# Final Project
# Caravan Insurance Policy Purchase Prediction

Anutida Sangkla

Predictive Risk Modeling, Spring 2022

**Abstract**

Customers who are interested in purchasing a caravan insurance policy are predicted using different techniques such as logistic regression, forward stepwise selection, backward stepwise selection, and others. Data were trained and validated to select best model to predict an interest in buying a caravan. The lasso regression model was the selected model we used to predict who would be interested in buying a caravan insurance policy and explain why people would buy this insurance policy.

## 1 Introduction

Caravans are commonly used as temporary accommodation while traveling. However, because of advantages such as easily towable units, low fuel consumption, lower maintenance and insurance costs, and depreciation value, some people use them as their primary residence. In 2021, the caravan market was valued at $48.49 billion, and it is expected to increase over the forecast period between 2022 and 2027 due to the COVID-19 outbreak causing the demand for recreational vehicles increased [3].As a result, this may potentially increase the demand for a caravan insurance policy.

Today, the use of predictive modeling has permanently changed the way insurance policies are priced. Innovative tools allow insurers to use datasets to design sophisticated models that accurately determine the amount charged to each customer [1]. However, predictive modeling could also help marketing by researching customer buying behaviors. We believe that predictive modeling can help carriers to identify customers who require a caravan insurance policy. More advanced data insights will help insurers identify customers who is intent to buy this insurance policy [1]. Therefore, this project aims to discover the features influencing a caravan insurance policy buying and predict a customer who is potentially interested in buying this insurance policy. The predictive modeling was applied to select best model with the highest prediction accuracy.

In this project, we used three datasets provided by The Insurance Company (TIC) Benchmark which contains 85 variables and 1 target variable [4]. The dataset comprises all kinds of customer information of an insurance company, including product usage data and socio-demographic data derived from zip area codes [4]. These datasets allows us to create a process used statistical technique to predict future behaviors or outcomes by analyzing patterns in given sets of input data. In the past, these datasets were used to predict who would be interested in buying a specific insurance product and to explain why people would buy applying the bias-variance analysis [5]. For our project, the algorithms for a classification problem, such as logistic regression model, were used in order to achieve the goals of this project.

# 2 Data Exploration

There are three datasets using to predict a caravan insurance policy:

1. TICDATA2000.txt: Dataset to train and validate prediction models containing 5,822 customer records, and each record contain 86 attributes.
2. TICEVAL2000.txt: Dataset for predictions containing 4,000 customers, and each record contains 85 attibutes, target variable is missing.
3. TICTGTS2000.txt: Targets for the model evaluation

## 2.1 Importing data

All three datasets were imported to analyze and predict an interest in buying a caravan insurance policy of a customer as follow:

```
ticdata2000 = read.delim("ticdata2000.txt",header=FALSE)
ticeval2000 = read.delim("ticeval2000.txt",header=FALSE)
tictgts2000 = read.delim("tictgts2000.txt",header=FALSE)
```

## 2.2 Data description

The table 1 shows the description for 86 attributes.

| | Name | Description |
|---|---|---|
| 1 | MOSTYPE | Customer subtype |
| 2 | MAANTHUI | Number of houses 1 - 10 |
| 3 | MGEMOMV | Avg size household 1 - 6 |
| 4 | MGEMLEEF | Average age |
| 5 | MOSHOOFD | Customer main type |
| 6 | MGODRK | Roman catholic |
| 7 | MGODPR | Protestant ... |
| 8 | MGODOV | Other religion |
| 9 | MGODGE | No religion |
| 10 | MRELGE | Married |
| 11 | MRELSA | Living together |
| 12 | MRELOV | Other relation |
| 13 | MFALLEEN | Singles |
| 14 | MFGEKIND | Household without children |
| 15 | MFWEKIND | Household with children |
| 16 | MOPLHOOG | High level education |
| 17 | MOPLMIDD | Medium level education |
| 18 | MOPLLAAG | Lower level education |
| 19 | MBERHOOG | High status |
| 20 | MBERZELF | Entrepreneur |
| 21 | MBERBOER | Farmer |
| 22 | MBERMIDD | Middle management |
| 23 | MBERARBG | Skilled labourers |
| 24 | MBERARBO | Unskilled labourers |
| 25 | MSKA | Social class A |

| | | |
|---|---|---|
| 26 | MSKB1 | Social class B1 |
| 27 | MSKB2 | Social class B2 |
| 28 | MSKC | Social class C |
| 29 | MSKD | Social class D |
| 30 | MHHUUR | Rented house |
| 31 | MHKOOP | Home owners |
| 32 | MAUT1 | 1 car |
| 33 | MAUT2 | 2 cars |
| 34 | MAUT0 | No car |
| 35 | MZFONDS | National Health Service |
| 36 | MZPART | Private health insurance |
| 37 | MINKM30 | Income >30.000 |
| 38 | MINK3045 | Income 30-45.000 |
| 39 | MINK4575 | Income 45-75.000 |
| 40 | MINK7512 | Income 75-122.000 |
| 41 | MINK123M | Income <123.000 |
| 42 | MINKGEM | Average income |
| 43 | MKOOPKLA | Purchasing power class |
| 44 | PWAPART | Contribution private third party insurance |
| 45 | PWABEDR | Contribution third party insurance (firms) |
| 46 | PWALAND | Contribution third party insurance (agriculture) |
| 47 | PPERSAUT | Contribution car policies |
| 48 | PBESAUT | Contribution delivery van policies |
| 49 | PMOTSCO | Contribution motorcycle/scooter policies |
| 50 | PVRAAUT | Contribution lorry policies |
| 51 | PAANHANG | Contribution trailer policies |
| 52 | PTRACTOR | Contribution tractor policies |
| 53 | PWERKT | Contribution agricultural machines policies |
| 54 | PBROM | Contribution moped policies |
| 55 | PLEVEN | Contribution life insurances |
| 56 | PPERSONG | Contribution private accident insurance policies |
| 57 | PGEZONG | Contribution family accidents insurance policies |
| 58 | PWAOREG | Contribution disability insurance policies |
| 59 | PBRAND | Contribution fire policies |
| 60 | PZEILPL | Contribution surfboard policies |
| 61 | PPLEZIER | Contribution boat policies |
| 62 | PFIETS | Contribution bicycle policies |
| 63 | PINBOED | Contribution property insurance policies |
| 64 | PBYSTAND | Contribution social security insurance policies |
| 65 | AWAPART | Number of private third party insurance 1 - 12 |
| 66 | AWABEDR | Number of third party insurance (firms) ... |
| 67 | AWALAND | Number of third party insurance (agriculture) |
| 68 | APERSAUT | Number of car policies |
| 69 | ABESAUT | Number of delivery van policies |
| 70 | AMOTSCO | Number of motorcycle/scooter policies |
| 71 | AVRAAUT | Number of lorry policies |
| 72 | AAANHANG | Number of trailer policies |
| 73 | ATRACTOR | Number of tractor policies |

| | | |
|---|---|---|
| 74 | AWERKT | Number of agricultural machines policies |
| 75 | ABROM | Number of moped policies |
| 76 | ALEVEN | Number of life insurances |
| 77 | APERSONG | Number of private accident insurance policies |
| 78 | AGEZONG | Number of family accidents insurance policies |
| 79 | AWAOREG | Number of disability insurance policies |
| 80 | ABRAND | Number of fire policies |
| 81 | AZEILPL | Number of surfboard policies |
| 82 | APLEZIER | Number of boat policies |
| 83 | AFIETS | Number of bicycle policies |
| 84 | AINBOED | Number of property insurance policies |
| 85 | ABYSTAND | Number of social security insurance policies |
| 86 | CARAVAN | Number of mobile home policies 0 - 1 |

Table 1: Data Description

Each column was changed the names from the numbers to names.

```
colnames(ticdata2000) = c('MOSTYPE','MAANTHUI','MGEMOMV','MGEMLEEF','
   MOSHOOFD','MGODRK','MGODPR','MGODOV','MGODGE','MRELGE','MRELSA','MRELOV
   ','MFALLEEN','MFGEKIND','MFWEKIND','MOPLHOOG','MOPLMIDD','MOPLLAAG','
   MBERHOOG','MBERZELF','MBERBOER','MBERMIDD','MBERARBG','MBERARBO','MSKA'
   ,'MSKB1','MSKB2','MSKC','MSKD','MHHUUR','MHKOOP','MAUT1','MAUT2','MAUT0
   ','MZFONDS','MZPART','MINKM30','MINK3045','MINK4575','MINK7512','
   MINK123M','MINKGEM','MKOOPKLA','PWAPART','PWABEDR','PWALAND','PPERSAUT'
   ,'PBESAUT','PMOTSCO','PVRAAUT','PAANHANG','PTRACTOR','PWERKT','PBROM','
   PLEVEN','PPERSONG','PGEZONG','PWAOREG','PBRAND','PZEILPL','PPLEZIER','
   PFIETS','PINBOED','PBYSTAND','AWAPART','AWABEDR','AWALAND','APERSAUT','
   ABESAUT','AMOTSCO','AVRAAUT','AAANHANG','ATRACTOR','AWERKT','ABROM','
   ALEVEN','APERSONG','AGEZONG','AWAOREG','ABRAND','AZEILPL','APLEZIER','
   AFIETS','AINBOED','ABYSTAND','CARAVAN')
colnames(ticeval2000) = c('MOSTYPE','MAANTHUI','MGEMOMV','MGEMLEEF','
   MOSHOOFD','MGODRK','MGODPR','MGODOV','MGODGE','MRELGE','MRELSA','MRELOV
   ','MFALLEEN','MFGEKIND','MFWEKIND','MOPLHOOG','MOPLMIDD','MOPLLAAG','
   MBERHOOG','MBERZELF','MBERBOER','MBERMIDD','MBERARBG','MBERARBO','MSKA'
   ,'MSKB1','MSKB2','MSKC','MSKD','MHHUUR','MHKOOP','MAUT1','MAUT2','MAUT0
   ','MZFONDS','MZPART','MINKM30','MINK3045','MINK4575','MINK7512','
   MINK123M','MINKGEM','MKOOPKLA','PWAPART','PWABEDR','PWALAND','PPERSAUT'
   ,'PBESAUT','PMOTSCO','PVRAAUT','PAANHANG','PTRACTOR','PWERKT','PBROM','
   PLEVEN','PPERSONG','PGEZONG','PWAOREG','PBRAND','PZEILPL','PPLEZIER','
   PFIETS','PINBOED','PBYSTAND','AWAPART','AWABEDR','AWALAND','APERSAUT','
   ABESAUT','AMOTSCO','AVRAAUT','AAANHANG','ATRACTOR','AWERKT','ABROM','
   ALEVEN','APERSONG','AGEZONG','AWAOREG','ABRAND','AZEILPL','APLEZIER','
   AFIETS','AINBOED','ABYSTAND')
colnames(tictgts2000) = 'CARAVAN'
```

### 2.2.1 TICDATA2000

```
str(ticdata2000)

'data.frame': 5822 obs. of  86 variables:
```

```
$ MOSTYPE : int  33 37 37 9 40 23 39 33 33 11 ...
$ MAANTHUI: int  1 1 1 1 1 1 2 1 1 2 ...
$ MGEMOMV : int  3 2 2 3 4 2 3 2 2 3 ...
$ MGEMLEEF: int  2 2 2 3 2 1 2 3 4 3 ...
$ MOSHOOFD: int  8 8 8 3 10 5 9 8 8 3 ...
$ MGODRK  : int  0 1 0 2 1 0 2 0 0 3 ...
$ MGODPR  : int  5 4 4 3 4 5 2 7 1 5 ...
$ MGODOV  : int  1 1 2 2 1 0 0 0 3 0 ...
$ MGODGE  : int  3 4 4 4 4 5 5 2 6 2 ...
$ MRELGE  : int  7 6 3 5 7 0 7 7 6 7 ...
$ MRELSA  : int  0 2 2 2 1 6 2 2 0 0 ...
$ MRELOV  : int  2 2 4 2 2 3 0 0 3 2 ...
$ MFALLEEN: int  1 0 4 2 2 3 0 0 3 2 ...
$ MFGEKIND: int  2 4 4 3 4 5 3 5 3 2 ...
$ MFWEKIND: int  6 5 2 4 4 2 6 4 3 6 ...
$ MOPLHOOG: int  1 0 0 3 5 0 0 0 0 0 ...
$ MOPLMIDD: int  2 5 5 4 4 5 4 3 1 4 ...
$ MOPLLAAG: int  7 4 4 2 0 4 5 6 8 5 ...
$ MBERHOOG: int  1 0 0 4 0 2 0 2 1 2 ...
$ MBERZELF: int  0 0 0 0 5 0 0 0 1 0 ...
$ MBERBOER: int  1 0 0 0 4 0 0 0 0 0 ...
$ MBERMIDD: int  2 5 7 3 0 4 4 2 1 3 ...
$ MBERARBG: int  5 0 0 1 0 2 1 5 8 3 ...
$ MBERARBO: int  2 4 2 2 0 2 5 2 1 3 ...
$ MSKA    : int  1 0 0 3 9 2 0 2 1 1 ...
$ MSKB1   : int  1 2 5 2 0 2 1 1 1 2 ...
$ MSKB2   : int  2 3 0 1 0 2 4 2 0 1 ...
$ MSKC    : int  6 5 4 4 0 4 5 5 8 4 ...
$ MSKD    : int  1 0 0 0 0 2 0 2 1 2 ...
$ MHHUUR  : int  1 2 7 5 4 9 6 0 9 0 ...
$ MHKOOP  : int  8 7 2 4 5 0 3 9 0 9 ...
$ MAUT1   : int  8 7 7 9 6 5 8 4 5 6 ...
$ MAUT2   : int  0 1 0 0 2 3 0 4 2 1 ...
$ MAUT0   : int  1 2 2 0 1 3 1 2 3 2 ...
$ MZFONDS : int  8 6 9 7 5 9 9 6 7 6 ...
$ MZPART  : int  1 3 0 2 4 0 0 3 2 3 ...
$ MINKM30 : int  0 2 4 1 0 5 4 2 7 2 ...
$ MINK3045: int  4 0 5 5 0 2 3 5 2 3 ...
$ MINK4575: int  5 5 0 3 9 3 3 3 1 3 ...
$ MINK7512: int  0 2 0 0 0 0 0 0 0 1 ...
$ MINK123M: int  0 0 0 0 0 0 0 0 0 0 ...
$ MINKGEM : int  4 5 3 4 6 3 3 3 2 4 ...
$ MKOOPKLA: int  3 4 4 4 3 3 5 3 3 7 ...
$ PWAPART : int  0 2 2 0 0 0 0 0 0 2 ...
$ PWABEDR : int  0 0 0 0 0 0 0 0 0 0 ...
$ PWALAND : int  0 0 0 0 0 0 0 0 0 0 ...
$ PPERSAUT: int  6 0 6 6 0 6 6 0 5 0 ...
$ PBESAUT : int  0 0 0 0 0 0 0 0 0 0 ...
$ PMOTSCO : int  0 0 0 0 0 0 0 0 0 0 ...
$ PVRAAUT : int  0 0 0 0 0 0 0 0 0 0 ...
$ PAANHANG: int  0 0 0 0 0 0 0 0 0 0 ...
$ PTRACTOR: int  0 0 0 0 0 0 0 0 0 0 ...
$ PWERKT  : int  0 0 0 0 0 0 0 0 0 0 ...
$ PBROM   : int  0 0 0 0 0 0 0 3 0 0 ...
```

```
 $ PLEVEN   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PPERSONG: int  0 0 0 0 0 0 0 0 0 0 ...
 $ PGEZONG : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PWAOREG : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PBRAND  : int  5 2 2 2 6 0 0 0 0 3 ...
 $ PZEILPL : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PPLEZIER: int  0 0 0 0 0 0 0 0 0 0 ...
 $ PFIETS  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PINBOED : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PBYSTAND: int  0 0 0 0 0 0 0 0 0 0 ...
 $ AWAPART : int  0 2 1 0 0 0 0 0 0 1 ...
 $ AWABEDR : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AWALAND : int  0 0 0 0 0 0 0 0 0 0 ...
 $ APERSAUT: int  1 0 1 1 0 1 1 0 1 0 ...
 $ ABESAUT : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AMOTSCO : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AVRAAUT : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AAANHANG: int  0 0 0 0 0 0 0 0 0 0 ...
 $ ATRACTOR: int  0 0 0 0 0 0 0 0 0 0 ...
 $ AWERKT  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ABROM   : int  0 0 0 0 0 0 0 1 0 0 ...
 $ ALEVEN  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ APERSONG: int  0 0 0 0 0 0 0 0 0 0 ...
 $ AGEZONG : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AWAOREG : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ABRAND  : int  1 1 1 1 1 0 0 0 0 1 ...
 $ AZEILPL : int  0 0 0 0 0 0 0 0 0 0 ...
 $ APLEZIER: int  0 0 0 0 0 0 0 0 0 0 ...
 $ AFIETS  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AINBOED : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ABYSTAND: int  0 0 0 0 0 0 0 0 0 0 ...
 $ CARAVAN : int  0 0 0 0 0 0 0 0 0 0 ...
```

### 2.2.2 TICEVAL2000

```
str(ticeval2000)
'data.frame': 4000 obs. of  85 variables:
 $ MOSTYPE : int  33 6 39 9 31 30 35 6 4 10 ...
 $ MAANTHUI: int  1 1 1 1 1 1 1 1 1 1 ...
 $ MGEMOMV : int  4 3 3 2 2 2 2 3 2 4 ...
 $ MGEMLEEF: int  2 2 3 3 4 4 4 3 4 2 ...
 $ MOSHOOFD: int  8 2 9 3 7 7 8 2 1 3 ...
 $ MGODRK  : int  0 0 1 2 0 1 2 3 0 0 ...
 $ MGODPR  : int  6 5 4 3 2 4 5 4 7 7 ...
 $ MGODOV  : int  0 0 2 2 0 2 1 2 2 0 ...
 $ MGODGE  : int  3 4 3 4 7 3 2 2 0 2 ...
 $ MRELGE  : int  5 5 5 5 9 5 8 9 9 9 ...
 $ MRELSA  : int  0 2 2 4 0 0 0 0 0 0 ...
 $ MRELOV  : int  4 2 3 1 0 4 1 0 0 0 ...
 $ MFALLEEN: int  1 1 2 2 0 4 2 0 1 0 ...
 $ MFGEKIND: int  1 4 3 4 6 3 5 5 7 2 ...
 $ MFWEKIND: int  8 5 6 4 3 2 3 4 2 7 ...
 $ MOPLHOOG: int  2 5 2 2 0 1 1 4 3 2 ...
 $ MOPLMIDD: int  2 4 4 4 0 2 5 4 4 3 ...
```

```
 $ MOPLLAAG: int  6 0 4 4 9 6 4 2 2 5 ...
 $ MBERHOOG: int  0 5 2 2 0 1 2 4 2 0 ...
 $ MBERZELF: int  0 0 1 1 0 0 0 3 0 0 ...
 $ MBERBOER: int  1 0 1 1 0 1 0 0 0 0 ...
 $ MBERMIDD: int  2 4 3 5 2 3 3 2 4 5 ...
 $ MBERARBG: int  6 0 2 1 4 3 3 0 3 2 ...
 $ MBERARBO: int  1 0 2 2 4 3 3 2 1 3 ...
 $ MSKA    : int  0 4 1 3 0 1 1 6 2 0 ...
 $ MSKB1   : int  2 3 1 1 0 1 1 1 3 4 ...
 $ MSKB2   : int  1 0 5 3 0 2 5 0 1 0 ...
 $ MSKC    : int  5 2 2 2 7 5 4 2 4 5 ...
 $ MSKD    : int  3 1 1 2 2 1 0 0 1 0 ...
 $ MHHUUR  : int  1 3 1 3 9 5 8 0 7 0 ...
 $ MHKOOP  : int  8 6 8 6 0 4 1 9 2 9 ...
 $ MAUT1   : int  8 9 6 7 7 5 8 5 7 6 ...
 $ MAUT2   : int  1 0 2 2 2 1 1 4 0 1 ...
 $ MAUT0   : int  1 0 2 1 0 4 1 0 2 2 ...
 $ MZFONDS : int  8 7 6 7 9 9 4 3 7 6 ...
 $ MZPART  : int  1 2 3 2 0 0 5 6 2 3 ...
 $ MINKM30 : int  3 1 2 2 5 2 2 1 3 0 ...
 $ MINK3045: int  3 1 4 5 4 5 5 3 3 7 ...
 $ MINK4575: int  3 5 3 3 0 2 2 4 3 2 ...
 $ MINK7512: int  0 4 1 1 0 1 0 2 1 0 ...
 $ MINK123M: int  0 0 0 0 0 0 0 2 0 0 ...
 $ MINKGEM : int  3 6 3 4 3 4 3 6 4 4 ...
 $ MKOOPKLA: int  3 8 5 4 1 2 5 8 6 8 ...
 $ PWAPART : int  1 2 2 2 2 0 2 2 2 2 ...
 $ PWABEDR : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PWALAND : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PPERSAUT: int  0 6 6 5 0 0 6 0 0 0 ...
 $ PBESAUT : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PMOTSCO : int  0 4 0 0 0 0 0 0 0 0 ...
 $ PVRAAUT : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PAANHANG: int  0 0 0 0 0 0 0 0 0 0 ...
 $ PTRACTOR: int  0 0 0 0 0 0 0 0 0 0 ...
 $ PWERKT  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PBROM   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PLEVEN  : int  0 3 4 0 0 0 0 0 0 0 ...
 $ PPERSONG: int  0 0 0 0 0 0 0 0 0 0 ...
 $ PGEZONG : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PWAOREG : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PBRAND  : int  4 4 4 3 1 4 2 0 2 4 ...
 $ PZEILPL : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PPLEZIER: int  0 0 0 0 0 0 0 0 0 0 ...
 $ PFIETS  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PINBOED : int  0 0 0 0 0 0 0 0 0 0 ...
 $ PBYSTAND: int  0 0 0 0 0 0 0 0 0 3 ...
 $ AWAPART : int  1 1 1 1 1 0 1 1 1 1 ...
 $ AWABEDR : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AWALAND : int  0 0 0 0 0 0 0 0 0 0 ...
 $ APERSAUT: int  0 1 1 1 0 0 1 0 0 0 ...
 $ ABESAUT : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AMOTSCO : int  0 1 0 0 0 0 0 0 0 0 ...
 $ AVRAAUT : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
 $ AAANHANG: int  0 0 0 0 0 0 0 0 0 0 ...
 $ ATRACTOR: int  0 0 0 0 0 0 0 0 0 0 ...
 $ AWERKT  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ABROM   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ALEVEN  : int  0 2 1 0 0 0 0 0 0 0 ...
 $ APERSONG: int  0 0 0 0 0 0 0 0 0 0 ...
 $ AGEZONG : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AWAOREG : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ABRAND  : int  1 1 1 1 1 2 1 0 1 1 ...
 $ AZEILPL : int  0 0 0 0 0 0 0 0 0 0 ...
 $ APLEZIER: int  0 0 0 0 0 0 0 0 0 0 ...
 $ AFIETS  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AINBOED : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ABYSTAND: int  0 0 0 0 0 0 0 0 0 1 ...
```

#### 2.2.3 TICTGTS2000

```
str(tictgts2000)

'data.frame': 4000 obs. of  1 variable:
 $ CARAVAN: int  0 1 0 0 0 0 0 0 0 0 ...
```

### 2.3 Target exploration

Number of mobile home policies is the target variable for our prediction. We explored this variable to learn more about our target as below:

```
caravan = table(ticdata2000$CARAVAN)
caravan

 0    1
5474  348
```

The data contains 5,474 customer records purchased 0 caravan policy and 348 customer records purchased 1 caravan policy. Therefore, 99% of customers did not purchase a mobile home policy.

#### 2.3.1 Correlation between target and predictors

We explored the relationships between the target variable and independent variables to determine the feature highly correlated to a caravan insurance policy.

```
#### Top 20 highly correlated between features and target variable:
cor_target = cor(ticdata2000[-86], ticdata2000$CARAVAN, method = "pearson"
   ) %>%
  as_tibble(rownames = "Variable") %>%
  mutate(abs_cor = abs(V1)) %>%
  arrange(-abs_cor)
topcor_target= cor_target[1:20,]
topcor_target$Variable = factor(topcor_target$Variable,
levels = topcor_target$Variable[order(topcor_target$abs_cor, decreasing =
   FALSE)])
colnames(topcor_target) = c("Variable","Correlation","Absolute_Correlation
   ")
```

```
# Plotting
p = ggplot(topcor_target, aes(x=Variable, y=Absolute_Correlation, fill =
   Absolute_Correlation))
p = p + geom_bar(stat = 'identity') + coord_flip()
p = p + scale_fill_gradient2(low = "green", mid = "yellow", high = "
   darkred", midpoint = max(topcor_target$Absolute_Correlation)/2)
p = p + labs(y = "Correlation", fill = "Correlation")
p
```

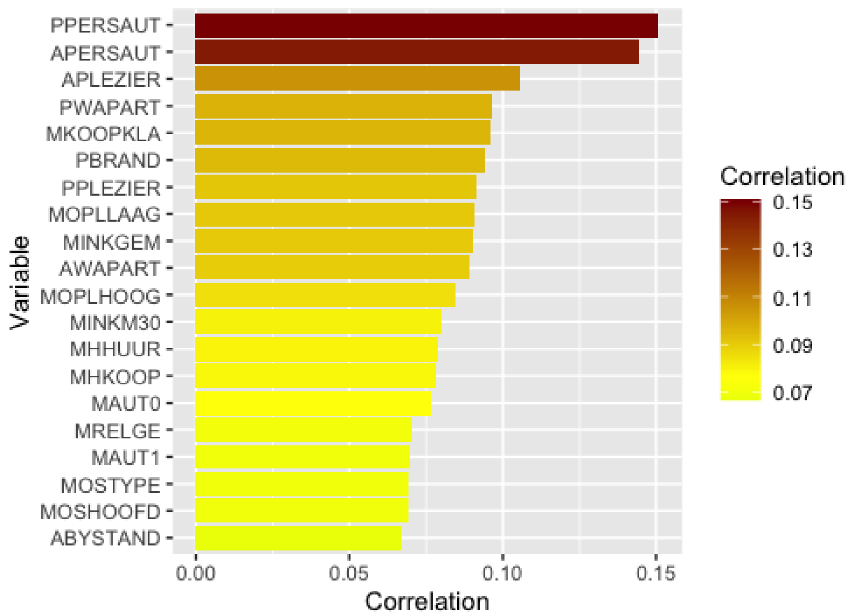


Figure 1: Correlation Between Target and Predictors.

The figure 1 indicates that **PPERSAUT**, Contribution car policies, is the most correlated to CARAVAN.

### 2.3.2 The most correlated variable

We explored the contribution car policies variable since this variable is the highest correlated to our target variable.

```
ggplot(ticdata2000, aes(x = reorder(PPERSAUT, PPERSAUT, function(x) -
   length(x)), fill = as.factor(CARAVAN))) +
  geom_bar() +
  scale_fill_brewer(palette = "Set2")+
  labs(x = "Contribution Car Policies", fill = "# of CARAVAN")
```

The figure 2 shows zero contribution car policies is the most frequency. However, 6 contributions car policies are also high. With 6 contributions car policies, customers were most likely to purchase a caravan insurance policy.

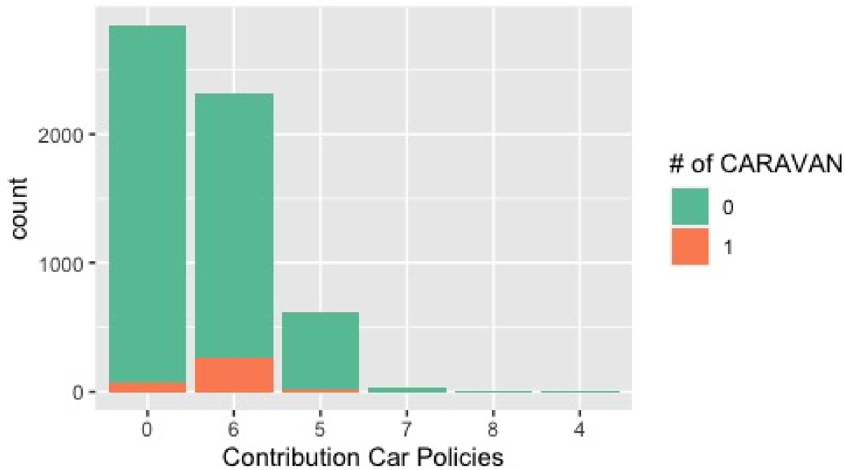


Figure 2: PPERSAUT: Contribution Car Policies

## 2.4 Feature exploration

We analyzed the relationships between independent variables which include 85 features containing in TICDATA2000 set to determine the high correlation variables.

```
corr_sig <- function(data = ticdata2000,sig = 0.9){
  df_cor <- data %>% mutate_if(is.character, as.factor)
  df_cor <- df_cor %>% mutate_if(is.factor, as.numeric)
  corr <- cor(df_cor)
  corr[lower.tri(corr,diag=TRUE)] <- NA
  #drop perfect correlations
  corr[corr == 1] <- NA
  #turn into a 3-column table
  corr <- as.data.frame(as.table(corr))
  #remove the NA values from above
  corr <- na.omit(corr)
  #select significant values
  corr <- subset(corr, abs(Freq) > sig)
  #sort by highest correlation
  corr <- corr[order(-abs(corr$Freq)),]
  return(corr)
}
corr.df = as.data.frame(corr_sig())
rownames(corr.df) = NULL

# Create table for report
tab2 = xtable(corr.df, , digits = 4, caption = "Correlation Between
   Predictors", label = "tab:table2", table.placement = "h!" )
print(tab2, tabular.environment = "longtable")
```

| | Var1 | Var2 | Freq |
|---|---|---|---|
| 1 | MHHUUR | MHKOOP | -0.9996 |
| 2 | MZFONDS | MZPART | -0.9992 |
| 3 | MOSTYPE | MOSHOOFD | 0.9927 |

| | | | |
|---|---|---|---|
| 4 | PWALAND | AWALAND | 0.9876 |
| 5 | PWAPART | AWAPART | 0.9814 |
| 6 | PGEZONG | AGEZONG | 0.9800 |
| 7 | PBROM | ABROM | 0.9697 |
| 8 | PBYSTAND | ABYSTAND | 0.9662 |
| 9 | PAANHANG | AAANHANG | 0.9661 |
| 10 | PVRAAUT | AVRAAUT | 0.9487 |
| 11 | PWAOREG | AWAOREG | 0.9484 |
| 12 | PFIETS | AFIETS | 0.9359 |
| 13 | PTRACTOR | ATRACTOR | 0.9298 |
| 14 | PPERSAUT | APERSAUT | 0.9162 |
| 15 | PWERKT | AWERKT | 0.9097 |
| 16 | PMOTSCO | AMOTSCO | 0.9049 |
| 17 | PPLEZIER | APLEZIER | 0.9044 |
| 18 | PBESAUT | ABESAUT | 0.9030 |

Table 2: Correlation Between Predictors

The table 2 shows paired variables having a correlation greater than 0.90. Especially, rented house (MHHUUR) variable and home owners (MHKOOP) have the highest correlation, which is equal to -0.9996. However, it is clear that rented house and home owners have a negative relationship.

### 2.5 Customer exploration

Customers related data are analyzed to provide insights related to customers information.

#### 2.5.1 Customer subtype

```
### Customer Subtype and their interests in buying a caravan insurance
   policy:
mostype =  ticdata2000
mostype$MOSTYPE = as.factor(mostype$MOSTYPE)
ggplot(mostype, aes(x = reorder(MOSTYPE, MOSTYPE, function(x) - length(x))
   , fill = as.factor(CARAVAN))) +
  geom_bar() +
  labs(x = "Customer Subtype", fill = "# of CARAVAN")
```

The figure 3 shows most customers were lower class large families, type 33. However, middle class families, type 8, were more likely to purchase a caravan insurance policy.

#### 2.5.2 Customer main type

Customer main type and their interests in buying a caravan insurance policy:

```
moshoofd = ticdata2000
moshoofd$MOSHOOFD = as.factor(moshoofd$MOSHOOFD)
ggplot(moshoofd, aes(x = reorder(MOSHOOFD, MOSHOOFD, function(x) - length(
   x)), fill = as.factor(CARAVAN))) +
  geom_bar() +
  scale_fill_brewer(palette = "Dark2")+
  labs(x = "Customer Main Type", fill = "# of CARAVAN")
```

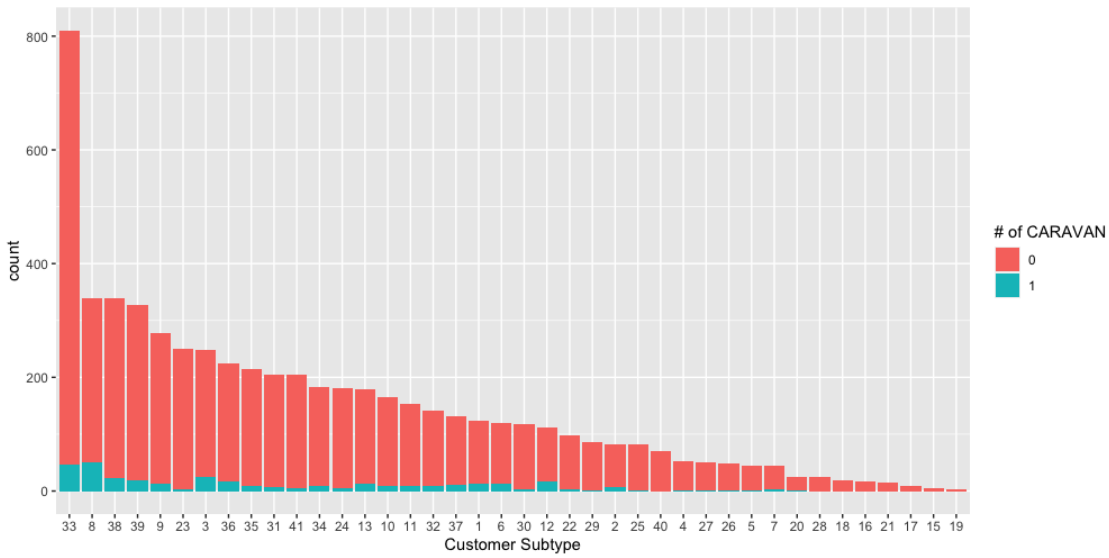


Figure 3: Customer Subtype and Customers Purchased a Caravan Insurance Policy

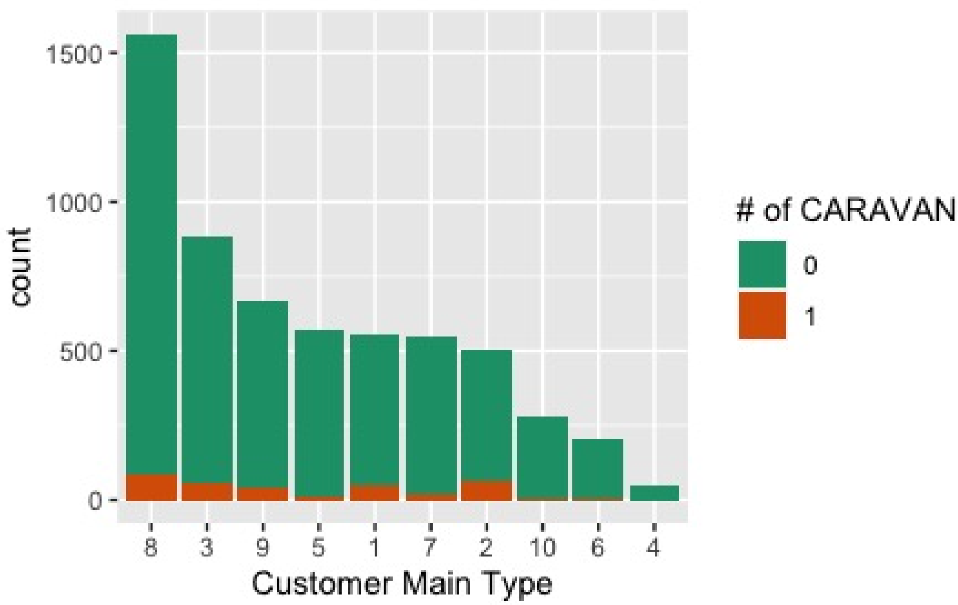


Figure 4: Customer Main Type and Customers Purchased a Caravan Insurance Policy

The figure 4 shows most customers were family with grown ups, main type 8, and the most caravan policies were bought by this main type.

### 2.5.3 Customer age group

```
CARAVAN_1 = ticdata2000[ticdata2000$CARAVAN == 1,]
CARAVAN_1$MGEMLEEF = as.factor(CARAVAN_1$MGEMLEEF)
ggplot(CARAVAN_1, aes(x = reorder(MGEMLEEF, MGEMLEEF, function(x) - length
    (x)))) +
  geom_bar() +
```

```
  labs(x = "Average Age") + theme(legend.position="none")
```

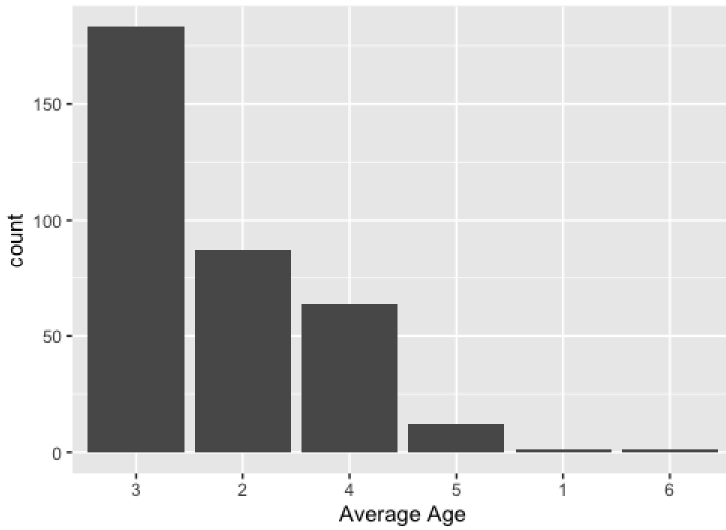


Figure 5: Age Group and Customers Purchased a Caravan Insurance Policy

The figure 5 shows a caravan insurance policy were interested in buying for customers age group between 40 and 50 years.

# 3 Model selection

## 3.1 Splitting data

TICDATA2000 was divided into two sets with the train-test ratio of 70%. This means that we used 70% of the observations for training and the rest for validation.

```
### We will split data with the ratio 70:30
set.seed(1)
split = sample(c(rep(0, 0.7 * nrow(ticdata2000)), rep(1, 0.3 * nrow(
   ticdata2000))))
training <- ticdata2000[split == 0, ]
validation <- ticdata2000[split == 1, ]
dim(training)
[1] 4076   86

dim(validation)
[1] 1746   86
```

#### The example of training data

```
xtable(head(training[,1:5]), digits = 0, caption = " Training data", label
     = "tab:table2", table.placement = "h!")
```

The table 3 shows the first 5 customer records and 5 features in the training set used to train models to select the best model for predicting an interest in buying a caravan insurance policy of a customer.

| | MOSTYPE | MAANTHUI | MGEMOMV | MGEMLEEF | MOSHOOFD |
|---|---|---|---|---|---|
| 1 | 33 | 1 | 3 | 2 | 8 |
| 3 | 37 | 1 | 2 | 2 | 8 |
| 5 | 40 | 1 | 4 | 2 | 10 |
| 7 | 39 | 2 | 3 | 2 | 9 |
| 8 | 33 | 1 | 2 | 3 | 8 |
| 9 | 33 | 1 | 2 | 4 | 8 |

Table 3: Training data

**The example of validation data**

```
xtable(head(validation[,1:5]), digits = 0, caption = " Validation data",
   label = "tab:table4", table.placement = "h!")
```

| | MOSTYPE | MAANTHUI | MGEMOMV | MGEMLEEF | MOSHOOFD |
|---|---|---|---|---|---|
| 2 | 37 | 1 | 2 | 2 | 8 |
| 4 | 9 | 1 | 3 | 3 | 3 |
| 6 | 23 | 1 | 2 | 1 | 5 |
| 10 | 11 | 2 | 3 | 3 | 3 |
| 18 | 22 | 2 | 3 | 3 | 5 |
| 26 | 33 | 1 | 3 | 3 | 8 |

Table 4: Validation data

The table 4 shows the first 5 customer records and 5 features in the training set used to validate models to select the best model for predicting an interest in buying a caravan insurance policy.

## 3.2 Model training and validation

The training dataset was trained with different techniques, and the validation dataset was used to validate model to obtain model accuracy. We compared each model with predictive accuracy and chose the model having the highest accuracy as follows:

### 3.2.1 Logistic regression model

**Model fitting**

```
## Model fitting:
glm.fit = glm(CARAVAN ~ ., data = training, family = binomial)
sum = summary(glm.fit)

# Create table for the report
tab4 = xtable(sum, digits = 4, caption = "Logistic Regression Model",
   label = "tab:table4", table.placement = "h!")
print(tab4, tabular.environment = "longtable")
```

| | Estimate | Std. Error | z value | Pr(>\|z\|) |
|---|---|---|---|---|
| (Intercept) | 252.7201 | 13187.0282 | 0.0192 | 0.9847 |
| MOSTYPE | 0.0665 | 0.0558 | 1.1917 | 0.2334 |

| | | | | | |
|---|---|---|---|---|---|
| MAANTHUI | -0.1496 | 0.2230 | -0.6709 | 0.5023 | |
| MGEMOMV | -0.0520 | 0.1695 | -0.3070 | 0.7588 | |
| MGEMLEEF | 0.1122 | 0.1217 | 0.9219 | 0.3566 | |
| MOSHOOFD | -0.2885 | 0.2515 | -1.1472 | 0.2513 | |
| MGODRK | -0.1306 | 0.1292 | -1.0108 | 0.3121 | |
| MGODPR | -0.0657 | 0.1408 | -0.4670 | 0.6405 | |
| MGODOV | -0.0348 | 0.1258 | -0.2765 | 0.7822 | |
| MGODGE | -0.1184 | 0.1322 | -0.8953 | 0.3706 | |
| MRELGE | 0.2783 | 0.1861 | 1.4957 | 0.1347 | |
| MRELSA | 0.1650 | 0.1756 | 0.9397 | 0.3474 | |
| MRELOV | 0.1631 | 0.1857 | 0.8783 | 0.3798 | |
| MFALLEEN | -0.1305 | 0.1598 | -0.8168 | 0.4141 | |
| MFGEKIND | -0.1780 | 0.1611 | -1.1050 | 0.2692 | |
| MFWEKIND | -0.1312 | 0.1716 | -0.7644 | 0.4446 | |
| MOPLHOOG | -0.1119 | 0.1606 | -0.6970 | 0.4858 | |
| MOPLMIDD | -0.1689 | 0.1680 | -1.0053 | 0.3148 | |
| MOPLLAAG | -0.2514 | 0.1684 | -1.4929 | 0.1355 | |
| MBERHOOG | 0.1724 | 0.1140 | 1.5126 | 0.1304 | |
| MBERZELF | 0.0725 | 0.1223 | 0.5924 | 0.5536 | |
| MBERBOER | -0.1188 | 0.1368 | -0.8683 | 0.3852 | |
| MBERMIDD | 0.1652 | 0.1120 | 1.4746 | 0.1403 | |
| MBERARBG | 0.0648 | 0.1108 | 0.5853 | 0.5583 | |
| MBERARBO | 0.1481 | 0.1119 | 1.3243 | 0.1854 | |
| MSKA | 0.0050 | 0.1243 | 0.0405 | 0.9677 | |
| MSKB1 | -0.0036 | 0.1221 | -0.0293 | 0.9766 | |
| MSKB2 | 0.0401 | 0.1108 | 0.3620 | 0.7174 | |
| MSKC | 0.0622 | 0.1209 | 0.5141 | 0.6072 | |
| MSKD | -0.0696 | 0.1178 | -0.5909 | 0.5546 | |
| MHHUUR | -14.6780 | 970.7529 | -0.0151 | 0.9879 | |
| MHKOOP | -14.6452 | 970.7529 | -0.0151 | 0.9880 | |
| MAUT1 | 0.2700 | 0.1855 | 1.4555 | 0.1455 | |
| MAUT2 | 0.1667 | 0.1676 | 0.9947 | 0.3199 | |
| MAUT0 | 0.1498 | 0.1746 | 0.8575 | 0.3912 | |
| MZFONDS | -14.2061 | 1097.5164 | -0.0129 | 0.9897 | |
| MZPART | -14.2590 | 1097.5164 | -0.0130 | 0.9896 | |
| MINKM30 | 0.0907 | 0.1191 | 0.7613 | 0.4465 | |
| MINK3045 | 0.0442 | 0.1144 | 0.3863 | 0.6993 | |
| MINK4575 | 0.0163 | 0.1155 | 0.1410 | 0.8878 | |
| MINK7512 | 0.0913 | 0.1215 | 0.7516 | 0.4523 | |
| MINK123M | -0.1420 | 0.1695 | -0.8375 | 0.4023 | |
| MINKGEM | 0.0936 | 0.1125 | 0.8323 | 0.4052 | |
| MKOOPKLA | 0.0843 | 0.0561 | 1.5025 | 0.1330 | |
| PWAPART | 0.5225 | 0.4525 | 1.1546 | 0.2483 | |
| PWABEDR | -0.5155 | 0.8444 | -0.6105 | 0.5416 | |
| PWALAND | -1.4032 | 1.5576 | -0.9009 | 0.3676 | |
| PPERSAUT | 0.1984 | 0.0515 | 3.8505 | 0.0001 | *** |
| PBESAUT | 11.1491 | 409.8225 | 0.0272 | 0.9783 | |
| PMOTSCO | -0.2047 | 0.1388 | -1.4746 | 0.1403 | |

| | | | | | |
|---|---|---|---|---|---|
| PVRAAUT | -2.7313 | 2438.9492 | -0.0011 | 0.9991 | |
| PAANHANG | 0.2379 | 1.4096 | 0.1688 | 0.8659 | |
| PTRACTOR | 1.1172 | 0.7898 | 1.4144 | 0.1572 | |
| PWERKT | -6.1844 | 4591.9914 | -0.0013 | 0.9989 | |
| PBROM | -0.0479 | 0.7092 | -0.0675 | 0.9462 | |
| PLEVEN | -0.2744 | 0.1428 | -1.9218 | 0.0546 | |
| PPERSONG | -0.4464 | 2.3459 | -0.1903 | 0.8491 | |
| PGEZONG | 0.9177 | 1.1247 | 0.8160 | 0.4145 | |
| PWAOREG | 0.8809 | 0.5637 | 1.5626 | 0.1182 | |
| PBRAND | 0.2706 | 0.0939 | 2.8823 | 0.0039 | ** |
| PZEILPL | -5.1678 | 2174.2129 | -0.0024 | 0.9981 | |
| PPLEZIER | -0.4551 | 0.4677 | -0.9731 | 0.3305 | |
| PFIETS | -0.1113 | 0.9248 | -0.1203 | 0.9042 | |
| PINBOED | -0.2620 | 0.8574 | -0.3056 | 0.7599 | |
| PBYSTAND | -0.6676 | 0.4377 | -1.5251 | 0.1272 | |
| AWAPART | -0.9239 | 0.9067 | -1.0190 | 0.3082 | |
| AWABEDR | 0.6951 | 2.4345 | 0.2855 | 0.7753 | |
| AWALAND | 2.7427 | 4.7940 | 0.5721 | 0.5672 | |
| APERSAUT | 0.1834 | 0.2183 | 0.8399 | 0.4010 | |
| ABESAUT | -67.0725 | 2458.9318 | -0.0273 | 0.9782 | |
| AMOTSCO | 0.3531 | 0.3801 | 0.9288 | 0.3530 | |
| AVRAAUT | -1.1088 | 10215.0365 | -0.0001 | 0.9999 | |
| AAANHANG | 0.0001 | 2.4055 | 0.0000 | 1.0000 | |
| ATRACTOR | -4.2004 | 3.1144 | -1.3487 | 0.1774 | |
| AWERKT | 0.6237 | 9171.8688 | 0.0001 | 0.9999 | |
| ABROM | -0.3009 | 2.1443 | -0.1403 | 0.8884 | |
| ALEVEN | 0.5535 | 0.2810 | 1.9699 | 0.0489 | * |
| APERSONG | 0.8135 | 4.8316 | 0.1684 | 0.8663 | |
| AGEZONG | -1.8510 | 2.9160 | -0.6348 | 0.5256 | |
| AWAOREG | -2.8146 | 3.0516 | -0.9223 | 0.3563 | |
| ABRAND | -0.3879 | 0.3379 | -1.1479 | 0.2510 | |
| AZEILPL | NA | NA | NA | NA | |
| APLEZIER | 2.9115 | 1.3090 | 2.2242 | 0.0261 | * |
| AFIETS | 0.6717 | 0.5991 | 1.1213 | 0.2622 | |
| AINBOED | 0.0695 | 1.9121 | 0.0363 | 0.9710 | |
| ABYSTAND | 2.7145 | 1.4673 | 1.8501 | 0.0643 | |

Table 5: Logistic Regression Model

The table 5 shows the results from fitting logistic regression model. From the result, we can observe that AZEILPL, number of surfboard policies, is removed from our logistic regression model as it produces null value. With a significance level of 0.05, only four variables are statistically significance as follows:

1. **PPERSAUT**: Contribution car policies
2. **PBRAND**: Contribution fire policies

3. **ALEVEN**: Number of life insurance
4. **APLEZIER**: Number of boat policies

**Model Evaluation**

```
glm.probs = predict(glm.fit,  newdata = validation, type = "response")
glm.pred = rep(0, length(glm.probs))
glm.pred[glm.probs > 0.5] = 1
y.val = validation$CARAVAN
# Get confusion matrix
ConfusionMatrix = confusionMatrix(table(glm.pred, y.val))
logistic.acc = as.numeric(ConfusionMatrix$overall[1])
print(ConfusionMatrix)

Confusion Matrix and Statistics

        y.val
glm.pred    0    1
       0 1634  103
       1    7    2

               Accuracy : 0.937
                 95% CI : (0.9246, 0.9479)
    No Information Rate : 0.9399
    P-Value [Acc > NIR] : 0.7136

                  Kappa : 0.0258

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.99573
            Specificity : 0.01905
         Pos Pred Value : 0.94070
         Neg Pred Value : 0.22222
             Prevalence : 0.93986
         Detection Rate : 0.93585
   Detection Prevalence : 0.99485
      Balanced Accuracy : 0.50739

       'Positive' Class : 0
```

From the result, the accuracy of logistic regression model is 93.7%.

### 3.2.2 Logistic regression model applying forward stepwise selection

Forward stepwise selection method was used to obtain a subset.

**Model fitting**

```
# Applying forward stepwise selection using StepAIC:
# Full Model Fitting
full.model = formula(glm(CARAVAN~., data = training, family = 'binomial'))

# Fitting Logistic Regression with no predictors, only intercept included
glm.model0 = glm(CARAVAN ~1, data = training, family = 'binomial')
```

```
# Logistic Regression Model
glm.forward = stepAIC(glm.model0, direction = 'forward', scope = full.
   model, k= log(nrow(training)), trace = 0)
forward.anova = glm.forward$anova
```

```
# Create table for the report
xtable(forward.anova, digits = 4, caption = "Analysis of Deviance", label
   = "tab:table6", table.placement = "h!")
```

| | Step | Df | Deviance | Resid. Df | Resid. Dev | AIC |
|---|---|---|---|---|---|---|
| 1 | | | | 4075.0000 | 1841.6438 | 1849.9567 |
| 2 | + PPERSAUT | 1.0000 | 103.8871 | 4074.0000 | 1737.7568 | 1754.3825 |
| 3 | + MKOOPKLA | 1.0000 | 47.9881 | 4073.0000 | 1689.7687 | 1714.7073 |
| 4 | + PBRAND | 1.0000 | 18.8505 | 4072.0000 | 1670.9182 | 1704.1697 |
| 5 | + MBERBOER | 1.0000 | 14.8075 | 4071.0000 | 1656.1107 | 1697.6751 |
| 6 | + APLEZIER | 1.0000 | 11.1603 | 4070.0000 | 1644.9505 | 1694.8277 |
| 7 | + PWALAND | 1.0000 | 8.8476 | 4069.0000 | 1636.1029 | 1694.2930 |
| 8 | + MAUT1 | 1.0000 | 8.9474 | 4068.0000 | 1627.1555 | 1693.6584 |
| 9 | + MINK7512 | 1.0000 | 9.2531 | 4067.0000 | 1617.9024 | 1692.7182 |

Table 6: Analysis of Deviance

The table 6 shows the Analysis of Deviance. The results indicate that 8 variables are added to the final model. As a result, the best subset chosen by forward stepwise selection method contains 8 variables as follows:

1. **PPERSAUT**: Contribution car policies
2. **MKOOPKLA**: Purchasing power class
3. **PBRAND**: Contribution fire policies
4. **MBERBOER**: Farmer
5. **APLEZIER**: Number of boat policies
6. **PWALAND**: Contribution third party insurance (agriculture)
7. **MAUT1**: 1 Car
8. **MINK7512**: Income 75 - 122.000

**Fitting logistic regression model with the chosen subset**

We fitted a logistic regression model with the subset containing 8 variables chosen from the forward stepwise selection method and obtained coefficients from fitted model as shown below.

```
# Summarize the final selected model
best.forward = coef(glm.forward)

# Create table for the report
xtable(best.forward, digits = 4, caption = "Logistic Regression Model
   Applying Forward Stepwise Selection ", label = "tab:table7", table.
   placement = "h!")
```

| | Estimate | Std. Error | z value | Pr(>\|z\|) | |
|---|---|---|---|---|---|
| (Intercept) | -5.7633 | 0.3695 | -15.5986 | 0.0000 | *** |
| PPERSAUT | 0.2405 | 0.0284 | 8.4544 | 0.0000 | *** |
| MKOOPKLA | 0.1352 | 0.0367 | 3.6890 | 0.0002 | *** |
| PBRAND | 0.1950 | 0.0374 | 5.2199 | 0.0000 | *** |
| MBERBOER | -0.2433 | 0.0996 | -2.4432 | 0.0146 | * |
| APLEZIER | 1.7996 | 0.4625 | 3.8907 | 0.0001 | *** |
| PWALAND | -0.6494 | 0.3200 | -2.0293 | 0.0424 | * |
| MAUT1 | 0.1537 | 0.0483 | 3.1854 | 0.0014 | ** |
| MINK7512 | 0.1589 | 0.0506 | 3.1391 | 0.0017 | ** |

Table 7: Logistic Regression Model Applying Forward Stepwise Selection

The table 7 shows the results from fitting logistic regression model with the variables obtained from forward stepwise selection method. It can be observed that all variables are statistically significance.

**Model evaluation**

The model was evaluated using validation set and obtaining confusion matrix to get the model accuracy.

```
prob = predict(glm.forward, validation, type = "response")
pred = ifelse(prob > 0.5, 1, 0)
y.val = validation$CARAVAN

# Get confusion matrix
ConfusionMatrix = confusionMatrix(table(pred, y.val))
forward.acc <- as.numeric(ConfusionMatrix$overall[1])
print(ConfusionMatrix)

Confusion Matrix and Statistics

    y.val
pred    0    1
   0 1638  104
   1    3    1

               Accuracy : 0.9387
                 95% CI : (0.9264, 0.9495)
    No Information Rate : 0.9399
    P-Value [Acc > NIR] : 0.6047

                  Kappa : 0.014

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.998172
            Specificity : 0.009524
         Pos Pred Value : 0.940299
         Neg Pred Value : 0.250000
             Prevalence : 0.939863
         Detection Rate : 0.938144
```

```
   Detection Prevalence : 0.997709
      Balanced Accuracy : 0.503848

       'Positive' Class : 0
```

The accuracy of the logistic regression model applying forward stepwise selection method is equal to 93.87 %, which is greater than the logistic regression model including all variables. Therefore, applying forward stepwise selection method helps improve our prediction.

#### 3.2.3 Logistic regression model applying backward stepwise selection

Backward stepwise selection method was used to obtain a subset.

**Model fitting**

```
# Applying backward stepwise selection using StepAIC:
# Model Formula
# Full Model Fitting
full.model = formula(glm(CARAVAN~., data = training, family = 'binomial'))

# Fitting Logistic Regression with all predictors
glm.model.all = glm(CARAVAN ~ ., data = training, family = 'binomial')

# Backward Stepwise Selection using StepAIC
# Logistic Regression Model with BIC
glm.backward = stepAIC(glm.model.all, direction = 'backward', scope = full
   .model, k= log(nrow(training)), trace = 0)
backward.acc <- as.numeric(ConfusionMatrix$overall[1])
sum3 = glm.backward$anova
```

```
library(xtable)
tab8 = xtable(sum3, digits = 4, caption = "Backward Stepwise Selection",
   label = "tab:table8", table.placement = "h!")
print(tab8, tabular.environment = "longtable")
```

| | Step | Df | Deviance | Resid. Df | Resid. Dev | AIC |
|---|---|---|---|---|---|---|
| 1 | | | | 3991.0000 | 1542.9819 | 2249.5760 |
| 2 | - AZEILPL | 0.0000 | 0.0000 | 3991.0000 | 1542.9819 | 2249.5760 |
| 3 | - PVRAAUT | 1.0000 | 0.0000 | 3992.0000 | 1542.9819 | 2241.2631 |
| 4 | - AWERKT | 1.0000 | 0.0000 | 3993.0000 | 1542.9819 | 2232.9503 |
| 5 | - AAANHANG | 1.0000 | 0.0000 | 3994.0000 | 1542.9819 | 2224.6374 |
| 6 | - MSKB1 | 1.0000 | 0.0009 | 3995.0000 | 1542.9828 | 2216.3254 |
| 7 | - AINBOED | 1.0000 | 0.0013 | 3996.0000 | 1542.9841 | 2208.0138 |
| 8 | - PBROM | 1.0000 | 0.0046 | 3997.0000 | 1542.9887 | 2199.7055 |
| 9 | - MSKA | 1.0000 | 0.0081 | 3998.0000 | 1542.9968 | 2191.4008 |
| 10 | - PFIETS | 1.0000 | 0.0151 | 3999.0000 | 1543.0119 | 2183.1030 |
| 11 | - MINK4575 | 1.0000 | 0.0224 | 4000.0000 | 1543.0343 | 2174.8125 |
| 12 | - APERSONG | 1.0000 | 0.0366 | 4001.0000 | 1543.0709 | 2166.5362 |
| 13 | - PPERSONG | 1.0000 | 0.0273 | 4002.0000 | 1543.0981 | 2158.2506 |
| 14 | - MGODOV | 1.0000 | 0.0756 | 4003.0000 | 1543.1737 | 2150.0134 |
| 15 | - AWABEDR | 1.0000 | 0.0881 | 4004.0000 | 1543.2618 | 2141.7885 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 16 | - PZEILPL | 1.0000 | 0.0910 | 4005.0000 | 1543.3528 | 2133.5667 |
| 17 | - MGEMOMV | 1.0000 | 0.1035 | 4006.0000 | 1543.4563 | 2125.3573 |
| 18 | - MGODPR | 1.0000 | 0.1574 | 4007.0000 | 1543.6137 | 2117.2018 |
| 19 | - AWALAND | 1.0000 | 0.2921 | 4008.0000 | 1543.9058 | 2109.1811 |
| 20 | - MINK3045 | 1.0000 | 0.3060 | 4009.0000 | 1544.2118 | 2101.1741 |
| 21 | - PAANHANG | 1.0000 | 0.3065 | 4010.0000 | 1544.5183 | 2093.1678 |
| 22 | - PBESAUT | 1.0000 | 0.3892 | 4011.0000 | 1544.9075 | 2085.2441 |
| 23 | - ABESAUT | 1.0000 | 0.3030 | 4012.0000 | 1545.2105 | 2077.2342 |
| 24 | - MBERZELF | 1.0000 | 0.4294 | 4013.0000 | 1545.6399 | 2069.3508 |
| 25 | - MBERARBG | 1.0000 | 0.1734 | 4014.0000 | 1545.8133 | 2061.2113 |
| 26 | - MOPLHOOG | 1.0000 | 0.2290 | 4015.0000 | 1546.0424 | 2053.1275 |
| 27 | - MSKD | 1.0000 | 0.3399 | 4016.0000 | 1546.3823 | 2045.1546 |
| 28 | - AGEZONG | 1.0000 | 0.4252 | 4017.0000 | 1546.8075 | 2037.2669 |
| 29 | - MAANTHUI | 1.0000 | 0.4847 | 4018.0000 | 1547.2922 | 2029.4388 |
| 30 | - MINKGEM | 1.0000 | 0.5737 | 4019.0000 | 1547.8659 | 2021.6996 |
| 31 | - MINKM30 | 1.0000 | 0.4118 | 4020.0000 | 1548.2777 | 2013.7985 |
| 32 | - PINBOED | 1.0000 | 0.5636 | 4021.0000 | 1548.8413 | 2006.0493 |
| 33 | - MFALLEEN | 1.0000 | 0.5788 | 4022.0000 | 1549.4202 | 1998.3152 |
| 34 | - MFWEKIND | 1.0000 | 0.1628 | 4023.0000 | 1549.5830 | 1990.1651 |
| 35 | - MFGEKIND | 1.0000 | 0.5290 | 4024.0000 | 1550.1120 | 1982.3813 |
| 36 | - PGEZONG | 1.0000 | 0.6324 | 4025.0000 | 1550.7443 | 1974.7008 |
| 37 | - AMOTSCO | 1.0000 | 0.6652 | 4026.0000 | 1551.4096 | 1967.0531 |
| 38 | - APERSAUT | 1.0000 | 0.6074 | 4027.0000 | 1552.0169 | 1959.3476 |
| 39 | - MRELSA | 1.0000 | 0.7753 | 4028.0000 | 1552.7922 | 1951.8100 |
| 40 | - MRELOV | 1.0000 | 0.1968 | 4029.0000 | 1552.9890 | 1943.6939 |
| 41 | - ABROM | 1.0000 | 0.7994 | 4030.0000 | 1553.7884 | 1936.1805 |
| 42 | - MINK123M | 1.0000 | 0.8720 | 4031.0000 | 1554.6604 | 1928.7396 |
| 43 | - MGEMLEEF | 1.0000 | 0.7897 | 4032.0000 | 1555.4501 | 1921.2164 |
| 44 | - MZFONDS | 1.0000 | 0.9794 | 4033.0000 | 1556.4295 | 1913.8830 |
| 45 | - MZPART | 1.0000 | 0.9033 | 4034.0000 | 1557.3328 | 1906.4734 |
| 46 | - PPLEZIER | 1.0000 | 0.9296 | 4035.0000 | 1558.2624 | 1899.0901 |
| 47 | - MOPLMIDD | 1.0000 | 0.9621 | 4036.0000 | 1559.2244 | 1891.7393 |
| 48 | - MSKB2 | 1.0000 | 1.0500 | 4037.0000 | 1560.2745 | 1884.4764 |
| 49 | - MSKC | 1.0000 | 0.9673 | 4038.0000 | 1561.2418 | 1877.1309 |
| 50 | - MAUT2 | 1.0000 | 1.0431 | 4039.0000 | 1562.2849 | 1869.8611 |
| 51 | - MAUT0 | 1.0000 | 0.5757 | 4040.0000 | 1562.8606 | 1862.1240 |
| 52 | - AVRAAUT | 1.0000 | 1.0984 | 4041.0000 | 1563.9591 | 1854.9096 |
| 53 | - PWERKT | 1.0000 | 1.2658 | 4042.0000 | 1565.2248 | 1847.8624 |
| 54 | - ABRAND | 1.0000 | 1.2586 | 4043.0000 | 1566.4834 | 1840.8081 |
| 55 | - MBERARBO | 1.0000 | 1.3554 | 4044.0000 | 1567.8387 | 1833.8506 |
| 56 | - MBERHOOG | 1.0000 | 0.8377 | 4045.0000 | 1568.6764 | 1826.3754 |
| 57 | - MGODRK | 1.0000 | 1.1096 | 4046.0000 | 1569.7860 | 1819.1722 |
| 58 | - MBERMIDD | 1.0000 | 1.1098 | 4047.0000 | 1570.8958 | 1811.9691 |
| 59 | - PMOTSCO | 1.0000 | 1.3859 | 4048.0000 | 1572.2817 | 1805.0421 |
| 60 | - AWAPART | 1.0000 | 1.5234 | 4049.0000 | 1573.8051 | 1798.2526 |
| 61 | - PWAPART | 1.0000 | 0.1945 | 4050.0000 | 1573.9996 | 1790.1343 |
| 62 | - MOSHOOFD | 1.0000 | 1.6398 | 4051.0000 | 1575.6394 | 1783.4612 |
| 63 | - MOSTYPE | 1.0000 | 0.0856 | 4052.0000 | 1575.7249 | 1775.2339 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 64 | - AWAOREG | 1.0000 | 1.8071 | 4053.0000 | 1577.5320 | 1768.7281 |
| 65 | - PTRACTOR | 1.0000 | 1.8699 | 4054.0000 | 1579.4019 | 1762.2851 |
| 66 | - ATRACTOR | 1.0000 | 1.2784 | 4055.0000 | 1580.6804 | 1755.2507 |
| 67 | - MGODGE | 1.0000 | 2.1196 | 4056.0000 | 1582.8000 | 1749.0574 |
| 68 | - PBYSTAND | 1.0000 | 2.0077 | 4057.0000 | 1584.8077 | 1742.7523 |
| 69 | - ABYSTAND | 1.0000 | 1.9490 | 4058.0000 | 1586.7567 | 1736.3884 |
| 70 | - MKOOPKLA | 1.0000 | 2.5899 | 4059.0000 | 1589.3466 | 1730.6654 |
| 71 | - PWABEDR | 1.0000 | 2.6851 | 4060.0000 | 1592.0317 | 1725.0376 |
| 72 | - MRELGE | 1.0000 | 3.1964 | 4061.0000 | 1595.2281 | 1719.9211 |
| 73 | - MHKOOP | 1.0000 | 3.2686 | 4062.0000 | 1598.4966 | 1714.8768 |
| 74 | - PWAOREG | 1.0000 | 3.6027 | 4063.0000 | 1602.0993 | 1710.1667 |
| 75 | - ALEVEN | 1.0000 | 3.6431 | 4064.0000 | 1605.7424 | 1705.4968 |
| 76 | - PLEVEN | 1.0000 | 0.6409 | 4065.0000 | 1606.3833 | 1697.8249 |
| 77 | - MHHUUR | 1.0000 | 3.8997 | 4066.0000 | 1610.2830 | 1693.4117 |
| 78 | - MBERBOER | 1.0000 | 7.4630 | 4067.0000 | 1617.7459 | 1692.5618 |
| 79 | - AFIETS | 1.0000 | 7.5539 | 4068.0000 | 1625.2999 | 1691.8028 |

Table 8: Backward Stepwise Selection

The table 8 shows the Analysis of Deviance. The results indicate that 79 variables were removed from the final model; therefore, the best subset chosen by backward stepwise selection method contains 7 variables as follows:

1. **MOPLLAAG**: Lower level education
2. **MAUT1**: 1 Car
3. **MINK7512**: Income 75 - 122.000
4. **PWALAND**: Contribution third party insurance (agriculture)
5. **PPERSAUT**: Contribution car policies
6. **PBRAND**: Contribution fire policies
7. **APLEZIER**: Number of boat policies

**Fitting Logistic regression model with the chosen subset**

We fitted a logistic regression model with the subset containing 7 variables chosen from the backward forward stepwise selection method and obtained coefficients from the fitted model as shown below:

```
# Summarize the final selected model
best.backward = summary(glm.backward)

# Create table for the report
xtable(best.backward, digits = 4, caption = "Logistic Regression Model
   Applying Backward Stepwise Selection", label = "tab:table9", table.
   placement = "h!")
```

The table 9 shows fitted logistic regression model with the variables obtained from backward stepwise selection method. It can be observed that all variables are statistically significance.

| | Estimate | Std. Error | z value | Pr(>\|z\|) | |
|---|---|---|---|---|---|
| (Intercept) | -4.9211 | 0.3979 | -12.3685 | 0.0000 | *** |
| MOPLLAAG | -0.1292 | 0.0319 | -4.0519 | 0.0001 | *** |
| MAUT1 | 0.1880 | 0.0472 | 3.9869 | 0.0001 | *** |
| MINK7512 | 0.1685 | 0.0505 | 3.3396 | 0.0008 | *** |
| PWALAND | -0.7300 | 0.3203 | -2.2791 | 0.0227 | * |
| PPERSAUT | 0.2408 | 0.0284 | 8.4907 | 0.0000 | *** |
| PBRAND | 0.1976 | 0.0367 | 5.3787 | 0.0000 | *** |
| APLEZIER | 1.7639 | 0.4644 | 3.7981 | 0.0001 | *** |

Table 9: Logistic regression model applying backward stepwise selection

### Model evaluation

```
prob =  predict(glm.backward, validation, type = "response")
pred = ifelse(prob > 0.5, 1, 0)
y.val = validation$CARAVAN
ConfusionMatrix =  confusionMatrix(table(pred, y.val))
backward.acc = as.numeric(ConfusionMatrix$overall[1])
print(ConfusionMatrix)

Confusion Matrix and Statistics

    y.val
pred    0    1
   0 1638  104
   1    3    1

               Accuracy : 0.9387
                 95% CI : (0.9264, 0.9495)
    No Information Rate : 0.9399
    P-Value [Acc > NIR] : 0.6047

                  Kappa : 0.014

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.998172
            Specificity : 0.009524
         Pos Pred Value : 0.940299
         Neg Pred Value : 0.250000
             Prevalence : 0.939863
         Detection Rate : 0.938144
   Detection Prevalence : 0.997709
      Balanced Accuracy : 0.503848

       'Positive' Class : 0
```

From the result, the accuracy of the logistic regression model applying backward stepwise selection is 93.87%, which equals to the logistic regression model applying forward stepwise selection even though some selected variables are different.

### 3.2.4 Ridge regression

**Model fitting Step 1: Find the best lambda using cross-validation**

```
# Find the best lambda using cross-validation
library(glmnet)
set.seed(1)
x.train = model.matrix(CARAVAN~., training)[,-1]
y.train = training$CARAVAN
cv.ridge = cv.glmnet(x.train, y.train, alpha = 0, family = "binomial")
bestlam.ridge = cv.ridge$lambda.min
sprintf('%s is %f', 'The best lambda obtained from the cross-validation',
   bestlam.ridge)
[1] "The best lambda obtained from the cross-validation is 0.037422"
```

With cross-validation method, the best lambda obtained is 0.037422.

**Step 2: Fit a ridge regression model with chosen lambda**

```
ridge.model = glmnet(x.train, y.train, alpha = 0, family = "binomial",
   lambda = bestlam.ridge)
```

**Ridge regression model results**

```
# Display regression coefficients
coefs = coef(ridge.model)
var = rownames(coefs)
rownames(coefs) = NULL
ridge.coefs = cbind(data.frame(Variable =var, Estimate = coefs[,1]))

# Create table for the report
xtable(ridge.coefs, digits = 4, caption = "Ridge Regression Model:
   Coefficients", label = "tab:table10", table.placement = "h!")
```

| | Variable | Estimate |
|---|---|---|
| 1 | (Intercept) | -4.5410 |
| 2 | MOSTYPE | -0.0009 |
| 3 | MAANTHUI | -0.0691 |
| 4 | MGEMOMV | -0.0135 |
| 5 | MGEMLEEF | 0.0459 |
| 6 | MOSHOOFD | -0.0080 |
| 7 | MGODRK | -0.0258 |
| 8 | MGODPR | 0.0164 |
| 9 | MGODOV | 0.0261 |
| 10 | MGODGE | -0.0258 |
| 11 | MRELGE | 0.0289 |
| 12 | MRELSA | -0.0148 |
| 13 | MRELOV | -0.0197 |
| 14 | MFALLEEN | -0.0074 |
| 15 | MFGEKIND | -0.0223 |
| 16 | MFWEKIND | 0.0109 |

| | | |
|---|---|---|
| 17 | MOPLHOOG | 0.0358 |
| 18 | MOPLMIDD | 0.0134 |
| 19 | MOPLLAAG | -0.0309 |
| 20 | MBERHOOG | 0.0244 |
| 21 | MBERZELF | 0.0111 |
| 22 | MBERBOER | -0.0977 |
| 23 | MBERMIDD | 0.0368 |
| 24 | MBERARBG | -0.0161 |
| 25 | MBERARBO | 0.0070 |
| 26 | MSKA | 0.0043 |
| 27 | MSKB1 | 0.0096 |
| 28 | MSKB2 | 0.0123 |
| 29 | MSKC | 0.0093 |
| 30 | MSKD | -0.0373 |
| 31 | MHHUUR | -0.0130 |
| 32 | MHKOOP | 0.0116 |
| 33 | MAUT1 | 0.0563 |
| 34 | MAUT2 | -0.0195 |
| 35 | MAUT0 | -0.0234 |
| 36 | MZFONDS | 0.0056 |
| 37 | MZPART | -0.0082 |
| 38 | MINKM30 | -0.0031 |
| 39 | MINK3045 | -0.0009 |
| 40 | MINK4575 | -0.0013 |
| 41 | MINK7512 | 0.0700 |
| 42 | MINK123M | -0.0907 |
| 43 | MINKGEM | 0.0453 |
| 44 | MKOOPKLA | 0.0432 |
| 45 | PWAPART | 0.0783 |
| 46 | PWABEDR | -0.0692 |
| 47 | PWALAND | -0.1072 |
| 48 | PPERSAUT | 0.0888 |
| 49 | PBESAUT | -0.0295 |
| 50 | PMOTSCO | -0.0488 |
| 51 | PVRAAUT | -0.0862 |
| 52 | PAANHANG | 0.0502 |
| 53 | PTRACTOR | -0.0465 |
| 54 | PWERKT | -0.1335 |
| 55 | PBROM | -0.0536 |
| 56 | PLEVEN | -0.0539 |
| 57 | PPERSONG | -0.0254 |
| 58 | PGEZONG | 0.1307 |
| 59 | PWAOREG | 0.2270 |
| 60 | PBRAND | 0.0715 |
| 61 | PZEILPL | -0.2123 |
| 62 | PPLEZIER | 0.0473 |
| 63 | PFIETS | 0.2362 |
| 64 | PINBOED | -0.1096 |

| | | |
|---|---|---|
| 65 | PBYSTAND | 0.0141 |
| 66 | AWAPART | 0.1078 |
| 67 | AWABEDR | -0.0948 |
| 68 | AWALAND | -0.3469 |
| 69 | APERSAUT | 0.3268 |
| 70 | ABESAUT | -0.1398 |
| 71 | AMOTSCO | 0.0039 |
| 72 | AVRAAUT | -0.2823 |
| 73 | AAANHANG | 0.0846 |
| 74 | ATRACTOR | -0.2007 |
| 75 | AWERKT | -0.1800 |
| 76 | ABROM | -0.1639 |
| 77 | ALEVEN | 0.1261 |
| 78 | APERSONG | -0.0006 |
| 79 | AGEZONG | 0.0500 |
| 80 | AWAOREG | 0.1117 |
| 81 | ABRAND | 0.0721 |
| 82 | AZEILPL | -0.6368 |
| 83 | APLEZIER | 1.2793 |
| 84 | AFIETS | 0.3115 |
| 85 | AINBOED | -0.2269 |
| 86 | ABYSTAND | 0.3971 |

Table 10: Ridge Regression Model: Coefficients

The table 10 indicate that all variables are included in the ridge regression model.

**Model evaluation**

```
# Make predictions on the validation data
x.test = model.matrix(CARAVAN~., validation)[,-1]
probs = predict(ridge.model,newx = x.test)
pred = ifelse(probs > 0.5, 1, 0)
```

```
# Model accuracy
y.val = validation$CARAVAN
ConfusionMatrix = confusionMatrix(table(pred, y.val))
ridge.acc = as.numeric(ConfusionMatrix$overall[1])
print(ConfusionMatrix)
Confusion Matrix and Statistics

    y.val
pred    0    1
   0 1641  104
   1    0    1

               Accuracy : 0.9404
                 95% CI : (0.9283, 0.9511)
    No Information Rate : 0.9399
    P-Value [Acc > NIR] : 0.4858
```

```
                  Kappa : 0.0178

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 1.000000
            Specificity : 0.009524
         Pos Pred Value : 0.940401
         Neg Pred Value : 1.000000
             Prevalence : 0.939863
         Detection Rate : 0.939863
   Detection Prevalence : 0.999427
      Balanced Accuracy : 0.504762

       'Positive' Class : 0
```

From the result, the accuracy of ridge regression model is 94.04%.

### 3.2.5 Lasso model

**Model fitting Step 1: Find the best lambda using cross-validation**

```
# Find the best lambda using cross-validation
set.seed(1)
x.train = model.matrix(CARAVAN~., training)[,-1]
y.train = training$CARAVAN
cv.lasso = cv.glmnet(x.train, y.train, alpha = 1, family = "binomial")
bestlam.lasso = cv.lasso$lambda.min
sprintf('%s is %f', 'The best lambda obtained from the cross-validation',
   bestlam.lasso)
[1] "The best lambda obtained from the cross-validation is 0.004303"
```

**Step 2: Fit a lasso model with chosen lambda ¡¡¿¿= @**

```
# Fit the final model on the training data
lasso.model = glmnet(x.train, y.train, alpha = 1, family = "binomial",
   lambda = bestlam.lasso)
```

**Non-zero coefficients**

Non-zero coefficients from the lasso model are selected.

```
# Display non-zero coefficients
coefs = coef(lasso.model)
var = rownames(coefs)
rownames(coefs) = NULL
coefs = cbind(data.frame(Variable =var, Estimate = coefs[,1]))
lasso.coefs = coefs[with(coefs, Estimate != 0),]

# Create table for the report
tab11 = xtable(lasso.coefs, digits = 4, caption = "Lasso Model: Non-Zero
   Coefficients", label = "tab:table11", table.placement = "h!")
print(tab11, tabular.environment = "longtable")
```

| | Variable | Estimate |
|---|---|---|
| 1 | (Intercept) | -4.8518 |
| 10 | MGODGE | -0.0219 |
| 11 | MRELGE | 0.0364 |
| 17 | MOPLHOOG | 0.0112 |
| 19 | MOPLLAAG | -0.0533 |
| 22 | MBERBOER | -0.1157 |
| 23 | MBERMIDD | 0.0179 |
| 31 | MHHUUR | -0.0081 |
| 33 | MAUT1 | 0.0888 |
| 41 | MINK7512 | 0.0756 |
| 43 | MINKGEM | 0.0319 |
| 44 | MKOOPKLA | 0.0721 |
| 45 | PWAPART | 0.0783 |
| 47 | PWALAND | -0.1418 |
| 48 | PPERSAUT | 0.1869 |
| 59 | PWAOREG | 0.1676 |
| 60 | PBRAND | 0.1118 |
| 69 | APERSAUT | 0.0624 |
| 74 | ATRACTOR | -0.1334 |
| 83 | APLEZIER | 1.4074 |
| 84 | AFIETS | 0.4445 |
| 86 | ABYSTAND | 0.3055 |

Table 11: Lasso Model: Non-Zero Coefficients

The table 11 shows non-zero coefficients obtained from the lasso model. There are 21 variables not shrinking to zero, while 64 variables shrink to zero.

### Model evaluation

```
# Make predictions on the validation data
x.test = model.matrix(CARAVAN~., validation)[,-1]
probs = predict(lasso.model,newx = x.test)
pred = ifelse(probs > 0.5, 1, 0)
```

```
# Model accuracy
y.val = validation$CARAVAN
ConfusionMatrix = confusionMatrix(table(pred, y.val))
lasso.acc = as.numeric(ConfusionMatrix$overall[1])
print(ConfusionMatrix)

Confusion Matrix and Statistics
  y.val
pred    0    1
   0 1641  104
   1    0    1

               Accuracy : 0.9404
```

```
                95% CI : (0.9283, 0.9511)
    No Information Rate : 0.9399
    P-Value [Acc > NIR] : 0.4858

                  Kappa : 0.0178

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 1.000000
            Specificity : 0.009524
         Pos Pred Value : 0.940401
         Neg Pred Value : 1.000000
             Prevalence : 0.939863
         Detection Rate : 0.939863
   Detection Prevalence : 0.999427
      Balanced Accuracy : 0.504762

       'Positive' Class : 0
```

From the result, the accuracy of lasso model is 94.04%.

#### 3.2.6 Linear Discriminant Analysis Model: LDA

**Model fitting**

```
lda.fit = lda(CARAVAN~., data = training)

# coefficients of LDA model
lda.coefs = lda.fit$scaling
var = rownames(lda.coefs)
rownames(lda.coefs) = NULL
lda.coefs = cbind(data.frame(Variable =var, LD1 = lda.coefs[,1]))

# Create table for report
tab12 = xtable(lda.coefs, digits = 4, caption = "Linear Discriminant
   Analysis Model: LDA", label = "tab:table12", table.placement = "h!")
print(tab12, tabular.environment = "longtable")
```

| | Variable | LD1 |
|---|---|---|
| 1 | MOSTYPE | 0.0468 |
| 2 | MAANTHUI | -0.0806 |
| 3 | MGEMOMV | -0.0426 |
| 4 | MGEMLEEF | 0.1118 |
| 5 | MOSHOOFD | -0.2167 |
| 6 | MGODRK | -0.1077 |
| 7 | MGODPR | -0.0462 |
| 8 | MGODOV | -0.0231 |
| 9 | MGODGE | -0.0987 |
| 10 | MRELGE | 0.1692 |
| 11 | MRELSA | 0.1001 |
| 12 | MRELOV | 0.0933 |
| 13 | MFALLEEN | -0.0949 |

| | | |
|---|---|---|
| 14 | MFGEKIND | -0.1431 |
| 15 | MFWEKIND | -0.0822 |
| 16 | MOPLHOOG | -0.0376 |
| 17 | MOPLMIDD | -0.1168 |
| 18 | MOPLLAAG | -0.2076 |
| 19 | MBERHOOG | 0.0829 |
| 20 | MBERZELF | 0.0124 |
| 21 | MBERBOER | -0.0648 |
| 22 | MBERMIDD | 0.0752 |
| 23 | MBERARBG | 0.0031 |
| 24 | MBERARBO | 0.0522 |
| 25 | MSKA | -0.0261 |
| 26 | MSKB1 | -0.0486 |
| 27 | MSKB2 | 0.0066 |
| 28 | MSKC | 0.0312 |
| 29 | MSKD | -0.0210 |
| 30 | MHHUUR | -0.8000 |
| 31 | MHKOOP | -0.7657 |
| 32 | MAUT1 | 0.2055 |
| 33 | MAUT2 | 0.1329 |
| 34 | MAUT0 | 0.1226 |
| 35 | MZFONDS | -0.8963 |
| 36 | MZPART | -0.9520 |
| 37 | MINKM30 | 0.0812 |
| 38 | MINK3045 | 0.0304 |
| 39 | MINK4575 | -0.0015 |
| 40 | MINK7512 | 0.0763 |
| 41 | MINK123M | -0.1661 |
| 42 | MINKGEM | 0.1340 |
| 43 | MKOOPKLA | 0.0604 |
| 44 | PWAPART | 0.3662 |
| 45 | PWABEDR | -0.2447 |
| 46 | PWALAND | -0.4593 |
| 47 | PPERSAUT | 0.1192 |
| 48 | PBESAUT | -0.0046 |
| 49 | PMOTSCO | -0.1703 |
| 50 | PVRAAUT | -0.3224 |
| 51 | PAANHANG | 0.1471 |
| 52 | PTRACTOR | 0.1293 |
| 53 | PWERKT | -0.3403 |
| 54 | PBROM | 0.0330 |
| 55 | PLEVEN | -0.2828 |
| 56 | PPERSONG | 0.0709 |
| 57 | PGEZONG | 2.0686 |
| 58 | PWAOREG | 1.4925 |
| 59 | PBRAND | 0.2333 |
| 60 | PZEILPL | -0.2322 |
| 61 | PPLEZIER | -1.0494 |

| | | |
|---|---|---|
| 62 | PFIETS | -1.0890 |
| 63 | PINBOED | -0.1370 |
| 64 | PBYSTAND | -1.1625 |
| 65 | AWAPART | -0.5009 |
| 66 | AWABEDR | 0.2812 |
| 67 | AWALAND | 0.6754 |
| 68 | APERSAUT | 0.2466 |
| 69 | ABESAUT | -0.2711 |
| 70 | AMOTSCO | 0.3764 |
| 71 | AVRAAUT | 0.7094 |
| 72 | AAANHANG | -0.0570 |
| 73 | ATRACTOR | -0.6363 |
| 74 | AWERKT | 0.3035 |
| 75 | ABROM | -0.1373 |
| 76 | ALEVEN | 0.6742 |
| 77 | APERSONG | -0.1552 |
| 78 | AGEZONG | -4.0971 |
| 79 | AWAOREG | -5.2523 |
| 80 | ABRAND | -0.3482 |
| 81 | AZEILPL | -0.6966 |
| 82 | APLEZIER | 6.4779 |
| 83 | AFIETS | 1.4865 |
| 84 | AINBOED | -0.1154 |
| 85 | ABYSTAND | 4.9815 |

Table 12: Linear Discriminant Analysis Model: LDA

The table 12 shows the calculated coefficients for all variables including in the LDA model. Because our target contains two groups, the results indicate only LD1 in this case.

**Model evaluation**

```
# Make predictions on the validation data
lda.pred = predict(lda.fit,newdata = validation)
lda.class = lda.pred$class
head(lda.class)

# Model accuracy
y.val = validation$CARAVAN
ConfusionMatrix = confusionMatrix(table(lda.class, y.val))
lda.acc = as.numeric(ConfusionMatrix$overall[1])
print(ConfusionMatrix)

Confusion Matrix and Statistics

          y.val
lda.class    0    1
        0 1621   99
        1   20    6
```

```
               Accuracy : 0.9318
                 95% CI : (0.919, 0.9432)
    No Information Rate : 0.9399
    P-Value [Acc > NIR] : 0.9257

                  Kappa : 0.0694

 Mcnemar's Test P-Value : 8.662e-13

            Sensitivity : 0.98781
            Specificity : 0.05714
         Pos Pred Value : 0.94244
         Neg Pred Value : 0.23077
             Prevalence : 0.93986
         Detection Rate : 0.92841
   Detection Prevalence : 0.98511
      Balanced Accuracy : 0.52248

       'Positive' Class : 0
```

The confusion matrix shows that the accuracy of LDA model is 93.18%, lower than other models, therefore, LDA might not be a model for predicting an interest in buying a caravan insurance policy.

#### 3.2.7 K-Nearest Neighbor: KNN

**K optimal value selection**

```
# Select X from training set
train.X = cbind(training[-ncol(training)])

# Select X from validation set
test.X = cbind(validation[-ncol(validation)])

# Select y from training set
train.y = training$CARAVAN

# Compute optimal value of k
i = 1
k.optm = 1
for (i in 1:100){
  knn.mod =  knn(train.X,test.X,train.y, k = i)
  k.optm[i] = 100 * sum(y.val == knn.mod)/NROW(y.val)
  k=i
}
sprintf("%s is %i", "The maximum k",which.max(k.optm))
[1] "The maximum k is 20"
```

**K optimal value**

```
plot(k.optm, type="b", xlab="K-Value",ylab="Accuracy level", xlim = c(0,30
   ))
points(20, k.optm[20], col = 'red', pch = 19)
```

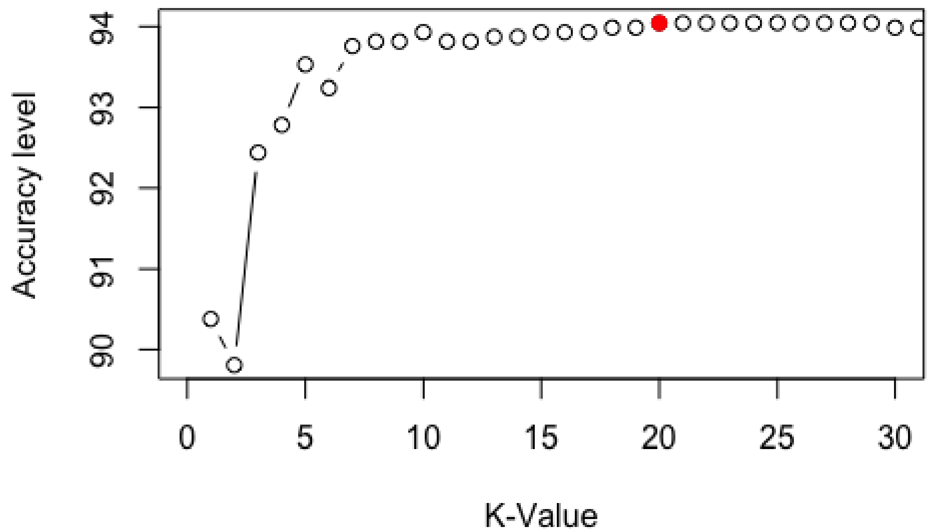


Figure 6: K-Optimal Value.

The figure 6 shows the optimal value of k is 20.

**Model fitting**
We fitted KNN model with k = 20 as follow:

```
set.seed(1)
knn.pred = knn(train.X,test.X,train.y, k = 20)
```

**Model evaluation**

```
# Model accuracy
y.val = validation$CARAVAN
ConfusionMatrix = confusionMatrix(table(knn.pred, y.val))
knn.acc = as.numeric(ConfusionMatrix$overall[1])
print(ConfusionMatrix)

Confusion Matrix and Statistics

        y.val
knn.pred    0    1
       0 1641  104
       1    0    1

               Accuracy : 0.9404
                 95% CI : (0.9283, 0.9511)
    No Information Rate : 0.9399
    P-Value [Acc > NIR] : 0.4858

                  Kappa : 0.0178

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 1.000000
            Specificity : 0.009524
```

```
         Pos Pred Value : 0.940401
         Neg Pred Value : 1.000000
             Prevalence : 0.939863
         Detection Rate : 0.939863
   Detection Prevalence : 0.999427
      Balanced Accuracy : 0.504762

       'Positive' Class : 0
```

The confusion matrix shows that the accuracy of KNN model with k-optimal value of 20 is 94.04%, which is high.

#### 3.2.8 Bagging technique model

We used bagging technique to construct a more powerful prediction model.

**Model fitting**

```
set.seed(1)
bag.caravan = randomForest(CARAVAN~., data = training, mtry = 85, ntree =
   1000)
bag.caravan

Call:
 randomForest(formula = CARAVAN ~ ., data = training, mtry = 85,
    ntree = 1000)
               Type of random forest: regression
                     Number of trees: 1000
No. of variables tried at each split: 85

          Mean of squared residuals: 0.06147344
                    % Var explained: -9.65
```

**Variable importance**

The most 20 important variables are selected from the bagging approach model can be shown as follow:

```
# Variable Important Measure: 20 important variables from bagging
   technique
imp = as.data.frame(importance(bag.caravan))
imp = data.frame(variable = rownames(imp),importance = imp$IncNodePurity)
imp = imp[order(imp$importance,decreasing = T),]
imp = imp[1:20,]
imp$variable = factor(imp$variable, levels = imp$variable[order(imp$
   importance, decreasing = FALSE)])

# Feature importance plot
ggplot(imp, aes(x=variable, y= importance, fill = importance))+
  geom_bar(stat = 'identity') + coord_flip() +
  scale_fill_gradient2(low = "yellow", mid = "skyblue", high = "blue",
     midpoint = max(imp$importance)/2)+
  labs(y = "Correlation") +
  xlab("Variable") + ylab("IncNodePurity") + theme(legend.position = 'none
     ')
```

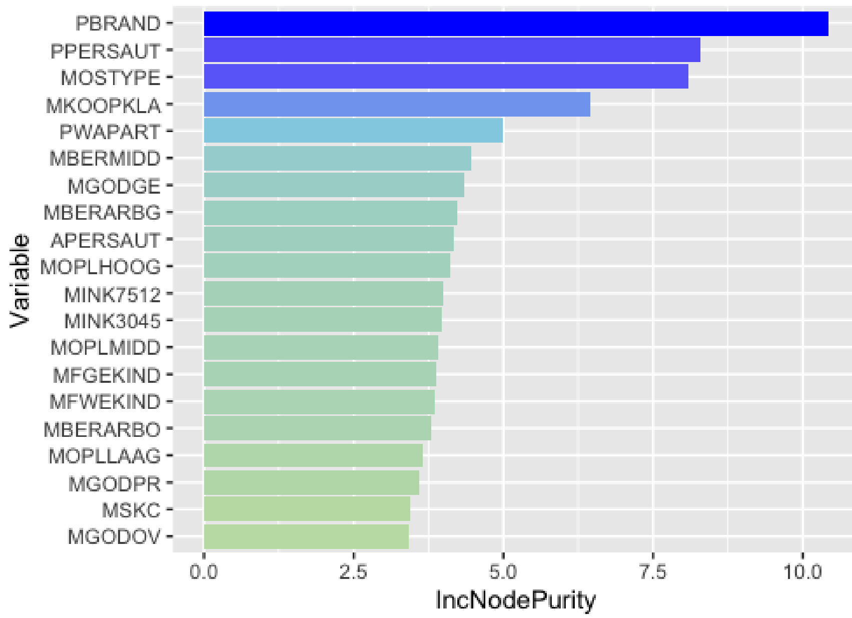


Figure 7: Variable Importance Using Bagging Technique.

The figure 7 shows top 20 variables importance. It can be observed that there are four variables seem more important than others. Therefore, we chose these four variables to re-fit our model to improve model performance as follows:

1. **PBRAND**: Contribution fire policies
2. **PPERSAUT**: Contribution car policies
3. **MOSTYPE**: Customer subtype
4. **MKOOPKLA**: Purchasing power class

**Re-fitting bagging approach model with importance variables**

```
# Fitting a random forest of classification tree model with importance
   variables from bagging method having importance level equal or greater
   than 5.
bag.caravan.imp = randomForest(CARAVAN~PBRAND + PPERSAUT + MOSTYPE +
   MKOOPKLA, data = training, mtry = 4, ntree = 1000)
bag.caravan.imp

Call:
 randomForest(formula = CARAVAN ~ PBRAND + PPERSAUT + MOSTYPE +
    MKOOPKLA, data = training, mtry = 4, ntree = 1000)
               Type of random forest: regression
                     Number of trees: 1000
No. of variables tried at each split: 4

          Mean of squared residuals: 0.05578325
                    % Var explained: 0.5
```

**Model evaluation**

```
# Make predictions on the validation data
bag.probs = predict(bag.caravan.imp,newdata = validation)
bag.pred = ifelse(bag.probs > 0.5, 1, 0)

# Model accuracy
y.val = validation$CARAVAN
ConfusionMatrix = confusionMatrix(table(bag.pred, y.val))
bag.acc = as.numeric(ConfusionMatrix$overall[1])
print(ConfusionMatrix)

Confusion Matrix and Statistics

        y.val
bag.pred    0    1
       0 1640  104
       1    1    1

               Accuracy : 0.9399
                 95% CI : (0.9277, 0.9506)
    No Information Rate : 0.9399
    P-Value [Acc > NIR] : 0.5259

                  Kappa : 0.0165

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.999391
            Specificity : 0.009524
         Pos Pred Value : 0.940367
         Neg Pred Value : 0.500000
             Prevalence : 0.939863
         Detection Rate : 0.939290
   Detection Prevalence : 0.998855
      Balanced Accuracy : 0.504457

       'Positive' Class : 0
```

The confusion matrix shows that the accuracy of bagging approach model using high variable importance is 93.99%.

### 3.2.9 Model comparison

```
# Model comparison
Accuracy = c(logistic.acc, forward.acc, backward.acc, ridge.acc, lasso.acc
   , lda.acc, knn.acc,  bag.acc)
Model = c("Logistic Regression", "Forward Selection", "Backward Selection"
   , "Ridge Regression", "Lasso Regresion", "LDA", "KNN",  "Bagging")
Model.selection = data.frame(Model = Model, Accuracy = Accuracy)
Model.selection = Model.selection[order(Model.selection$Accuracy,
   decreasing = T),]
Model.selection
```

```
# Create table for report
xtable(Model.selection, digits = 4, caption = "Model Comparison", label =
  "tab:table13", table.placement = "h!")
```

| | Model | Accuracy |
|---|---|---|
| 4 | Ridge Regression | 0.9404 |
| 5 | Lasso Regresion | 0.9404 |
| 7 | KNN | 0.9404 |
| 8 | Bagging | 0.9399 |
| 2 | Forward Selection | 0.9387 |
| 3 | Backward Selection | 0.9387 |
| 1 | Logistic Regression | 0.9370 |
| 6 | LDA | 0.9318 |

Table 13: Model Comparison

The table 13 indicates all models performance. From the results, it can be observed that the ridge regression, lasso, and k-nearest neighbor models have the same accuracy rates, which is equal to 94.04%, and perform the best. The model using bagging approach has the accuracy of 93.99%. The models applying forward stepwise selection and backward stepwise selection are not different, while the logistic regression model with all predictors is not good as much as the logistic regression model with selected variables using forward and backward stepwise selection methods. In addition, linear discriminant analysis performs worst. This may be concluded that LDA is not good for predicting customers who are more likely to buy a caravan policy. Since there are 3 models performing the best and having the same accuracy, we may need to compare the algorithm that is the best for prediction to consider the model that we would use to predict a customer purchases a caravan insurance policy. The followings are each model advantage and disadvantage:

- **KNN classifier** predicts the class of a given test observation by identifying the observations that are nearest to it; therefore, the KNN algorithm can compete with the most accurate models because it makes highly accurate predictions. We can use the KNN algorithm for applications that require high accuracy. However, we will not able to extract feature importance from this model [2].
- **Ridge regression** is the method used for the analysis of multicollinearity in multiple regression data. Ridge regression can still perform well by trading off a small increase in bias for a large decrease in variance when predictors are greater than observations. Therefore, ridge regression is most suitable when a data set contains a higher number of predictor variables than number of observations. However, ridge regression includes all predictors in the final model. This may not be a problem for prediction accuracy, but it can create a challenge in model interpretation in settings in which the number of variables is quite large as our dataset [2].
- **Lasso regression** is introduced in order to improve the prediction accuracy and interoperability of regression models. Lasso regression will automatically select those features that are useful, discarding the useless or redundant features by making its coefficient equal to zero [2].

Therefore, we chose the lasso regression model to be the model used to predict an interest in buying a caravan policy of a customer since it can help us improve the prediction accuracy as well as allowing us to explain why people would buy a caravan insurance policy based on selected variables, because it overcomes the disadvantage of ridge regression and select useful features that we cannot find from using KNN model.

# 4 Prediction

We used TICDATA2000 to find the optimal value of lambda and train the lasso model.

### Find the best lambda using cross-validation

```
# Find the best lambda using cross-validation
set.seed(1)
x.train = model.matrix(CARAVAN~., ticdata2000)[,-1]
y.train = ticdata2000$CARAVAN
cv.lasso = cv.glmnet(x.train, y.train, alpha = 1, family = "binomial")
bestlam.lasso = cv.lasso$lambda.min
sprintf('%s is %f', 'The best lambda obtained from the cross-validation',
   bestlam.lasso)
[1] "The best lambda obtained from the cross-validation is 0.002902"
```

### Model fitting

```
# Fit the final model on the training data
lasso.model = glmnet(x.train, y.train, alpha = 1, family = "binomial",
   lambda = bestlam.lasso)
```

### Non-zero coefficients

```
# Display non-zero coefficients
coefs = coef(lasso.model)
var = rownames(coefs)
rownames(coefs) = NULL
coefs = cbind(data.frame(Variable =var, Estimate = coefs[,1]))
lasso.coefs = coefs[with(coefs, Estimate != 0),]
lasso.coefs

# Create table for the report
tab14 = xtable(lasso.coefs, digits = 4, caption = "Lasso Model: Non-Zero
   Coefficients", label = "tab:table14", table.placement = "h!")
print(tab14, tabular.environment = "longtable")
```

| | Variable | Estimate |
|---|---|---|
| 1 | (Intercept) | -4.8319 |
| 5 | MGEMLEEF | 0.0382 |
| 7 | MGODRK | -0.0092 |
| 8 | MGODPR | 0.0187 |
| 10 | MGODGE | -0.0102 |
| 11 | MRELGE | 0.0494 |
| 12 | MRELSA | -0.0152 |
| 17 | MOPLHOOG | 0.0491 |
| 19 | MOPLLAAG | -0.0510 |
| 22 | MBERBOER | -0.1263 |

| | | |
|---|---|---|
| 23 | MBERMIDD | 0.0306 |
| 30 | MSKD | -0.0028 |
| 31 | MHHUUR | -0.0192 |
| 33 | MAUT1 | 0.0458 |
| 38 | MINKM30 | -0.0057 |
| 41 | MINK7512 | 0.0236 |
| 42 | MINK123M | -0.0933 |
| 43 | MINKGEM | 0.0446 |
| 44 | MKOOPKLA | 0.0421 |
| 45 | PWAPART | 0.1229 |
| 47 | PWALAND | -0.1279 |
| 48 | PPERSAUT | 0.2035 |
| 54 | PWERKT | -0.0389 |
| 58 | PGEZONG | 0.0926 |
| 59 | PWAOREG | 0.1535 |
| 60 | PBRAND | 0.1039 |
| 63 | PFIETS | 0.1047 |
| 74 | ATRACTOR | -0.0610 |
| 82 | AZEILPL | 0.8141 |
| 83 | APLEZIER | 1.8316 |
| 84 | AFIETS | 0.2741 |
| 86 | ABYSTAND | 0.3898 |

Table 14: Lasso Model: Non-Zero Coefficients

The table 14 shows non-zero coefficients from the lasso regression model. There are 31 variables that are not equal to zero.

**Model prediction**

After fitting the chosen model to the TICDATA2000 dataset, we applied the fitted model to the TICEVAL2000 dataset to obtain the predictions.

```
# Make predictions on the TICEVAL2000 data
# Make predictions on the TICEVAL2000 data
x.test = as.matrix(ticeval2000)
probs = predict(lasso.model,newx = x.test)
prediction = ifelse(probs > 0.5, 1, 0)
table(prediction)

prediction
   0    1
3997    3
```

Our predictions contain 3,997 customer records did not purchase a caravan insurance policy, and only 3 customer records purchased a caravan insurance policy. Therefore, only 0.08% of customers purchased insurance.

## 5 Evaluation

We evaluated model using the TICTGTS2000 set to obtain the prediction accuracy.

```
# Evaluating model
# Model accuracy
target = tictgts2000$CARAVAN
ConfusionMatrix = confusionMatrix(table(prediction, target))
print(ConfusionMatrix)

Confusion Matrix and Statistics

          target
prediction    0    1
         0 3760  237
         1    2    1

               Accuracy : 0.9402
                 95% CI : (0.9325, 0.9474)
    No Information Rate : 0.9405
    P-Value [Acc > NIR] : 0.5438

                  Kappa : 0.0068

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.999468
            Specificity : 0.004202
         Pos Pred Value : 0.940706
         Neg Pred Value : 0.333333
             Prevalence : 0.940500
         Detection Rate : 0.940000
   Detection Prevalence : 0.999250
      Balanced Accuracy : 0.501835

       'Positive' Class : 0
```

The results show that the prediction accuracy is 94.02%, which is not much different than the training accuracy.

## 6 Conclusion

To predict whether customers would buy a caravan insurance policy, we used different prediction techniques to select best performed model including logistic regression, forward stepwise selection, backward stepwise selection, ridge regression, lasso regression, linear discriminant analysis, k-nearest neighbor, and bagging approach. Each technique has different method to choose a set of features that is useful for predicting customers purchase insurance.

Among these models, the lasso regression is the model we chose to predict an interest in buying a caravan insurance policy although there are three models having highest accuracy, which is 94.04%, since the lasso regression model using regularization which is a technique that can be used to improve a model and also good for feature selection as it tries to minimize the cost function and select those useful features. It is believed that with the lasso regression model, we would be able to predict who would be interested in buying a caravan insurance policy and why people would buy this insurance policy based on the selected variables as followings:

1. **MGEMLEEF:** Average age
2. **MGODPK:** Roman catholic
3. **MGODPR:** Protestant
4. **MGODGE:** No religion
5. **MRELGE:** Married
6. **MRELSA:** Living together
7. **MOPLHOOG:** High level education
8. **MOPLLAAG:** Lower level education
9. **MBERBOER:** Farmer
10. **MBERMIDD:** Middle management
11. **MSKD:** Social class D
12. **MHHUUR:** Rented house
13. **MAUTI:** 1 Car
14. **MINKM30:** Income $> 30.000$
15. **MINK7512:** Income $75 - 122.000$
16. **MINK123M:** Income $< 123.000$
17. **MINKGEM:** Average income
18. **MKOOPKLA:** Purchasing power class
19. **PWAPART:** Contribution private third party insurance
20. **PWALAND:** Contribution third party insurance (agriculture)
21. **PPERSAUT:** Contribution car policies
22. **PWERKT:** Contribution agricultural machine policies
23. **PGEZONG:** Contribution private accident insurance policies
24. **PWAOREG:** Contribution disability insurance policies
25. **PBRAND:** Contribution fire policies
26. **PFIETS:** Contribution boat policies
27. **ATRACTOR:** Number of tractor policies
28. **AZEILPL:** Number of surfboard policies
29. **APLEZIER:** Number of boat policies
30. **AFIETS:** Number of bicycle policies

31. **ABYSTAND:** Number of social security insurance policies

The number of selected variables when all customer records from the TICDATA2000 set used is greater than the number of variables when customer records were split into training and validation sets because the features increased from 21 variables to 31 variables. Based on the lasso regression model, the socio-demographic related variables are selected, such as age, education level, and marriage status. The income level of the customer including income and purchasing power related variables is also selected to be important.

The product usage related variables are also included in the lasso regression model. The features measuring the number and contribution of insurance policies are predictors for caravan insurance policy purchases, such as contribution private third party insurance, contribution car policies, contribution fire policies, number of boat policies, and number of bicycle policies. In some sense, people who buy any kind of insurance are more likely to buy a caravan insurance policy.

However, PPERSAUT variable was found to be relevant by all selected methods we used to select the best model for prediction. Based on the correlation coefficient between target variable and feature, PPERSAUT is the most correlated variable. The lasso regression model used for prediction includes this variable as well. Clearly, the strong predictor of caravan insurance policy purchases is the feature measuring the contribution to car policy purchase. This may be concluded that we might be able to provide more accurate prediction for whether a customer purchases a caravan insurance policy if we include this variable in our model.

After we chose the lasso regression model, we fitted the model with training set and made predictions using test set. After we got our predictions, we evaluated these predictions with the provided targets. The model accuracy is 94.02% meaning that our model is able to classify 3,761 customer records from 4,000 customer records. Below is showing our prediction results compared to target data.

```
pred.sum = table(prediction)
target.sum = table(target)
# Create two-way table
data = matrix(c(3997, 3762, 3, 238), ncol = 2)
rownames(data) = c("Prediction","Target")
colnames(data) = c("0", "1")
barplot(data, legend = TRUE, beside = TRUE, ylim = c(0,5000), col = c("#
   eb8060","#b9e38d"),xlab = "CARAVAN")
text(x= 1.5, y = 4200, labels = "3997")
text(x= 2.5, y = 4000, labels = "3762")
text(x= 4.5, y = 200, labels = "3")
text(x= 5.5, y = 500, labels = "238")
```

The figure 8 shows the comparison between prediction results and given target data. We can see that the number of 0 caravan insurance policy for both prediction and target is not much different, but our model seems not to predict people who are more likely to but a caravan insurance policy correctly. This may be caused by the imbalanced class as 0 is the majority class and 1 is the minority class in these data. Therefore, our model seems over-classify the larger class like 0 due to its increased prior probability.

Although our model predicted there are 3 customers would buy a caravan insurance policy, only one customer was predicted correctly compared to our target data. From the result, a customer from customer record of 576 from the test set is interested in purchasing a caravan insurance policy

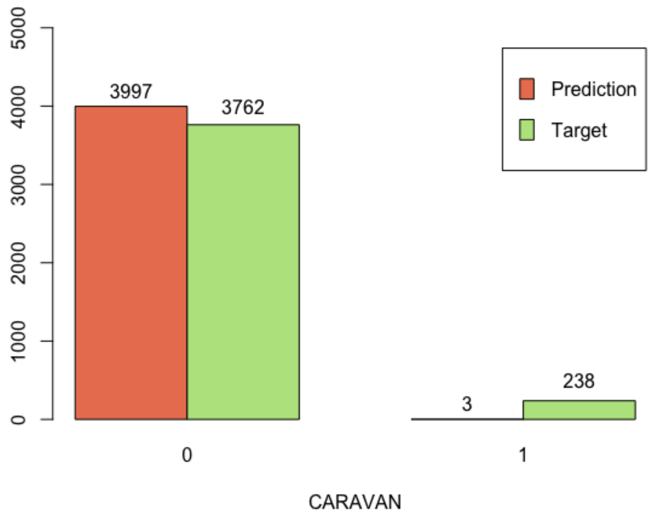


Figure 8: Prediction vs. Target

as shown below.

```
# Create table for test set containing target and prediction
pred.target = cbind(ticeval2000,prediction, target)

# Select only useful features from lasso
print(lasso.coefs[,1])
pred.target= pred.target[, c( "MGEMLEEF","MGODRK","MGODPR",  "MGODGE","
   MRELGE","MRELSA","MOPLHOOG","MOPLLAAG","MBERBOER",  "MBERMIDD","MSKD","
   MHHUUR","MAUT1","MINKM30","MINK7512", "MINK123M","MINKGEM","MKOOPKLA","
   PWAPART","PWALAND","PPERSAUT", "PWERKT","PGEZONG","PWAOREG","PBRAND","
   PFIETS","ATRACTOR", "AZEILPL","APLEZIER","AFIETS","ABYSTAND","
   prediction","target" )]

# Select only target of 1
target1 = subset(pred.target, target == 1)

# Select only the prediction of 1
pred1 = subset(target1, prediction == 1)
# Create table for report
data = unlist(pred1[1,], use.names = FALSE)
data = matrix(data, ncol = 1)
colnames(data) = "Data"
rownames(data) = c( "MGEMLEEF","MGODRK","MGODPR",  "MGODGE","MRELGE","
   MRELSA","MOPLHOOG","MOPLLAAG","MBERBOER",  "MBERMIDD","MSKD","MHHUUR","
   MAUT1","MINKM30","MINK7512", "MINK123M","MINKGEM","MKOOPKLA","PWAPART",
   "PWALAND","PPERSAUT", "PWERKT","PGEZONG","PWAOREG","PBRAND","PFIETS","
   ATRACTOR", "AZEILPL","APLEZIER","AFIETS","ABYSTAND","prediction","
   target" )
descrb = ticdatadescr[c(4,6,7,9,10,11,16,18,21,22,29,30,32,37,40,41,42,43,
   44,46,47,53,56,58,59,62,73,81,82,83,85),]
descrb = as.data.frame(descrb[,2])
```

```
descrb[nrow(descrb) + 1,] = "Number of mobile home policies"
descrb[nrow(descrb) + 1,] = "Number of mobile home policies"
data = cbind(data,descrb)

tab15 = xtable(data, digits = 0, caption = "Customer who purchases a
   caravan insurance policy", label = "tab15")
print(tab15, tabular.environment = "longtable")
```

| | Data | Description |
|---|---|---|
| MGEMLEEF | 4 | Average age |
| MGODRK | 2 | Roman catholic |
| MGODPR | 2 | Protestant ... |
| MGODGE | 5 | No religion |
| MRELGE | 7 | Married |
| MRELSA | 1 | Living together |
| MOPLHOOG | 5 | High level education |
| MOPLLAAG | 2 | Lower level education |
| MBERBOER | 0 | Farmer |
| MBERMIDD | 0 | Middle management |
| MSKD | 0 | Social class D |
| MHHUUR | 0 | Rented house |
| MAUT1 | 3 | 1 car |
| MINKM30 | 0 | Income >30.000 |
| MINK7512 | 2 | Income 75-122.000 |
| MINK123M | 1 | Income <123.000 |
| MINKGEM | 6 | Average income |
| MKOOPKLA | 7 | Purchasing power class |
| PWAPART | 0 | Contribution private third party insurance |
| PWALAND | 0 | Contribution third party insurance (agriculture) |
| PPERSAUT | 6 | Contribution car policies |
| PWERKT | 0 | Contribution agricultural machines policies |
| PGEZONG | 0 | Contribution private accident insurance policies |
| PWAOREG | 0 | Contribution disability insurance policies |
| PBRAND | 5 | Contribution fire policies |
| PFIETS | 0 | Contribution bicycle policies |
| ATRACTOR | 0 | Number of tractor policies |
| AZEILPL | 0 | Number of surfboard policies |
| APLEZIER | 2 | Number of boat policies |
| AFIETS | 0 | Number of bicycle policies |
| ABYSTAND | 0 | Number of social security insurance policies |
| prediction | 1 | Number of mobile home policies |
| target | 1 | Number of mobile home policies |

Table 15: Customer who purchases a caravan policy

The table 15 indicate customer data based on selected features from the lasso regression model. In some sense, this may be said that people with these features are more likely to buy a caravan insurance policy. For example, the customer with 6 contributions car policies were most likely to purchase a caravan insurance policy, but it cannot be concluded that people with 6 contributions car policies would buy a caravan insurance policy since there are other 30 features as mentioned, which include both product usage and socio-demographic data, we need to consider to analyze whether a customer will purchase this insurance policy.

Therefore, we can predict who would be interested in buying a caravan insurance policy using predictive modeling techniques since the prediction accuracy of 94.02% indicates that we have a highly accurate model. Of course, using this model to predict is better than a random prediction. Those predictive modeling techniques also enable us to choose best performance model like the lasso regression and found most relevant features. With the lasso regression model, we can not only predict people who would buy a caravan insurance policy but also explain why those people buy a caravan insurance policy based on their given information. As a result, our model would benefit insurance companies because insurers can use this model to discover customer characteristics and predict which customers are potentially interested in an insurance policy.

## References

[1] Insurance Journal.(2012). How Predictive Modeling Has Revolutionized Insurance. Insurance Journal. Retrieved April 5, 2022, from https://www.insurancejournal.com/news/national/2012/06/18/251957.htm.

[2] James, G., Witten, D., Hastie, T., amp; Tibshirani, R. (2013). An introduction to statistical learning. Springer Texts in Statistics. https://doi.org/10.1007/978-1-4614-7138-7

[3] Mordor Intelligence. (2022, January 21). Caravan and motorhome market growth, trends, Industry Analysis (2022 - 27). Retrieved April 5, 2022, from https://www.mordorintelligence.com/industry-reports/caravan-and-motorhome-market

[4] van der Putten, P. (2000). Coil Challenge 2000 . Retrieved April 26, 2022, from https://liacs.leidenuniv.nl/ puttenpwhvander/library/cc2000/

[5] van der Putten, P., amp; van Someren, M. (2004). A bias-variance analysis of a real world learning problem: The Coil Challenge 2000. Machine Learning, 57, 177–195. https://doi.org/10.1023/b:mach.0000035476.95130.99