STAT 52900 Final Report

Solution for whether the car insurance claim should be passed or not using Bayesian Statistics?

Ayush Manojkumar Lodha
(amlodha@iu.edu)

Table of Contents

Abstract

The premium of the claim and whether the claim should be passed (accepted) or not(dismissed) for the car insurance case determines the revenue, profit, and overall bread butter of the company. If these decisions are not appropriately taken company might have to incur losses. Also, as a car owner and customer of the insurance company, it is an exciting thing to know what the chances of my claim are being accepted by the company. Nowadays, there are many state-of-the-art machine learning algorithms and statistical methods to solve this decision-making problem.

In this paper, we discuss whether this car insurance claim should be passed or not using a dataset that has various independent variables determining this decision. We aim to compare the analysis of the Bayesian way of modeling this problem with the baseline algorithm logistic regression, which is commonly used to classify the "Yes"/ "No" problem. Variable Selection and deciding whether the variable should be included in our analysis, choosing priors and experimenting with them, understanding the settings of diagnostics, and deciding the metrics to rate each model on are some of the essential steps we will understand further in this paper.

Problem Statement and Dataset Description

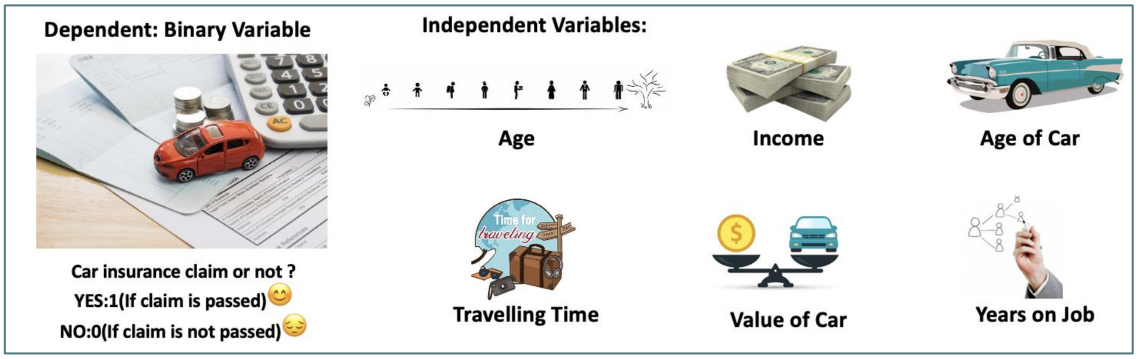
To illustrate our analysis on this topic of interest, we sourced a dataset from Kaggle, which can be accessed through this link. Initially, this dataset had 25 variables for determining the two target variables as follows:

a) Binary Variable: Whether the claim is passed (Yes or No).
b) Claim Amount: Given for the claim if it has been given.

There are 25 independent variables (assumed in the dataset description), for determining these two target variables. For our analysis and scope of this project, we have chosen to determine only the binary variable of whether the claim is passed (Yes or No). We have taken six interesting independent variables to help us assess our decision based on the following criteria.

a) The correlation among the variables should be as low as possible
b) Interesting variables for the analysis.

Following is the image illustrating the dependent variable and independent variables in our analysis.


Dependent: Binary Variable
Car insurance claim or not ?
YES:1(If claim is passed)😊
NO:0(If claim is not passed)😔
Independent Variables:
Age
Income
Age of Car
Time for traveling
Travelling Time
Value of Car
Years on Job

Before moving toward our analysis, we need to understand the kind of information each variable bring in, which is described in the following table.

Independent Variables	**Description**	**Hypothesis (Hypothetical Effects/Presumptions)**
AGE	Age of the Drive	Young Drivers tend to be riskier
INCOME	Income	High-income people tend to have advanced cars. (General trend may not be true)
CAR_AGE	Age of Car	Newer cars have more safety features
TRAVTIME	Travelling Time (Distance to work)	Longer journey means chance of accident increases
BLUEBOOK	Value of Vehicle	This may affect the claim amount and hence related to claim must be passed or not
YOJ	Years on Job	More the time spend on job, more they drive safely usually

Preprocessing of Dataset

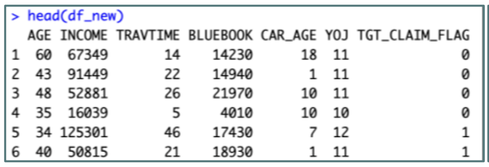
We do the following things to clean the dataset.

a) Drop the duplicate rows

b) Replace the null values with interpolation and mean values of the data.

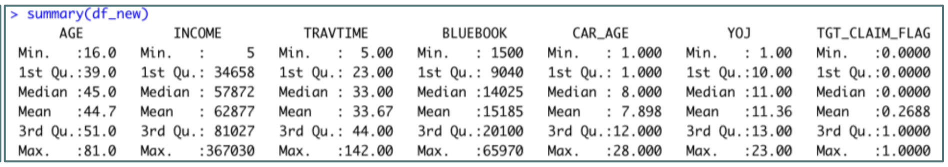
We obtain the dataset of the 7 columns and 7656 rows after preprocessing.

Below are the images of the first few rows of the dataset and a summary of the dataset.



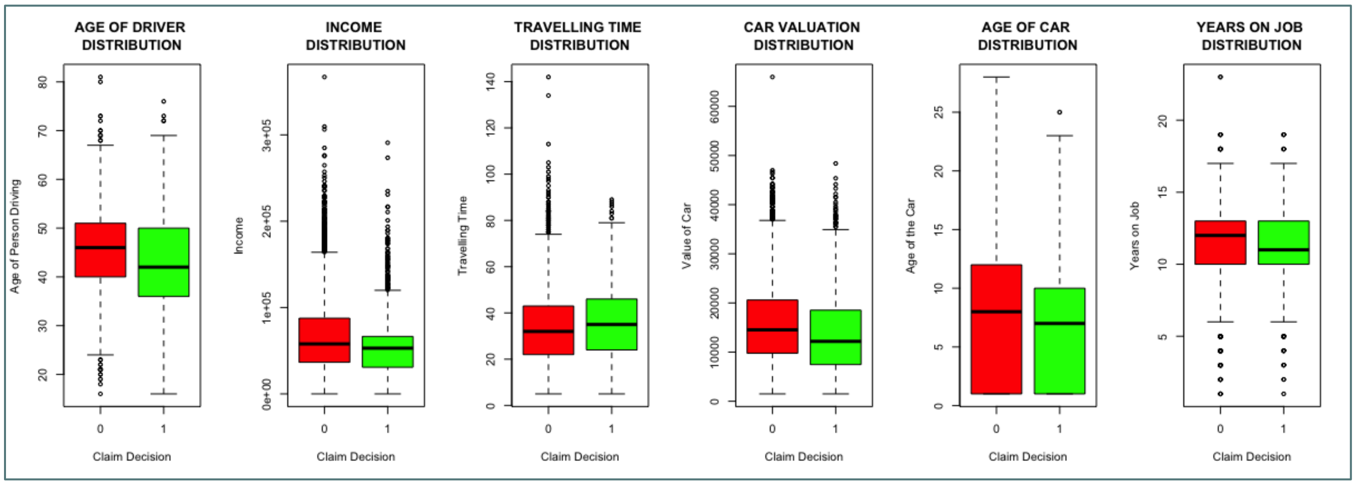
```
> head(df_new)
  AGE INCOME TRAVTIME BLUEBOOK CAR_AGE YOJ TGT_CLAIM_FLAG
1  60  67349       14    14230      18  11              0
2  43  91449       22    14940       1  11              0
3  48  52881       26    21970      10  11              0
4  35  16039        5     4010      10  10              0
5  34 125301       46    17430       7  12              1
6  40  50815       21    18930       1  11              1
```

Rows of the Dataframe



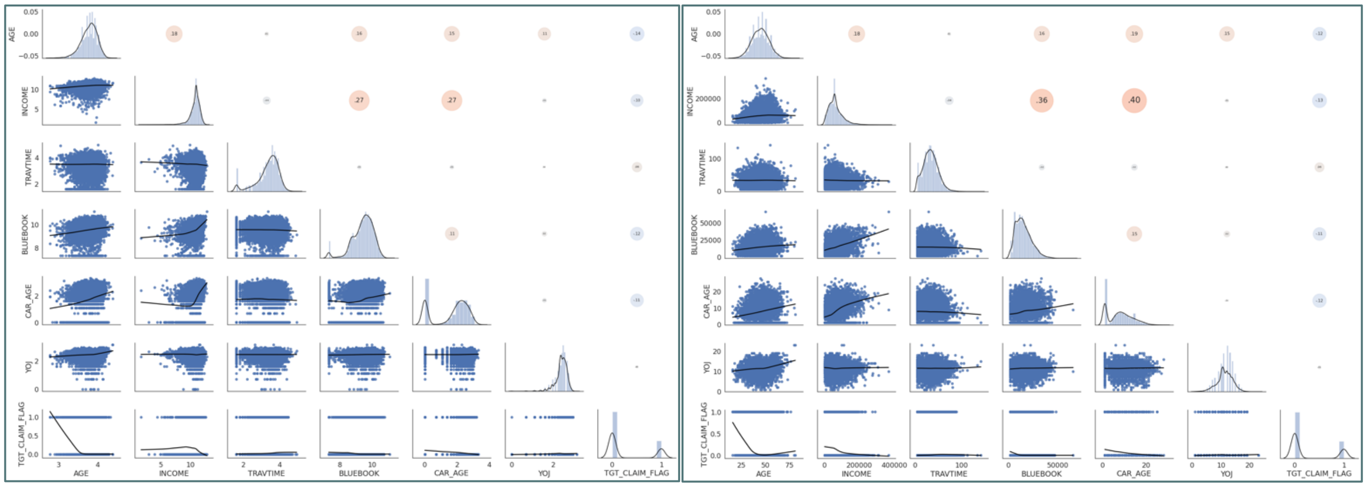
```
> summary(df_new)
      AGE           INCOME          TRAVTIME         BLUEBOOK        CAR_AGE            YOJ         TGT_CLAIM_FLAG
 Min.   :16.0   Min.   :     5   Min.   :  5.00   Min.   : 1500   Min.   : 1.000   Min.   : 1.00   Min.   :0.0000
 1st Qu.:39.0   1st Qu.: 34658   1st Qu.: 23.00   1st Qu.: 9040   1st Qu.: 1.000   1st Qu.:10.00   1st Qu.:0.0000
 Median :45.0   Median : 57872   Median : 33.00   Median :14025   Median : 8.000   Median :11.00   Median :0.0000
 Mean   :44.7   Mean   : 62877   Mean   : 33.67   Mean   :15185   Mean   : 7.898   Mean   :11.36   Mean   :0.2688
 3rd Qu.:51.0   3rd Qu.: 81027   3rd Qu.: 44.00   3rd Qu.:20100   3rd Qu.:12.000   3rd Qu.:13.00   3rd Qu.:1.0000
 Max.   :81.0   Max.   :367030   Max.   :142.00   Max.   :65970   Max.   :28.000   Max.   :23.00   Max.   :1.0000
```

Summary Statistics of Dataframe



Boxplots of the Distribution

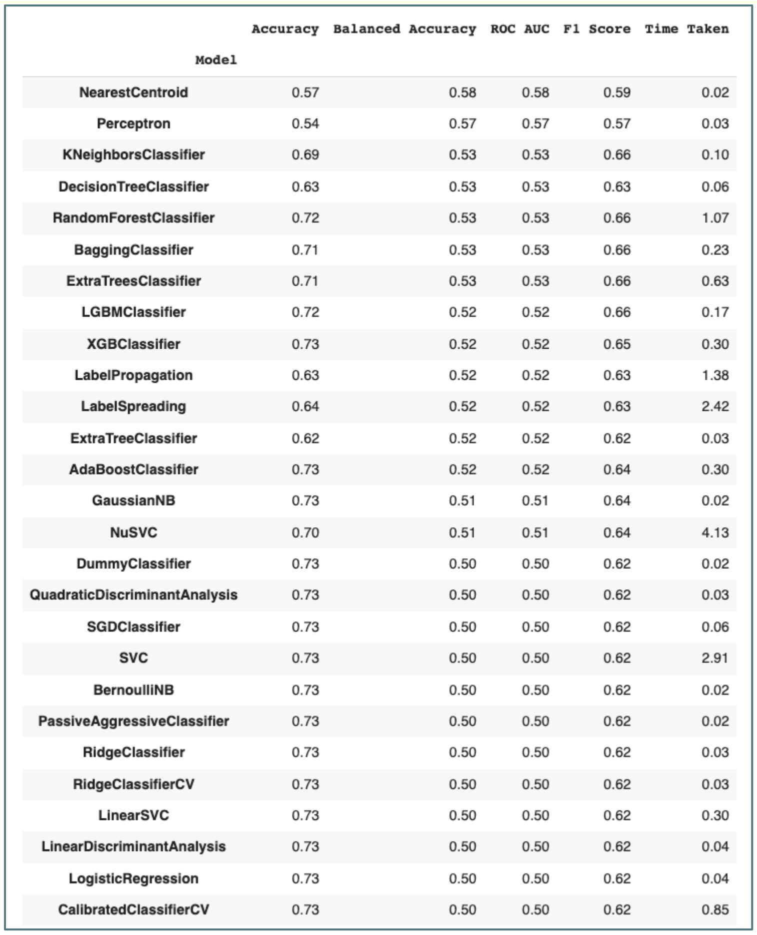
From the above boxplot of the distribution, we can see many outliers in the dataset. There was a challenge here: we removed the outliers from our data without losing the crucial information from our dataset. After eliminating these outliers from our datasets, there are around 300 data points that we lose.



(a)On the Left Plot, Pairplots, Histogram, and Correlation Matrix.(b) On the Right, Log Transform of the Plot Pairplots, Histogram and Correlation Matrix.

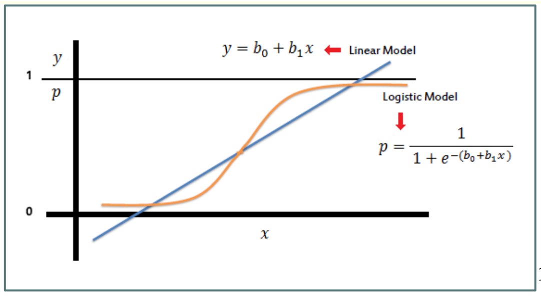
From the above figure (a), we can see that some variables are skewed. To fix the skewness, we do the Log Transform of all the independent variables and see the effect of this fix in the above figure (b).

Baseline Model



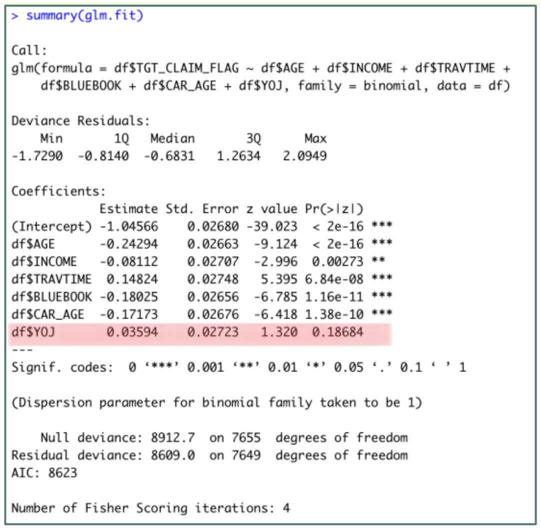
Model	Accuracy	Balanced Accuracy	ROC AUC	F1 Score	Time Taken
NearestCentroid	0.57	0.58	0.58	0.59	0.02
Perceptron	0.54	0.57	0.57	0.57	0.03
KNeighborsClassifier	0.69	0.53	0.53	0.66	0.10
DecisionTreeClassifier	0.63	0.53	0.53	0.63	0.06
RandomForestClassifier	0.72	0.53	0.53	0.66	1.07
BaggingClassifier	0.71	0.53	0.53	0.66	0.23
ExtraTreesClassifier	0.71	0.53	0.53	0.66	0.63
LGBMClassifier	0.72	0.52	0.52	0.66	0.17
XGBClassifier	0.73	0.52	0.52	0.65	0.30
LabelPropagation	0.63	0.52	0.52	0.63	1.38
LabelSpreading	0.64	0.52	0.52	0.63	2.42
ExtraTreeClassifier	0.62	0.52	0.52	0.62	0.03
AdaBoostClassifier	0.73	0.52	0.52	0.64	0.30
GaussianNB	0.73	0.51	0.51	0.64	0.02
NuSVC	0.70	0.51	0.51	0.64	4.13
DummyClassifier	0.73	0.50	0.50	0.62	0.02
QuadraticDiscriminantAnalysis	0.73	0.50	0.50	0.62	0.03
SGDClassifier	0.73	0.50	0.50	0.62	0.06
SVC	0.73	0.50	0.50	0.62	2.91
BernoulliNB	0.73	0.50	0.50	0.62	0.02
PassiveAggressiveClassifier	0.73	0.50	0.50	0.62	0.02
RidgeClassifier	0.73	0.50	0.50	0.62	0.03
RidgeClassifierCV	0.73	0.50	0.50	0.62	0.03
LinearSVC	0.73	0.50	0.50	0.62	0.30
LinearDiscriminantAnalysis	0.73	0.50	0.50	0.62	0.04
LogisticRegression	0.73	0.50	0.50	0.62	0.04
CalibratedClassifierCV	0.73	0.50	0.50	0.62	0.85

(p)State-of-Art Model Results



[1]

(q)Logistic Regression Overview



```
> summary(glm.fit)

Call:
glm(formula = df$TGT_CLAIM_FLAG ~ df$AGE + df$INCOME + df$TRAVTIME +
    df$BLUEBOOK + df$CAR_AGE + df$YOJ, family = binomial, data = df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7290  -0.8140  -0.6831   1.2634   2.0949

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.04566    0.02680 -39.023  < 2e-16 ***
df$AGE      -0.24294    0.02663  -9.124  < 2e-16 ***
df$INCOME   -0.08112    0.02707  -2.996  0.00273 **
df$TRAVTIME  0.14824    0.02748   5.395 6.84e-08 ***
df$BLUEBOOK -0.18025    0.02656  -6.785 1.16e-11 ***
df$CAR_AGE  -0.17173    0.02676  -6.418 1.38e-10 ***
df$YOJ       0.03594    0.02723   1.320  0.18684
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 8912.7  on 7655  degrees of freedom
Residual deviance: 8609.0  on 7649  degrees of freedom
AIC: 8623

Number of Fisher Scoring iterations: 4
```

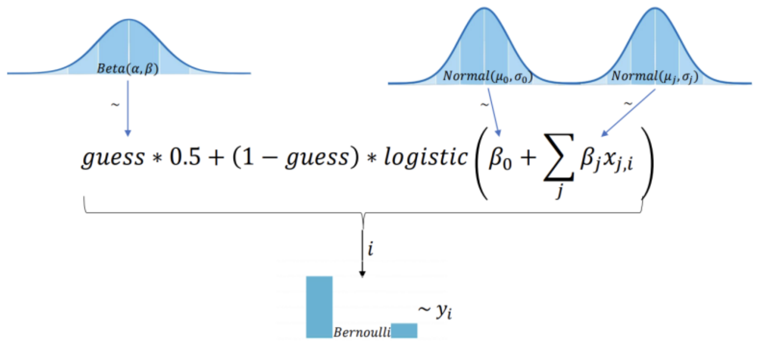
(r)Results of Logistic Regression

From figure (p) State-of-Art Model Results, we can see the test results from all the major classification machine learning models on our dataset. From the above illustration, we understand that the maximum accuracy any model can achieve in this dataset is 73%. Still, since this is the classification problem, accuracy cannot be the only metric for judging the best model because accuracy is hugely dependent on the threshold value we set (generally, we put it as 0.5). Still, changing the point can affect accuracy significantly above or below based on the skewness of the target-dependent variable. We can learn this fact from figure (q).

We choose logistic regression as our baseline model, and figure(r) depicts the summary of the fit of the logistic regression. We can see that the independent variable in logistic regression, “YOJ,” is insignificant in our model because it fails the p-value test and its coefficient distribution would have zero in it. We deliberately keep this variable in our dataset while modeling our first bayesian model to examine whether the bayesian belief can convey the same fact.

[1] https://www.saedsayad.com/logistic_regression.htm

Proposed Model during Midterm Presentation.



Proposed Model during the Midterm Presentation

Mathematical Model

Since our target variable Y (TGT_CLAIM_FLAG) is 0,1 variable, we take the Bernoulli model as follows,

$$Y \sim Bernoulli(\theta)$$

Where, $\theta = guess * 0.5 + (1 - guess) * logit(\beta_0 + \beta_1 AGE + \beta_2 INCOME + \beta_3 CARAGE + \beta_4 TRAVTIME + \beta_5 BLUEBOOK + \beta_6 YOJ)$

Where $logit(p_i) = \log\left(\frac{p_i}{1-p_i}\right)$

For the bayesian belief of priors, we choose the “normal distribution” as the priors for the coefficients of the variables and the “beta distribution” for the guessing parameter. We may have several different priors for these coefficients. We will discuss them in our further analysis.

Since we had many outliers in our data, I assumed this guess parameter to be a non-zero value. But now, after figuring out how to remove these outliers without losing much information. We can assume the “guess” parameter to be zero. Hence we can update our model.

$$Y \sim Bernoulli(\theta)$$

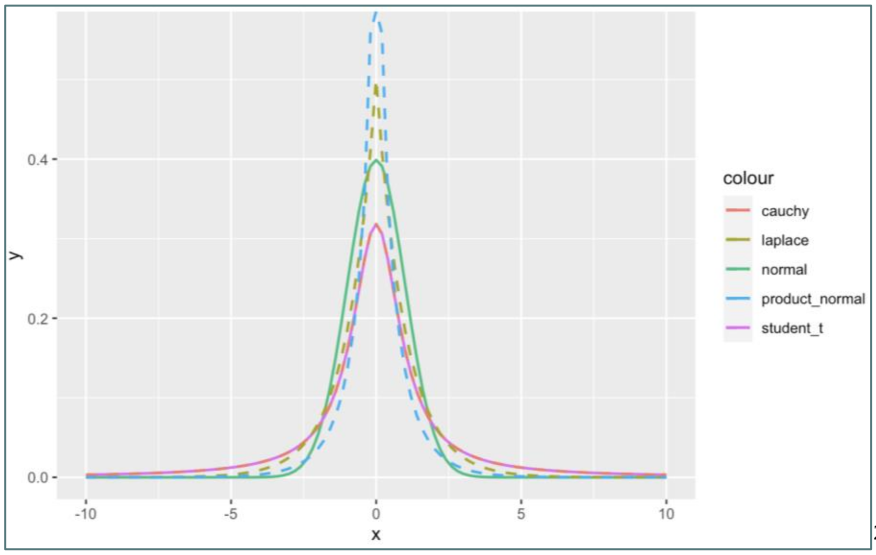
Where, $\theta = guess * 0.5 + (1 - guess) * logit(\beta_0 + \beta_1 AGE + \beta_2 INCOME + \beta_3 CARAGE + \beta_4 TRAVTIME + \beta_5 BLUEBOOK + \beta_6 YOJ)$

$\theta = 0 * 0.5 + (1 - 0) * logit(\beta_0 + \beta_1 AGE + \beta_2 INCOME + \beta_3 CARAGE + \beta_4 TRAVTIME + \beta_5 BLUEBOOK + \beta_6 YOJ)$

$\theta = logistic(\beta_0 + \beta_1 AGE + \beta_2 INCOME + \beta_3 CARAGE + \beta_4 TRAVTIME + \beta_5 BLUEBOOK + \beta_6 YOJ)$ Where, $logit(p_i) = \log\left(\frac{p_i}{1-p_i}\right)$

Before moving to the different models, we would highlight the choice of priors and metrics of the confusion matrix used to evaluate upcoming models.

Choice of Priors



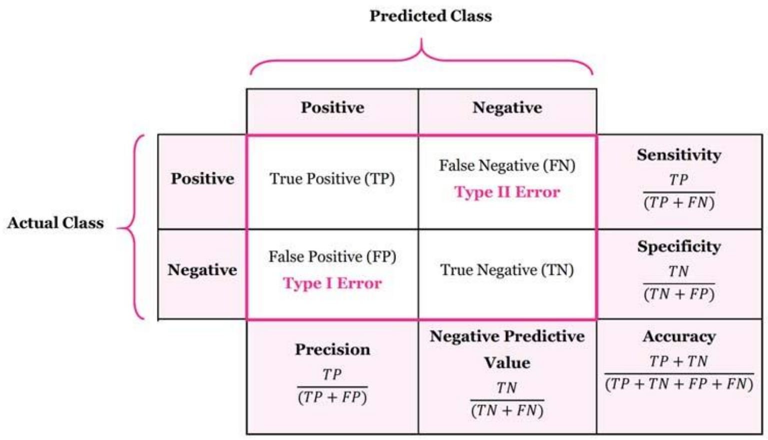
[2]

Comparison of Prior Distribution

From the above illustration, we see the different priors generally used when we use a binomial family of priors.

Based on the spread and considering the nature of being the less pin-pointed type of distribution, we considered “normal” and “student_t” as the prior for the Bayesian modeling technique.

Confusion Matrix



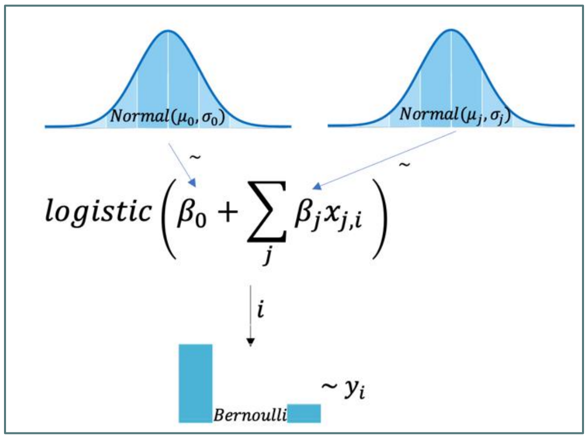
		Predicted Class		
		Positive	**Negative**	
Actual Class	**Positive**	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

Confusion Matrix

Above figure of confusion matrix is the metric we would use to assess our models. For our models, actual class would be “TGT_CLAIM_FLAG” column of the test set for validation and predicted class is value we would predict using our Bayesian models.

[2] *https://mc-stan.org/rstanarm/reference/priors.html*

Model 1. (Normal and Uninformative Priors)



$Normal(\mu_0, \sigma_0)$
$Normal(\mu_j, \sigma_j)$
$logistic\left(\beta_0 + \sum_j \beta_j x_{j,i}\right)$
i
$\sim y_i$
Bernoulli

Binomial Logistic Regression model with normal priors.

We use normal uninformative priors for running the above Bayesian model; below is the table of uninformative priors used.

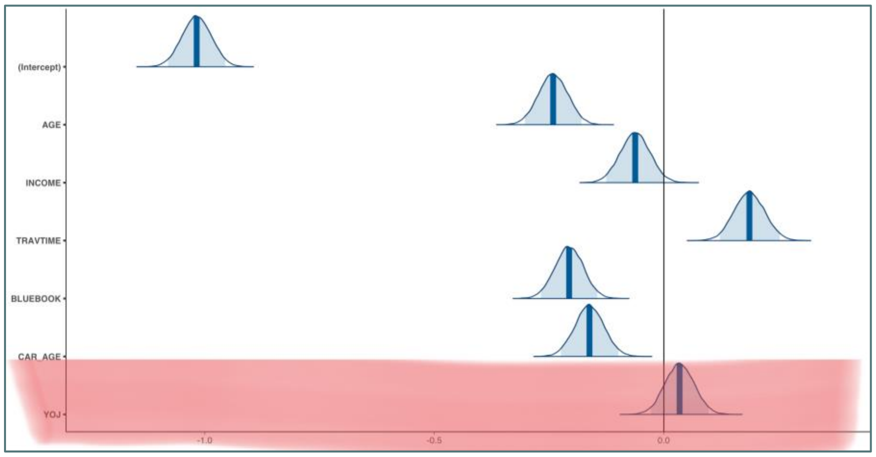
β_1	β_2	β_3	β_4	β_5	β_6
$N(0,10)$	$N(0,10)$	$N(0,10)$	$N(0,10)$	$N(0,10)$	$N(0,10)$

We use the "rstanarm" library in R for running our models. We use the MCMC techniques to estimate the coefficients, and the Markov chains need to converge to estimate the coefficients successfully. Following are the settings which we should decide for the MCMC to run.

Settings of the Model 1:

Train Test Split	70:30 (randomly)
No of iterations	50,000
Warmup Iterations	10,000
Thinning Steps	5
No of Chains	3

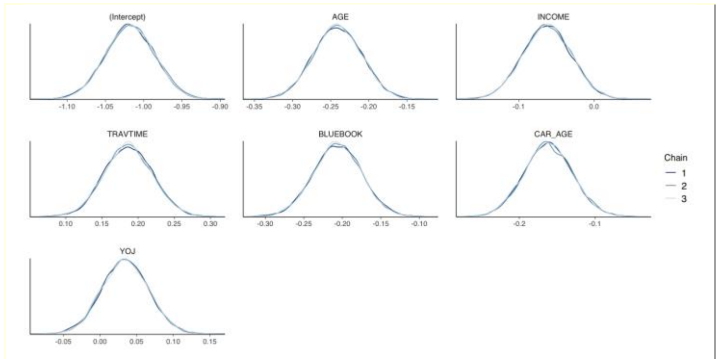
From the settings, we can understand that 3 MCMC chains would be randomly initialized and run for 50,000 iterations (including the 10,000 warmup iterations to be discarded, these initial values are discarded, which Markov chains don't converge because they have just initialized. They will start to converge after these warmup iterations are completed). 5 Thinning steps mean every 5th value from these iterations would be kept and rest all would be discarded; this is used to ensure that all the values we randomly sample/draw from the distribution should not be dependent on each other.



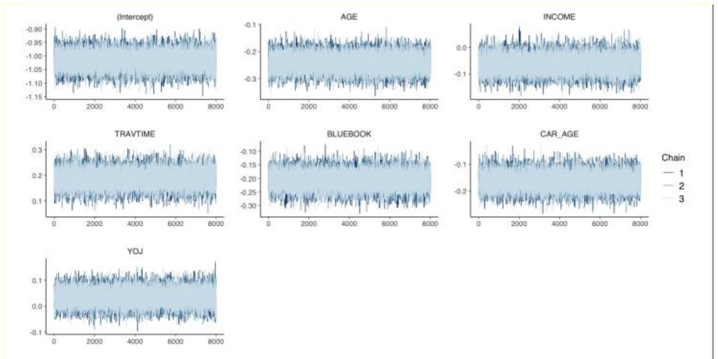
High Probability Density of the Coefficients of Variables from the results of Model 1

From the above figure, we can see that the HPD (95% confidence interval) of β_6 which is the coefficient of "YOJ," has zero in its interval, which says that "YOJ" is significant for determining the target variable. Our conclusion from the figure is similar to the discussion in logistic regression analysis; the "YOJ" should not be included in the model. Hence we can see that the Bayesian way of modeling can show the significance of each variable in the model same as in the logistic regression.

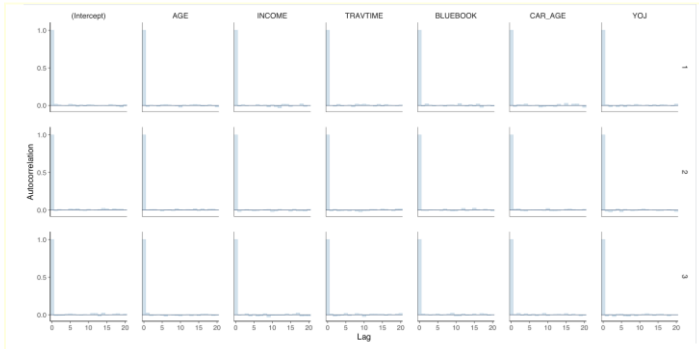
MCMC Diagnostics



Density Plot of the Posterior Distribution of Coefficients from MCMC



Trace Plot of the Posterior Distribution of Coefficients from MCMC



Autocorrelation Plot

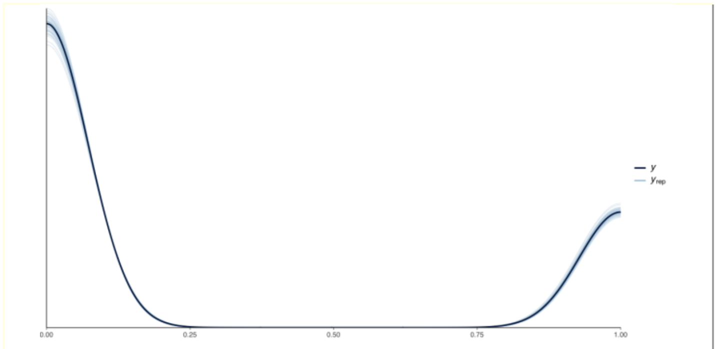


Illustration of the posterior predictive checks

From the above figure, we can understand that the MCMC chains have converged successfully. ACF plot shows that after certain lags the autocorrelation diminishes for each of the variables. We obtain the following confusion matrix and accuracy for the testing data,

$$Confusion\ Matrix = \begin{bmatrix} 1689 & 26 \\ 559 & 23 \end{bmatrix}; Accuracy = 0.7453$$

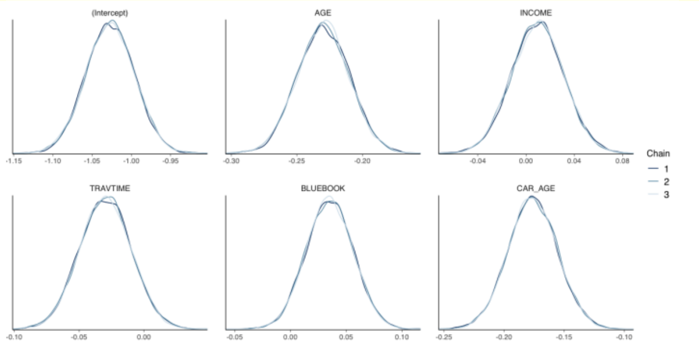
Model 2. (Normal and Informative Priors)

We keep the same “Normal” Distribution for the Priors as for Model 1, for this Model 2. There are 5 variables in this model since we removed “YOJ” from our data for the further analysis. We use the informative priors as the mean and standard deviations of the coefficients obtained from the baseline Logistic regression model as shown in the following table,

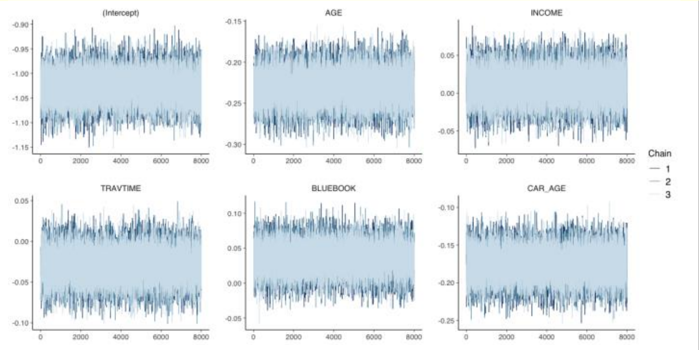
β_1	β_2	β_3	β_4	β_5
$N(-0.239, 0.026)$	$N(-0.082, 0.027)$	$N(0.148, 0.027)$	$N(-0.180, 0.027)$	$N(-0.172, 0.027)$

Settings of the Model 2: Same as Model 1

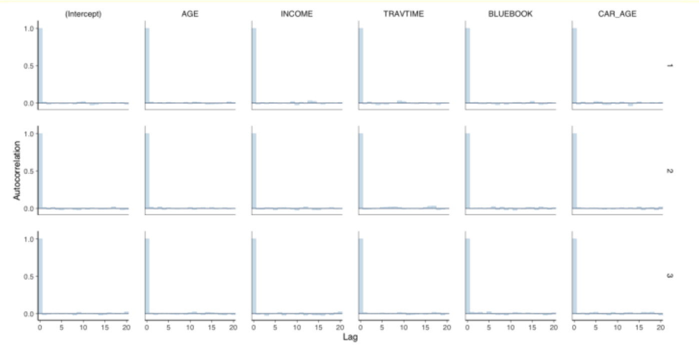
MCMC Diagnostics



Density Plot of the Posterior Distribution of Coefficients from MCMC



Trace Plot of the Posterior Distribution of Coefficients from MCMC



Autocorrelation Plot

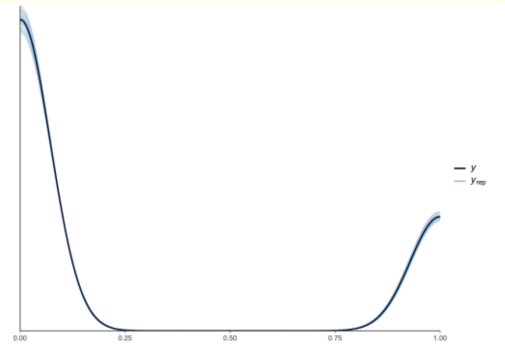
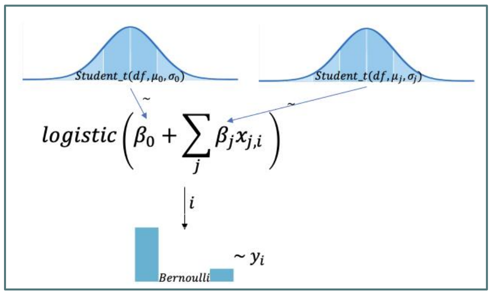


Illustration of the posterior predictive checks

From the above figure, we can understand that the MCMC chains have converged successfully. ACF plot shows that after certain lags the autocorrelation diminishes for each of the variables. We obtain the following confusion matrix and accuracy for the testing data,

$$Confusion\ Matrix = \begin{bmatrix} 1676 & 1 \\ 618 & 2 \end{bmatrix}; Accuracy = 0.7305$$

Model 3. (Student_t and Uninformative Priors)


$Student\_t(df, \mu_0, \sigma_0)$
$Student\_t(df, \mu_j, \sigma_j)$
~
~
$logistic\left(\beta_0 + \sum_j \beta_j x_{j,i}\right)$
i
Bernoulli
$\sim y_i$

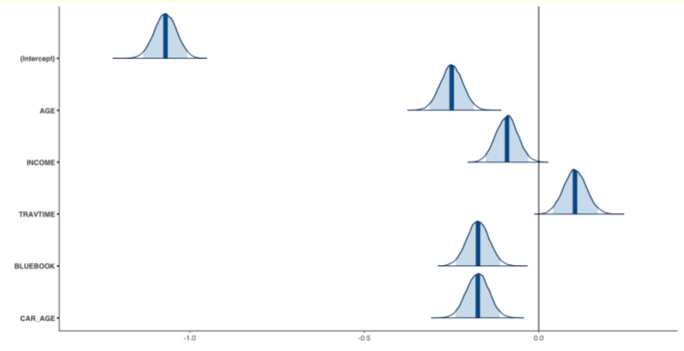
Binomial Logistic Regression model with student_t priors.

We use student_t uninformative priors for running the above Bayesian model; below is the table of uninformative priors used.

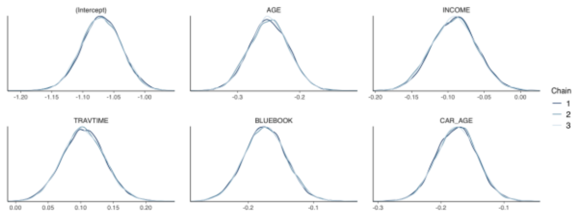
β_1	β_2	β_3	β_4	β_5	β_6
$student\_t(0,10)$	$student\_t(0,10)$	$student\_t(0,10)$	$student\_t(0,10)$	$student\_t(0,10)$	$student\_t(0,10)$

Settings of the Model 3: Same as Model 1

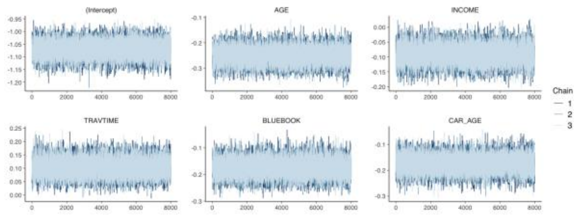
MCMC Diagnostics:



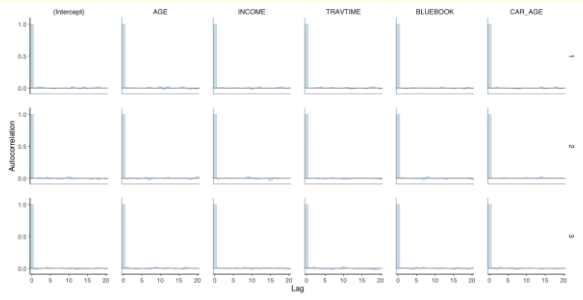
High Probability High Probability Density of the Coefficients of Variables from the results of Model 3



Density Plot of the Posterior Distribution of Coefficients from MCMC



Trace Plot of the Posterior Distribution of Coefficients from MCMC



Autocorrelation Plot

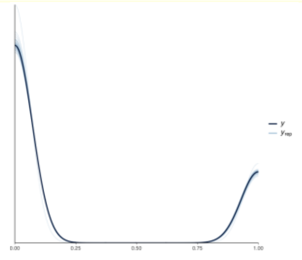


Illustration of the posterior predictive checks

From the above figure, we can understand that the MCMC chains have converged successfully. ACF plot shows that after certain lags the autocorrelation diminishes for each of the variables. We obtain the following confusion matrix and accuracy for the testing data,

$$Confusion\ Matrix = \begin{bmatrix} 1642 & 15 \\ 622 & 18 \end{bmatrix}; Accuracy = 0.7226$$

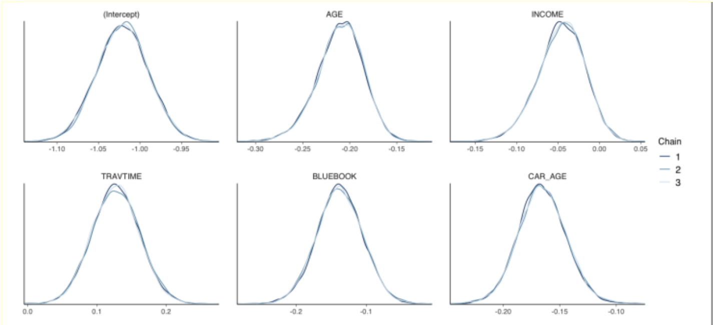
Model 4. (Student_t and Informative Priors)

Keeping the same distribution of "student_t" from Model 3, we take informative priors from baseline logistic regression model

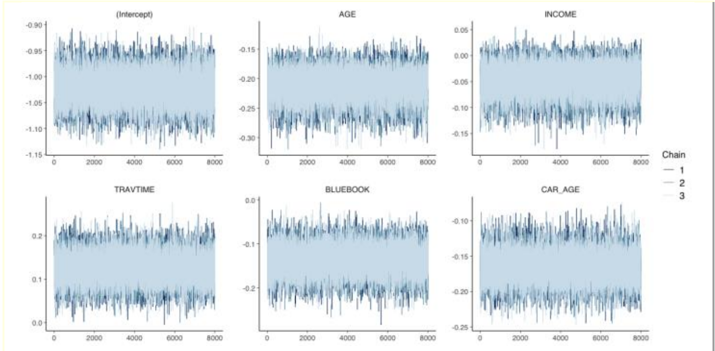
β_1	β_2	β_3	β_4	β_5
$student\_t(-0.239,0.02$	$student\_t(-0.082,0.02$	$student\_t(0.148,0.02$	$student\_t(-0.180,0.02$	$student\_t(-0.172,0.02$

Settings of the Model 4: Same as Model 1

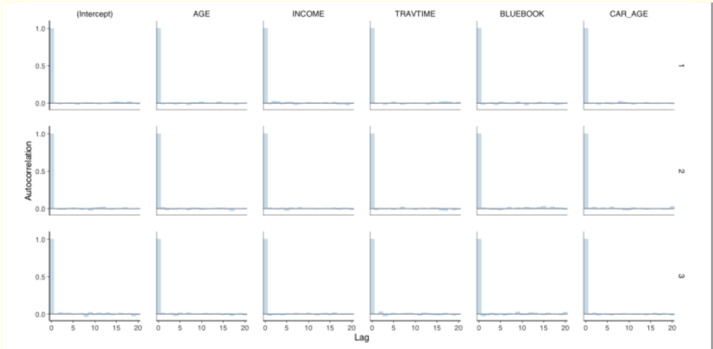
MCMC Diagnostics



Density Plot of the Posterior Distribution of Coefficients from MCMC



Trace Plot of the Posterior Distribution of Coefficients from MCMC



Autocorrelation Plot

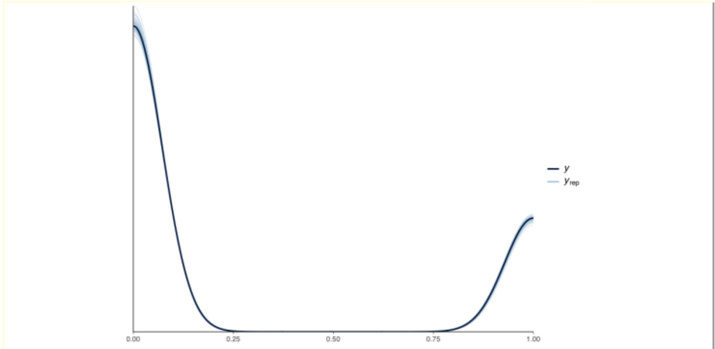


Illustration of the posterior predictive checks

From the above figure, we can understand that the MCMC chains have converged successfully. ACF plot shows that after certain lags the autocorrelation diminishes for each of the variables. We obtain the following confusion matrix and accuracy for the testing data,

$$Confusion\ Matrix = \begin{bmatrix} 1684 & 7 \\ 597 & 9 \end{bmatrix}; Accuracy = 0.7370$$

Conclusion and Discussion

Model	Prior	Prior Type	Accuracy	Confusion Matrix
Baseline Logistic Regression	-	-	0.73	$\begin{bmatrix}1641 & 16\\ 618 & 22\end{bmatrix}$
Model 1	Uninformative	Normal	0.7453	$\begin{bmatrix}1689 & 26\\ 559 & 23\end{bmatrix}$
Model 2	Informative	Normal	0.7305	$\begin{bmatrix}1676 & 1\\ 618 & 2\end{bmatrix}$
Model 3	Uninformative	Student_t	0.7226	$\begin{bmatrix}1642 & 15\\ 622 & 18\end{bmatrix}$
Model 4	Informative	Student_t	0.7370	$\begin{bmatrix}1684 & 7\\ 597 & 9\end{bmatrix}$

From the above table, we can see the comparative analysis of the various models we build throughout the project.
Some of the major learning which I have learned from this analysis are as follows:

- Bayesian Models are capable of finding out the coefficients which are not useful for the determining the dependent variables (illustrated while eliminating "YOJ" variable from the data)
- Bayesian Models are sensitive to the choice of priors.
- Sometimes these uninformative priors are better than using informative once, because we bias ourselves by using informative, they should be taken from some previous studies and information available from domain experts.
- Uninformative works very well but with slight restriction over type of distribution and spread of the variance.
- Thinning Steps and Burn In steps are the important parameters in MCMC diagnosis because it would eventually decide the no of iterations which will directly affect the posterior distribution, which is our main focus during Bayesian analysis.
- Whenever we deal with real world dataset, it comes lots of bias, which require extensive work over removing outliers without actual losing any critical information from data. Important steps of data cleaning and data preprocessing should not be missed and should be done thoroughly otherwise there can be mistakes in assumptions of the model. (We experienced this while concluding "guess" parameter to be zero)

References

1. *Class Notes and handout by Prof. Ben Boukai in Class: 52900 and 51200*
2. *Bayesian Logistic Regression Modelfor Credit Score Data-Presented by Hongfei Li, Lai Wei*
3. *https://avehtari.github.io/modelselection/diabetes.html*
4. *https://www.bayesrulesbook.com/chapter-13.html*
5. *https://www.rdocumentation.org/packages/rstanarm/versions/2.21.3/topics/stan_glm*
6. *https://www.kaggle.com/code/kerneler/starter-car-insurance-claim-data-62f4f91c-d/comments*
7. *https://www.kaggle.com/code/kerneler/starter-car-insurance-claim-data-62f4f91c-d/comments*
8. *https://www.kaggle.com/code/dllim1/end-to-end-ml-on-motor-insurance-with-xgboost*
9. *https://hbiostat.org/doc/rms/lrm-brms.pdf*
10. *https://www.statology.org/confusion-matrix-in-r/*

End of Report (R Code for reference)[Main Files]:

EDA.R	**normal_prior_informative.R**	**normal_prior_noninformative.R**	**student_t_informative.R**	**student_t_noninformative.R**

library(tidyverse)
library(caret)
library(GGally)
library(ggplot2)
library(corrplot)
library(bayesplot)
theme_set(bayesplot::theme_default(base_family = "sans"))
library(rstanarm)
options(mc.cores = 1)
library(loo)
library(projpred)
SEED=14124869
 	
df_new <- read.csv("df_new.csv", header = TRUE)
df_new <- df_new[,c(2:8)]
head(df_new)
summary(df_new)
str(df_new)
 	
par(mfrow=c(2,3))
boxplot(df_new$AGE~df_new$TGT_CLAIM_FLAG,
col=rainbow(ncol(df)),
xlab="CLAIM PASSED OR NOT",
ylab="AGE",
main="AGE DISTRIBUTION")
boxplot(df_new$INCOME~df_new$TGT_CLAIM_FLAG,
col=rainbow(ncol(df_new)),
xlab="CLAIM PASSED OR NOT",
ylab="INCOME",
main="INCOME DISTRIBUTION")
boxplot(df_new$TRAVTIME~df_new$TGT_CLAIM_FLAG,
col=rainbow(ncol(df_new)),
xlab="CLAIM PASSED OR NOT",
ylab="TRAVELLING TIME",
main="TRAVELLING TIME DISTRIBUTION")
boxplot(df_new$BLUEBOOK~df_new$TGT_CLAIM_FLAG,
col=rainbow(ncol(df_new)),
xlab="CLAIM PASSED OR NOT",
ylab="VALUE OF CAR",
main="CAR VALUATION")
boxplot(df_new$CAR_AGE~df_new$TGT_CLAIM_FLAG,
col=rainbow(ncol(df_new)),
xlab="CLAIM PASSED OR NOT",
ylab="AGE OF CAR",
main="AGE OF CAR DISTRIBUTION")
boxplot(df_new$YOJ~df_new$TGT_CLAIM_FLAG,
col=rainbow(ncol(df_new)),
xlab="CLAIM PASSED OR NOT",
ylab="YEARS ON JOB",
main="YEARS ON JOB DISTRIBUTION")
 	
par(mar=c(2,3))
hist(df_new$AGE,xlab="Age",main="Histogram Distribution of Age")
hist(df_new$INCOME,xlab="Income",main="Histogram Distribution of Income")
hist(df_new$TRAVTIME,xlab="Travelling Time",main="Histogram Distribution of Travelling Time")
hist(df_new$BLUEBOOK,xlab="Car Valuation",main="Histogram Distribution of Car Valuation")
hist(df_new$CAR_AGE,xlab="Age of Car",main="Histogram Distribution of Age of Car")
hist(df_new$YOJ,xlab="Years on Job",main="Histogram Distribution of Years on Job")
hist(df_new$TGT_CLAIM_FLAG)
	library(tidyverse)
	library(caret)
	library(GGally)
	library(ggplot2)
	library(corrplot)
	library(bayesplot)
	library(bayesrules)
	theme_set(bayesplot::theme_default(base_family = "sans"))
	library(rstanarm)
	options(mc.cores = 2)
	library(loo)
	library(projpred)
	
	df <- read.csv("df_log_preprocess.csv", header = TRUE)
	df <- df[,c("AGE","INCOME","TRAVTIME","BLUEBOOK","CAR_AGE","TGT_CLAIM_FLAG")]
	summary(df)
	
	for (i in 1:5) {
	df[i] <- scale(df[i])
	}
	
	df$TGT_CLAIM_FLAG<- factor(df$TGT_CLAIM_FLAG)
	
	sample_size = floor(0.70*nrow(df))
	train_sample = sample(seq_len(nrow(df)),size = sample_size)
	train_Data = df[train_sample,]
	test_Data = df[-train_sample,]
	
	summary(train_Data)
	summary(test_Data)
	
	y_test = test_Data$TGT_CLAIM_FLAG
	x_test = model.matrix(TGT_CLAIM_FLAG ~ . - 1, data = test_Data)
	
	# preparing the inputs
	x_train <- model.matrix(TGT_CLAIM_FLAG ~ . - 1, data = train_Data)
	y_train <- train_Data$TGT_CLAIM_FLAG
	(reg_formula <- formula(paste("TGT_CLAIM_FLAG ~", paste(names(train_Data)[1:5], collapse = " + "))))
	
	normal_prior <- normal(location =c(-0.239,-0.082,0.148,-0.180,-0.172),
	scale = c(0.026,0.027,0.027,0.027,0.027))
	post_norm_in <- stan_glm(reg_formula, data = train_Data,
	family = binomial(link = "logit"),
	prior = normal_prior,
	QR=TRUE,
	chain=3,
	iter=50000,
	warmup=10000,
	thin = 5,
	cores = 8)
	
	summary(post_norm_in)
	prior_summary(post_norm_in)
	
	# pplot<-plot(post_norm_in, "areas", prob = 0.95, prob_outer = 1)
	# pplot+ geom_vline(xintercept = 0)
	
	mcmc_dens_overlay(post_norm_in)
	mcmc_trace(post_norm_in)
	plot(post_norm_in,"acf_bar")
	
	classification_summary(model = post_norm_in_norm_in,data = train_Data,cutoff = 0.5)
	classification_summary(model = post_norm_in,data = test_Data,cutoff = 0.5)
	
	pp_check(post_norm_in)
	pp_check(post_norm_in,plotfun = "hist",nreps=5)
	pp_check(post_norm_in, plotfun = "error_binned")
		library(tidyverse)
		library(caret)
		library(GGally)
		library(ggplot2)
		library(corrplot)
		library(bayesplot)
		library(bayesrules)
		theme_set(bayesplot::theme_default(base_family = "sans"))
		library(rstanarm)
		options(mc.cores = 2)
		library(loo)
		library(projpred)
		
		df <- read.csv("df_log_preprocess.csv", header = TRUE)
		df <- df[,c("AGE","INCOME","TRAVTIME","BLUEBOOK","CAR_AGE","YOJ","TGT_CLAIM_FLAG")]
		summary(df)
		
		for (i in 1:6) {
		df[i] <- scale(df[i])
		}
		
		df$TGT_CLAIM_FLAG<- factor(df$TGT_CLAIM_FLAG)
		
		sample_size = floor(0.70*nrow(df))
		train_sample = sample(seq_len(nrow(df)),size = sample_size)
		train_Data = df[train_sample,]
		test_Data = df[-train_sample,]
		
		summary(train_Data)
		summary(test_Data)
		
		y_test = test_Data$TGT_CLAIM_FLAG
		x_test = model.matrix(TGT_CLAIM_FLAG ~ . - 1, data = test_Data)
		
		# preparing the inputs
		x_train <- model.matrix(TGT_CLAIM_FLAG ~ . - 1, data = train_Data)
		y_train <- train_Data$TGT_CLAIM_FLAG
		(reg_formula <- formula(paste("TGT_CLAIM_FLAG ~", paste(names(train_Data)[1:6], collapse = " + "))))
		
		normal_prior <- normal(location =0,scale = 10)
		post_norm_uni <- stan_glm(reg_formula, data = train_Data,
		family = binomial(link = "logit"),
		prior = normal_prior,
		prior_intercept = normal_prior,
		QR=TRUE,
		chain=3,
		iter=50000,
		warmup=10000,
		thin = 5,
		cores = 8)
		
		summary(post_norm_uni)
		prior_summary(post_norm_uni)
		
		pplot<-plot(post_norm_uni, "areas", prob = 0.95, prob_outer = 1)
		pplot+ geom_vline(xintercept = 0)
		
		mcmc_dens_overlay(post_norm_uni)
		mcmc_trace(post_norm_uni)
		plot(post_norm_uni,"acf_bar")
		
		classification_summary(model = post_norm_uni,data = train_Data,cutoff = 0.5)
		classification_summary(model = post_norm_uni,data = test_Data,cutoff = 0.5)
		
		pp_check(post_norm_uni)
		pp_check(post_norm_uni,plotfun = "hist",nreps=5)
		pp_check(post_norm_uni, plotfun = "error_binned")
			library(tidyverse)
			library(caret)
			library(GGally)
			library(ggplot2)
			library(corrplot)
			library(bayesplot)
			library(bayesrules)
			theme_set(bayesplot::theme_default(base_family = "sans"))
			library(rstanarm)
			options(mc.cores = 2)
			library(loo)
			library(projpred)
			
			df <- read.csv("df_log_preprocess.csv", header = TRUE)
			df <- df[,c("AGE","INCOME","TRAVTIME","BLUEBOOK","CAR_AGE","TGT_CLAIM_FLAG")]
			summary(df)
			
			for (i in 1:5) {
			df[i] <- scale(df[i])
			}
			
			df$TGT_CLAIM_FLAG<- factor(df$TGT_CLAIM_FLAG)
			
			sample_size = floor(0.70*nrow(df))
			train_sample = sample(seq_len(nrow(df)),size = sample_size)
			train_Data = df[train_sample,]
			test_Data = df[-train_sample,]
			
			summary(train_Data)
			summary(test_Data)
			
			y_test = test_Data$TGT_CLAIM_FLAG
			x_test = model.matrix(TGT_CLAIM_FLAG ~ . - 1, data = test_Data)
			
			# preparing the inputs
			x_train <- model.matrix(TGT_CLAIM_FLAG ~ . - 1, data = train_Data)
			y_train <- train_Data$TGT_CLAIM_FLAG
			(reg_formula <- formula(paste("TGT_CLAIM_FLAG ~", paste(names(train_Data)[1:5], collapse = " + "))))
			
			t_prior <- student_t(df=6,location =c(-0.239,-0.082,0.148,-0.180,-0.172),
			scale = c(0.026,0.027,0.027,0.027,0.027))
			post_s_in <- stan_glm(reg_formula, data = train_Data,
			family = binomial(link = "logit"),
			prior = t_prior,
			QR=TRUE,
			chain=3,
			iter=50000,
			warmup=10000,
			thin = 5,
			cores = 8)
			
			summary(post_s_in)
			prior_summary(post_s_in)
			
			pplot<-plot(post_s_in, "areas", prob = 0.95, prob_outer = 1)
			pplot+ geom_vline(xintercept = 0)
			
			mcmc_dens_overlay(post_s_in)
			mcmc_trace(post_s_in)
			plot(post_s_in,"acf_bar")
			
			classification_summary(model = post_s_in,data = train_Data,cutoff = 0.5)
			classification_summary(model = post_s_in,data = test_Data,cutoff = 0.5)
			
			pp_check(post_s_in)
			pp_check(post_s_in,plotfun = "hist",nreps=5)
			pp_check(post_s_in_s_in, plotfun = "error_binned")
				library(tidyverse)
				library(caret)
				library(GGally)
				library(ggplot2)
				library(corrplot)
				library(bayesplot)
				library(bayesrules)
				theme_set(bayesplot::theme_default(base_family = "sans"))
				library(rstanarm)
				options(mc.cores = 2)
				library(loo)
				library(projpred)
				
				df <- read.csv("df_log_preprocess.csv", header = TRUE)
				df <- df[,c("AGE","INCOME","TRAVTIME","BLUEBOOK","CAR_AGE","TGT_CLAIM_FLAG")]
				summary(df)
				
				for (i in 1:5) {
				df[i] <- scale(df[i])
				}
				
				df$TGT_CLAIM_FLAG<- factor(df$TGT_CLAIM_FLAG)
				
				sample_size = floor(0.70*nrow(df))
				train_sample = sample(seq_len(nrow(df)),size = sample_size)
				train_Data = df[train_sample,]
				test_Data = df[-train_sample,]
				
				summary(train_Data)
				summary(test_Data)
				
				y_test = test_Data$TGT_CLAIM_FLAG
				x_test = model.matrix(TGT_CLAIM_FLAG ~ . - 1, data = test_Data)
				
				# preparing the inputs
				x_train <- model.matrix(TGT_CLAIM_FLAG ~ . - 1, data = train_Data)
				y_train <- train_Data$TGT_CLAIM_FLAG
				(reg_formula <- formula(paste("TGT_CLAIM_FLAG ~", paste(names(train_Data)[1:5], collapse = " + "))))
				
				t_prior <- student_t(df=6,location =0,scale = 10)
				post_s_uni <- stan_glm(reg_formula, data = train_Data,
				family = binomial(link = "logit"),
				prior = t_prior,
				QR=TRUE,
				chain=3,
				iter=50000,
				warmup=10000,
				thin = 5,
				cores = 8)
				
				summary(post_s_uni)
				prior_summary(post_s_uni)
				
				pplot<-plot(post_s_uni, "areas", prob = 0.95, prob_outer = 1)
				pplot+ geom_vline(xintercept = 0)
				
				mcmc_dens_overlay(post_s_uni)
				mcmc_trace(post_s_uni)
				plot(post_s_uni,"acf_bar")
				
				classification_summary(model = post_s_uni,data = train_Data,cutoff = 0.5)
				classification_summary(model = post_s_uni,data = test_Data,cutoff = 0.5)
				
				pp_check(post_s_uni)
				pp_check(post_s_uni,plotfun = "hist",nreps=5)
				pp_check(post_s_uni, plotfun = "error_binned")