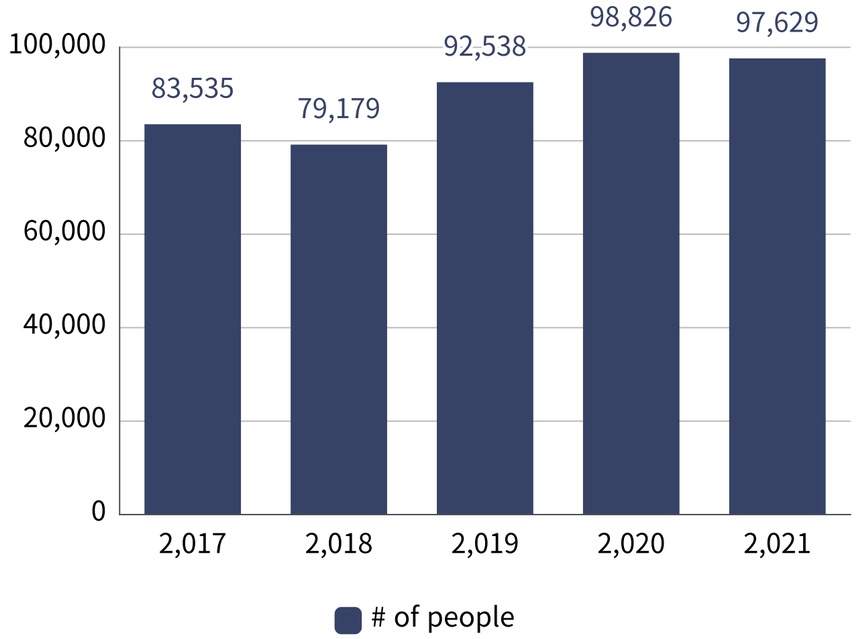
Predictive Analytics for Insurance Fraud Detection

Kunyoon Kim

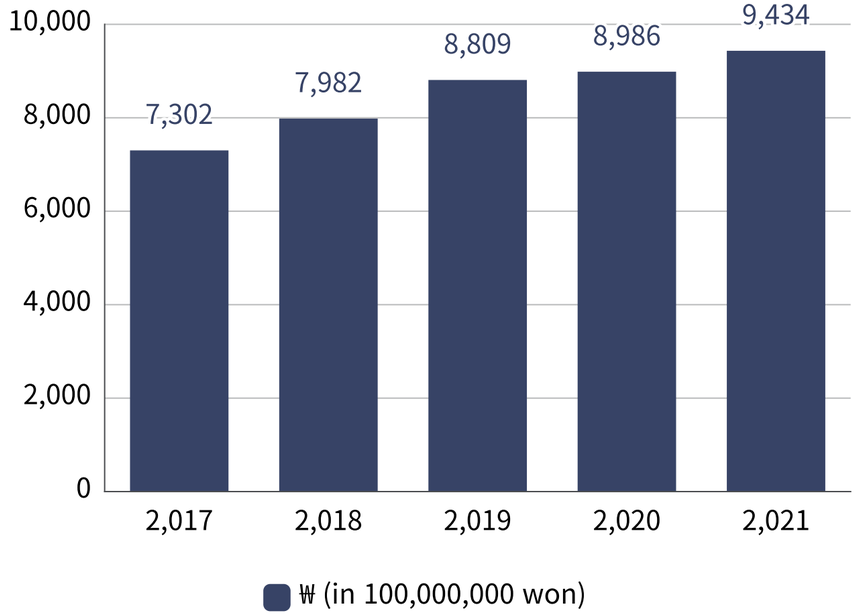
CONTENTS

Objective

Number of people caught in insurance fraud



Amount of insurance fraud detected



(Financial Supervisory Service, 2022)

- 5 trillion KRW in insurance fraud detections in Korea over the past five years
- Increasing trends in both number of fraudsters and amount of damage
- Non-life insurance (4.556 trillion won) was damaged significantly larger than life insurance (384 billion won)
  - ○ common schemes: false injuries, staged accidents, self-inflicted injuries, multiple claims, etc.

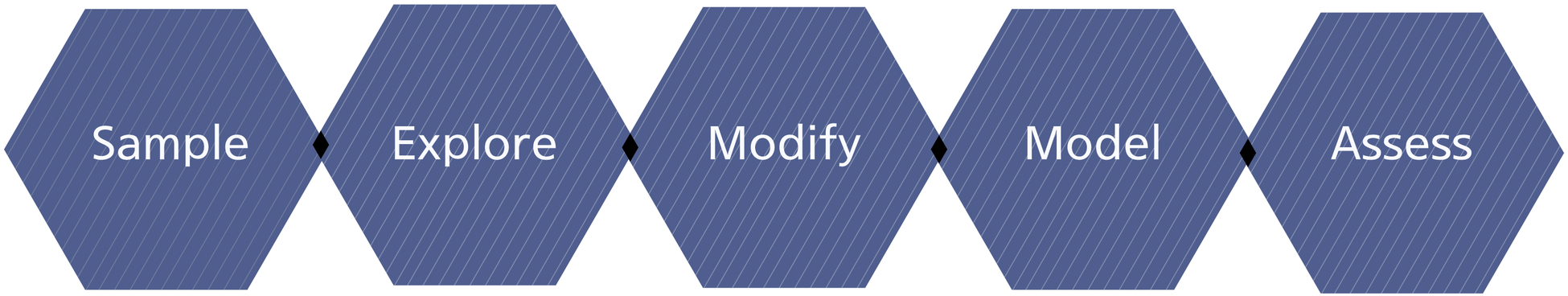
Objective

Objective	To classify people who are likely to commit insurance fraud
Plan Point	Insurance fraud prediction
Goal	To increase fraud prediction rate
Analytic	Classification
Assess Criteria	F-measure
Required Data	Customer and Claim Data provided by H Insurance

SEMMA Methodology

visualization and basic description of the data

use variety of statistical and machine learning models



generate a representative sample of the data

select variables, transform variable representations

evaluate the accuracy and usefulness of the models

Data Preview

No.	Variable	Variable Type	Explanation
1	CUST_ID	N	Unique ID of each customer (provider)
2	DIVIDED_SET	N	1: training set // 2: test set
3	SIU_CUST_YN	C	Whether fraud (Y/N) // N/A for test set
4	SEX	N	1: male // 2: female
5	AGE	N	Customer's age
6	RESI_CODE	N	Estimated cost of the customer's house
7	RESI_TYPE_CODE	C	Type of the provider's house - 11: detached house, 12: multi-household house, 13: detached house with shop, 20: condo, 30: flat, 40: shopping mall, 50: studio apartment, 60: accommodation, 70: residence, 99: etc.
8	FP_CAREER	C	FP career (Y/N)
9	CUST_RGST	C	Customer's first registration date
10	CTPR	C	Customer's city & province
11	OCCP_GRP1	C	Code classified into 8 occupational groups
12	OCCP_GRP2	C	Code classified into 25 occupational groups
13	TOTALPREM	N	Total premium paid
14	MINCRDT	C	Minimum credit rating
15	MAXCRDT	C	Maximum credit rating
16	WEDD_YN	C	Marital status (Y/N)
17	MATE_OCCP_GRP1	C	Spouse occupational code (8 groups)
18	MATE_OCCP_GRP2	C	Spouse occupational code (25 groups)
19	CHLD_CNT	N	Number of children
20	LTBN_CHLD_AGE	N	Age of the youngest child
21	MAX_PAYM_YM	C	Year & Month of paying the largest premium
22	MAX_PRM	N	Maximum monthly premium paid
23	CUST_INCM	N	Customer estimated income
24	RCBASE_HSHD_INCM	N	Household income estimated by the customer's house price
25	JPBASE_HSHD_INCM	N	Household income estimated by customer's occupation and premium level

Customer Data (CUST_DATA) :
– 22,400 rows, 25 columns
– Wide format

Data Preview

No.	Variable	Variable Type	Explanation
1	CUST_ID	N	Unique ID of each customer (provider)
2	POLY_NO	N	Insurance contract number
3	ACCI_OCCP_GRP1	C	Code classified into 8 occupational groups
4	ACCI_OCCP_GRP2	C	Code classified into 25 occupational groups
5	CHANG_FP_YN	C	Whether the financial planner changed or not
6	CNTT_RECP_SQNO	C	Contract receipt serial number
7	RECP_DATE	C	Date of incident
8	ORIG_RESN_DATE	C	Date of accident
9	RESN_DATE	C	Insurance payment date
10	CRNT_PROG_DVSN	C	Current progress - Reception (11), Examination Assignment (21), Examination (22), Examination Approval (23), Investigation (32), Investigation Approval (33)
11	ACCI_DVSN	C	Cause of accident - Disaster(1), traffic accident(2), disease(3)
12	CAUS_CODE	C	Accident cause code
13	CAUS_CODE_DTAL	C	Accident cause code (in details)
14	DSAS_NAME	C	Disease name
15	DMND_RESN_CODE	C	Reason code causing the claim for payment - Death(01), Hospitalization(02), Outpatient Hospital(3), Disability(04), Surgery(05), Diagnosis(06), Treatment(07), Termination/Invalidation(09)
16	CMND_RSCD_SQNO	N	Claim reason code
17	HOSP_OTPA_STDT	C	Hospitalization/outpatient start date
18	HOSP_OTPA_ENDT	C	Hospitalization/outpatient end date
19	RESL_CD1	C	Result code for the cause of the accident
20	RESL_NM1	C	Result
21	VLID_HOSP_OTDA	N	Effective number of hospitalization/outpatient days
22	HOUSE_HOSP_DIST	N	Distance from customer residence to hospital
23	HOSP_CODE	N	Hospital code
24	ACCI_HOSP_ADDR	C	Hospital regional code
25	HOSP_SPEC_DVSN	C	Type of hospital
26	CHME_LICE_NO	N	Physician license number
27	PAYM_DATE	C	Insurance payment date
28	DMND_AMT	N	Insurance claim amount
29	PAYM_AMT	N	Insurance payment amount
30	PMMI_DLNG_YN	C	Loss insurance (Y/N)
31	SELF_CHAM	N	Co-payment after applying National Health Care
32	NON_PAY	N	Amount not covered by health insurance
33	TAMT_SFCA	N	Total amount not covered by health insurance
34	PATT_CHRG_TOTA	N	Total patient charged amount
35	DSCT_AMT	N	Hospital discount amount
36	COUNT_TRMT_ITEM	N	Number of medical courses
37	NON_PAY_RATIO	N	Non-paid and paid ratio
38	HEED_HOSP_YN	C	Whether the hospital is monitored by the Financial Supervisory Service

Insurance Claim Data (CLAIM_DATA) :
– 119,020 rows, 39 columns
– Long format

Age → Age Group

CUST_ID	DIVIDED_SET	SIU_CUST_YN	SEX	AGE	RESI_COST
1	1	N	2	47	21111
2	1	N	1	53	40000
3	1	N	1	60	0
4	1	N	2	64	12861
5	1	N	2	54	0
6	1	N	1	62	6218
7	1	Y	2	60	11388
8	1	N	1	57	86527
9	1	N	1	54	22638
12	1	N	1	58	37222
13	1	Y	1	63	8140
15	1	N	1	59	23055
16	1	N	2	52	6684
17	1	N	1	68	14027
18	1	N	1	48	33333

CUST_ID	DIVIDED_SET	SIU_CUST_YN	SEX	AGE	RESI_COST
1	1	0	2	4	21111
2	1	0	1	5	40000
3	1	0	1	6	0
4	1	0	2	6	12861
5	1	0	2	5	0
6	1	0	1	6	6218
7	1	1	2	6	11388
8	1	0	1	5	86527
9	1	0	1	5	22638
12	1	0	1	5	37222
13	1	1	1	6	8140
15	1	0	1	5	23055
16	1	0	2	5	6684
17	1	0	1	6	14027
18	1	0	1	4	33333

code:

```
age_to_gen <- function(row){
  row = floor(row/10)
}
data_cust$AGE <- sapply(data_cust$AGE, age_to_gen)
```

Missing Values in Martial Status

TOTALPREM	MINCRDT	MAXCRDT	WEDD_YN
NA	7	8	N
12372648	3	6	Y
1729850	NA	NA	
27594800	NA	NA	Y
1700876	NA	NA	N
1659104	NA	NA	N
6095040	NA	NA	N
3523480	NA	NA	
5622009	NA	NA	
38706064	NA	NA	Y
1789500	NA	NA	
24858168	NA	NA	
1295500	NA	NA	
20741427	NA	NA	N
18544722	NA	NA	Y
15098742	6	6	N

TOTALPREM	MINCRDT	MAXCRDT	WEDD_YN
NA	7	8	N
12372648	3	6	Y
1729850	NA	NA	N
27594800	NA	NA	Y
1700876	NA	NA	N
1659104	NA	NA	N
6095040	NA	NA	N
3523480	NA	NA	N
5622009	NA	NA	N
38706064	NA	NA	Y
1789500	NA	NA	N
24858168	NA	NA	N
1295500	NA	NA	N
20741427	NA	NA	N
18544722	NA	NA	Y
15098742	6	6	N

→ Single people are more likely to leave blank in marital status
→ Replace 'N/A' with 'N'

code:

```
null_to_N <- function(row){
  if(row=="N"){
    row = "N"
  } else if (row == "Y") {
    row = "Y"
  } else {
    row = "N"
  }
}
data_cust$WEDD_YN <- sapply(data_cust$WEDD_YN, null_to_N)
```

Y/N → 1/0

WEDD_YN	SIU_CUST_YN	FP_CAREER
Y	N	N
Y	N	N
N	N	N
N	N	Y
Y	N	Y
Y	N	N
Y	Y	N
Y	N	Y
Y	N	N
Y	N	N
Y	Y	N
Y	N	N
Y	N	N
Y	N	N
Y	N	N

WEDD_YN	SIU_CUST_YN	FP_CAREER
1	0	0
1	0	0
0	0	0
0	0	1
1	0	1
1	0	0
1	1	0
1	0	1
1	0	0
1	0	0
1	1	0
1	0	0
1	0	0
1	0	0
1	0	0

code:

```
yn_to_10 <- function(row){
  if(row=="Y")
    row = 1
  else if(row=="N")
    row = 0
  else
    row =""
}
data_cust$SIU_CUST_YN <- sapply(data_cust$SIU_CUST_YN, yn_to_10)
data_cust$FP_CAREER <- sapply(data_cust$FP_CAREER, yn_to_10)
data_cust$WEDD_YN <- sapply(data_cust$WEDD_YN, yn_to_10)
```

NULL → 0

RESI_TYPE_CODE	TOTALPREM
40	53792972
30	110373302
NA	*NA*
30	30132998
20	86678063
13	19038547
20	38270564
99	15267626
99	9854625
11	24116656
20	254732084
20	25613176
11	*NA*
11	38213095
99	5027013

RESI_TYPE_CODE	TOTALPREM
40	53792972
30	110373302
0	0
30	30132998
20	86678063
13	19038547
20	38270564
99	15267626
99	9854625
11	24116656
20	254732084
20	25613176
11	0
11	38213095
99	5027013

code:

```
na_to_0 <- function(row){
  if(is.na(row))
    row=0
  else
    row = row
}
data_cust$RESI_TYPE_CODE <- sapply(data_cust$RESI_TYPE_CODE, na_to_0)
data_cust$TOTALPREM <- sapply(data_cust$TOTALPREM, na_to_0)
```

NULL → 6 (avg. value)

MINCRDT	MAXCRDT
NA	NA
1	6
NA	NA
2	99
8	8
6	6
NA	NA
NA	NA
6	6
6	8
6	7
NA	NA
6	6
NA	NA
10	8

MINCRDT	MAXCRDT
6	6
1	6
6	6
2	99
8	8
6	6
6	6
6	6
6	6
6	8
6	7
6	6
6	6
6	6
10	8

code:

```
na_to_6 <- function(row){
  if(is.na(row))
    return (6)
  else
    return (row)
}
data_cust$MINCRDT <- sapply(data_cust$MINCRDT, na_to_6)
data_cust$MAXCRDT <- sapply(data_cust$MAXCRDT, na_to_6)
```

Name → Code

CTPR	OCCP_GRP_1	MATE_OCCP_GRP_1
충북	3.사무직	3.사무직
서울	3.사무직	1.주부
서울	5.서비스	
경기	2.자영업	
광주	2.자영업	3.사무직
충남	3.사무직	1.주부
서울	5.서비스	
서울	2.자영업	2.자영업
서울	4.전문직	4.전문직
서울	4.전문직	4.전문직
경기	6.제조업	5.서비스
서울	3.사무직	1.주부
전북	1.주부	3.사무직
대구	5.서비스	1.주부
경기	5.서비스	3.사무직

CTPR	OCCP_GRP_1	MATE_OCCP_GRP_1
18	3	3
10	3	1
10	5	0
3	2	0
6	2	3
17	3	1
10	5	0
10	2	2
10	4	4
10	4	4
3	6	5
10	3	1
15	1	3
7	5	1
3	5	3

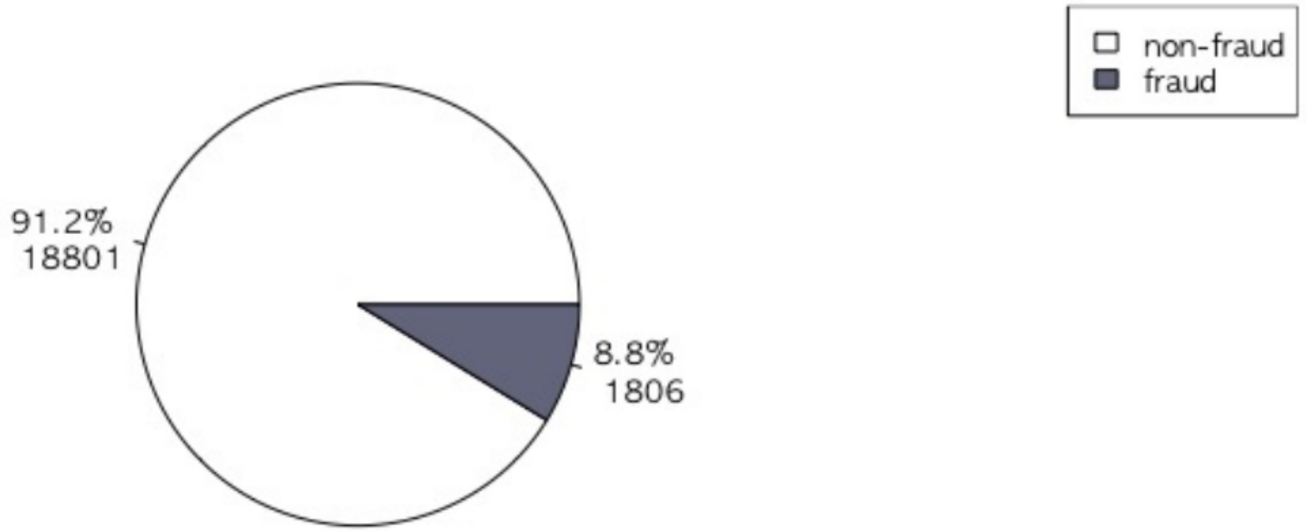
code:

```
# region name -> code
data_cust$CTPR <- as.factor(data_cust$CTPR)
is.factor(data_cust$CTPR)
area_name <- levels(data_cust$CTPR)
data_cust$CTPR <- as.numeric(data_cust$CTPR)
```

```
# occupation name -> code
occp_to_no <- function(row){
  row = substr(row,1,1)
  if(row == " "){
    return(0)
  } else {
    return (as.integer(row))
  }
}
data_cust$OCCP_GRP_1 <- sapply(data_cust$OCCP_GRP_1, occp_to_no)
data_cust$MATE_OCCP_GRP_1 <- sapply(data_cust$MATE_OCCP_GRP_1, occp_to_no)
```

Fraud Rate

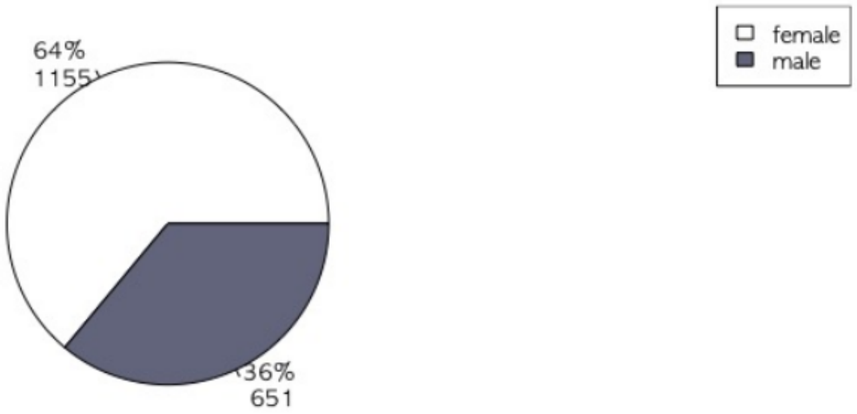
Number of Fraudulent Providers



Total: 22,400
> Non-fraud cases: 18,801 (91.2%)
> Fraud cases: 1,806 (8.8%)

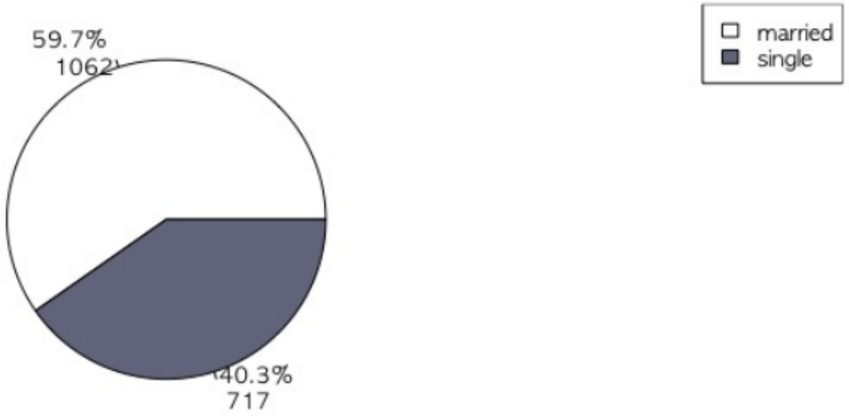
Fraud Rate by Gender and Marital Status

Fraud Rate by Gender



> Female: 1,155 (64.0%)
> Male: 651 (36.0%)

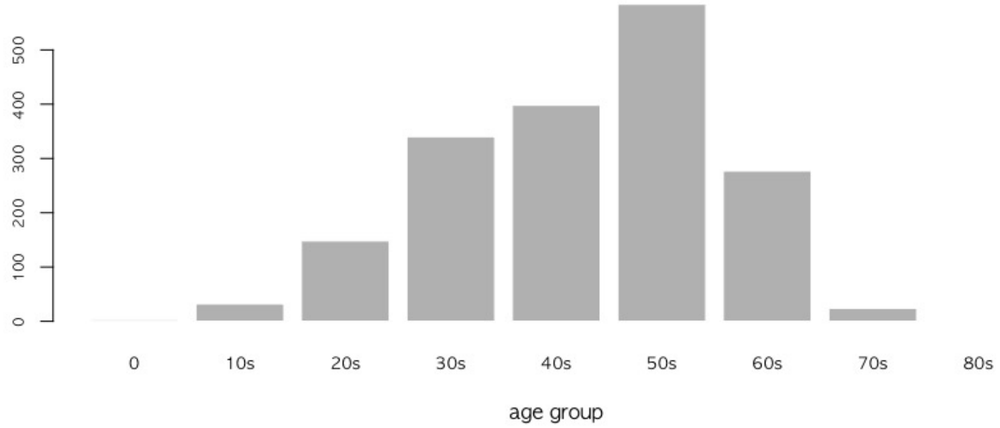
Fraud Rate by Marital Status



> Married: 1,062 (59.7%)
> Single: 717 (40.3%)

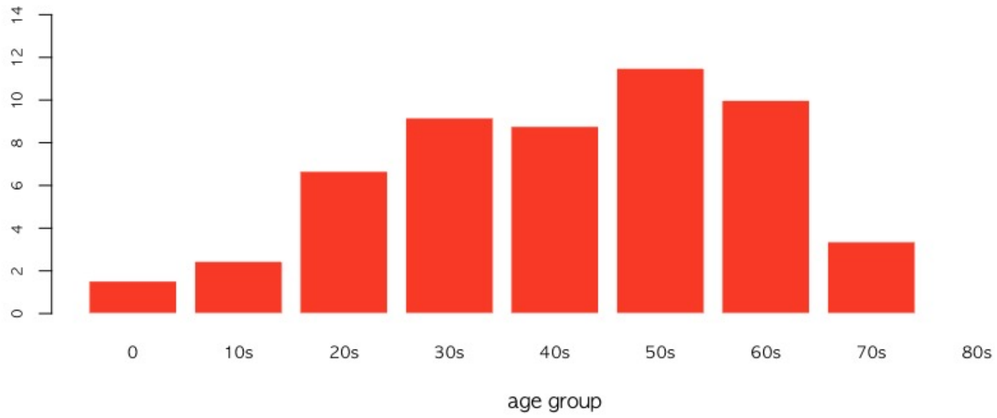
Fraud Rate by Age Group

Number of Insurance Fraud by Age Group



1st: 50s (584)
2nd: 40s (398)
3rd: 30s (340)

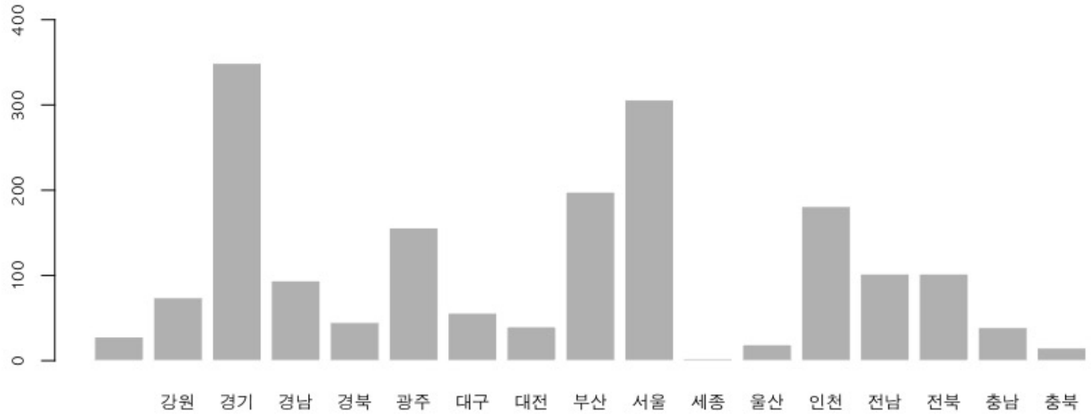
Insurance Fraud Rate by Age Group



1st: 50s (11.50%)
2nd: 60s (10.00%)
3rd: 30s (9.17%)

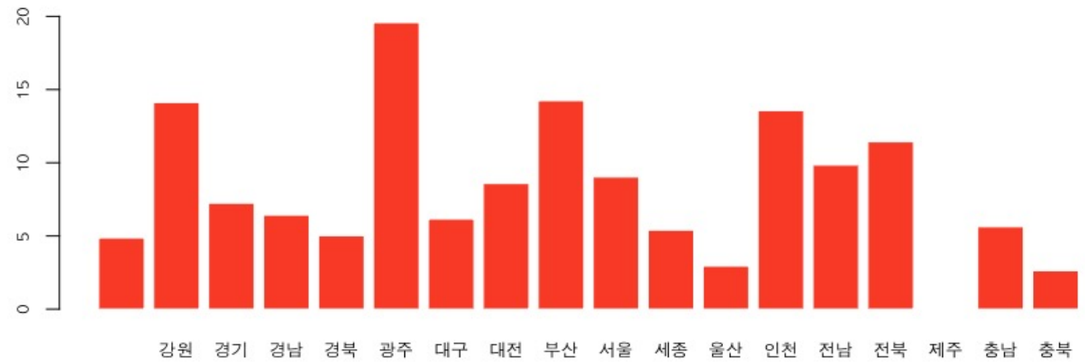
Fraud Rate by Area

Number of Insurance Fraud by Area



1st: 경기 (349)
2nd: 서울 (306)
3rd: 부산 (198)

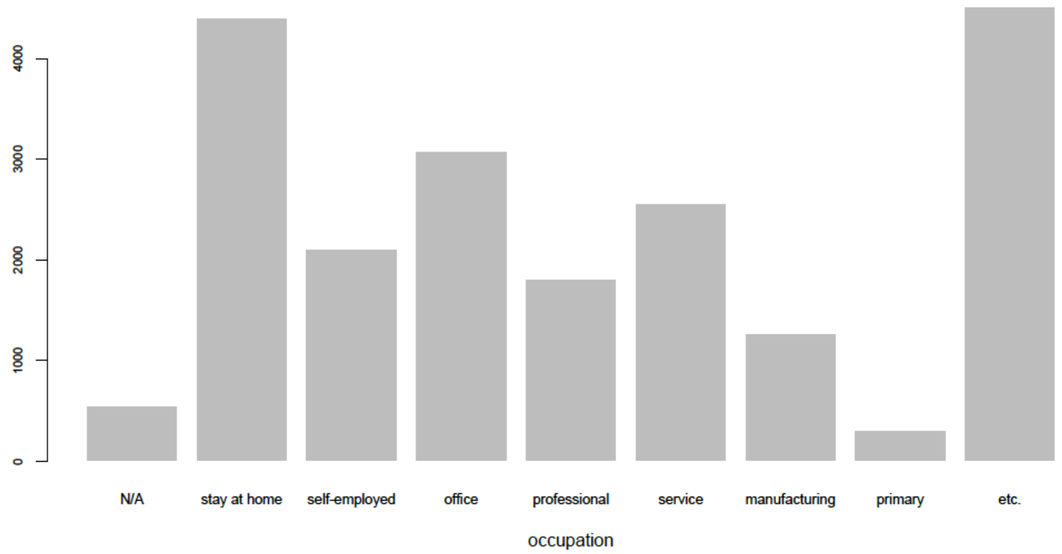
Insurance Fraud Rate by Area



1st: 광주 (19.57%)
2nd: 부산 (14.24%)
3rd: 강원 (14.12%)

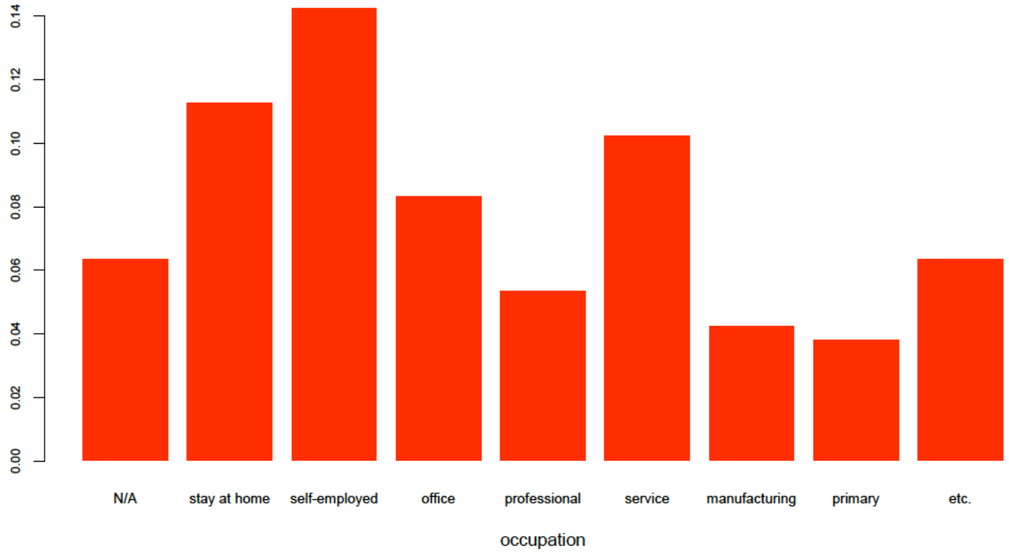
Fraud Rate by Occupation

Number of Insurance Fraud by Occupation



1st: Stay-at-home (498)
2nd: Self-employed (301)
3rd: etc. (288)

Insurance Fraud Rate by Occupation



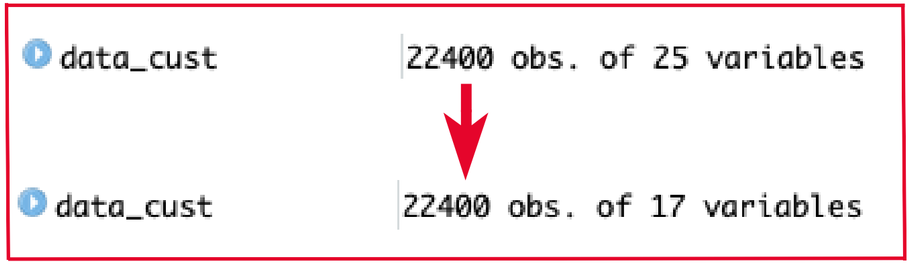
1st: Self-employed (14.27%)
2nd: Stay-at-home (11.31%)
3rd: Service (10.25%)

Remove Variables

→ Remove variables to be excluded from analysis after selecting variables through data EDA

Variables to be removed:

- Occupational Code 2 (OCCP_GRP_2)
- Spouse Occupational Code 2 (MATE_OCCP_GRP_2)
- Year & Month of paying the largest premium (MAX_PAYM_YM)
- Maximum monthly premium paid (MAX_PRM)
- Customer's first registration date (CUST_RGST)
- Number of children (CHLD_CNT)
- Age of the youngest child (LTBN_CHLD_AGE)
- Household income estimated by customer's occupation and premium level (JPBASE_HSHD_INCM)



code:

```
data_cust <- subset(data_cust, select=-c(OCCP_GRP_2, MATE_OCCP_GRP_2))
data_cust <- subset(data_cust, select=-c(MAX_PAYM_YM, MAX_PRM))
data_cust <- subset(data_cust, select=-CUST_RGST)
data_cust <- subset(data_cust, select=-c(CHLD_CNT, LTBN_CHLD_AGE))
data_cust <- subset(data_cust, select=-JPBASE_HSHD_INCM)
```

VLID_HOSP_OTDA

CUST_INCM	RCBASE_HSHD_INCM	HOSP_DAYS
4879.0000	10094	1
6509.0000	9143	3
4180.0000	0	16
4621.0813	4270	0
3894.0000	0	25
4319.3765	0	2
3611.0000	0	32
6465.0000	12219	2
4975.0000	7553	2
4169.1220	0	60
4319.3765	6301	7
8780.0000	10466	5
4181.9310	4066	15
4621.0813	0	20
4362.0000	6654	6

→ Count the average number of hospitalization / outpatient days per customer

→ Append VLID_HOSP_OTDA from CLAIM_DATA to CUST_DATA

code:

```
data_claim <- read.csv("CLAIM_DATA.csv", header = T, sep=",",
                                encoding="CP949", fileEncoding = "UCS-2")
data_claim
hosp_day_per_cust <- aggregate(data_claim$VLID_HOSP_OTDA,
                                        by=list(data_claim$CUST_ID), mean)
names(hosp_day_per_cust) <- c("CUST_ID", "HOSP_DAYS")
hosp_day_per_cust$HOSP_DAYS <- round(hosp_day_per_cust$HOSP_DAYS)
data_cust <- merge(data_cust, hosp_day_per_cust)
```

ACCI_DVSN & DMND_RESN_CODE

code:

```
library(reshape2)
table(data_claim$ACCI_DVSN, data_claim$DMND_RESN_CODE)
acci_dmnd_count <- table(data_claim$CUST_ID, data_claim$ACCI_DVSN, data_claim$DMND_RESN_CODE)
acci_dmnd_count <- as.data.frame(acci_dmnd_count)
names(acci_dmnd_count) <- c("CUST_ID", "ACCI_DVSN", "DMND_RESN_CODE", "value")

acci_dmnd_count
acci_dmnd_count <- dcast(data=acci_dmnd_count, CUST_ID ~ ACCI_DVSN + DMND_RESN_CODE, sum)
data_cust <- merge(data_cust, acci_dmnd_count)
data_cust <- data_cust[, sapply(data_cust, function(v) (var(v, na.rm=TRUE)!=0))]
```

| Add Variables from DATA_CLAIM

ACCI_DVSN & DMND_RESN_CODE

HOSP_DAYS	1_1	1_2	1_3	1_4	1_5	1_6	1_7	1_9	2_1	2_2	2_3	2_4	2_5	2_6
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	9	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	1	0	0	0	0
2	0	0	0	0	1	1	0	0	0	0	0	0	0	0
32	0	4	0	0	0	0	0	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	2	0	0	0	0	0	0	0	0	0	0	0
60	1	2	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	1	0	0	0	0
5	0	0	1	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	1	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0

→ Columns used from CLAIM_DATA:

1. ACCI_DVSN (cause of accident): disaster(1), traffic accident(2), disease(3)
2. DMND_RESN_CODE (reason code causing the claim for payment): death(01), hospitalization(02), outpatient hospital(03), disability(04), surgery(05), diagnosis(06), treatment(07), termination.invalidation(09)

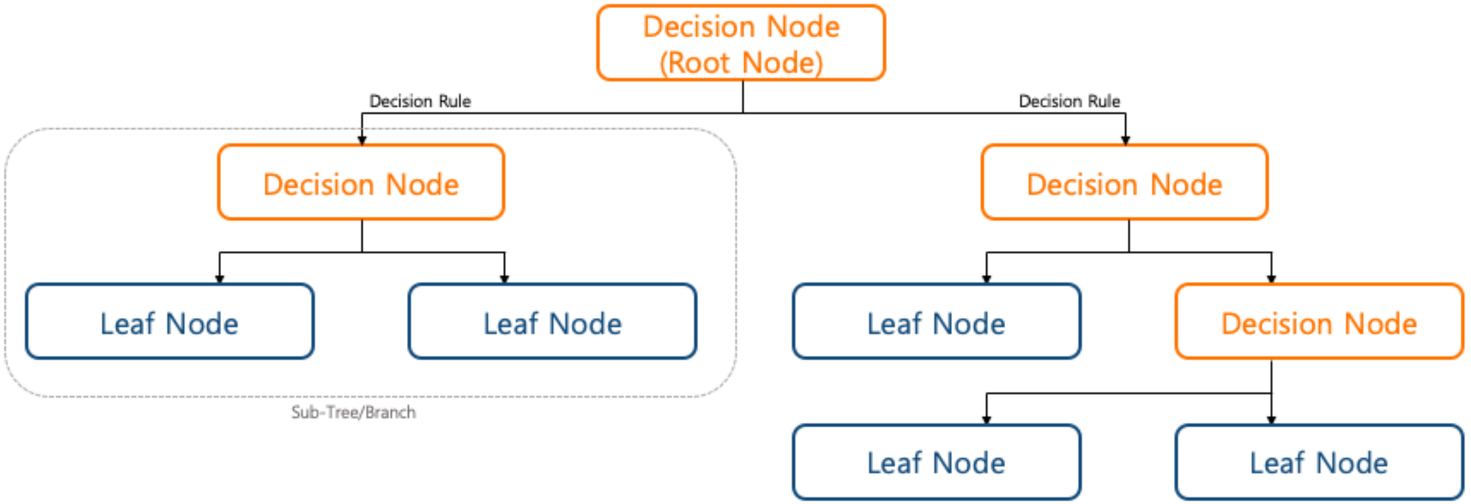
→ Create new columns in a format of **(ACCI_DVSN)_(DMND_RESN_CODE)** then record the number of claims by each customer

code:

```
library(reshape2)
table(data_claim$ACCI_DVSN, data_claim$DMND_RESN_CODE)
acci_dmnd_count <- table(data_claim$CUST_ID, data_claim$ACCI_DVSN, data_claim$DMND_RESN_CODE)
acci_dmnd_count <- as.data.frame(acci_dmnd_count)
names(acci_dmnd_count) <- c("CUST_ID", "ACCI_DVSN", "DMND_RESN_CODE", "value")

acci_dmnd_count
acci_dmnd_count <- dcast(data=acci_dmnd_count, CUST_ID ~ ACCI_DVSN + DMND_RESN_CODE, sum)
data_cust <- merge(data_cust, acci_dmnd_count)
data_cust <- data_cust[, sapply(data_cust, function(v) (var(v, na.rm=TRUE)!=0))]
```

Decision Tree



code:

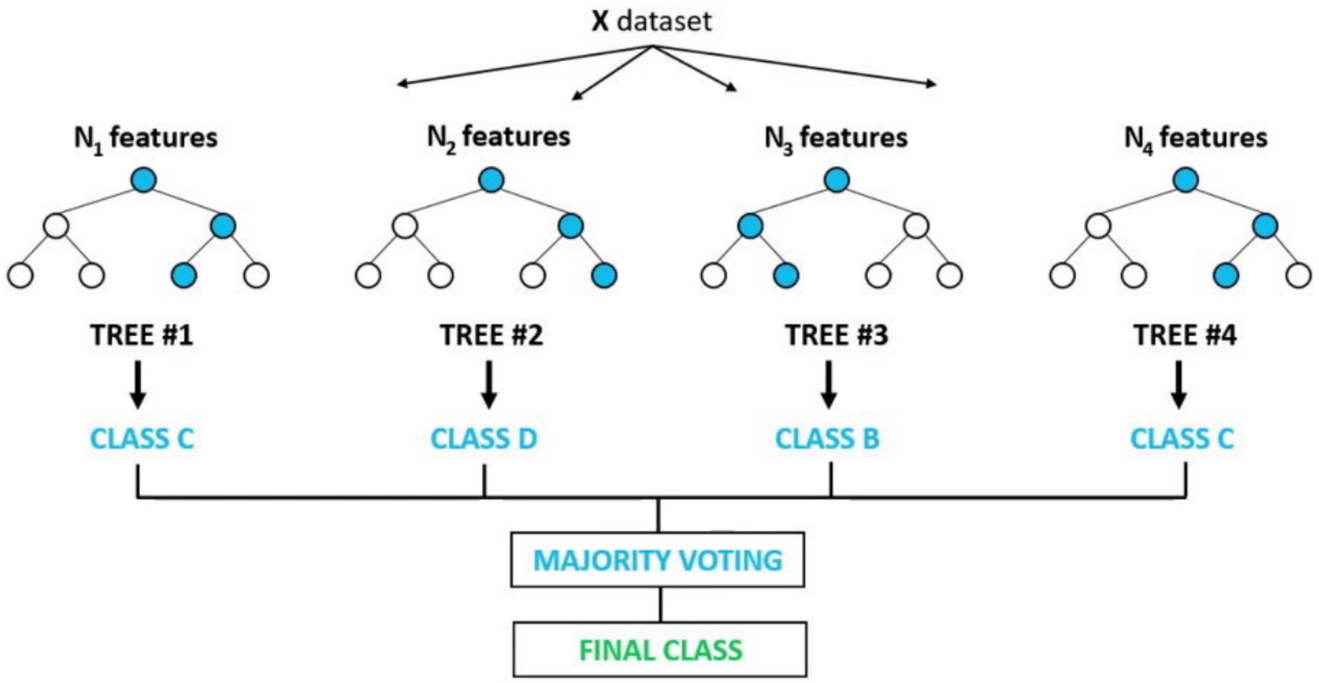
```
> library(party)
> tree_model <- ctree(SIU_CUST_YN~., data=train)
> tree_pred <- predict(tree_model, test)
> tree_t <- table(test$SIU_CUST_YN, round(tree_pred))
> tree_t

     0   1
0 5500 135
1  312 236
```

```
> # Accuracy
> mean(test$SIU_CUST_YN==round(tree_pred))
[1] 0.927705
> # Precision
> (tree_t[2,2])/(tree_t[1,2]+tree_t[2,2])
[1] 0.6361186
> # Recall
> (tree_t[2,2])/(tree_t[2,1]+tree_t[2,2])
[1] 0.4306569
```

- Technique of deriving classification results in the form of a tree structure
- Visualizes results by binary classification based on the most influential variable among input variables
- Uses Entropy and Gini Index as the evaluation method

Random Forest



code:

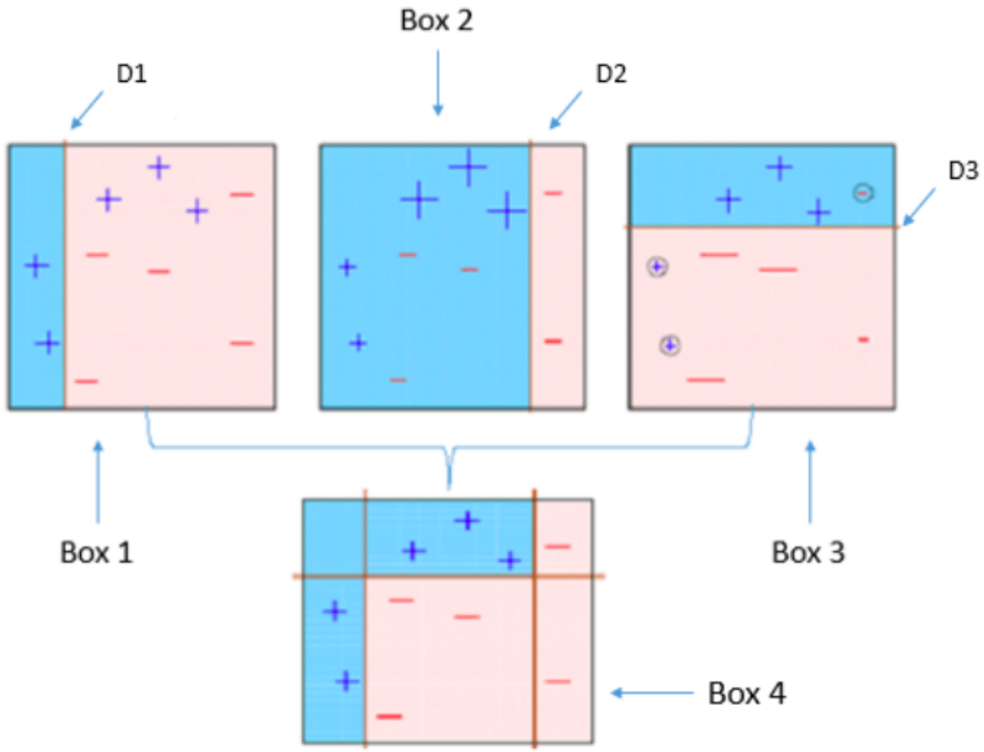
```
> library(randomForest)
> rf_model <- randomForest(SIU_CUST_YN~., data=train, ntree=100)
> rf_pred <- predict(rf_model, test)
> rf_t <- table(test$SIU_CUST_YN, round(rf_pred))
> rf_t

     0   1
0 5558  77
1  323 225
```

```
> # Accuracy
> mean(test$SIU_CUST_YN==round(rf_pred))
[1] 0.9353065
> # Precision
> (rf_t[2,2])/(rf_t[1,2]+rf_t[2,2])
[1] 0.7450331
> # Recall
> (rf_t[2,2])/(rf_t[2,1]+rf_t[2,2])
[1] 0.4105839
```

- Uses multiple decision trees as base learning models, trained with the "bagging" method
- Runs multiple models in parallel at the same time then selects a model by means of voting (or averaging)
- Typically results in higher prediction accuracy than a single decision tree

XGBoost



code:

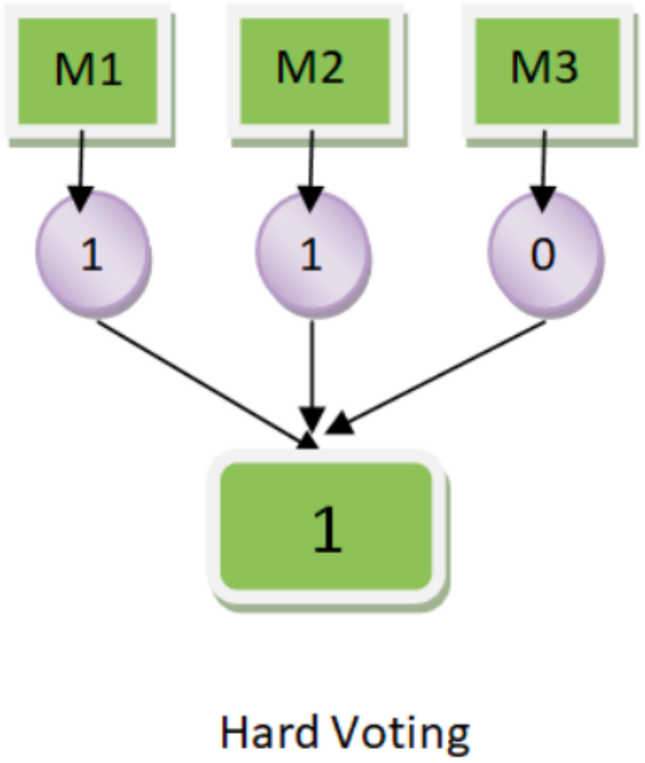
```
> library(xgboost)
> test_mat <- xgb.DMatrix(as.matrix(test[, -c(1,2,3)]))
> xgb_model <- xgboost(data=dtrain, objective='binary:logistic',
  nrounds =100, eta=1)
> xgb_pred <- predict(xgb_model, testmat)
> xgb_t <- table(test$SIU_CUST_YN, round(xgb_pred))
> xgb_t

        0    1
0  5495  140
1   300  248
```

```
> # Accuracy
> mean(test$SIU_CUST_YN==round(xgb_pred))
[1] 0.9288371
> # Precision
> (xgb_t[2,2])/(xgb_t[1,2]+xgb_t[2,2])
[1] 0.6391753
> # Recall
> (xgb_t[2,2])/(xgb_t[2,1]+xgb_t[2,2])
[1] 0.4525547
```

- ensemble learning technique which works by combining a number of “weak learners” to form a “strong learner”
- Updates the model by applying high weights to parts with low accuracy
- Highly efficient and precise so it's a widely used method in the field

Voting



Hard Voting

code:

```
> total_pred <- data.frame(list(round(tree_pred), round(rf_pred),
  round(xgb_pred)))
> names(total_pred) <- c("Tree", "RF", "XGB")
> voting <- function(x) {
    value_count = sort(table(x), decreasing = TRUE)
    label = names(value_count)[1]
  }
> voting_pred <- apply(total_pred, MARGIN=1, FUN=voting)
> vot_t <- table(test$SIU_CUST_YN, round(vot_pred))
> vot_t
    vot_pred
        0    1
  0  5495  140
  1   300  248
```

```
> # Accuracy
> mean(test$SIU_CUST_YN==round(vot_pred))
[1] 0.9343361
> # Precision
> (vot_t[2,2])/(vot_t[1,2]+vot_t[2,2])
[1] 0.7138554
> # Recall
> (vot_t[2,2])/(vot_t[2,1]+vot_t[2,2])
[1] 0.4324818
```

- In the form of majority rule
- Selected as the highest value among the three models
- It is divided into Hard Voting and Soft Voting, and Hard Voting method is used here

F1-score

Confusion Matrix

		Predicted	
		0	1
Actual	0	True Negative	False Positive
	1	False Negative	True Positive

	Accuracy	Precision	Recall	F1-score
Decision Tree	0.927705	0.6361186	0.4306569	0.5136017
Random Forest	**0.9353065**	**0.7450331**	0.4105839	0.5294118
XGBoost	0.9288371	0.6391753	**0.4525547**	0.5299145
Voting	0.9343361	0.7138554	0.4324818	**0.5386364**

- To increase insurance fraud prediction, finding fraudsters is important
- Low Precision raises customer exit rates, low Recall raises company spendings
- Both are important, but because of the trade-off relationship, calculate the ratio with the harmonic mean
- As a result of F1 Score, the number obtained by Voting is the highest, so choose the Voting model

Thank you!

Predictive Analytics for Insurance Fraud Detection

Kunyoon Kim