Milestone 2
STOCK PRICE PREDICTION

Team 10
Srikanth Parvathala
Vijay Arni
Ya-Ting Yang

Research Question

Can future stock prices be predictable using historical stock price of specific company and macroeconomic indicators like the nominal GDP index, real GDP index, unemployment rate, and federal funds rate?

Changes considered after Milestone 1:

- Based on the milestone 1 feedback, we added Federal fund rate (Interest rates) which might play a significant role in stock price prediction.
- We consider it as Independent variable.
- We also considered company source as Independent variable and converted into dummy continuous variable.
- All the datasets are merged based on month and year.

Linear Regression

A. Split Data into Training and Testing data
B. Linear regression on individual variables
C. Multivariate Regression
D. Regularization
E. Repeating experiments with different random splits

Q1. a. Linear Regression

Intercept and Coefficient for each individual feature:

Feature	Intercept	Coefficient	p-value
Monthly_Nominal_GDP_Index	-3.530e+03	2.068e-01	< 2.2e-16
Monthly_Real_GDP_Index	-6.683e+03	4.001e-01	< 2.2e-16
Unemployment_Rate	432.629	49.565	< 2.2e-16
Federal_Funds_Rate	736.28	-61.29	3.668e-07
Sourceamazon	443.038	1216.589	< 2.2e-16
Sourceapple	839.68	-778.38	< 2.2e-16
Sourcefacebook	808.63	-627.69	< 2.2e-16
Sourcegoogle	544.84	684.65	< 2.2e-16
Sourcenetflix	784.91	502.19	< 2.2e-16

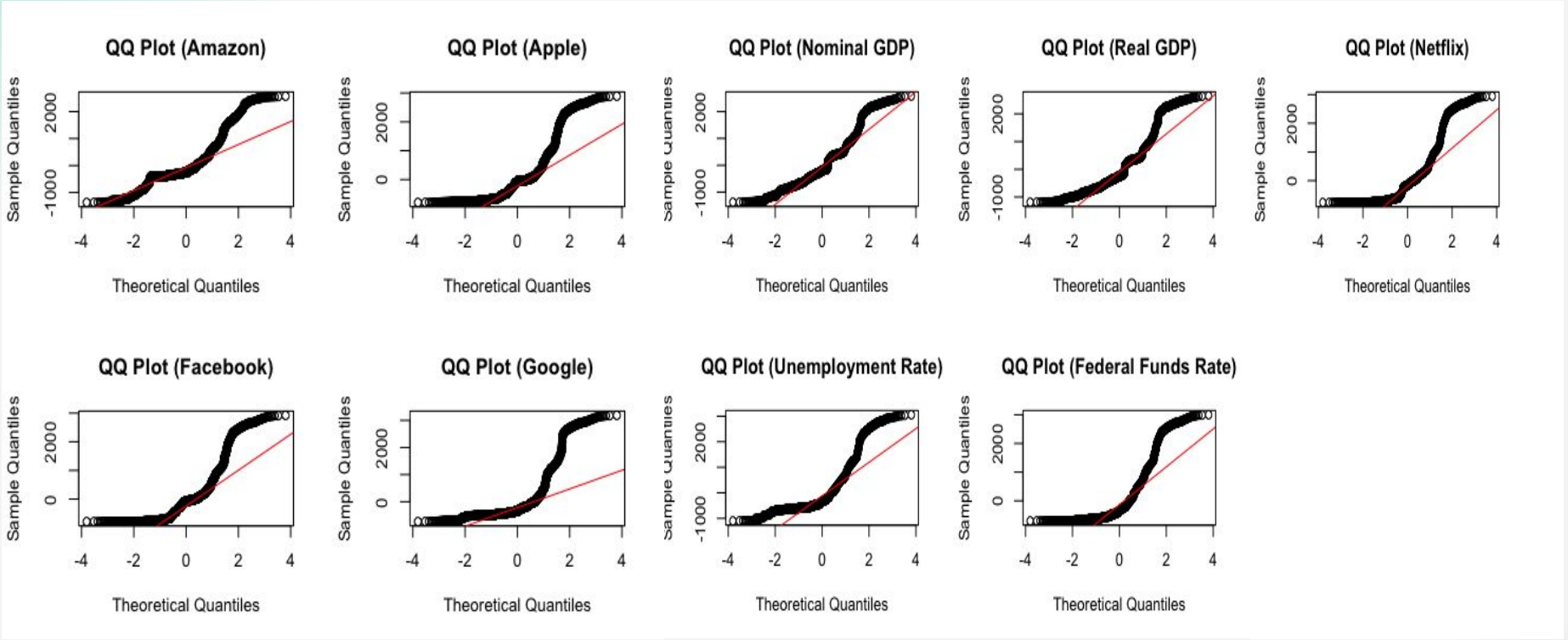
Predictive feature :

- A p-value less than 0.05 typically suggests that the predictor is statistically significant.
- All the individual features have p-values less than 0.05, indicating that all are statistically significant predictive features.

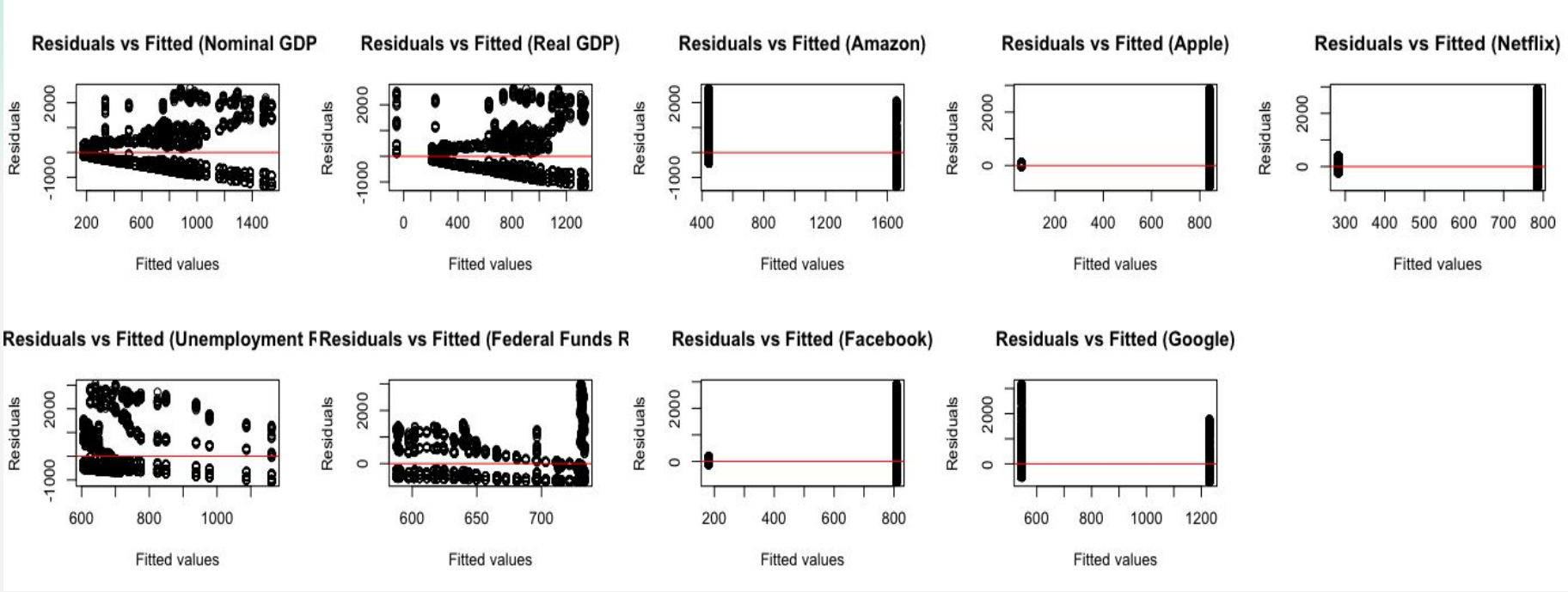
Most Predictive feature :

- Based on the coefficient value, Source Amazon has a highest coefficient value.
- Followed by Source Apple and Source Facebook also have high magnitude coefficients but in the negative direction.
- Monthly_Nominal_GDP_Index" and "Monthly_Real_GDP_Index have the strongest relationships with the dependent variable.

QQ plot of actual and predictive variables



Residual Plots



Prediction accuracy on testing data

- The correlation (R-squared) between the predicted and real values is 0.99999983
- This indicating that the model can explain almost 100% of the variation in the closing stock prices with the provided features.
- The mean squared error (MSE) between the predicted and real closing stock prices is 0.4030519
- MSE value suggesting that the model's predictions are very close to the actual values, with minor deviations.

Multivariate Regressions

- Source Amazon, has an adjusted R^2 of 0.3392 and has an AIC value of 14684.94. However, by the time we include eight predictors in the model, the adjusted R^2 increases to 0.78584 and the AIC value decreases to 8.
- Considering combinations of independent features indeed improves the prediction results.

Coefficient values

Predictor	Coefficient (Estimate)	p-value
Monthly_Nominal_GDP_Index	2.710e-01	< 2e-16
Monthly_Real_GDP_Index	-1.347e-01	0.00779
Unemployment_Rate	3.595e+01	5.88e-15
Federal_Funds_Rate	-5.453e+01	8.12e-14
Sourceamazon	1.368e+03	< 2e-16
Sourceapple	-2.295e+02	< 2e-16
Sourcefacebook	-1.075e+02	1.66e-13
Sourcegoogle	9.337e+02	< 2e-16
Sourcenetflix	NA	NA

Most predictive feature according to training data

- Based on the magnitude of coefficient, Source Amazon stands out as the most predictive feature, this feature alone explains about 33.92% of the variance in the stock Close price. Source Google is another key predictor.
- Monthly_Nominal_GDP_Index and Unemployment_Rate also play significant roles, as indicated by their coefficients values.

Prediction accuracy for Multivariate Regression

- To predict the accuracy, the correlation value should be closer to 1 and mean square error value less.
- Correlation value: 0.882386292602172
- Mean Square Error : 150745.609776844
- Correlation value is high and close to 1 which suggests multivariante model is performing well in prediction close stock prices.
- When compared with individual features, multivariate regression has less Mean Square Error which makes this model prediction accuracy high and might be a better fit for the data.

Regularization

- For individual predictors, the highest correlation in both Ridge and Lasso regressions is achieved using Source Amazon with a value of 0.6026. This is closely followed by Monthly_Nominal_GDP_Index with a value of 0.4016
- The multivariate regression, which considers all the predictors together, achieves a much higher correlation of approximately 0.8827. This is significantly better than any of the individual predictors.
- Among the individual predictors, Source Amazon has the lowest MSE for both Ridge and Lasso regressions, with values 434,621.4 and 433,717, respectively.
- Again, the multivariate regression outperforms all the individual predictors with a much lower MSE of 151,123.1 for Ridge and 150,329.7 for Lasso.

Repeating experiments with different random splits

Regression Type	Average Correlation	Average MSE
Linear Regression	0.8856	149,495.67
Ridge Regression	0.8827	151,123.13
Lasso Regression	0.8827	150,329.68

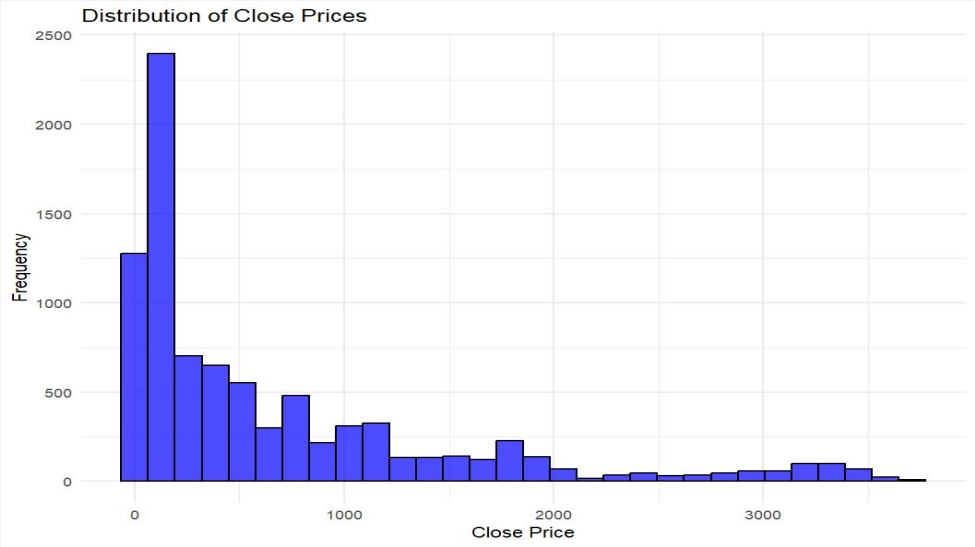
- All three regression types have very similar correlation values, with Linear Regression having a slightly higher correlation.
- Across the 10 iterations with different random splits, the models' performance metrics are quite consistent.
- This indicates the robustness of the models, and the random splitting of the data doesn't introduce a large variability in the performance metrics.

Logistic Regression and Naive Bayes

A. Logistic Regression
B. Naive Bayes
C. Confusion Matrix
D. Log Odds and Odd Ratio

Logistic Regression Analysis

- Intercept: 0.33549
- Key Features: Monthly_Real_GDP_Index, Unemployment_Rate, Prev_Close
- The log odds of stock price increase with all variables constant



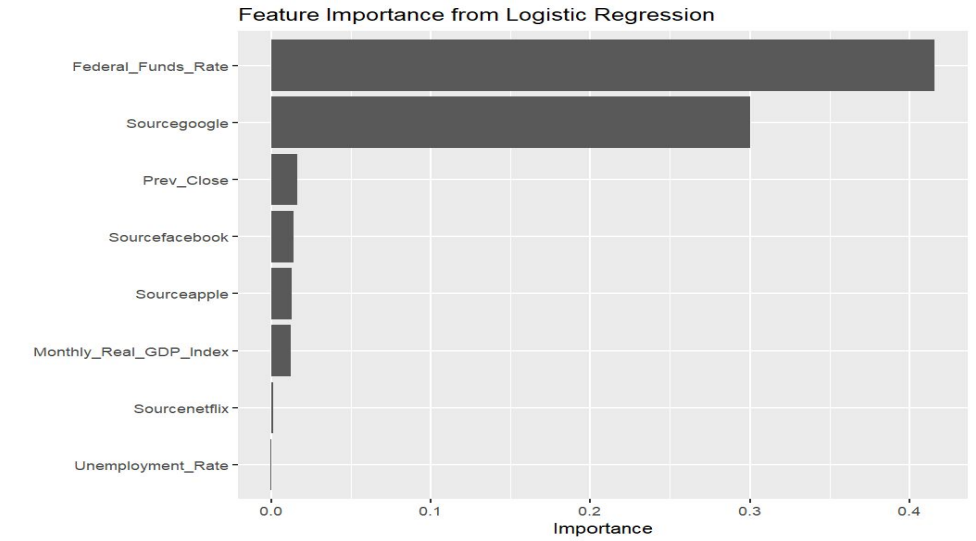
Logistic Regression

Intercept and Coefficient for each individual feature:

Feature	Estimate	Std. Error	p-value
Monthly_Real_GDP_Index	0.335	0.080	0.012
Unemployment_Rate	0.11022	0.030	0.0002
Federal_Funds_Rate	0.024	0.029	0.415
Source apple	-0.307	0.123	0.012
Source facebook	-0.288	0.117	0.014
Source google	-0.830	0.080	0.300
Source netflix	-0.364	0.112	0.001

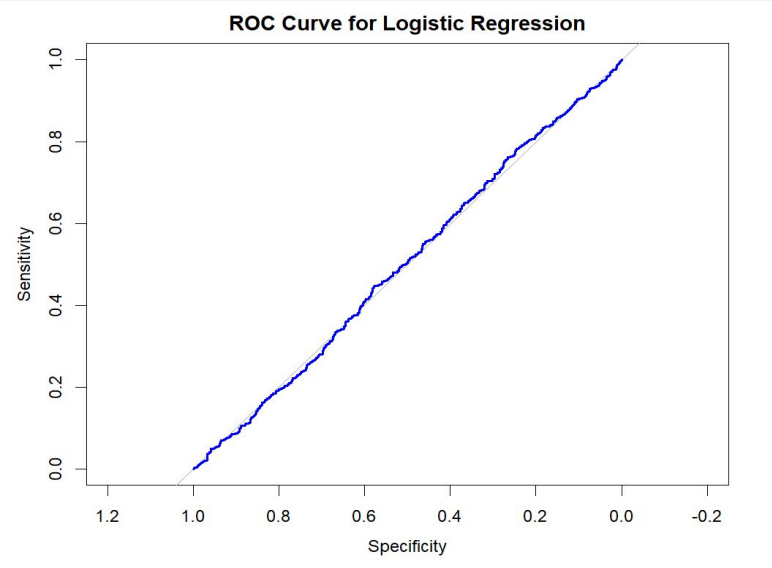
Logistic Regression

- Coefficients represent change in log-odds for a unit increase in predictors.
- Odds ratios can be computed by exponentiating coefficients.
- Most Predictive Features: Unemployment_Rate, Prev_Close, and specific company sources.



Logistic Regression

- Model tested on a separate testing dataset.
- Performance metric: AUC = 0.5014185.
- Indicates performance better than random guessing.



Naive Bayes

Naive Bayes Classifier:

- Based on Bayes theorem with an assumption of independence between predictors.
- Variables were categorized before model computation.
- Dataset was divided into training and testing sets.
- Classifier was trained and tested.
- Confusion matrix was used to evaluate performance.

Laplace Estimator:

- Laplace estimator was used to handle zero probabilities.
- Results were compared with and without the Laplace estimator.
- No significant improvement was observed with the estimator.

Decision Trees and Random Forests

A. Split Data into Training and Testing data
B. Decision Tree and the Confusion Matrices
C. Boosting
D. Bagging and Random Forests

Split Data into Training and Testing data

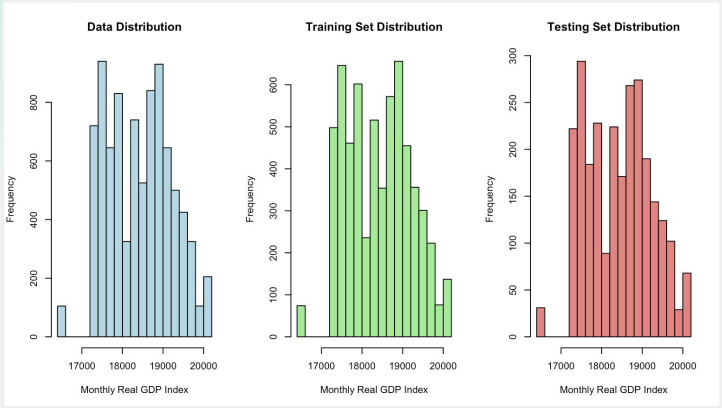
```r
# Randomize data
set.seed(1234)
data <- data[order(runif(nrow(data))), ]

# Determine and create classes
data_range <- range(data$Close)
min_value <- data_range[1]
max_value <- data_range[2]
num_folds <- 5
range_width <- (max_value - min_value) / 270
data$Class <- cut(data$Close,
                  breaks = seq(min_value, max_value, by = range_width),
                  labels = FALSE)

data <- data[!is.na(data$Class), ]

# Perform K-fold cross-validation
folds <- createFolds(data$Class, k = num_folds, list = TRUE)
train_datasets <- list()
test_datasets <- list()

for (i in 1:num_folds) {
  # Extract the indices for the current fold
  train_indices <- unlist(folds[-i])
  test_indices <- unlist(folds[i])

  # Create train and test datasets based on the indices
  train_set <- data[train_indices, ]
  test_set <- data[test_indices, ]

  # Store the datasets in the lists
  train_datasets[[i]] <- train_set
  test_datasets[[i]] <- test_set
}
```

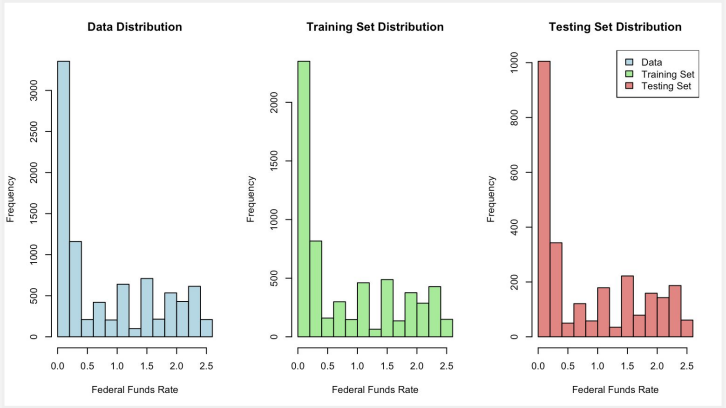
→ Randomize data

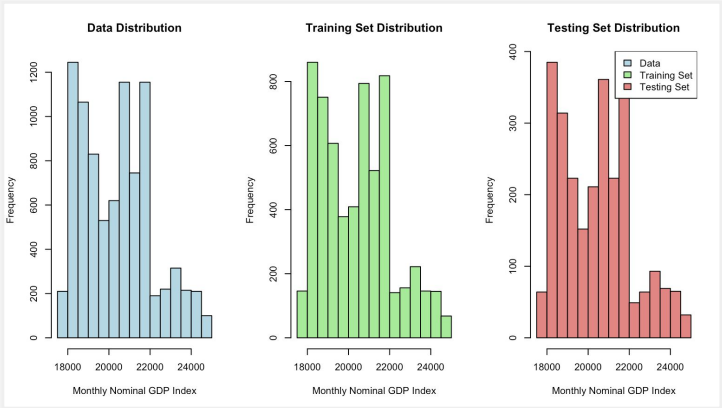
→ Create Classes for “Close” column value

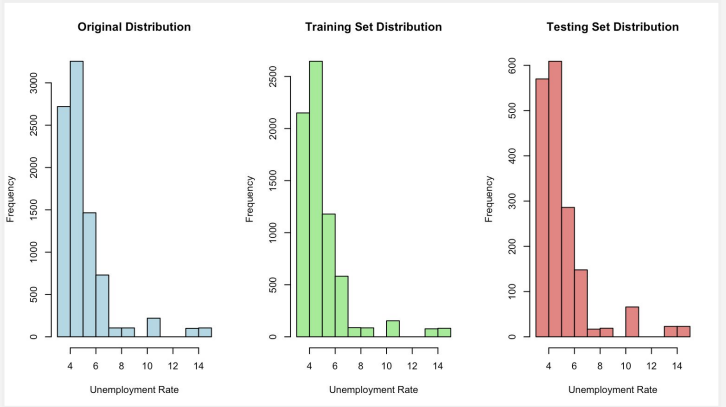
→ K-fold cross-validation for refines data division

The Distribution of Original Data, Training Data, and Testing Data









The Decision Tree Incorporating 4 Variables

```
Federal_Funds_Rate <= 0.9:
:...Monthly_Real_GDP_Index > 18274.32:
:   :...Unemployment_Rate <= 5.6:
:   :   :...Unemployment_Rate > 4.6: 10 (286/254)
:   :   :   Unemployment_Rate <= 4.6:
:   :   :   :...Federal_Funds_Rate <= 0.36: 23 (281/252)
:   :   :       Federal_Funds_Rate > 0.36: 3 (100/90)
:   :   Unemployment_Rate > 5.6:
:   :   :...Unemployment_Rate <= 6.7: 8 (758/654)
:   :       Unemployment_Rate > 6.7:
:   :       :...Unemployment_Rate <= 8.4: 7 (294/249)
:   :           Unemployment_Rate > 8.4: 6 (94/79)
:   Monthly_Real_GDP_Index <= 18274.32:
:   :...Unemployment_Rate <= 5.2:
:       :...Monthly_Real_GDP_Index > 17644.14:
:       :   :...Unemployment_Rate <= 4.6: 9 (273/179)
:       :   :   Unemployment_Rate > 4.6:
:       :   :   :...Monthly_Nominal_GDP_Index <= 18774.47: 55 (103/77)
:       :   :       Monthly_Nominal_GDP_Index > 18774.47: 8 (465/357)
:       :   Monthly_Real_GDP_Index <= 17644.14:
:       :   :...Monthly_Real_GDP_Index <= 17520.27:
:       :       :...Monthly_Nominal_GDP_Index <= 18346.29: 1 (378/299)
:       :       :   Monthly_Nominal_GDP_Index > 18346.29: 6 (284/212)
:       :       Monthly_Real_GDP_Index > 17520.27:
:       :       :...Monthly_Nominal_GDP_Index <= 18512.66: 6 (189/149)
:       :           Monthly_Nominal_GDP_Index > 18512.66:
:       :           :...Unemployment_Rate <= 5: 1 (281/226)
:       :               Unemployment_Rate > 5: 7 (93/69)
:       Unemployment_Rate > 5.2:
:       :...Unemployment_Rate > 7.9:
:           :...Monthly_Nominal_GDP_Index <= 19095.95: 4 (93/77)
:           :   Monthly_Nominal_GDP_Index > 19095.95: 5 (188/158)
:           Unemployment_Rate <= 7.9:
:           :...Monthly_Nominal_GDP_Index > 18161.75: 5 (187/125)
:               Monthly_Nominal_GDP_Index <= 18161.75:
:               :...Unemployment_Rate > 5.6: 1 (91/71)
:                   Unemployment_Rate <= 5.6:
:                   :...Unemployment_Rate <= 5.4: 5 (197/149)
:                       Unemployment_Rate > 5.4: 4 (86/61)
```

Decision Tree

- Variables:
  - Monthly_Nominal_GDP_Index
  - Monthly_Real_GDP_Index
  - Unemployment_Rate
  - Federal_Funds_Rate
- Tree size: 33
- Key decision points: "Federal_Funds_Rate" and “Monthly_Real_GDP_Index

Performance

- Accuracy rate is lower than expected
  - Training dataset: 20.04% (0.2004291)
  - Testing dataset: 18.72% (0.1872872)
- Error rate
  - Training data: 79.95% (0.7995709)
  - Testing data: 81.27% (0.8127128)
- The decision tree model did not deliver satisfactory performance.

The Decision Tree Incorporating 5 Variables

```
Source in {amazon,google}:
:...Monthly_Nominal_GDP_Index <= 19748.54:
:   :...Monthly_Nominal_GDP_Index <= 18659.54:
:   :   :...Unemployment_Rate > 5.1:
:   :   :   :...Source = google:
:   :   :   :   :...Monthly_Real_GDP_Index <= 17287.58:
:   :   :   :   :   :...Monthly_Nominal_GDP_Index <= 17930.52: 35 (18/10)
:   :   :   :   :   :   Monthly_Nominal_GDP_Index > 17930.52: 39 (18/6)
:   :   :   :   :   Monthly_Real_GDP_Index > 17287.58:
:   :   :   :   :   :...Unemployment_Rate <= 5.2: 37 (18/13)
:   :   :   :   :       Unemployment_Rate > 5.2: 38 (72/28)
:   :   :   :   Source = amazon:
:   :   :   :   :...Monthly_Nominal_GDP_Index > 18161.75:
:   :   :   :       :...Monthly_Nominal_GDP_Index <= 18241.99: 30 (36/8)
:   :   :   :       :   Monthly_Nominal_GDP_Index > 18241.99: 34 (19/14)
:   :   :   :       Monthly_Nominal_GDP_Index <= 18161.75:
:   :   :   :       :...Unemployment_Rate > 5.6: 20 (17/10)
:   :   :   :           Unemployment_Rate <= 5.6:
:   :   :   :           :...Federal_Funds_Rate <= 0.11: 26 (36/13)
:   :   :   :               Federal_Funds_Rate > 0.11: 27 (21/10)
:   :   :   Unemployment_Rate <= 5.1:
:   :   :   :...Source = amazon:
:   :   :       :...Monthly_Real_GDP_Index <= 17520.27:
:   :   :       :   :...Monthly_Nominal_GDP_Index > 18346.29: 38 (58/43)
:   :   :       :   :   Monthly_Nominal_GDP_Index <= 18346.29:
:   :   :       :   :   :...Monthly_Nominal_GDP_Index <= 18286.02: 38 (18/11)
:   :   :       :   :       Monthly_Nominal_GDP_Index > 18286.02: 47 (36/20)
:   :   :       :   Monthly_Real_GDP_Index > 17520.27:
:   :   :       :   :...Federal_Funds_Rate > 0.37:
:   :   :       :       :...Monthly_Nominal_GDP_Index <= 18648.3: 51 (20/10)
:   :   :       :       :   Monthly_Nominal_GDP_Index > 18648.3: 53 (19/10)
:   :   :       :       Federal_Funds_Rate <= 0.37:
:   :   :       :       :...Federal_Funds_Rate <= 0.36: 41 (39/26)
:   :   :       :           Federal_Funds_Rate > 0.36:
:   :   :       :           :...Monthly_Nominal_GDP_Index <= 18584.12: 50 (18/11)
:   :   :       :               Monthly_Nominal_GDP_Index > 18584.12: 42 (19/13)
```

Decision Tree

- Variables:
  - Source
  - Monthly_Nominal_GDP_Index
  - Monthly_Real_GDP_Index
  - Unemployment_Rate
  - Federal_Funds_Rate
- Tree size: 236
- Key decision points: Source in {amazon, google}, Source in {facebook,apple,netflix}, and Monthly_Nominal_GDP_Index

Performance

- Accuracy rate significantly improved
  - Training dataset: 55% (0.5537044)
  - Testing dataset: 47% (0.4721907)
- Error rate
  - Training data: 44.6% (0.4462956)
  - Testing data: 52.27% (0.5278093)
- "Source" variable has an critical influence in refining the model's predictive capabilities

The Code of Training The Decision Tree

```r
# Define the target variable
train_set$Class <- as.factor(train_set$Class)

# Train a decision tree using C50
tree_model <- C5.0(Class ~ Source + Monthly_Nominal_GDP_Index + Monthly_Real_GDP_Index + Unemployment_Rate + Federal_Funds_Rate, data = train_set)
# tree_model <- C5.0(Class ~ Monthly_Nominal_GDP_Index + Monthly_Real_GDP_Index + Unemployment_Rate + Federal_Funds_Rate, data = train_set)

# display simple facts about the tree
tree_model

# display detailed information about the tree
summary(tree_model)

# Evaluating Model Performance
train_predictions <- predict(tree_model, train_set, type = "class")
test_predictions <- predict(tree_model, test_set, type = "class")

CrossTable(test_set$Class, test_predictions,
           prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
           dnn = c('actual default', 'predicted default'))

# Create confusion matrices
train_confusion <- table(train_predictions, train_set$Class)
test_confusion <- table(test_predictions, test_set$Class)

# Compute percentages of correctly and incorrectly classified samples
train_accuracy <- sum(diag(train_confusion)) / sum(train_confusion)
test_accuracy <- sum(diag(test_confusion)) / sum(test_confusion)

cat("Accuracy on Training Data:", train_accuracy, "\n")
cat("Accuracy on Testing Data:", test_accuracy, "\n")

# Calculate the error rate
train_error_rate <- 1 - train_accuracy
test_error_rate <- 1 - test_accuracy

cat("Error Rate on Training Data:", train_error_rate, "\n")
cat("Error Rate on Testing Data:", test_error_rate, "\n")
```

Boosting

- The training and testing accuracy rates remain relatively consistent
- The error rate on the test data also remains fairly stable, ranging from 51.53% to 52.21%
- The accuracy and error rates in bagging are not significantly affected by changes in trials

Trials	Average tree size	Accuracy on Training Data	Accuracy on Testing Data
1(original)	236	53% (0.5537044)	47% (0.4721907)
5	177.4	52% (0.5241701)	45% (0.4517594)
10	164.8	51% (0.5196264)	45% (0.4517594)
30	159.9	51% (0.5183643)	44% (0.4494892)
50	158.9	51% (0.5183643)	44% (0.4494892)
100	158.3	51% (0.5177332)	44% (0.4483541)

Evaluation on Boosting

Trial	Decision Tree		
	Size	Errors	
0	236	3536(44.6%)	
1	168	3823(48.3%)	
2	166	3827(48.3%)	
3	165	3822(48.2%)	
4	152	4011(50.6%)	
boost		3768(47.6%)	<<

Trial	Decision Tree		
	Size	Errors	
0	236	3536(44.6%)	
1	168	3823(48.3%)	
2	166	3827(48.3%)	
3	166	3827(48.3%)	
4	166	3827(48.3%)	
5	166	3827(48.3%)	
6	163	3839(48.5%)	
7	153	3890(49.1%)	
8	146	4023(50.8%)	
9	118	4185(52.8%)	
boost		3806(48.0%)	<<

Trial	Decision Tree	
	Size	Errors
0	236	3536(44.6%)
1	168	3823(48.3%)
2	166	3827(48.3%)
3	166	3827(48.3%)
4	166	3827(48.3%)
5	166	3827(48.3%)
6	166	3827(48.3%)
7	166	3827(48.3%)
8	166	3827(48.3%)
9	166	3827(48.3%)
10	166	3827(48.3%)
11	166	3827(48.3%)
12	166	3827(48.3%)
13	166	3827(48.3%)
14	166	3827(48.3%)
15	166	3827(48.3%)
16	166	3827(48.3%)
17	165	3834(48.4%)
18	166	3830(48.3%)
19	163	3858(48.7%)
20	157	3890(49.1%)
21	157	3941(49.7%)
22	149	3954(49.9%)
23	155	3956(49.9%)
24	155	3999(50.5%)
25	146	4136(52.2%)
26	130	4157(52.5%)
27	126	4204(53.1%)
28	119	4174(52.7%)
29	115	4336(54.7%)
boost		3816(48.2%)

Trial	Decision Tree	
	Size	Errors
0	236	3536(44.6%)
1	168	3823(48.3%)
2	166	3827(48.3%)
3	166	3827(48.3%)
4	166	3827(48.3%)
5	166	3827(48.3%)
6	166	3827(48.3%)
7	166	3827(48.3%)
8	166	3827(48.3%)
9	166	3827(48.3%)
10	166	3827(48.3%)
11	166	3827(48.3%)
12	166	3827(48.3%)
13	166	3827(48.3%)
14	166	3827(48.3%)
15	166	3827(48.3%)
16	166	3827(48.3%)
17	166	3827(48.3%)
18	166	3827(48.3%)
19	166	3827(48.3%)
20	166	3827(48.3%)
21	166	3827(48.3%)
22	166	3827(48.3%)
23	166	3827(48.3%)
24	166	3827(48.3%)
25	166	3827(48.3%)
26	166	3827(48.3%)
27	165	3827(48.3%)
28	168	3820(48.2%)
29	164	3831(48.4%)
30	166	3837(48.4%)
31	164	3856(48.7%)
32	163	3862(48.7%)
33	160	3892(49.1%)
34	157	3886(49.0%)
35	156	3929(49.6%)
36	155	3947(49.8%)
37	158	3983(50.3%)
38	155	3985(50.3%)
39	152	4025(50.8%)
40	157	3996(50.4%)
41	153	4092(51.6%)
42	142	4137(52.2%)
43	134	4184(52.8%)
44	128	4169(52.6%)
45	120	4296(54.2%)
46	117	4267(53.9%)
47	114	4252(53.7%)
48	119	4271(53.9%)
49	122	4238(53.5%)
boost		3816(48.2%)

Random Forests

- Training Data
  - Accuracy rates increased slightly from 55.79% with 10 trees to 56.22% with 200 trees
  - Error rates decreased slightly from 44.21% with 10 trees to 43.78% with 200 trees
- Testing Data
  - Accuracy rate remain relatively stable
  - Error rate show a decreasing trend from 51.99% with 10 trees to 48.52% with 200 trees
- Random Forest v.s Bagging
  - No significant improvement with bagging
  - Random forests show a slight improvement in the accuracy of the training data but the accuracy of the testing data remain stable

Trees	Accuracy Rate on Training Data	Error Rate on Training Data	Accuracy Rate on Testing Data	Error Rate on Testing Data
10	0.557869	0.442130	0.480136	0.519863
50	0.561403	0.438596	0.480136	0.519863
100	0.561529	0.438470	0.482406	0.517593
200	0.562160	0.437839	0.485811	0.514188

Comparative Analysis

- **Linear regression models** showed high predictive quality, with R-squared values close to 1 and low mean squared error. The multivariate linear regression model performed the best.
- **Logistic regression** had moderate predictive performance based on the AUC metric, slightly better than random guessing.
- **Naive Bayes classifier** performed similar with or without Laplace smoothing, suggesting it was not very effective for this dataset.
- **Boosting the decision tree** did not meaningfully improve accuracy. **Bagging and random forests** also showed limited improvements.
- Using the **multivariate linear regression** model for future prediction