RMSC 4002 – Financial Data Analytics with Machine Learning Group Project

CHOI Sen Hei (SID: 1155109412)
IEONG Hei (SID: 1155104271)
LAM Wai Chui (SID: 1155152095)
LAU Chiu Tan (SID: 1155108960)
LAW Yiu Leung Eric (SID: 1155149315)

December 13, 2021

Contents

1 Principal Component Analysis Factor Models

This section is contributed by CHOI Sen Hei (SID: 1155109412), IEONG Hei (SID: 1155104271) and LAU Chiu Tan (SID: 1155108960).

1.1 Introduction

Nowadays more and more people want to earn some money through investing in the stock market. There are various trading strategies which claim to generate profits but some of them fail if the market is performing worse. In this report, we aim at introducing a trading strategy that is able to outperform the market and reduce losses in economic downturns. The strategy is a combination of three components: stock selection by Principal Component Analysis (PCA), portfolio optimization, and a trading strategy called the Kelly's Formula.

In the project, we used historical stock data from 2000 to 2021 to test the performance of the above trading strategy. There were some financial crises that happened during the selected period such as the crisis in 2007-2008, and the crisis in 2020. The test is performed by R programming. Details of the procedures and results are included in the following sections in this report.

1.2 Method

1.2.1 Dataset

Our research is based on US Stock Market. For the dataset, we obtained the price data from each component in Russell3000 from Bloomberg terminal. The daily adjusted open price of the components from 2000/01/01 - 2021/11/12 are then collected using the BDH function in the Excel add-in. Among 3041 stocks collected, 833 are shortlisted which provide complete data among the 22 years time period.

1.2.2 Assumptions and Parameters

1. 10000 USD is used as the principal
2. Only Long is available, no short sales
3. Commission per trade : max{2.05, 1.3% × number of stock traded}
4. Window size of arithmetic return : 50-days
5. λ=0.94, estimated by J.P. Morgan Riskmetrics database(1)

1.2.3 Algorithm

Libraries used: `tseries, zoo, lubridate, fGarch`.

The overall procedure in a nutshell is as follow:

1. The adjusted open price data is splitted into 2 parts, the first years data (2000) are implemented as the historical price used for the estimated volatility in GARCH(1,1) model and EWMA model, also an initial portfolio with ten stocks is formed by applying PCA selection to the first five years prices. The next 21 years data are then implement with the investment strategy described below
2. A trading strategy through Kelly's Formula would be applied to maximize the trading profit regarding the principle of buy-low and sell-high, based on the threshold of buying and selling will be described in the later section of the report
3. When the sell signal is triggered, all stocks we are holding will be sold and PCA is implemented to find out the other ten best-performing stocks from Russell 3000 based on the one-year historical data from the trigger time point. Then, portfolio optimization would be performed to find out the best weighting of the stocks to compose the portfolio. A mean-variance efficient portfolio can be obtained as the solution of minimizing the portfolio variance under the constraint that the portfolio expected return equals a target return.

4. When the buying signal is triggered, all cash in hand will be spent to buy the portfolio based on the latest PCA selection and hold the portfolio until the next selling signal is triggered.

Algorithm 1 Trading Strategy Algorithm

```
Input: Stock price dataset $S_t, t = 1, 2, ..., T$.
Compute log-return $u_t = log(S_t) - log(S_{t-1})$;
Apply PCA on first 252 days log-return: $u_{1:252}$;
Select top 10 stocks with highest loadings;
Initialize: Window size = 50, N Obs = 252, Initial Cash = 10000;
for $t \leftarrow 252, T$ do
    GARCH(1,1):
    $\sigma_t^2 \leftarrow \gamma V_L + \alpha u_{t-1}^2 + \beta \sigma_{t-1}^2$;
    $\mu_t \leftarrow \frac{1}{50} \sum_{u_{t-50}}^{u_t}$;
    Goodness GARCH $\alpha_t \leftarrow \frac{\mu_t}{\sigma_t} - \frac{1}{2}$;
    EWMA:
    $\sigma_t^2 \leftarrow \lambda \sigma_{t-1}^2 + (1 - \lambda) u_{t-1}^2$;
    $\mu_t \leftarrow \frac{1}{50} \sum_{u_{t-50}}^{u_t}$;
    Goodness EWMA $\alpha_t \leftarrow \frac{\mu_t}{\sigma_t} - \frac{1}{2}$;
    if t ¡ 253 then
        Next;
    end if
    if No stocks and $\alpha_{t-1} < 0$ and $\alpha_t > 0$ then
        Buy;
    else if Have stocks and $\alpha_{t-1} > 0$ and $\alpha_t < 0$ then
        Sell;
        Update portfolio with PCA Algorithm;
    end if
end for
Return: Values of each model;
```

1.2.4 Pricipal Component Analysis (PCA)

Principal Component Analysis (PCA) is a multivariate statistical technique for dimension reduction. It is commonly used for image processing and variable selection(2). In this project, PCA is applied to stocks in the Russell3000 index as a stock selection method to build a portfolio of 10 stocks that closely represents the index. PCA decomposes the interrelated variables into uncorrelated principal components. The first PC is used for variable selection as it is the major market factor. It can represent the financial market performance. We select the 10 stocks with highest loadings in the first PC. The stock selection method works together with Kelly's formula, so we used a rolling window approach. For each time we sell our stocks at time t, we will apply PCA on the past t-253 to t-1 log-return to update our portfolio. Using PCA to extract the top 10 stocks can be obtained as follows:
Step 1 : Input stocks' price in the index, compute the log-return as u.
Step 2 : Calculate the covariance matrix of the first 252 days log-return u. Apply eigendecomposition on the covariance matrix, with eigenvalues σ and eigenvectors V.
Step 3 : Select the first PC which is the eigenvector with largest corresponding eigenvalue. Select the top 10 stocks with highest loadings in the first PC.
Step 4 : Repeat step 2 and step 3 with past 252 days log-return when sell signal is triggered.

The reason we use PCA for building our portfolio is that it is unwise to buy and hold a single stock. Building a portfolio with stocks in different sectors helps traders to diverse risk. Diversification is an important concept in investing. It prevents traders from losing all investment in a single asset. Variables selection by PCA will extract the uncorrelated components in the dataset. The reason for including securities that have low correlations or even negative correlations in our portfolio is that they will eliminate some risk which has become known as idiosyncratic or diversifiable risk. PCA for stock selection gives a good level of diversification and a reasonable performance. Also, the portfolio with PCA selected stock will replicate the index behavior except the financial crisis during pandemic. And the rolling windows approach to apply PCA on different periods of time updates our portfolio that gives out better performance in that time period.

The algorithm of updating portfolio at today using the past 252 days log-return is shown below:

Algorithm 2 PCA Algorithm

```
Input: Date T, N stocks log-return vectors in past 252 days $u_1, u_2, ..., u_N \in \mathbb{R}^{252}$ .
Compute mean vector;
$\bar{u} \leftarrow \frac{1}{N}\sum_{i=1}^{N} u_i$;
Subtract mean vector;
for $i \leftarrow 1, N$ do
    $\Phi_i \leftarrow u_i - \bar{u}$;
end for
$A \leftarrow [\Phi_1, \Phi_2, ..., \Phi_N]$;
Compute covariance matrix;
$C \leftarrow AA^T$;
Compute eigenvalues and eigenvectors of covariance matrix $C$ by SVD;
$C = V\Sigma V^{-1}$;
Eigenvectors:
$V = [v_1, v_2, ..., v_{252}]$;
Eigenvalues in descending order:
$\Sigma = diag(\sigma_1, \sigma_2, ..., \sigma_{252})$;
First PC: $v_1 \in \mathbb{R}^N$;
Sort $v_1$ with 10 highest loadings
Match stock price data of the 10 stocks with highest loadings;
Return: Top 10 stocks portfolio
```

1.2.5 Portfolio Optimization

A mean-variance efficient portfolio can be obtained as the solution of minimizing the portfolio variance under the constraint that the portfolio expected return equals a target return. A convenient R function for doing so is the function `portfolio.optim()` in the R package `tseries`. Its default implementation finds the mean-variance efficient portfolio weights under the constraint that the portfolio return equals the return on the equally-weighted portfolio. In this project, we have assumed that only long is possible, therefore the weighting will be all positive.

Algorithm 3 Mean-Variance Portfolio Optimization Algorithm

```
Input: Top 10 stocks log-return in past 252 days: $R = [r_1, r_2, .., r_{10}], r_i \in \mathbb{R}^{252}$
$m = [u_i]$, where $i = 1, 2, ..., 10$, $u_i = E[r_i]$;
$\mu = E[u_i]$;
Compute the covariance matrix S;
$S \leftarrow RR^T$;
Let weight of portfolio: $w = [w_1, w_2, ..., w_{10}]^T$;
Minimize $\frac{1}{2}w^T Sw$; Subject to $m^T w = \mu, \sum_{i=1}^{10} w_i = 1$;
Return: $w_i$, the weighting of each stocks in portfolio
```

1.2.6 Kelly's Formula

We all know that the best trading strategy is to "buy low, sell high", i.e. buy at the lowest price and sell at the highest price in the time frame. However, it is not practical in the real world. It would be better in practice to try to minimize the relative error between the selling price and the "true" highest price. That is the goal of

Kelly's Formula. Kelly's Formula assumed that the price process, $\{V_t\}_{(0\leq t\leq T)}$, of the underlying asset follows a Black-Scholes model:

$$(dV_t)/V_t = \mu dt + \sigma dW_t \tag{1.1}$$

where $V_0 = 1$, μ and σ are the rate of return and the volatility respectively that both are constant, and $\{W_t\}_{(0\leq t\leq T)}$ is the standard Brownian motion. After some calculation, Kelly's Formula suggests a goodness index as a trading signal:

$$\alpha_t = \frac{\mu_t}{\sigma_t^2} \tag{1.2}$$

where and are the rate of return and the volatility respectively at time t. According to this strategy, stocks should be bought with all cash on hand at time t and be held to time T, the end of the trading period, when $\alpha_{t-1} < 0.5$ and $\alpha_t > 0.5$ if we have no stock on hand. Otherwise, all stocks on hand should be sold at time t if $\alpha_{t-1} > 0.5$ and $\alpha_t < 0.5$ if we have some stocks on hand. Several methods are often being adopted for t and t estimation. Common estimations of tare the arithmetic, geometric, or log return of the latest few consecutive days. In our project, a 50 days time frame is adopted. Arithmetic return is calculated by $\frac{1}{50}\mu_s$. Geometric return is calculated by $\prod_{i=t-2}^{t}(1+\mu_i)^{(\frac{1}{50})}-1$. Log return is the log difference between price P_t and P_{t-1}, i.e. $log(P_t/P_{t-1})$. In this project, we estimated t using arithmetic calculations with log-return. Meanwhile, σ_{t} could be estimated by multiple methods. The volatility models adopted in this project are the Exponentially Weighted Moving Average (EWMA) and the GARCH(1,1). EWMA estimated ${\sigma_{\mathrm{t}}}^2$ with

$$\sigma_t^2 = \lambda\sigma_{t-1}^2 + (1-\lambda)u_{t-1}^2 \tag{1.3}$$

where λ is a parameter to be estimated. In the project, we adopted λ=0.94. GARCH (1,1) estimated ${\sigma_{\mathrm{t}}}^2$ with

$$\sigma_t^2 = \gamma V_L + \alpha u_{t-1}^2 + \beta\sigma_{t-1}^2 \tag{1.4}$$

where V_L is the long-run average variance rate and γ, α and β are parameters that summed to equal 1. The GARCH (1,1) model was performed by the garchFit function in the fGarch library in our R code.

1.3 Result

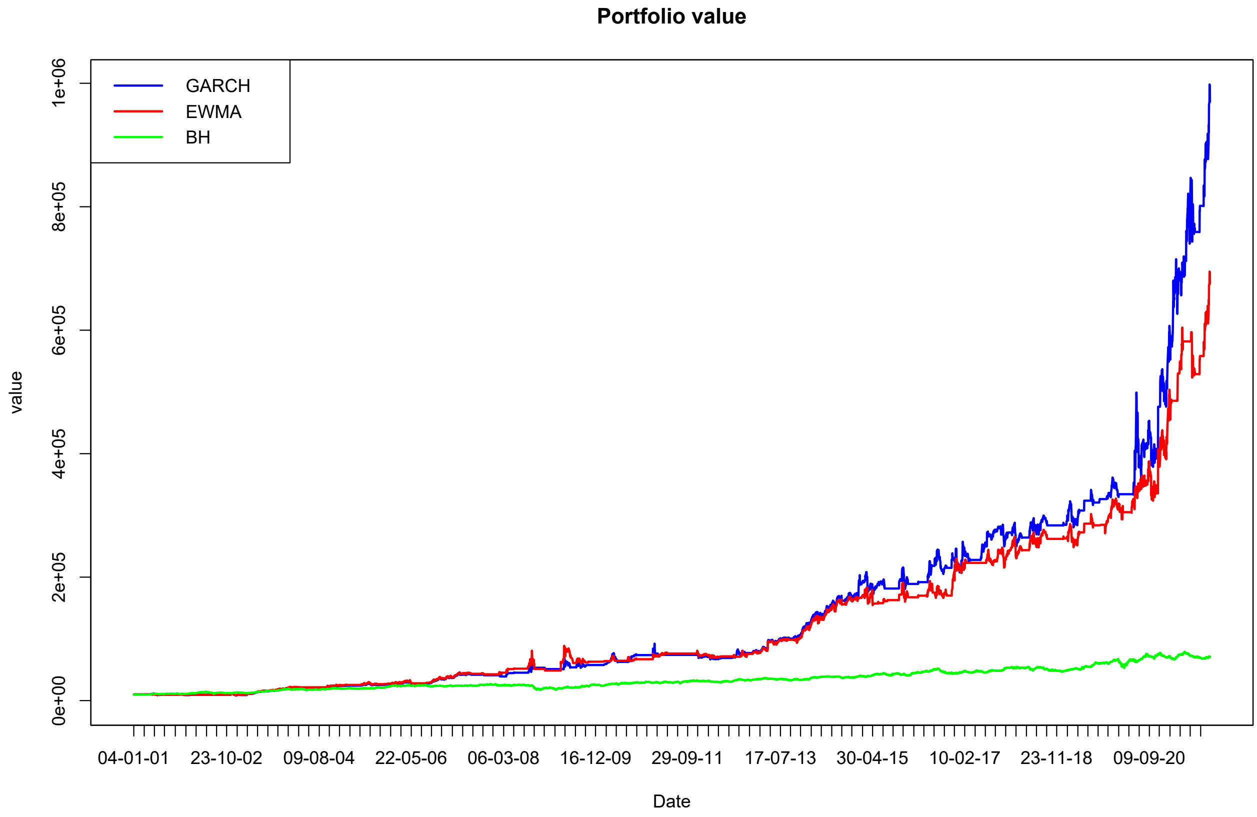


Figure 1: Portfolio value using different approach

Portfolio value at last trading date:
GARCH(1,1): 1486956 USD
EWMA: 690605 USD
Buy and Hold: 70415 USD
Figure 1 demonstrates the portfolio value after implementing our investment strategy and buy-and-hold our historical PCA selection in 2000. Blue line indicates our strategy using GARCH(1,1) as the volatility estimate, which outperforms the red lines using EWMA model and green lines(Buy-and Hold). Our strategy using GARCH(1,1) model has result in an around 150 times return from the principal 10000USD. While the EWMA model results in around 70 times of the principal. However Buy and Hold strategy just got 7 times of the principal.

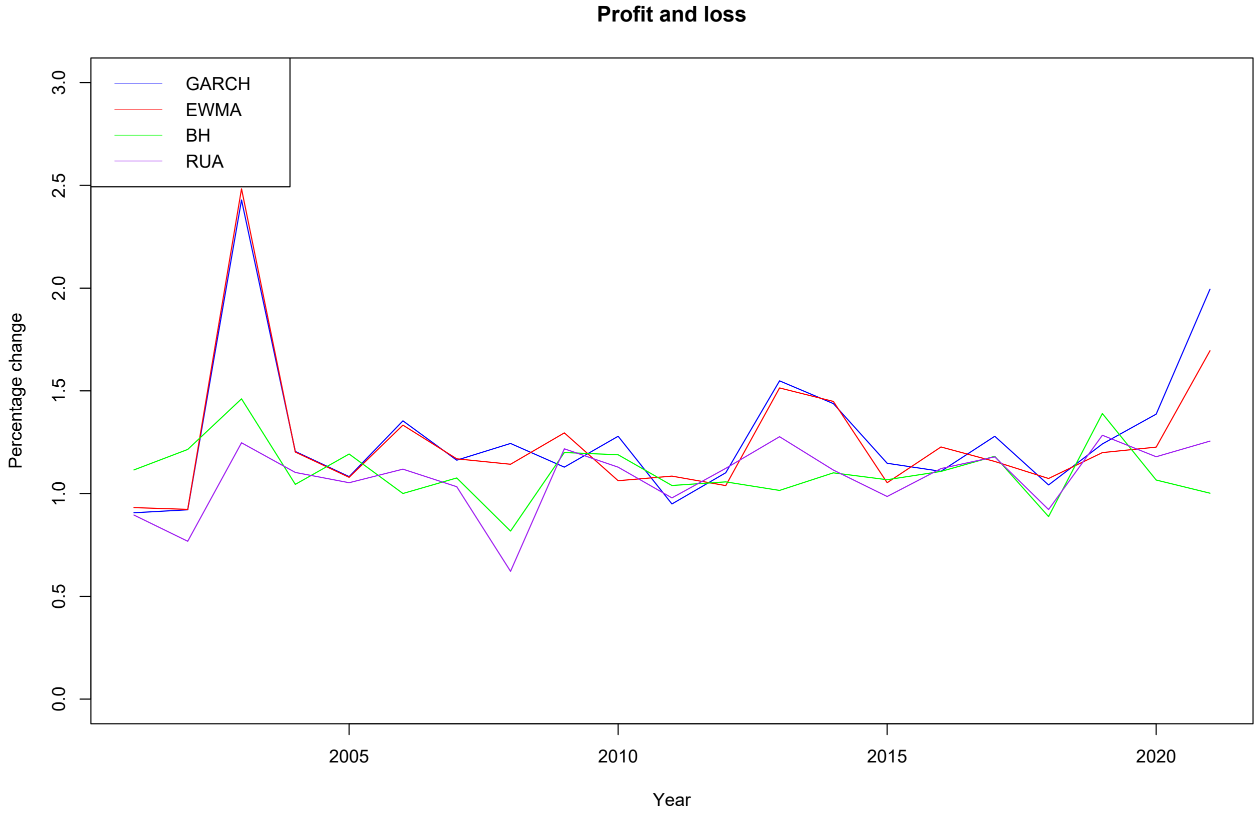


Figure 2: Year ended profit and loss using different approach

	Garch (%)	EWMA (%)	BH (%)	RUA (%)
2001	90.67	93.19	111.52	89.58
2002	92.13	92.33	121.49	76.81
2003	242.91	248.33	146.12	124.73
2004	120.51	120.22	104.5	110.3
2005	108.27	107.93	119.27	105.3
2006	135.41	133.33	100.06	111.9
2007	116.26	116.93	107.62	103.36
2008	124.41	114.28	81.76	62.19
2009	112.88	129.55	119.99	121.79
2010	127.9	106.28	118.9	112.86
2011	94.97	108.51	103.95	97.91
2012	110.15	103.9	105.71	112.29
2013	154.87	151.42	101.55	127.69
2014	143.8	144.86	110.1	111.46
2015	114.76	105.3	106.76	98.59
2016	110.84	122.69	110.8	112.18
2017	127.91	115.64	118.16	117.88
2018	104.21	107.35	88.83	92.25
2019	124.17	119.95	138.97	128.4
2020	138.67	122.62	106.61	117.94
2021	199.48	169.52	100.2	125.55

Table 1: Year ended percentage of each trading strategy

Figure 2 and Table 1 show the year ended percentage change for each approach. We can see that most of the year our proposed trading strategies are making profit. For GARCH(1,1), 18 out of 21 years are making profit. The losses have occurred in only 3 out of 21 years. The largest profit was +142.91% in 2003, and the largest loss was - 9.33% in 2001. For EWMA, 19 out of 21 years are making profit. The losses have occurred in only 2 out of 21 years. The largest profit was +148.33% in 2003, and the largest loss was -7.67% in 2002. From Figure 2, we can see that most of the years the performance of our proposed trading strategies are better than buy and hold strategy and Russell3000 index. Also, they can prevent losses from the global financial crisis in 2008. The Russell3000 index loss with -37.81% in 2008, buy and hold strategy loss with -12.24% in 2008. However, our proposed trading strategies were making profit in 2008.

GARCH(1,1)		
Stocks	Trading frequency	Sector
ATI	38	Basic Materials/Resources
CENX	36	Basic Materials/Resources
CPE	32	Energy Service
CLF	30	Basic Materials/Resources
EMKR	28	Technology
MTW	28	Industrial Goods
WRLD	28	Financial Services
YELL	28	Logistics
MTG	26	Financial Services
GERN	24	Health care

Table 2: Most frequently traded stocks using GARCH

EWMA		
Stocks	Trading frequency	Sector
CENX	52	Basic Materials/Resources
MTG	52	Financial Services
CLF	40	Basic Materials/Resources
SRPT	40	Health care
ATI	36	Basic Materials/Resources
MTW	36	Industrial Goods
SNBR	36	Consumer Goods
CPE	32	Energy Service
EMKR	32	Technology
YELL	32	Logistics

Table 3: Most frequently traded stocks using EWMA

	Buy triggered	Sell triggered
GARCH(1,1)	109	108
EWMA	131	130

Table 4: Trading frequency

1.4 Interpretation

1.4.1 Trading Frequency

Table 4 indicates that the number of trades in EWMA model is higher than GARCH(1,1) for about 30 trades, we can interpret that the goodness of using EWMA is easier to be triggered compared to GARCH(1,1). However when we are comparing the most frequent trade in Table 1 and 2, EWMA trades about two times more than GARCH(1,1) in their most traded stocks.

1.4.2 Diversification

Table 2 and 3 highlighted the sectors of the stocks in the correspondent portfolio, the sectors of these most frequently traded stocks are diversified over the portfolio in both GARCH(1,1) and EWMA model, with various types of sectors such as Basic Materials, Technology, Energy service, Financial service, etc. The PCA method extracts the independent components from the index. The good performing stocks and the uncorrelated variables are selected by PCA. The resulting portfolio can perform as principal components of the Russell3000 index, which can be used to represent the financial market and reduce the noise and underperforming stocks.

1.4.3 Major Market Events

There were two major financial crises included in our project, i.e. during 2008 - 2009 and during 2020. They were worth mentioning as we expected our strategy to outperform even during an economic downturn.

The 2008 to 2009 financial crisis was mainly due to the bursting of the housing bubble in U.S.. At that time, the default rate was high due to the drop in housing prices. Therefore, we assumed that the real estate market and bank stocks would perform worse and we believed that they would be automatically excluded by our trading strategy.

During that period, the return of our trading strategy was 24.41% and 14.28% respectively of GARCH and EWMA model. At that time, our strategy had traded 8 times with PCA selected portfolio. They were mainly health care, logistics stocks These types of stocks could perform well during the financial crisis because they are not correlated to the real estate market.

Simultaneously, the return of Russell3000 falls over 37% , much lower than our trading strategy.

On the other hand, the 2020 financial crisis was due to COVID-19. Because of the fear of pandemic, people were pessimistic about the economy and even some risk-free assets faced drop in prices. Therefore, we assumed that some stocks such as luxury goods, aviation, and tourism would perform worse and we believed that they would be automatically excluded by our trading strategy.

During 2020, the return of our trading strategy was 38% and 22.6% respectively. The virus affected the stock market mostly during late February to Early April, by our investment strategy, we sold all of our portfolio on 2020-02-04 and 2020-03-05 in GARCH and EWMA model while start buying on 2020-05-18 and 2020-05-04. Our strategy had dodged away the tremendous fall in the crisis and still had a positive return in the whole year.

Simultaneously, the return of buy-and-hold Russell3000 was only 18%, much lower than our trading strategy.

In short, the simulation proved that our trading strategy, which is a combination of PCA, portfolio optimization, and Kelly's Formula, performs better than the market, no matter whether the economy is in upturn or in downturn.

1.5 Conclusion

In this project, we introduce a stock selection method with PCA, a portfolio weighting method with Mean-Variance optimization method, and a trading strategy with Kelly's formula using GARCH model and EWMA method for forecasting. These three techniques combined together to form an optimal trading strategy to compare with buy and hold strategy and Russell3000 index. The testing period is between 2001-01-04 and 2021-11-12. The trading strategy we proposed can outperform the Russell3000 index and buy and hold strategy in the testing period.

1.5.1 Limitations

1. We proposed that we triggered the buy/sell signal by the threshold that is determined by the overall portfolio return and volatility, if one of the stocks in the portfolio performs badly, we might not able to notice that and change our portfolio

2. Due to the large data size, the total running time of the simulation was long. Meanwhile, since we run through all the trading days with GARCH(1,1) and EWMA, the for-loops in the program have further lengthened the running time.

3. In our project, we downloaded all required data first then did the simulation. However, in practice, data may need to be updated daily. It is daily additional work and time is needed to download all the components' price in Russell300 to start the analysis every day.

4. Liquidity, the simulation assumed that stock could be bought or sold immediately. But in the real world, there may be a time lag for the trading to be done, especially if the trading volume for that stock is not high. This may result in a reduction of return as price may have changed during the time lag. On the contrary, the buy-and-hold strategy would not have this problem because it only buys or sells once in the trading period.

1.5.2 Improvements

1. To shorten the running time in practice, we could choose only one among the GARCH(1,1) and EWMA volatility estimation methods. From the simulation, we could see that the portfolio return calculated by GARCH(1,1) and EWMA had very similar results from the start of the simulation period to the end of it.

2. The number of stocks in the pool could be smaller. In our simulation, we picked all components of Russell3000 into the pool for PCA. In practice, however, we may choose fewer stocks to put into the pool. This could shorten the time for data collection and extraction. It is suggested that the market value of stocks to be not too small, for example at least greater than 1 billion, and that the pool should contain various types of stocks so that the strategy would be able to cater different economic situations. It is important to note that since the pool is smaller, maybe we need to update the stocks in the pool regularly for more accurate results. There are trade-offs between accuracy and data set size.

3. To lower the chance that the stock is too hard to be traded due to a small trading volume in the market, we may avoid choosing less popular stocks.

4. Short Selling can be considered to be added to the strategy by selecting the least loadings in the PCA computed, also as our buy/sell strategy is based on volatility, short selling can also be possible in our strategy.

2 Classification / Decision and Regression Trees

This section is contributed by LAM Wai Chui (SID: 1155152095) and LAW Yiu Leung Eric (SID: 1155149315).

In this section, we are going to use Decision Tree and Random Forest methods to predict binary response variable. The source code is attached in `src/python/classification.ipynb`, notice that the output in this written report may be slightly different with the python notebook, due to random seed is involved.

2.1 Dataset

For the dataset, we use Bank Marketing Data Set from UCI Machine Learning Repository. The data is related with direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to access if the product (bank term deposit) would be ('yes') or not ('no') subscribed (3). There are total 21 variables (including y, the response variable), 41188 valid records.

Samples of the dataset:

	age	job	marital	education	default	housing	loan	contact	...	y
35666	58	management	married	university.degree	no	no	no	cellular	...	no
36531	48	admin.	married	university.degree	no	yes	no	cellular	...	yes
38676	73	retired	married	university.degree	no	no	yes	cellular	...	yes
15146	49	technician	divorced	high.school	no	no	no	cellular	...	no
33230	33	blue-collar	single	basic.6y	no	yes	no	cellular	...	no

Table 5: Dataset Samples

There are 41188 observations with 21 features

2.2 Feature Description

There are 20 explanatory variables, including numerical and categorical variables:

	Feature	Type
1	age	numerical
2	job	categorical
3	marital	categorical
4	education	categorical
5	default	categorical
6	housing	categorical
7	loan	categorical

Table 6: Bank client data

	Feature	Type
8	contact	categorical
9	month	categorical
10	day_of_week	categorical
11	duration	numerical

Table 7: Related with the last contact of the current campaign

	Feature	Type
12	campaign	numerical
13	pdays	numerical
14	previous	numerical
15	poutcome	categorical

Table 8: Other attributes

	Feature	Type
16	emp.var.rate	numerical
17	cons.price.idx	numerical
18	cons.conf.idx	numerical
19	euribor3m	numerical
20	nr.employed	numerical

Table 9: social and economic context attributes

2.3 Project Description

We are going to perform an analysis in **topic 6: Classification / Decision and Regression Trees**. We will use Decision Tree, Random Forest and also Logistic Regression to do classification on response variable **y**, i.e. whether the client subscribed a term deposit.

The following steps will be performed to complete this section:

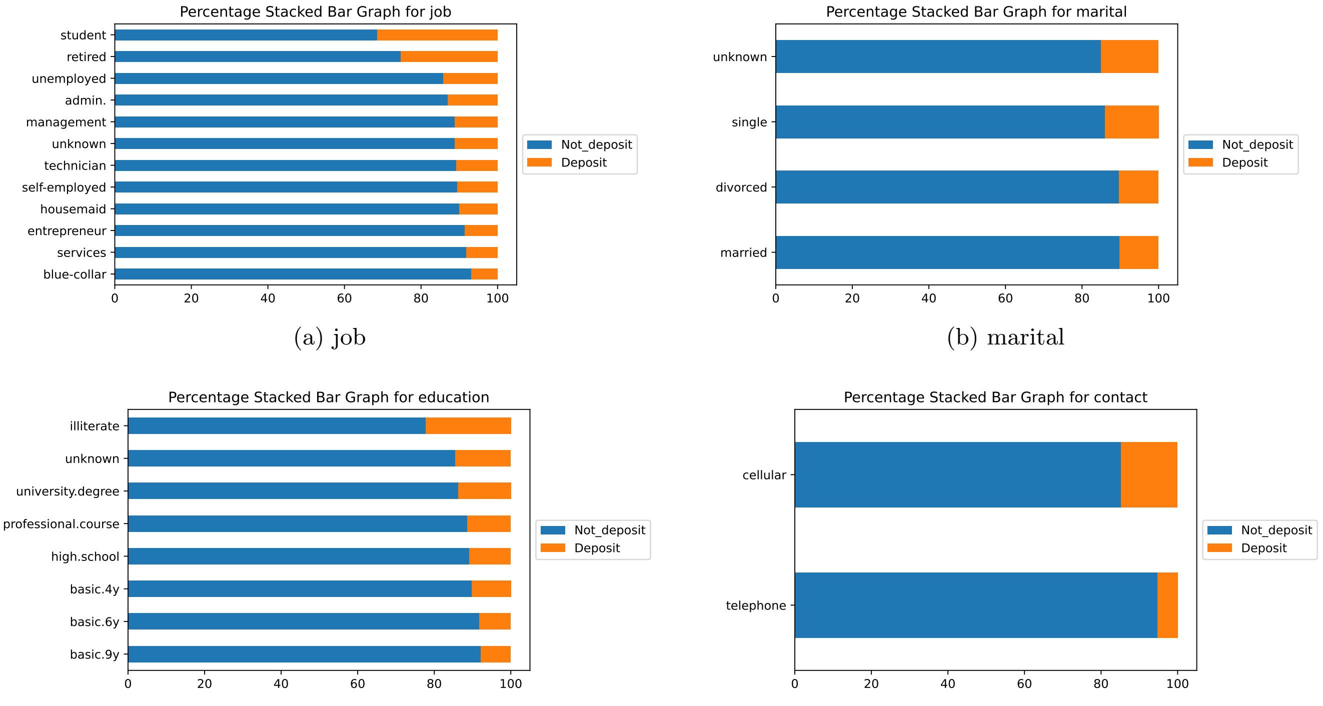
1. process data explanatory data analysis (EDA)
2. data preparation for modeling
3. visualization of random forest, decision tree and logistic regression
4. cross validation and grid search
5. logistic regression
6. comparison of performance of random forest, decision tree and logistic regression
7. conclusion

2.4 Process Data and Explanatory Data Analysis (EDA)

In order to easily implement machine learning algorithms, instead of writing from scratch, we imported `sklearn` library (4).

2.4.1 Categorical Variables

We plot percentage stacked bar graphs for 6 categorical variables in table 6 vs. `y`, to see the proportions of $\texttt{y} = 1$ given the client is different categories.

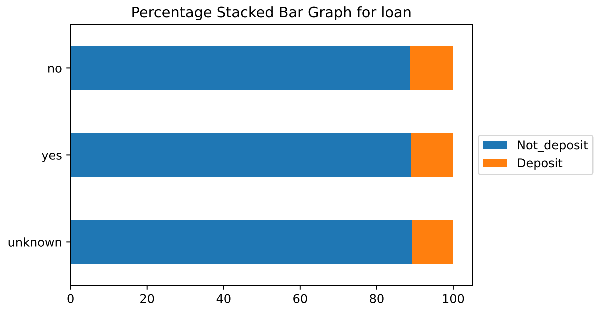


(a) job

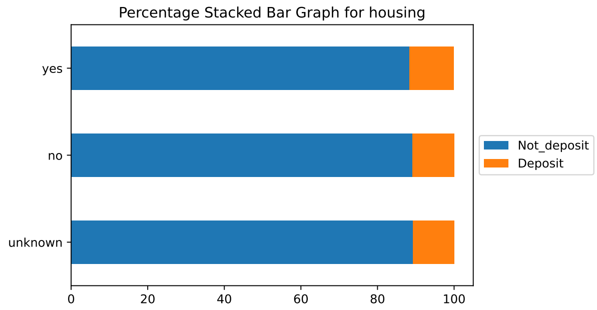
(b) marital

(c) education

(d) contact



(e) loan



(f) contact

Summary of Categorical Variables:

1. The proportion of opening deposit or not is different among different job type. When the job types are student, retired and unemployed, the proportions of opening the deposit are highest. When the job types are entrepreneur, services and blue-collar, the proportions of opening the deposit are lowest.

2. The proportion of opening deposit of unknown is close to that of single, while both of them have higher proportion than divorced and married.

3. It is surprised to see that illiterate has the highest proportion, other than that, generally higher education levels have higher proportion of opening deposit.

4. The proportion of opening deposit of using cellular is much higher than that of using telephone.

5. The proportion is basically no different among loan status.

6. The proportion is basically no different among contact status.

2.4.2 Numerical Variables

After exploring the categorical variables, we take a look into numerical variables, by calculate (Pearson) correlations of each variables, we can see which variables have significant relationship with `y`. The correlation matrix are as follow:

	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed	y
age	1.0000	-0.0009	0.0046	-0.0344	0.0244	-0.0004	0.0009	0.1294	0.0108	-0.0177	0.0304
duration	-0.0009	1.0000	-0.0717	-0.0476	0.0206	-0.0280	0.0053	-0.0082	-0.0329	-0.0447	0.4053
campaign	0.0046	-0.0717	1.0000	0.0526	-0.0791	0.1508	0.1278	-0.0137	0.1351	0.1441	-0.0664
pdays	-0.0344	-0.0476	0.0526	1.0000	-0.5875	0.2710	0.0789	-0.0913	0.2969	0.3726	-0.3249
previous	0.0244	0.0206	-0.0791	-0.5875	1.0000	-0.4205	-0.2031	-0.0509	-0.4545	-0.5013	0.2302
emp.var.rate	-0.0004	-0.0280	0.1508	0.2710	-0.4205	1.0000	0.7753	0.1960	0.9722	0.9070	-0.2983
cons.price.idx	0.0009	0.0053	0.1278	0.0789	-0.2031	0.7753	1.0000	0.0590	0.6882	0.5220	-0.1362
cons.conf.idx	0.1294	-0.0082	-0.0137	-0.0913	-0.0509	0.1960	0.0590	1.0000	0.2777	0.1005	0.0549
euribor3m	0.0108	-0.0329	0.1351	0.2969	-0.4545	0.9722	0.6882	0.2777	1.0000	0.9452	-0.3078
nr.employed	-0.0177	-0.0447	0.1441	0.3726	-0.5013	0.9070	0.5220	0.1005	0.9452	1.0000	-0.3547
y	0.0304	0.4053	-0.0664	-0.3249	0.2302	-0.2983	-0.1362	0.0549	-0.3078	-0.3547	1.0000

Table 10: Correlation Matrix

Summary of Numerical Variables:

- `duration` is the highest correlated variable with target feature (0.4053).

- `nr.employed, pdays, euribor3m` are also highly correlated with target feature.

2.5 Data Preparation for modeling

2.5.1 Transformation for Categorical Variables

Since this dataset contains a lot of categorical variables and the number of weakly correlated numeric variables is small, therefore, we use one-hot encoding to transform categorical data.

- For binary categorical variables (`"yes", "no"`), we transform it into binary number accordingly (`0, 1`).
- For `job, marital, education, month, day_of_week`, we use one-hot encoding to transform these variables since they have more than 3 types of possible options. Also, some variables contain missing data, labeled as `unknown` or `nonexistent`, we treat it as 0 (i.e. no) in general.
- For `pdays`, there exists many records of `999` which means the client was not previously contacted, we replaces those with `0`.
- At last, we remove duplicated records.

	age	marital	education	default	housing	loan	contact	duration	campaign	...
29851	52	0.103231	0.137219	0	1	0	1	70	4	...
15768	39	0.140090	0.113550	0	0	0	0	65	2	...
38033	76	0.101569	0.102515	0	1	0	0	259	2	...
27290	37	0.140090	0.137219	0	1	1	0	661	1	...
30685	34	0.101569	0.078246	0	1	0	0	332	1	...

Table 11: Dataset Samples after Encoding

After removing duplicates, there are 41173 observations with 44 features in the prepared dataset

2.5.2 Splitting Training Dataset and Testing Dataset

To test the performance of machine learning alogrithm, we split the dataset into training and testing datasets, 80% and 20% of the full dataset respectively. The models would be built using training dataset, and evaluated the performance using testing dataset accordingly. In order to make the prediction model reproducible, we fixed the random seed: `random_state=4002`.

Total records in training dataset = 32938, number of class `0` and 1 in training dataset: `[29267, 3671]`, noted that this is the numbers of true positive and true negative. These are used to calculate the support, confidence and capture in evaluating performance under training dataset.

2.6 Decision Tree

We set the depth of decision tree to 3, in order to plot a simplified decision tree.

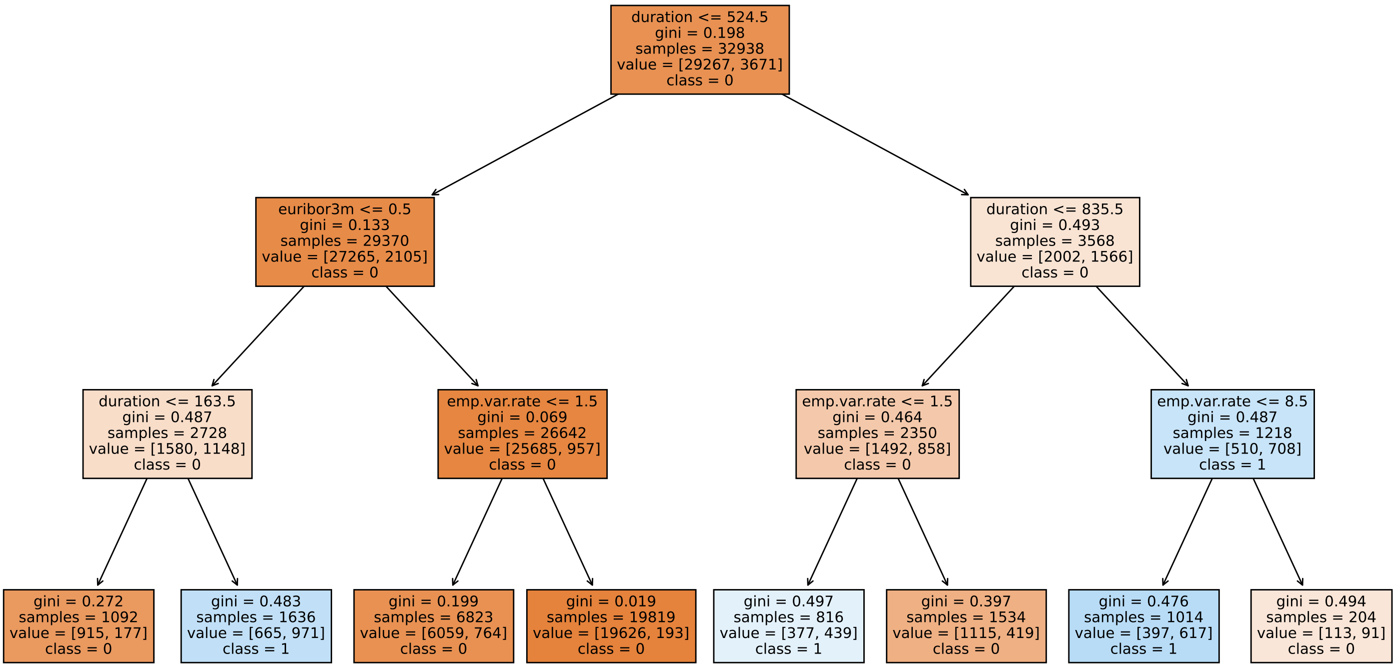


Figure 3: Decision Tree

Decision Rules

R1 If `duration` <= 524.5 and `euribor3m` <= 0.5 and `duration` <= 163.5, then `y` = 0
Gini = 0.272, Support = 0.0332, Confidence = 0.8379, Capture = 0.0313

R2 If `duration` <= 524.5 and `euribor3m` <= 0.5 and `duration` > 163.5, then `y` = 1
Gini = 0.483, Support = 0.0497, Confidence = 0.5935, Capture = 0.2645

R3 If `duration` <= 524.5 and `euribor3m` > 0.5 and `emp.var.rate` <= 1.5, then `y` = 0
Gini = 0.199, Support = 0.2071, Confidence = 0.8880, Capture = 0.2070

R4 If `duration` <= 524.5 and `euribor3m` > 0.5 and `emp.var.rate` > 1.5, then `y` = 0
Gini = 0.019, Support = 0.6017, Confidence = 0.9902, Capture = 0.6706

R5 If `duration` > 524.5 and `duration` <= 835.5 and `emp.var.rate` <= 1.5, then `y` = 1
Gini = 0.497, Support = 0.0247, Confidence = 0.5380, Capture = 0.1196

R6 If `duration` > 524.5 and `duration` <= 835.5 and `emp.var.rate` > 1.5, then `y` = 0
Gini = 0.397, Support = 0.0466, Confidence = 0.7269, Capture = 0.0381

R7 If `duration` > 524.5 and `duration` > 835.5 and `emp.var.rate` <= 8.5, then `y` = 1
Gini = 0.476, Support = 0.0308, Confidence = 0.6085, Capture = 0.1681

R8 If `duration` > 524.5 and `duration` > 835.5 and `emp.var.rate` > 8.5, then `y` = 0
Gini = 0.494, Support = 0.0062, Confidence = 0.5539, Capture = 0.0039

As the maximum depth is set to be only 3, there are only 8 rules, and quite a few redundant leafs, e.g., R3 and R4 could be combined into one rule: If `duration` <= 524.5 and `euribor3m` > 0.5, then `y` = 0

2.7 Random Forest

Again, the depth of decision trees are set to be 3, a simplified random forest is as followed:

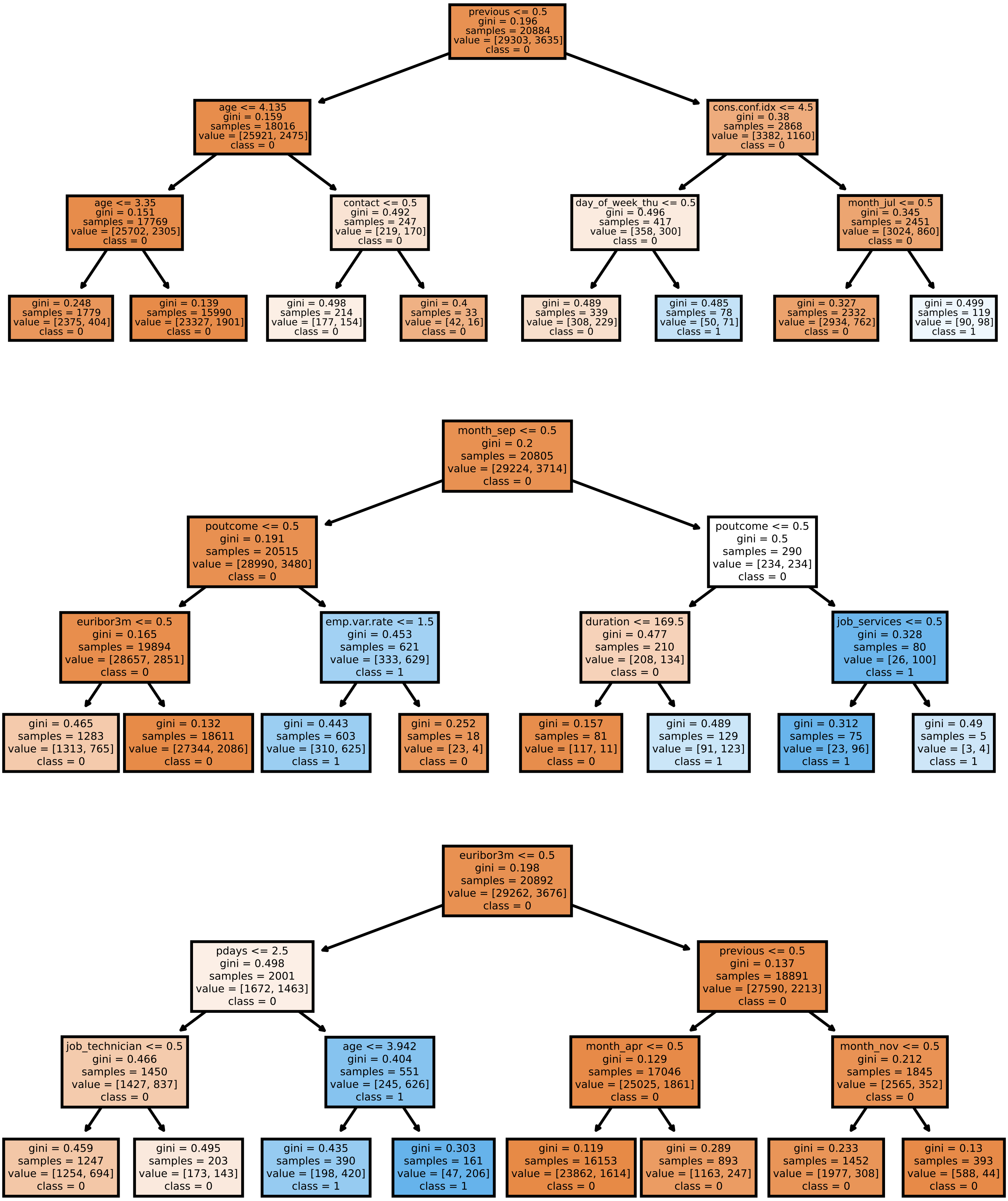

previous <= 0.5 gini = 0.196 samples = 20884 value = [29303, 3635] class = 0
age <= 4.135 gini = 0.159 samples = 18016 value = [25921, 2475] class = 0
cons.conf.idx <= 4.5 gini = 0.38 samples = 2868 value = [3382, 1160] class = 0
age <= 3.35 gini = 0.151 samples = 17769 value = [25702, 2305] class = 0
contact <= 0.5 gini = 0.492 samples = 247 value = [219, 170] class = 0
day_of_week_thu <= 0.5 gini = 0.496 samples = 417 value = [358, 300] class = 0
month_jul <= 0.5 gini = 0.345 samples = 2451 value = [3024, 860] class = 0
gini = 0.248 samples = 1779 value = [2375, 404] class = 0
gini = 0.139 samples = 15990 value = [23327, 1901] class = 0
gini = 0.498 samples = 214 value = [177, 154] class = 0
gini = 0.4 samples = 33 value = [42, 16] class = 0
gini = 0.489 samples = 339 value = [308, 229] class = 0
gini = 0.485 samples = 78 value = [50, 71] class = 1
gini = 0.327 samples = 2332 value = [2934, 762] class = 0
gini = 0.499 samples = 119 value = [90, 98] class = 1
month_sep <= 0.5 gini = 0.2 samples = 20805 value = [29224, 3714] class = 0
poutcome <= 0.5 gini = 0.191 samples = 20515 value = [28990, 3480] class = 0
poutcome <= 0.5 gini = 0.5 samples = 290 value = [234, 234] class = 0
euribor3m <= 0.5 gini = 0.165 samples = 19894 value = [28657, 2851] class = 0
emp.var.rate <= 1.5 gini = 0.453 samples = 621 value = [333, 629] class = 1
duration <= 169.5 gini = 0.477 samples = 210 value = [208, 134] class = 0
job_services <= 0.5 gini = 0.328 samples = 80 value = [26, 100] class = 1
gini = 0.465 samples = 1283 value = [1313, 765] class = 0
gini = 0.132 samples = 18611 value = [27344, 2086] class = 0
gini = 0.443 samples = 603 value = [310, 625] class = 1
gini = 0.252 samples = 18 value = [23, 4] class = 0
gini = 0.157 samples = 81 value = [117, 11] class = 0
gini = 0.489 samples = 129 value = [91, 123] class = 1
gini = 0.312 samples = 75 value = [23, 96] class = 1
gini = 0.49 samples = 5 value = [3, 4] class = 1
euribor3m <= 0.5 gini = 0.198 samples = 20892 value = [29262, 3676] class = 0
pdays <= 2.5 gini = 0.498 samples = 2001 value = [1672, 1463] class = 0
previous <= 0.5 gini = 0.137 samples = 18891 value = [27590, 2213] class = 0
job_technician <= 0.5 gini = 0.466 samples = 1450 value = [1427, 837] class = 0
age <= 3.942 gini = 0.404 samples = 551 value = [245, 626] class = 1
month_apr <= 0.5 gini = 0.129 samples = 17046 value = [25025, 1861] class = 0
month_nov <= 0.5 gini = 0.212 samples = 1845 value = [2565, 352] class = 0
gini = 0.459 samples = 1247 value = [1254, 694] class = 0
gini = 0.495 samples = 203 value = [173, 143] class = 0
gini = 0.435 samples = 390 value = [198, 420] class = 1
gini = 0.303 samples = 161 value = [47, 206] class = 1
gini = 0.119 samples = 16153 value = [23862, 1614] class = 0
gini = 0.289 samples = 893 value = [1163, 247] class = 0
gini = 0.233 samples = 1452 value = [1977, 308] class = 0
gini = 0.13 samples = 393 value = [588, 44] class = 0

Figure 4: Random Forest

In this simplified random forest, we obtain 10 decision trees (only 3 were printed above), then the prediction is obtained by the majority vote in the case of classification.

2.8 Logistic Regression

Since random forest is expansion of decision tree, we also include logistic regression to compare with former two models. Notice that plotting the logistic regression logit plot requires extremely long time for plotting ten of thousands of points, so we only use `duration` as explanatory variable to demonstrate in plot.
From the graph, we can see that using only `duration` has low accuracy, so we introduce more explanatory variables in next section.

2.9 Cross Validation and Grid Search Procedures

To optimize our prediction model, we use 5-fold cross validation and grid search to find the optimized parameters. This step is implemented using `Pipeline` and `GridSearchCV` from `sklearn`. The training accuracy rate, testing accuracy rate and computation time are calculated base on the best model under grid search.

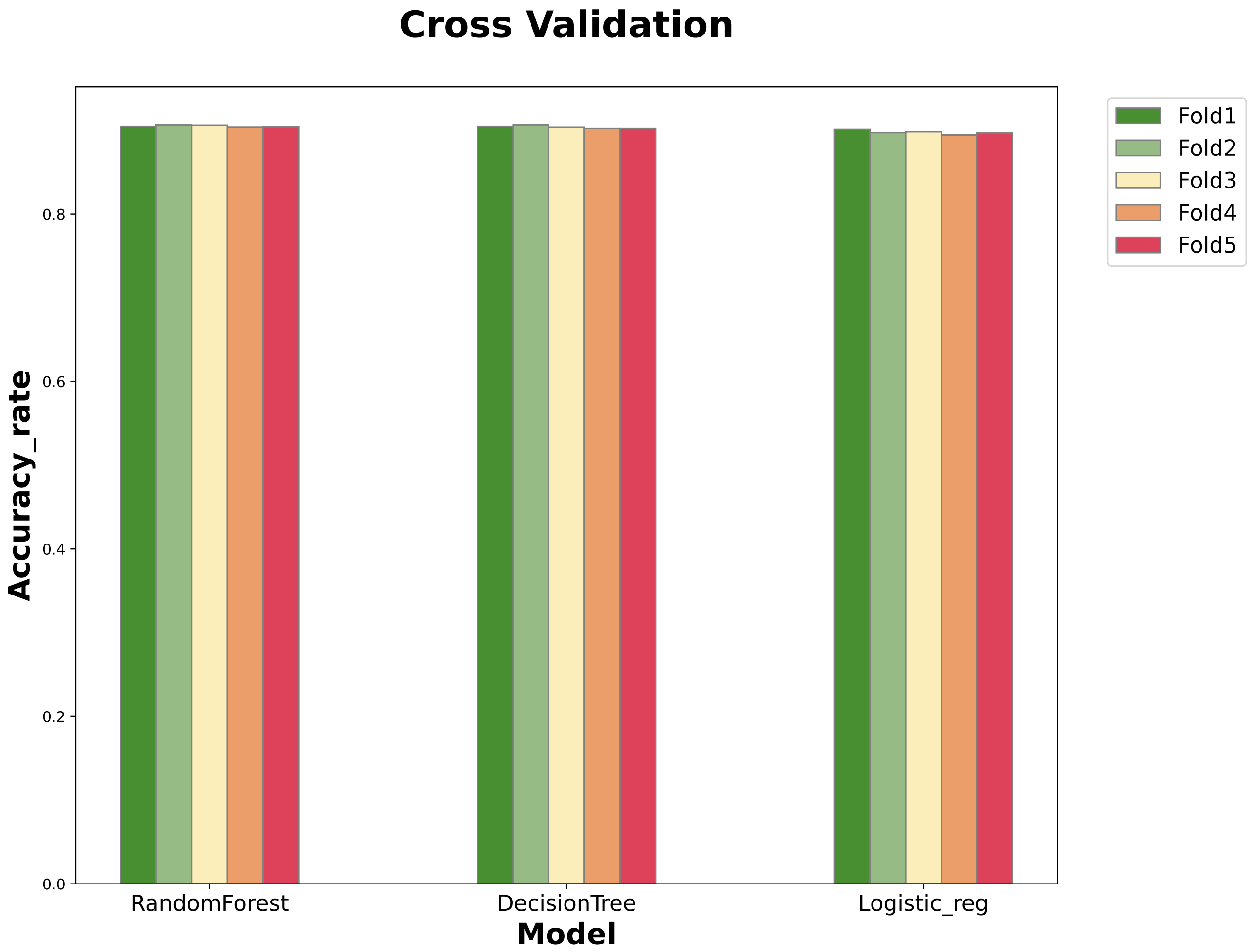


Figure 5: Performance in Cross Validation

Output: (refer to source code)

```
Using RandomForest model
training accuracy rate is 0.9048818398071896
ROC_AUC: 0.9397072069254537
best parms is {'rf__criterion':  'gini', 'rf__max_depth':  25, 'rf__min_samples_leaf':  60,
'rf__min_samples_split':  3, 'rf__n_estimators':  50}
Accuracy:  0.8990892531876138
Precision:  0.7546468401486989
Recall:  0.2097107438016529
It requires 1274.78 sec to compute

Using DecisionTree model
training accuracy rate is 0.9037888422385209
ROC_AUC: 0.8968310413769026
best parms is {'dt__criterion':  'gini', 'dt__max_depth':  10, 'dt__min_samples_leaf':  7}
Accuracy:  0.8993321190042501
Precision:  0.6026587887740029
Recall:  0.4214876033057851
It requires 3.91 sec to compute

Using Logistic_reg model
training accuracy rate is 0.8977168050738198
ROC_AUC: 0.9171117141112263
best parms is {'lr__C': 0.3, 'lr__penalty':  'l1', 'lr__solver':  'saga'}
Accuracy:  0.8970248937462052
Precision:  0.6578947368421053
Recall:  0.25826446280991733
It requires 244.53 sec to compute
```

Confusion matrix: Please refer to the confusion matrix heatmap on source code.

2.10 Comparison of Performance of Random Forest and Decision Tree

For reference, this running trial is performed using Intel Core$^{\text{TM}}$ i5-8250U[1](4C8T), 8-GB RAM.

2.10.1 Accuracy Rate, ROC rate and Computation Time

	Model_used	Accuracy	Precision	Recall	ROC_rate	Time_used
0	RandomForest	0.899089	0.754647	0.209711	0.939707	1274.78
1	DecisionTree	0.899332	0.602659	0.421488	0.896831	3.91
2	Logistic_reg	0.897025	0.657895	0.258264	0.917112	244.53

Table 12: Accuracy Rate, ROC rate and Computation Time

[1] https://ark.intel.com/content/www/us/en/ark/products/124967/intel-core-i58250u-processor-6m-cache-up-to-3-40-ghz.html

2.10.2 ROC Graph

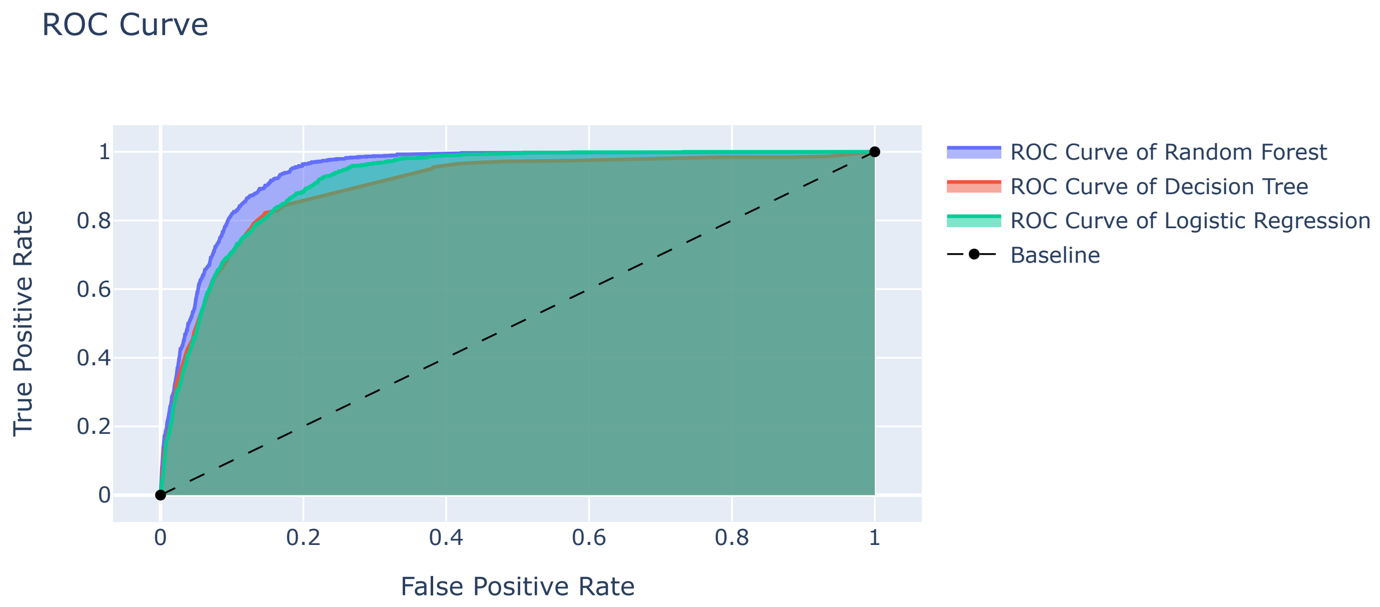


Figure 6: ROC Graph

2.11 Conclusion

From Table 12, random forest has the highest roc_rate, while the accuracy rate among three models are very close. Speaking of computation time, decision tree requires shortest time, random forest requires extremely longer time relatively to others, in which decision tree only requires less than 5 seconds.
From Figure 6, the roc curve of logistic regression is very close to random forest after FPR > 0.3

Our group believe that random forest performs better than decision tree in this case, in terms of the accuracy in roc rate. However, in order to balance the limit of computation time, we could use logistic regression in large dataset.

3 Bibliography

References

[1] J.P. Morgan. Risk metrics: technical document, 1996.

[2] William Rea, Alethea Rea, and Libin Yang. Stock selection with principal component analysis, 02 2015.

[3] Sérgio Moro, Paulo Cortez, and Paulo Rita. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62:22–31, 2014.

[4] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.