Credit Risk Project

Presented by Yu-Shiang Liao (Shawn)

Outline

01

Business Background

02

Data Highlights

03

Model Description

04

Findings & Insights

01

Business Background

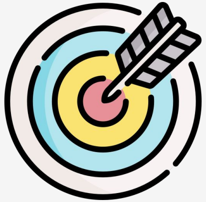
Business Background

Business Problem

Optimize the credit extension decision process to improve overall performance and secure a competitive advantage.

Project Goal

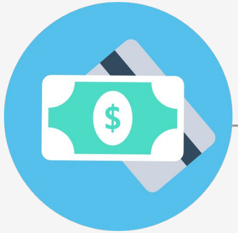
Develop a classification regression model to predict whether a new lender will default.



02

Data Highlights

Data Source



INDEPENDENT	6 Categorical Variables	CHK_ACCT, HISTORY, EMPLOYMENT, JOB
	6 Numerical Variables	DURATION, AMOUNT, AGE, NUM_CREDITS
	18 Dummy Variables	NEW_CAR, EDUCATION, RENT, GUARANTOR
DEPENDENT	Binary Variables	DEFAULT (70% of 0, 30% of 1)

Data Cleansing & Preparation

1 Check if there's any null value in the data set.

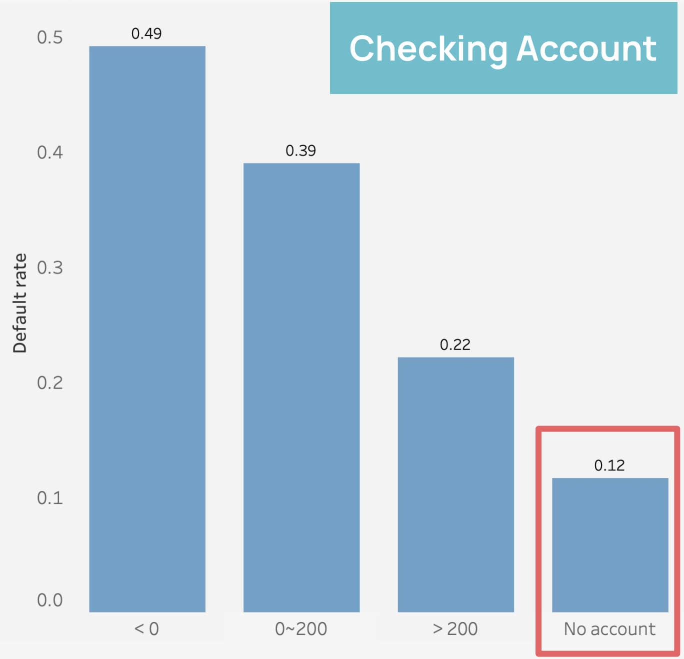
```{r}
na_count <-sapply(df, function(y) sum(length(which(is.na(y)))))
na_count <- data.frame(na_count)
sum(na_count)
```

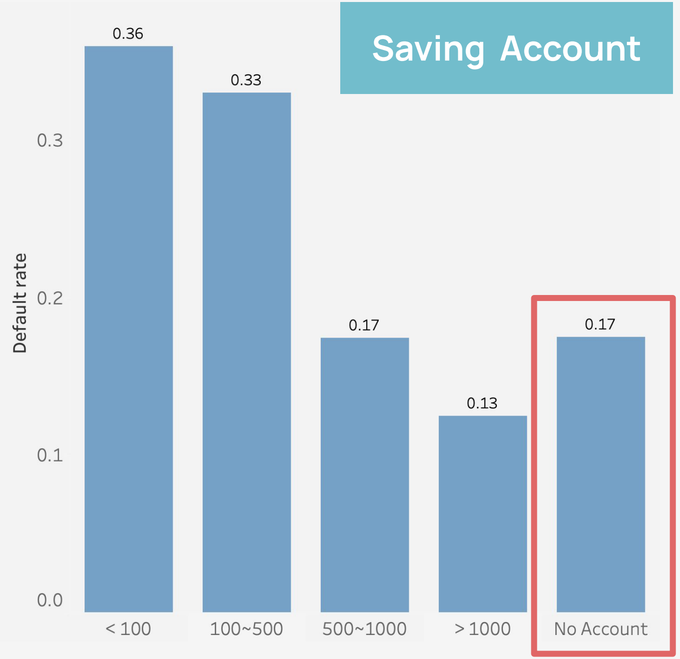
```
[1] 0
```

2 Delete **OBS#** column that cannot be used to classify risk of default.

3 Transform categorical variables into **factor**. Split data into training and validation samples.

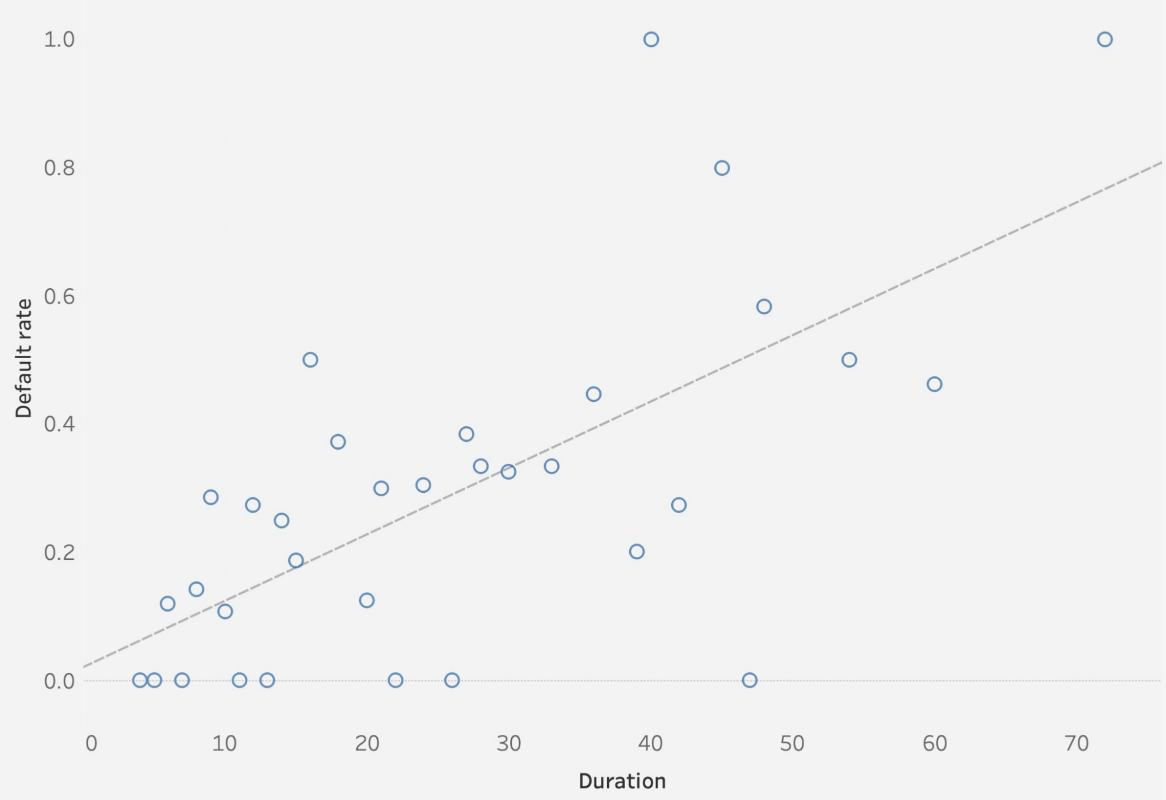
Default rate decreases as the account balance increases.





- One surprising finding is that default rate for those without an account is quite low.

Default rate becomes higher when duration of credit is higher.



Correlation: 0.67

Comparing binary variables with default rate:

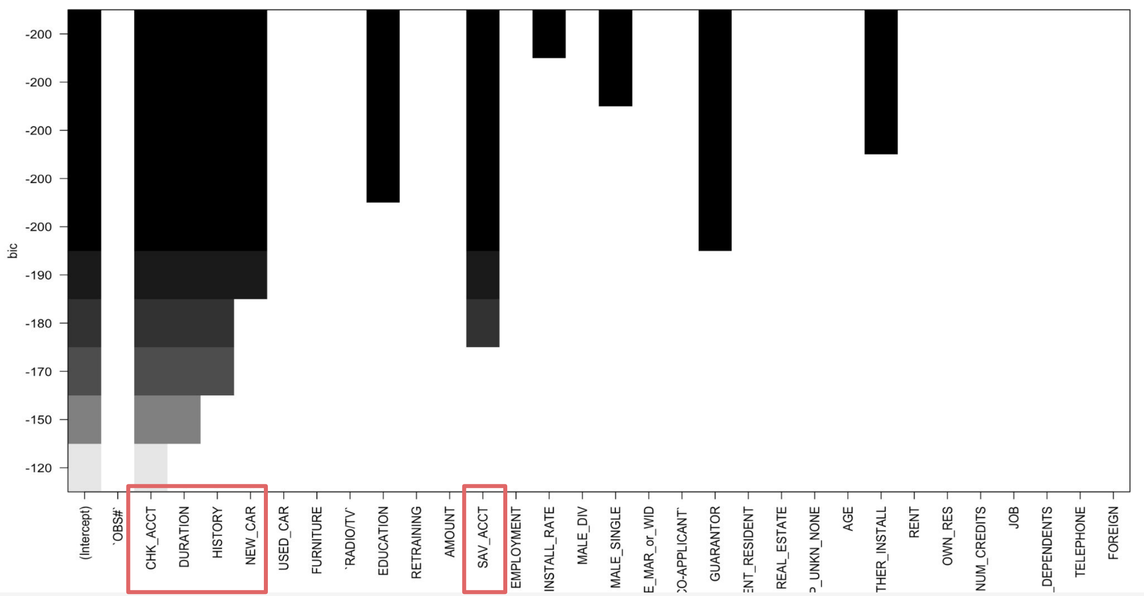
Purpose of Credit	Default Rate
New Car	0.38
Used Car	0.17
Furniture	0.32
Radio/TV	0.22
Education	0.44
Retraining	0.35

If the credit is intended for a new car or education, default rates tend to increase.

Variable	No	Yes
Guarantor	0.3	0.19
Co-applicant	0.29	0.44
Real Estate	0.33	0.21
No property	0.28	0.44
Other Installment	0.28	0.41
Rent	0.28	0.4
Own Residence	0.4	0.26
Telephone	0.31	0.28
Foreign Worker	0.31	$\boxed{0.11}$

Foreign workers tend to have low default rate.

Selecting best subset of features



- Adopted Best Subset Method with the selection criteria of BIC.
- Checking Account, Duration, History, New Car, and Saving Account are the top features.

03

Model Description

```
model_Linear <- lm(DEFAULT ~., data = train_Linear)
```

Model 1: Linear Probability Model

	GVIF	Df	GVIF^(1/(2*Df))
CHK_ACCT	1.459848	3	1.065086
DURATION	2.151960	1	1.466956
HISTORY	2.380836	4	1.114529
NEW_CAR	4.536855	1	2.129989
USED_CAR	2.947022	1	1.716689
FURNITURE	4.215161	1	2.053086
RADIO.TV	4.934990	1	2.221484
EDUCATION	1.998659	1	1.413739
RETRAINING	2.848484	1	1.687745
AMOUNT	2.587720	1	1.608639
SAV_ACCT	1.423551	4	1.045133
EMPLOYMENT	2.717334	4	1.133099
INSTALL_RATE	1.375517	1	1.172825
MALE_DIV	1.212195	1	1.100997
MALE_SINGLE	1.670193	1	1.292359
MALE_MAR_or_WID	1.276536	1	1.129839
CO.APPLICANT	1.093733	1	1.045817
GUARANTOR	1.137170	1	1.066382
PRESENT_RESIDENT	1.727825	3	1.095427
REAL_ESTATE	1.279959	1	1.131353
PROP_UNKN_NONE	2.885415	1	1.698651
AGE	1.551330	1	1.245524
OTHER_INSTALL	1.173354	1	1.083215
RENT	4.615894	1	2.148463
OWN_RES	6.019400	1	2.453446
NUM_CREDITS	1.622428	1	1.273746
JOB	2.472310	3	1.162832
NUM_DEPENDENTS	1.219145	1	1.104149
TELEPHONE	1.341291	1	1.158141
FOREIGN	1.105813	1	1.051577

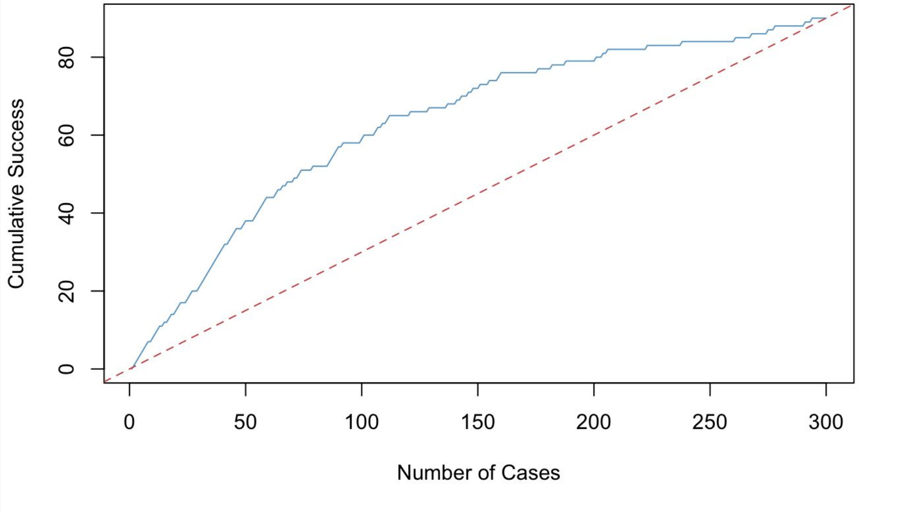
Compute GVIF to check multicollinearity

- GVIF ^ (1 / (2 * DF)) value of all variables are less than 5. Thus, multicollinearity is not a problem in this project.
- CHK_ACCT3 and SAV_ACCT4 are the most significant to the model.
- Model's accuracy was 0.783, AUC was 0.805.

```
model_Logi <- glm(DEFAULT ~ ., data = train_Logi, family = "binomial" )
```

Model 2: Logistic Regression Model

Gain Chart



- Just like the Linear Probability Model, CHK_ACCT3 and SAV_ACCT4 are the most significant for the model.
- Model's accuracy was 0.767, AUC was 0.792.

```
model_NB <- naiveBayes(DEFAULT~., data=train_NB)
```

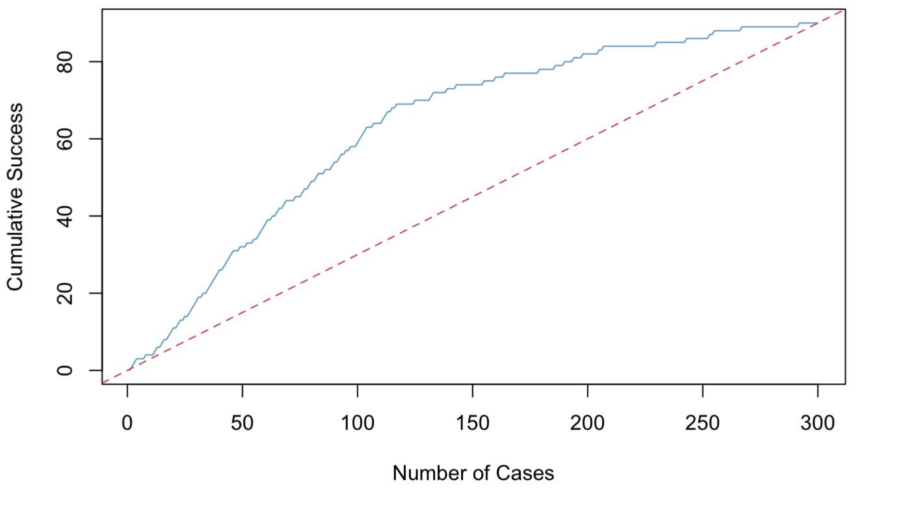
Model 3: Naïve Bayes Algorithm

- Transformed categorical variables into **factors**. Otherwise, the algorithm will fit a normal distribution to the data for conditional probabilities.

- Given that the person defaulted on loan, the probability that the checking account balance is less than $0 is 0.4667.

```
> model_NB[2]$tables$CHK_ACCT
   CHK_ACCT
Y            0          1          2          3
  0 0.20204082 0.23673469 0.06530612 0.49591837
  1 0.46666667 0.32380952 0.05238095 0.15714286
```

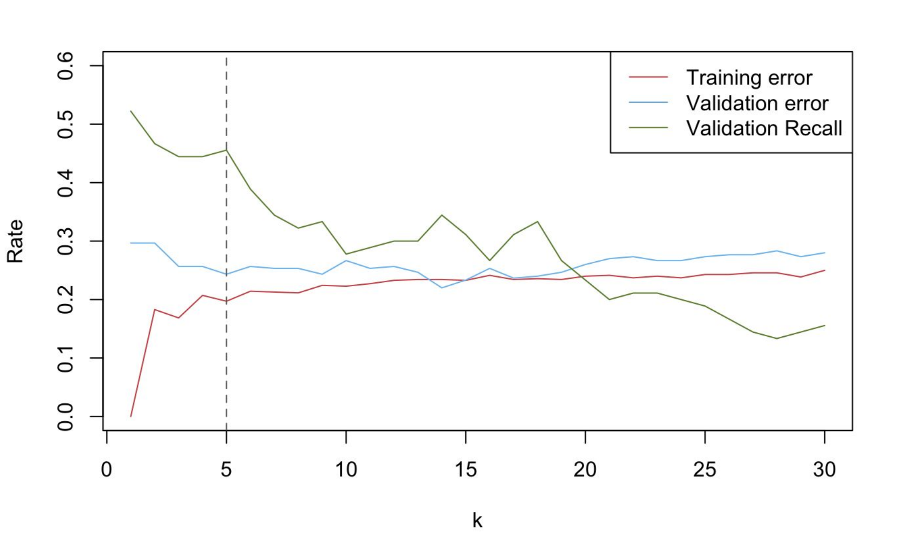
Gain Chart



```
prediction_best_k <- knn(train_input, validate_input,train_output, k=5)
```

Model 4: K-nearest Neighbors Algorithm

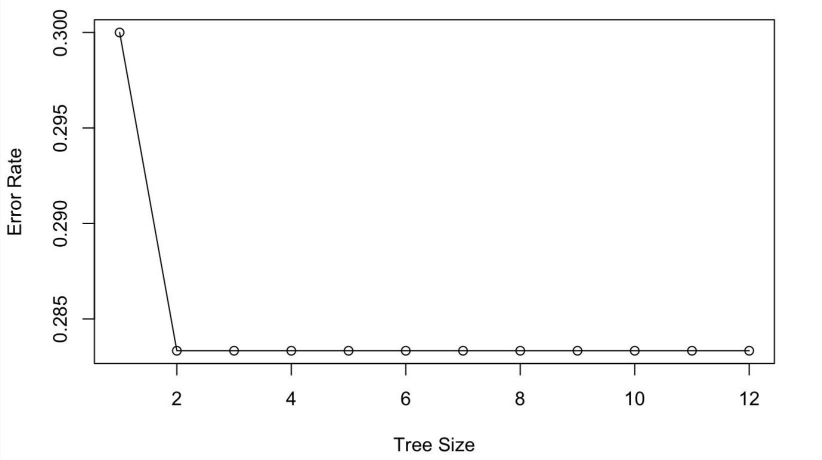
Determining the best K

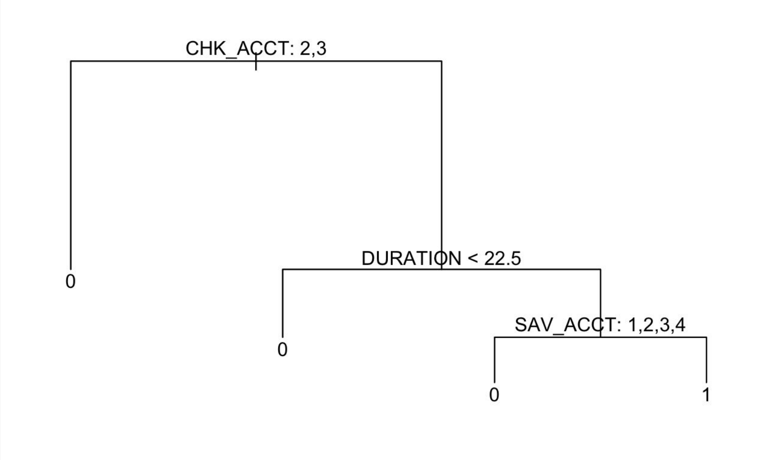


- Transformed categorical predictors into **dummy variables**.
- Due to a slight imbalance in the data, I considered both the error rate and sensitivity and built the model with k = 5.
- Model's accuracy was **0.76**, AUC was **0.746**.

```
prune.credit=prune.misclass(tree.credit,best=which.min(Error))
```

Model 5: Classification Tree





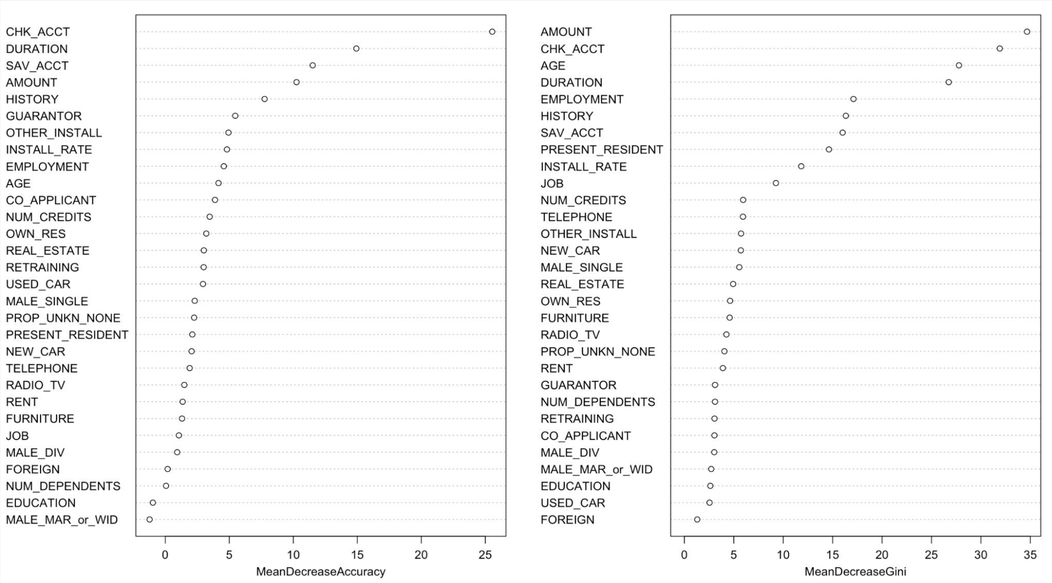
- **Prune the tree** by selecting the tree size with the lowest test error rate to avoid overfitting on training data.
- **Checking Account, Duration, and Saving Account** are the selected predictors.

```
bag.credit=randomForest(DEFAULT~.,data=Credit,subset=train,
                        mtry=best.m,importance=TRUE)
```

Model 6: Random Forest

```
> mtry
        mtry OOBError
4.OOB      4    0.241
5.OOB      5    0.237
7.OOB      7    0.228
10.OOB    10    0.242
```

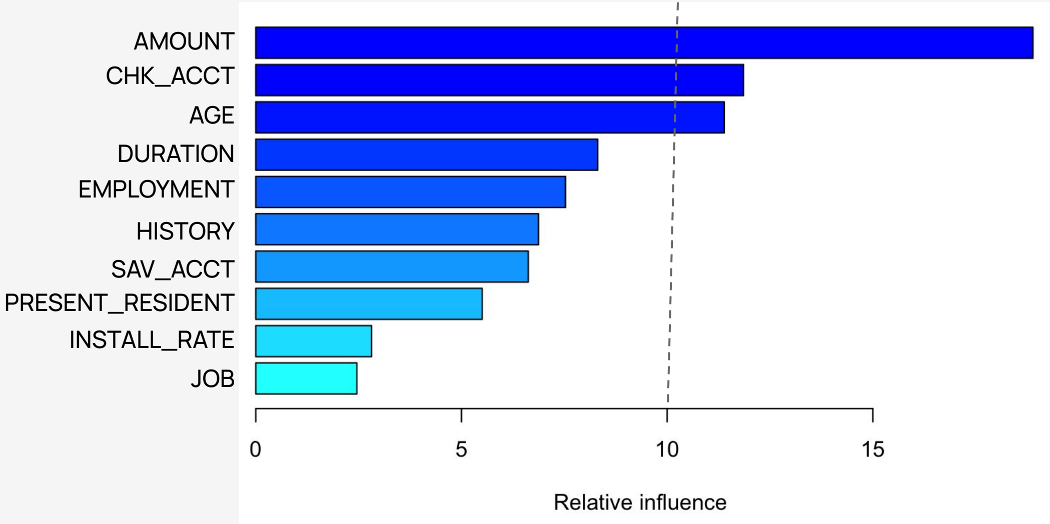
- Select the best mtry value with **minimum out-of-bag (OOB) error.**



- **Checking Account, Duration, Amount, and Saving Account** are the most significant variables.

```
boost.credit=gbm(DEFAULT~.,data=Credit_boost[train,],distribution="bernoulli",
        n.trees=500,shrinkage=0.1,interaction.depth=4)
```

Model 7: Generalized Boosted Model



Top 10 relatively important variables

- **Amount, Checking Account, Age** are the most important variables.
- Shrinkage, interaction.depth, and n.trees have been optimized manually by lowering validation error.
- Model's accuracy was **0.787**, AUC was **0.803**.

```r
bst <- xgboost(data = train_x, label = train_y, max.depth = 8, eta = 0.1,
               nround = 66, objective = "binary:logistic", eval_metric="error",
               verbose = 0)
```

Model 8: XGBoost

```
[58]	train-logloss:0.249338	test-logloss:0.482142
[59]	train-logloss:0.246341	test-logloss:0.480160
[60]	train-logloss:0.244666	test-logloss:0.479886
[61]	train-logloss:0.243282	test-logloss:0.480558
[62]	train-logloss:0.240792	test-logloss:0.479452
[63]	train-logloss:0.239009	test-logloss:0.479197
[64]	train-logloss:0.237003	test-logloss:0.481301
[65]	train-logloss:0.234495	test-logloss:0.478705
[66]	train-logloss:0.231906	test-logloss:0.475417
[67]	train-logloss:0.230245	test-logloss:0.476000
[68]	train-logloss:0.227675	test-logloss:0.477868
[69]	train-logloss:0.227019	test-logloss:0.479741
[70]	train-logloss:0.224611	test-logloss:0.480213
[71]	train-logloss:0.223082	test-logloss:0.480340
[72]	train-logloss:0.220917	test-logloss:0.478716
[73]	train-logloss:0.218477	test-logloss:0.480927
[74]	train-logloss:0.216508	test-logloss:0.480423
[75]	train-logloss:0.214362	test-logloss:0.481634
[76]	train-logloss:0.212332	test-logloss:0.480511
[77]	train-logloss:0.211509	test-logloss:0.481158
[78]	train-logloss:0.209271	test-logloss:0.483142
[79]	train-logloss:0.208777	test-logloss:0.483499
[80]	train-logloss:0.207024	test-logloss:0.485614
[81]	train-logloss:0.204189	test-logloss:0.492382
[82]	train-logloss:0.201851	test-logloss:0.497612
[83]	train-logloss:0.200486	test-logloss:0.497701
[84]	train-logloss:0.199171	test-logloss:0.497673
[85]	train-logloss:0.196828	test-logloss:0.497096
```

- Fit the model and display training and testing data in each of the 200 rounds.
- Minimum testing Log-loss is achieved at 66 rounds. Beyond this point, the number begins to increase, which could be a sign of overfitting.
- Model's accuracy was 0.787, AUC was 0.803.

Combined Model Method

Combine prediction result from multiple models

Exclude result from **Classification Tree** due to poor performance

Plurality voting

1st

Linear Probability	Logistic Regression	Naive Bayes	KNN	Random Forest	Boosting	XGBoost	Combined Model
0	0	1	0	1	1	1	1

⋮

300

Linear Probability	Logistic Regression	Naive Bayes	KNN	Random Forest	Boosting	XGBoost	Combined Model
1	0	0	1	0	0	0	0

04

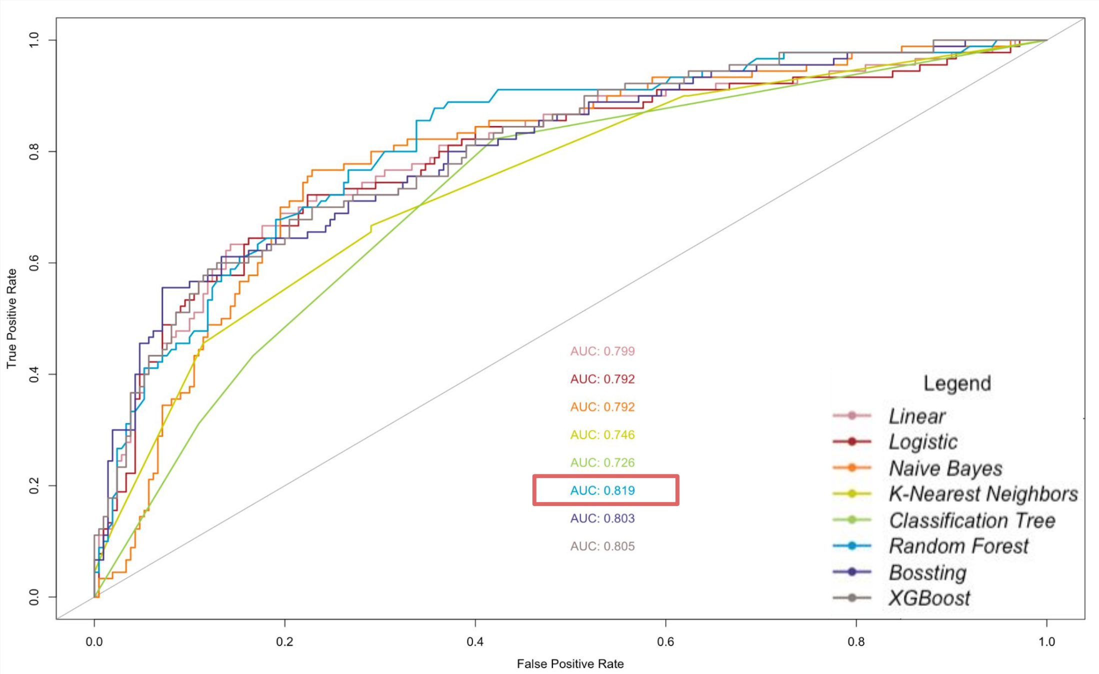
Findings & Insights

Model Evaluation

Model Name	Accuracy	Precision	Recall	F1 Score
Linear Probability	0.783	0.662	0.567	0.61
Logistic Regression	0.767	0.62	0.578	0.598
Naïve Bayes	0.757	0.586	0.644	0.614
K-nearest Neighbors	0.757	0.631	0.456	0.529
Classification Tree	0.717	0.549	0.311	0.397
Random Forest	0.78	0.722	0.433	0.542
Generalized Boosted Model	0.787	0.671	0.567	0.615
XGBoost	0.79	0.68	0.567	0.618
Combined Model	0.8	0.708	0.567	0.63

The worst!

ROC Curves



- **Classification Tree** and **K-Nearest Neighbors** have the lowest AUC
- **Ensemble Methods** have higher AUC, which are all greater than 0.8

Conclusions

Classification Tree performs poorly in predicting the likelihood of default.

In terms of Area Under Curve (AUC), **Random Forest** performs the best.

XGBoost has the highest accuracy and F1 Score, which is overall the best model.

Saving Account, **Amount**, **Checking Account**, and **Duration** are important predictors of default.

Thank you!