CALIFORNIA STATE UNIVERSITY
FULLERTON

ISDS 570
FINAL REPORT
MV PORTFOLIO OPTIMIZATION

Under the guidance of
Dr Pawel Kalczynski

Team T2

Eduardo Mejia
Keerthanaa Ellur
Annie Xu
Frank Meza Perales
Arundhathi Roy

TABLE OF CONTENTS

SUMMARY

This report provides a comprehensive overview of the process used to analyze stock market data, focusing on end-of-day (EOD) prices and index values. The analysis includes data completeness checks, data transformation, missing data imputation, return calculation, and portfolio optimization using Modern Portfolio Theory. Data was extracted from Yahoo Finance and cleaned using the ETL (Extract, Transform, Load) process, leveraging PostgreSQL for data management and R Studio for analysis. The main goal of this report is to minimize potential risks while maximizing potential returns.

The ETL process involved data preparation, data retrieval in R, data processing, return computation, and portfolio optimization. Data preparation included building a custom calendar and importing data for the SP500TR index and EOD quotes. Data gathered in R included indexes, EOD pricing, and custom calendar data, which were combined into a single dataset.

Portfolio optimization was performed using the PortfolioAnalytics package, with defined risk and return constraints. Test results were then subjected to the optimal weights, and annualized and cumulative returns were computed. Daily returns were calculated and checked for completeness.

Key observations include the high volatility and returns of certain stocks like POST and AGRO, and the stability of stocks like SRI and LSI. The predictive cumulative returns graphic, covering January to March 2021, showed the SP500TR index outperforming the optimal portfolio. With a higher annualized return (29.87%) compared to the portfolio (22.12%), the SP500TR index also demonstrated better risk-adjusted performance and greater volatility (standard deviation). Though its volatility is less, the portfolio underlined the trade-off between risk and return by demonstrating a noteworthy 7% underperformance against the SP500TR index. Risk-adjusted performance was evaluated using metrics including the Sharpe ratio; the SP500TR index showed a higher ratio than the portfolio, indicating improved investment quality.

BACKGROUND

ETL PROCESS

Extract Transform Load, or "ETL," is a process for aggregating data and incorporating information acquired from many sources, converting it into a usable format, and subsequently loading the data into a specified location. Since this technique enables data analysis from many platforms and sources integration, data management and analytics depend on it. Our ETL process takes advantage of both PostSQL and R Studio.

• **Extract:** Data is taken from several sources including files, database. In this first phase of the ETL process. This stage requires the specification of the data to be obtained together with the pertinent data sources to be located. Following extraction, the data is housed in a staging area or another makeshift storage space.

• **Transform:** This phase follows data extraction. This stage cleans, arranges, and formats the taken data so that it may be examined. Procedures including data cleansing, aggregation, enrichment, and mapping might be applied at this level. The revised data is kept in a temporary location for usage later on.

• **Load:** Loading, the last phase of the ETL process, moves the converted data into the target system. Data consistency and integrity are maintained in this phase by matching the converted data with the data model of the target system. Once imported into the target system, the data is then easily available for reporting, analysis, and decision-making.

ETL STEPS

1. Data Preparation

1.1 Database Creation and Data Import

Custom Calendar Table:

A custom calendar table has been added to the StockMarket database. This table, based on a CSV file (custom_calendar.csv), includes columns for:

- Dates
- Year
- Month
- Day
- Day of the week (DOW)
- Trading Day (indicating whether it is trading day or not)

Initially, the data covered dates from January 1, 2015, to December 31, 2020. To extend this data for portfolio backtesting, we added dates from January 1, 2021, to March 26, 2021. Custom formulas were used to populate the new data:

- Month: =MONTH()
- Year: =YEAR()
- Day: =DAY()
- Day of the Week: =TEXT(date,"ddd")
- Trading Day: A list of holidays was created, and the formula =NETWORKDAYS.INTL(Date,Date,,HolidayList) was used to determine trading days. This formula returns a value of 1 if the date is a trading day (i.e., not a weekend or holiday) and 0 otherwise.

Custom calender has two columns added using SQL: end-of-month markers (eom) and previous trading days (prev_trading_day).

- End-of-Month Markers (eom): An UPDATE SQL statement was used to identify the dates that mark the end of a month. For these dates, the eom column was set to 1.
- Previous Trading Days (prev_trading_day): Another SQL statement was used to determine the previous trading day for each date, and the prev_trading_day column was updated with these values.

End-of-Day Indices Table:

- Importing historical data for the SP500TR index from Yahoo Finance allowed us to generate the eod_indices table covering the period from December 31, 2015, to March 26, 2021.

End-of-Day Quotes Table:

- The eod_quotes table was populated using data from an eod.csv file.

1.2 Data Retrieval in R

Establishing Database Connection:

- Connected to the PostgreSQL database using the RPostgres and DBI packages.

Fetching Custom Calendar Data:

- Retrieved custom calendar data between December 31, 2015, and March 26, 2021, ordered by date.

Fetching EOD Prices and Indices Data:

- Retrieved end-of- day (EOD) quotations and indices data from December 31, 2015, to March 26, 2021, then combined the data into one dataset.

Disconnecting from the Database:

- Disconnected from the database and removed the connection object from memory.

2. Data Processing

2.1 Filtering Trading Days

Filtered out non-trading days from the custom calendar.

2.2 Completeness Check

Calculated the percentage of completeness for each symbol and selected those with data completeness of at least 99%.

2.3 Data Transformation

Transformed the data into a wide format with dates as rows and symbols as columns using the reshape2 package.

2.4 Merging with Calendar

Merged the transformed data with the custom calendar and set dates as row names.

2.5 Missing Data Imputation

Imputed missing data by carrying forward the last observation, with a maximum gap of 3 missing values allowed.

3. Return Calculation

3.1 Daily Returns

Calculated daily returns using the PerformanceAnalytics package and removed the first row (which contained NA).

3.2 Completeness Check for Daily Returns

Identified and selected symbols with maximum daily returns less than or equal to 1.00.

4. Portfolio Optimization

4.1 Data Preparation for Optimization

Set a random seed for reproducibility.

Selected tickers based on the assigned ticker list and converted data frames to xts objects for time series analysis.

There were 58 trading days between Jan 1 and Jan 26, hence split the data into training and testing sets and set aside the final 58 trading days for testing.

Table 1 : Selected Tickers

Name of the Team Member	Ticker	Name of the Company
Keerthanaa Ellur	LSI	Life Storage Inc. Common Stock
	CUZ	Cousins Properties Incorporated Common Stock
	AX	Axos Financial Inc. Common Stock
Eduardo Mejia	POST	Post Holdings Inc. Common Stock
	BKNG	Booking Holdings Inc. Common Stock
	VBTX	Veritex Holdings Inc. Common Stock
Frank Meza Perales	OFC	Corporate Office Properties Trust Common Stock
	CKX	CKX Lands Inc. Common Stock
	KT	KT Corporation Common Stock
Arundhathi Roy	BXP.PB	Boston Properties Inc. Common Stock
	AGRO	Adecoagro S.A. Common Shares
	SRI	Stoneridge Inc. Common Stock
Annie Xu	TU	Telus Corporation Ordinary Shares
	FISI	Financial Institutions Inc. Common Stock
	SFM	Sprouts Farmers Market Inc. Common Stock

4.2 Markowitz Portfolio Optimization

Specified and optimized the portfolio using the PortfolioAnalytics package, adding risk and return constraints, and using the ROI optimization method.

Retrieved the weights of the optimized portfolio and calculated the sum of these weights based on Project Range 1 (2016-2020).

4.3 Portfolio Testing

Applied the optimized weights to the test returns, calculated annualized returns, and plotted cumulative returns for the optimized portfolio and SP500TR index for all available 2021 data (Project Range 2).

5. Performance Evaluation

5.1 Cumulative Returns

Plotted cumulative return chart for Project Range #1 (2016-2020) for stock tickers assigned to our team.

5.2 Annualized Returns

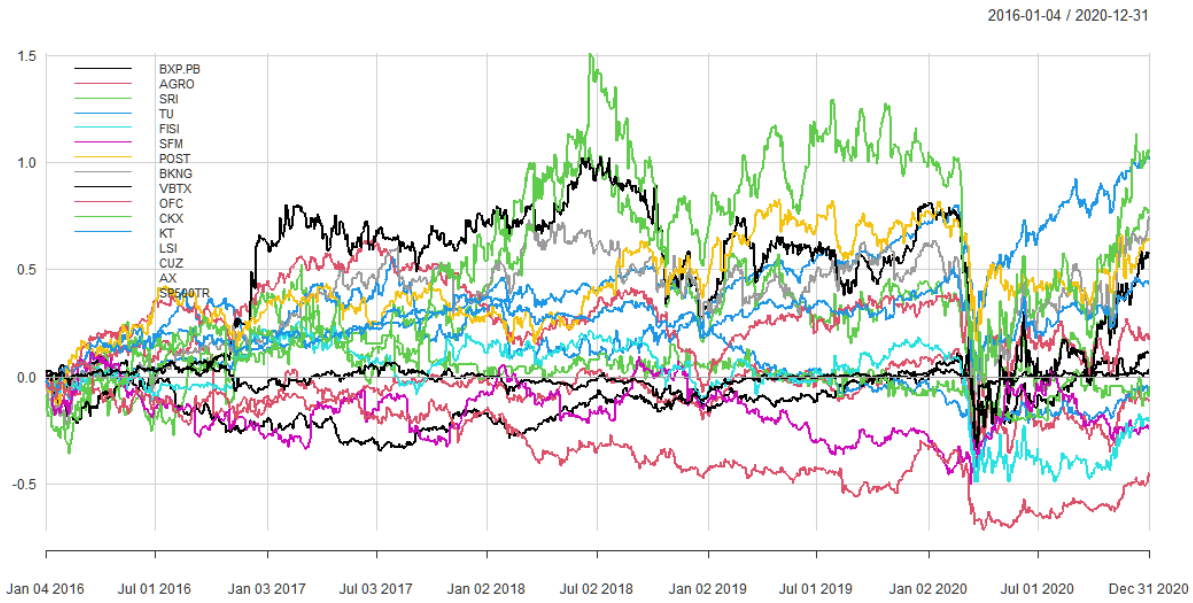
Calculated annualized returns for the portfolio and the SP500TR index for Project Range (2021).

RESULTS

Cumulative Returns Of Selected Tickers

The below chart showcases the cumulative returns of 15 selected stock tickers. The horizontal axis represents the timeline, while the vertical axis measures the cumulative returns in percentage. This graphic depiction helps one to examine the performance variability, stability, and related hazards of any ticker, therefore offering important information for strategic investment planning.

.Chart 1: Cumulative Returns Of Selected Tickers



Observations:

- **High Volatility**: High volatility stocks such as POST and AGRO show a greater risk even with their high returns. This is determined by stocks with wide fluctuations in their price over the given period.

- **Low Returns**: Stocks like KT, OFC, and AX exhibit minimal growth, suggesting underperformance compared to others. This is determined by stocks that have relatively flat lines or lines that do not show significant upward movement over time.
- **Stable Investment**: Less volatile stocks like SRI and LSI offer reasonable returns. This is found in equities with rather smooth lines displaying moderate but persistent upward curves.

Weights of optimized portfolio

Weights in a portfolio represent the proportion of the total value invested in each asset or security. This allocation is vital in shaping the portfolio's overall performance and highlights the impact each asset has on the portfolio's outcomes. Essentially, the greater the weight of an asset, the more significant its influence on the portfolio's returns, whether positive or negative. In our portfolio, TU holds the highest weight.

Table 2: Optimal Weights of the selected stocks

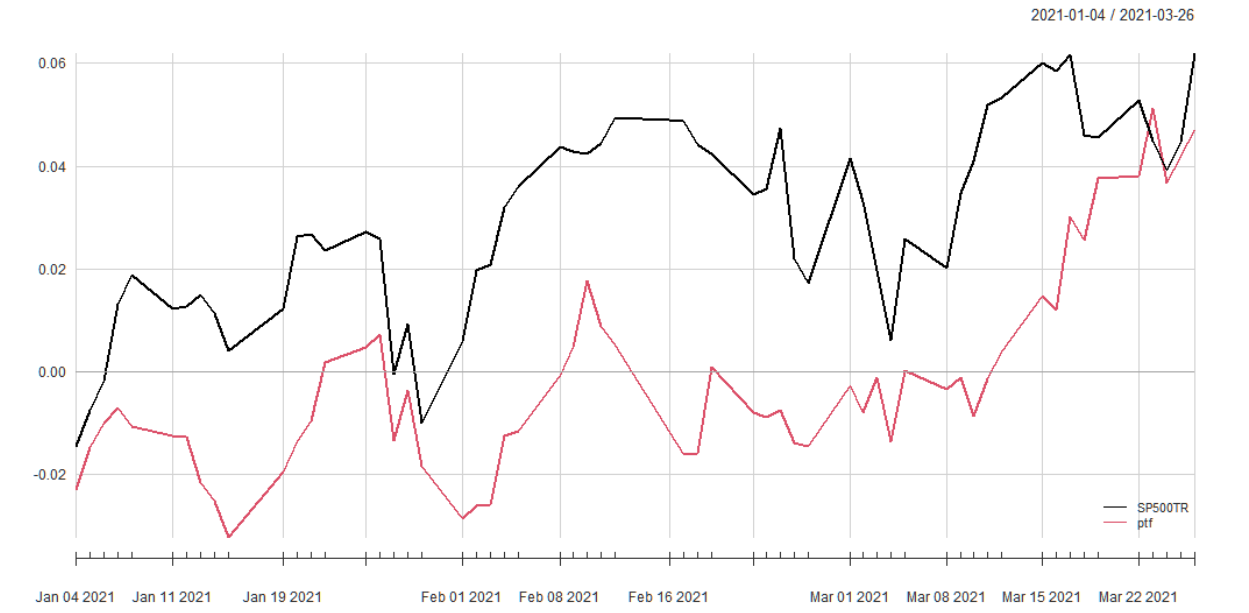
Stock	Weights
BXP.PB	0.2760
AGRO	-0.1226
SRI	0.1134
TU	0.3128
FISI	-0.2564
SFM	0.0225
POST	0.2194
BKNG	0.1397
VBTX	0.1227
OFC	0.0174
CKX	0.1247
KT	0.0216
LSI	0.0790
CUZ	-0.2111
AX	0.1410

Sum of all the weights	*1.0001*

Predictive Cumulative Returns

The below chart showcases the predicted cumulative returns for the optimized portfolio of the 15 selected stock tickers and the SP500TR index from January to March 2021. The horizontal axis represents the timeline and the vertical axis represents the measure of predicted cumulative return percentage. The black line is the SP500TR index while the red line represents the portfolio. Based on the chart, the SP500TR index shows a higher predicted cumulative return than the portfolio.

Chart 2: Predictive Cumulative Returns



Annualized Returns for SP500TR and Portfolio

The table below shows an annualized return of 29.87% for the SP500TR index and a 22.12% for the portfolio. In comparison to the SP500TR, the portfolio has a lower annualized return (about 7% lower). The annualized standard deviation and the annualized Sharpe for SP500TR index of 16.23% and 1.8399 are both higher than that of the portfolio's 14.92% and 1.4820. The standard deviation is a measure of how spread out the data is, and a higher standard deviation means that the data points are more spread out around the mean. Since the standard deviation is a measure of

risk, it means that the SP500TR has a higher risk/volatility than the portfolio. The Sharpe Ratio, on the other hand, evaluates the performance of an investment by comparing its returns to its risk, calculated by dividing the portfolio's excess returns by its volatility (standard deviation). A higher Sharpe Ratio indicates a better risk-adjusted performance, and since both SP500TR and the portfolio's Sharpe ratio is higher than 1, both are good investments.

Table 3 : Annualized Returns

	SP500TR	ptf
Annualized Return	0.2987	0.2212
Annualized Std Dev	0.1623	0.1492
Annualized Sharpe(Rf = 0%)	1.8399	1.4820

Comparative Analysis of Portfolio Performance Against the SP500TR Index

The performance comparison of the portfolio with the SP500TR index provides important insights. Although the portfolio delivered an annualized return of 22.12%, it falls short of the SP500TR index, which achieved a higher annualized return of 29.87%. This 7% difference indicates significant underperformance in comparison to the index. When analyzing volatility, the SP500TR index has a higher annualized standard deviation of 16.23% compared to the portfolio's 14.92%, suggesting that the index is more volatile. Despite this higher volatility, the index demonstrates superior performance, as evidenced by its Sharpe ratio of 1.8399, which exceeds the portfolio's ratio of 1.4820. A higher Sharpe ratio signifies better risk-adjusted returns, making the index a more attractive investment despite its increased volatility.

Furthermore, the composition of the portfolio, with heavy weighting towards specific stocks like BXP.PB, influences its overall performance and risk profile. Stocks such as POST and AGRO exhibit high volatility, contributing to increased risk, while stocks like SRI and LSI provide stability at the cost of moderate returns. This strategic allocation emphasizes the need to achieve a more effective balance between risk and return to enhance performance relative to benchmark indices like the SP500TR.

CONCLUSION

An in-depth analysis of stock market data was conducted, focusing on end-of-day prices and index values to ensure data integrity and optimize portfolio performance. The data was sourced from an SQL server, and a comprehensive ETL (Extract, Transform, Load) procedure was performed, followed by an analysis in R Studio. The report demonstrates how to achieve a balanced investment plan through portfolio optimization and data analysis. Ultimately, the optimized portfolio produced lower returns than the benchmark index, highlighting the trade-offs between risk and return in investment strategies. Key metrics such as the Sharpe ratio were used to evaluate risk-adjusted performance, revealing that while the benchmark index had higher returns, it also exhibited greater volatility. This analysis underscores the importance of balancing risk and returns to achieve optimal portfolio performance.

APPENDICES

Appendix A: SQL Code for Creating and Updating Tables

A.1 Custom Calendar Table

```
CREATE TABLE public.custom_calendar
(
    date date NOT NULL,
    y integer,
    m integer,
    d integer,
    dow character varying(3) COLLATE pg_catalog."default",
    trading smallint,
    CONSTRAINT custom_calendar_pkey PRIMARY KEY (date)
)
WITH (
    OIDS = FALSE
)

TABLESPACE pg_default;
ALTER TABLE public.custom_calendar
    OWNER to postgres;
ALTER TABLE public.custom_calendar
    ADD COLUMN eom smallint;
ALTER TABLE public.custom_calendar
    ADD COLUMN prev_trading_day date;


-- Update the table to add eom data
UPDATE custom_calendar
SET eom = EOMI.endofm
FROM (SELECT CC.date, CASE WHEN EOM.y IS NULL THEN 0 ELSE 1 END endofm
FROM custom_calendar CC LEFT JOIN
(SELECT y, m, MAX(d) lastd FROM custom_calendar WHERE trading=1 GROUP by y, m)
EOM
ON CC.y=EOM.y AND CC.m=EOM.m AND CC.d=EOM.lastd) EOMI
WHERE custom_calendar.date = EOMI.date;
```

```
-- Update the table to add prev_trading_day data
UPDATE custom_calendar
SET prev_trading_day = PTD.ptd
FROM (SELECT date, (SELECT MAX(CC.date) FROM custom_calendar CC WHERE
CC.trading=1 AND CC.date<custom_calendar.date) ptd FROM custom_calendar) PTD
WHERE custom_calendar.date = PTD.date;
```

A.2 EOD Indices Table

```
CREATE TABLE public.eod_indices
(
    symbol character varying(16) COLLATE pg_catalog."default" NOT NULL,
    date date NOT NULL,
    open real,
    high real,
    low real,
    close real,
    adj_close real,
    volume double precision,
    CONSTRAINT eod_indices_pkey PRIMARY KEY (symbol, date)
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;
ALTER TABLE public.eod_indices
    OWNER to postgres;

INSERT INTO custom_calendar VALUES('2015-12-31',2015,12,31,'Thu',1,1,NULL);
```

Appendix B: R Code for Data Retrieval

```
require(RPostgres)
require(DBI)
conn <- dbConnect(RPostgres::Postgres(),
                  user="stockmarketreader",
```

```
                password="read123",
                host="localhost",
                port=5432,
                dbname="stockmarket_HW2")
qry <- "SELECT * FROM custom_calendar WHERE date BETWEEN '2015-12-31' AND
'2021-03-26' ORDER by date"
ccal <- dbGetQuery(conn, qry)
qry1 <- "SELECT symbol, date, adj_close FROM eod_indices WHERE date BETWEEN
'2015-12-31' AND '2021-03-26'"
qry2 <- "SELECT ticker, date, adj_close FROM eod_quotes WHERE date BETWEEN
'2015-12-31' AND '2021-03-26'"
eod <- dbGetQuery(conn, paste(qry1, 'UNION', qry2))
dbDisconnect(conn)
rm(conn)
```

Appendix C: R Code for Data Preparation

```
tdays <- ccal[which(ccal$trading == 1), , drop = FALSE]
pct <- table(eod$symbol) / (nrow(tdays) - 1)
selected_symbols_daily <- names(pct)[which(pct >= 0.99)]
eod_complete <- eod[which(eod$symbol %in% selected_symbols_daily), , drop = FALSE]
require(reshape2)
eod_pvt <- dcast(eod_complete, date ~ symbol, value.var = 'adj_close', fun.aggregate = mean,
fill = NULL)
eod_pvt_complete <- merge.data.frame(x = tdays[, 'date', drop = FALSE], y = eod_pvt, by =
'date', all.x = TRUE)
rownames(eod_pvt_complete) <- eod_pvt_complete$date
eod_pvt_complete$date <- NULL
require(zoo)
eod_pvt_complete <- na.locf(eod_pvt_complete, na.rm = FALSE, fromLast = FALSE, maxgap =
3)
```

Appendix D: R Code for Return Calculation

```
require(PerformanceAnalytics)
eod_ret <- CalculateReturns(eod_pvt_complete)
eod_ret <- tail(eod_ret, -1)
```

```
colMax <- function(data) sapply(data, max, na.rm = TRUE)
max_daily_ret <- colMax(eod_ret)
selected_symbols_daily <- names(max_daily_ret)[which(max_daily_ret <= 1.00)]
eod_ret <- eod_ret[, which(colnames(eod_ret) %in% selected_symbols_daily), drop = FALSE]
```

Appendix E: R Code for Portfolio Optimization

```
set.seed(100)
tickers <- c('BXP.PB', 'AGRO', 'SRI', 'TU', 'FISI', 'SFM', 'POST', 'BKNG', 'VBTX', 'OFC', 'CKX',
'KT', 'LSI', 'CUZ', 'AX')
Ra <- as.xts(eod_ret[, tickers, drop = FALSE])
Rb <- as.xts(eod_ret[, 'SP500TR', drop = FALSE])
Ra_training <- head(Ra, -58)
Rb_training <- head(Rb, -58)
Ra_testing <- tail(Ra, 58)
Rb_testing <- tail(Rb, 58)
chart.CumReturns(cbind(Ra_training, Rb_training), legend.loc = 'topleft')
table.AnnualizedReturns(Rb_training)
mar <- mean(Rb_training)
require(PortfolioAnalytics)
require(ROI)
require(ROI.plugin.quadprog)
pspec <- portfolio.spec(assets = colnames(Ra_training))
pspec <- add.objective(portfolio = pspec, type = "risk", name = 'StdDev')
pspec <- add.constraint(portfolio = pspec, type = "full_investment")
pspec <- add.constraint(portfolio = pspec, type = "return", return_target = mar)
opt_p <- optimize.portfolio(R = Ra_training, portfolio = pspec, optimize_method = 'ROI')
opt_w <- opt_p$weights
Rp <- Rb_testing
Rp$ptf <- Ra_testing %*% opt_w
head(Rp)
tail(Rp)
table.AnnualizedReturns(Rp)
chart.CumReturns(Rp, legend.loc = 'bottomright')
```