.; Simpson, M.; Demner-Fushman, D.; Voorhees, E.; Hersh, W. State-of-the-art in biomedical literature retrieval for clinical cases: A survey of the TREC 2014 CDS track. *Inf. Retr. J.* **2016**, *19*, 113–148.
16. Alsulmi, M.; Carterette, B. Improving medical search tasks using learning to rank. In Proceedings of the IEEE CIBCB, St. Louis, MO, USA, 30 May 2018.
17. Alsulmi, M. Exploring Information Retrieval Approaches for Clinical Decision Support and Biomedical Search Tasks. Ph.D. Thesis, University of Delaware, Newark, DE, USA, 2018. UDSpace. Available online: https://udspace.udel.edu/handle/19716/24008 (accessed on 1 September 2022).

18. Croft, W.B.; Metzler, D.; Strohman, T. *Search Engines: Information Retrieval in Practice*, 1st ed.; Pearson: London, UK, 2009; pp. 1–9.
19. Salton, G.; Wong, A.; Yang, C. A vector space model for automatic indexing. *Commun. ACM* **1975**, *18*, 613–620.
20. Song, F.; Croft, W.B. A general language model for information retrieval. In Proceedings of the ACM CIKM, Kansas City, MO, USA, 1 November 1999.
21. Robertson, S.; Zaragoza, H. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.* **2009**, *16*, 333–389.
22. Zengin, M.; Carterette, B. Learning user preferences for topically similar documents. In Proceedings of the ACM CIKM, Melbourne, Australia, 17 October 2015.
23. Bah, A.; Carterette, B. PDF: A Probabilistic Data Fusion Framework for Retrieval and Ranking. In Proceedings of the ACM ICTIR, Newark, DE, USA,12 September 2016.
24. Bah, A.; Carterette, B. Using "Model" Pseudo-Documents to Improve Searching- as-Learning and Search over Sessions, In Proceedings of the Searching as Learning Workshop IIiX, Regensburg, Germany, 26 August 2014.
25. Su, X.; Yan, X.; Tsai, C.L. Linear regression. *Wiley Interdiscip. Rev. Comput. Stat.* **2012**, *4*, 275–369.
26. Friedman, J.H. Greedy function approximation: A gradient boosting machine. *Ann. Stat.* **2001**, *29*, 1189–1232.
27. Wu, Q.; Burges, C.J.; Svore, K.M.; Gao, J. Adapting boosting for information retrieval measures. *Inf. Retr.* **2010**, *13*, 254–270.
28. Burges, C.J.; Ragno, R.; Le., Q.V. Learning to rank with nonsmooth cost functions. In Proceedings of *Neural Information Processing Systems*; Vancouver BC, Canada, 4 December 2006.
29. Metzler, D.; Croft, W.B. Linear feature-based models for information retrieval. *Inf. Retr.* **2007**, *10*, 257–274.
30. Freund, Y.; Iyer, R.; Schapire, R.; Singer, Y. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.* **2003**, *4*, 933–969.
31. Yu, H.; Chen, R.; Zhang, G. A SVM Stock Selection Model within PCA. In Proceedings of the ITQM, Moscow, Russia, 3 June 2014.
32. Yuan, X.; Yuan, J.; Jiang, T.; Ain, Q.U. Integrated Long-Term Stock Selection Models Based on Feature Selection and Machine Learning Algorithms for China Stock Market. *IEEE Access* **2020**, *8*, 22672–22685.
33. Song, Q.; Liu, A.; Yang, S.Y. Stock portfolio selection using learning-to-rank algorithms with news sentiment. *Neurocomputing* **2017**, *264*, 20–28.
34. Saha, S.; Gao, J.; Gerlach, R. Stock Ranking Prediction Using List-Wise Approach and Node Embedding Technique. *IEEE Access* **2021**, *9*, 88981–88961.
35. About Saudi Exchange. Available online: https://www.saudiexchange.sa/wps/portal/tadawul/about/company/about-tadawul?locale=en (accessed on 9 September 2022).
36. jsoup: Java HTML Parser. Available online: https://jsoup.org (accessed on 9 September 2022).
37. EReference Data: Saudi Stock Exchange Historical Data. Available online: https://www.saudiexchange.sa/wps/portal/tadawul/knowledge-center/about/ereference-data (accessed on 1 February 2022).
38. Easton, P.D. PE ratios, PEG ratios, and estimating the implied expected rate of return on equity capital. *Account. Rev.* **2004**, *79*, 73–95.
39. P/E Ratio—Price-to-Earnings Ratio Formula, Meaning, and Examples by Jason Fernando. Available online: https://www.investopedia.com/terms/p/price-earningsratio.asp (accessed on 9 September 2022).
40. The Lemur Project-Wiki-RankLib: Lemur Project. Available online: https://sourceforge.net/p/lemur/wiki/RankLib (accessed on 1 February 2022).
41. Pasumarthi, R.K.; Bruch, S.; Wang, X.; Li, C.; Bendersky, M.; Najork, M.; Pfeifer, J.; Golbandi, N.; Anil, R.; Wolf, S. TF-Ranking: Scalable TensorFlow Library for Learning-to-Rank. In Proceedings of the ACM SIGKDD, Anchorage, AK, USA, 4 August 2019.
42. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32.
43. Burges, C.J.; Shaked, T.; Renshaw, E.; Lazier, A.; Deeds, M.; Hamilton, N.; Hullender, G. Learning to rank using gradient descent. In Proceedings of the ICML, Bonn, Germany, 7 August 2005.
44. Cao, Z.; Qin, T.; Liu, T.Y.; Tsai, M.; Li, H. Learning to Rank: From Pairwise Approach to Listwise Approach. In Proceedings of the ICML, OR, USA, 20 June 2007.
45. Mourao, A.; Martins, F.; Magalhaes, J. Multimodal medical information retrieval with unsupervised rank fusion. *Comput. Med. Imaging Graph.* **2015**, *39*, 35–45.
46. Cormack, G.V.; Clarke, C.; Buettcher, S. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In Proceedings of the ACM SIGIR, WA, USA, 6 August 2006.
47. Jarvelin, K.; Kekalainen, J. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.* **2002**, *20*, 422–466.
48. Clough, P.; Sanderson, M. Evaluating the performance of information retrieval systems using test collections. *Inf. Res.* **2013**, *18*, *1368-1613*.
49. Manning, C.D.; Raghavan, P.; Schutze, H. *Introduction to Information Retrieval*, 1st ed.; Cambridge University Press: New York, NY, USA, 2008, 151–175.
50. Demeester, T.; Trieschnigg, D.; Nguyen, D.; Hiemstra, D. Overview of the TREC 2013 Federated Web Search Track. In Proceedings of the TREC workshop, Gaithersburg, MD, UAS, 19 November 2013.
51. Zhu, D.; Wu, S.T.; Masanz, J.J.; Carterette, B.; Liu, H. Using Discharge Summaries to Improve Information Retrieval in Clinical Domain. In Proceedings of the CLEF workshop, Valencia, Spain, 23 September 2013.