Statistical Methods III
End-Semestral Project

Fall, 2022

[Predicting Breast-Cancer using Logistic, Probit and Similar Regression Techniques]

Alaap Kumar Mukhopadhyay(BS2125)
Srijan Chattopadhyay(BS2126)

Indian Statistical Institute, Kolkata

Contents

1 Aim of the Project **3**

2 Theoretical Background of the regression models **3**
- 2.1 **Logistic Regression:** . . . 3
- 2.2 **Probit Regression:** . . . 3
- 2.3 **Something Similar:** . . . 3

3 Our Data **3**
- 3.1 **Source:** . . . 4
- 3.2 **Data Description** . . . 4
- 3.3 **More details** . . . 5

4 Fitted Models **5**
- 4.1 **Logistic Regression Model:** . . . 5
- 4.2 **Probit Regression Model:** . . . 6
- 4.3 **T-Distribution Regression Model(df=43)[]will be explained later]:** . . . 6
- 4.4 **Double Exponential Regression Model:** . . . 6

5 Analysis and Inference **7**
- 5.1 **Confusion Matrix and generally used measures:** . . . 7
- 5.2 **Drawbacks:** . . . 7
- 5.3 **Two pathways:** . . . 7
- 5.4 **Confusion Matrices** . . . 8
  - 5.4.1 **Logistic Regression Model:** . . . 8
  - 5.4.2 **Probit Regression Model:** . . . 8
  - 5.4.3 **t-Distribution Regression Model:** . . . 9
  - 5.4.4 **Double Exponential Regression Model:** . . . 9
- 5.5 **Which one is the best model then??** . . . 9

6 Final Conclusion **11**

7 R-Codes **11**

8 Acknowledgement: **16**

1 Aim of the Project

The features of our data are computed from a digitized image of a Fine Needle Aspiration Cytology (FNAC) of a breast mass. We will try to develop a good model which will diagnose breast cancer on the basis of FNAC without doing biopsy. We haven't ourselves extracted the features from the image, the dataset of extracted features was already avaliable to us.
Our goal in this project is to find a suitable regression model to serve the above mentioned purpose. We fit our data into logit, probit, t-distribution and double exponential, and try to find out which model provides us with the best results in terms of (?) metric. We will define some criteria and use some result to reach at a conclusion in terms of fitting the best model(?) that we can also use for our future purpose. Whenever needed, we will do some estimation and hypothesis testing.

2 Theoretical Background of the regression models

2.1 Logistic Regression:

Here the model assumption is $y_i \sim \text{Bern}(p_i)$, where $E(y_i|x_{i1}, .., x_{i(k-1)}) = p_i$. So, $log(\frac{p_i}{1-p_i}) = x_i^T\beta$. We perform the mle estimation of β. Our log likelihood is

$$l(\beta|data) = \sum_{i=1}^{n} y_i ln p_i + (1-y_i)ln(1-p_i) = \sum_{i=1}^{n} y_i(x_i^T\beta) - ln(1+e^{x_i^T\beta})$$

Then we have to find the roots of $\frac{dl(\beta)}{d\beta} = 0$.For that different iteration method like Fisher-Scoring or IRLS etc. are used.

2.2 Probit Regression:

It is the same as that of logistic, except that here we use $p_i = \Phi(x_i^T\beta)$, where Φ is the cdf of standard normal distribution. Then we perform the same process as of the previous.

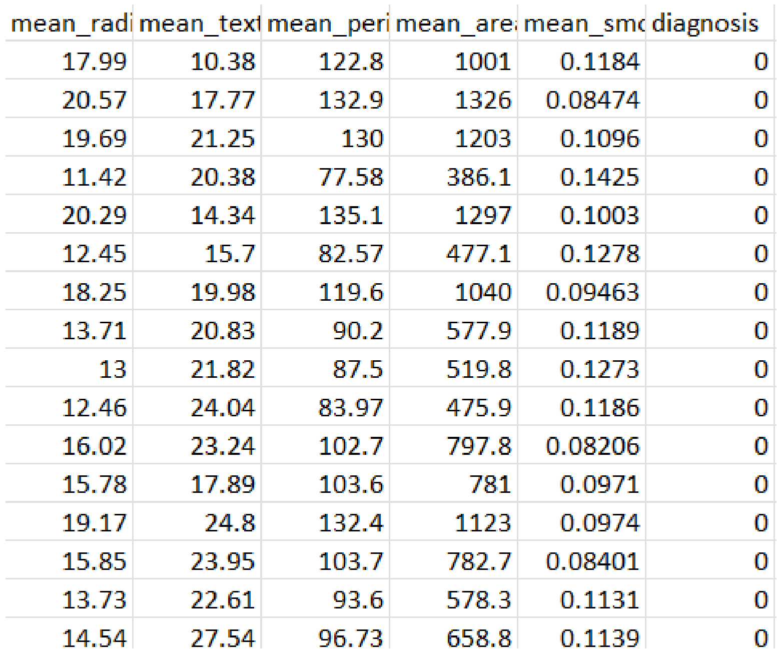
2.3 Something Similar:

Inspired by the idea of probit regression, it is very clear that in spite of normal, if we take any symmetric distribution, then that will also work. So, we also tried this just by replacing the normal assumption by another distributional assumption, i.e. replacing Φ by F in probit, where F is the corresponding CDF.So $p_i = F(x_i^T\beta)$ We tried with t-distribution with different degrees of freedom and double exponential. We didn't consider cauchy becuase it doesn't have any finite moments and hence the results won't be consistent any more.

3 Our Data

3.1 Source:

Our data consists of 569 rows and 6 columns namely ”mean radius”,”mean texture”,”mean perimeter”,”mean area”,”mean smoothness”,”diagnosis”. The dataset is available at https://www.kaggle.com/datasets/merishnasuwal/breast-cancer-prediction-dataset. Before fitting the model, we have to split the dataset first into two pieces- Training data and Testing data. To avoid this selection bias, we first took a random permutation of the rows and then took the first 469 rows as training data and rest 100 rows for testing. This gives us the splitting of the whole dataset. Now we move on to the fitting of the models.
The data-head looks like this:



mean_radi	mean_text	mean_peri	mean_are	mean_smo	diagnosis
17.99	10.38	122.8	1001	0.1184	0
20.57	17.77	132.9	1326	0.08474	0
19.69	21.25	130	1203	0.1096	0
11.42	20.38	77.58	386.1	0.1425	0
20.29	14.34	135.1	1297	0.1003	0
12.45	15.7	82.57	477.1	0.1278	0
18.25	19.98	119.6	1040	0.09463	0
13.71	20.83	90.2	577.9	0.1189	0
13	21.82	87.5	519.8	0.1273	0
12.46	24.04	83.97	475.9	0.1186	0
16.02	23.24	102.7	797.8	0.08206	0
15.78	17.89	103.6	781	0.0971	0
19.17	24.8	132.4	1123	0.0974	0
15.85	23.95	103.7	782.7	0.08401	0
13.73	22.61	93.6	578.3	0.1131	0
14.54	27.54	96.73	658.8	0.1139	0

Figure 1: Data

3.2 Data Description

Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass.

- **Mean Radius**: Mean of distances from center to points on the perimeter
- **Mean Texture**: Standard deviation of gray-scale values
- **Mean Perimeter**
- **Mean Area**

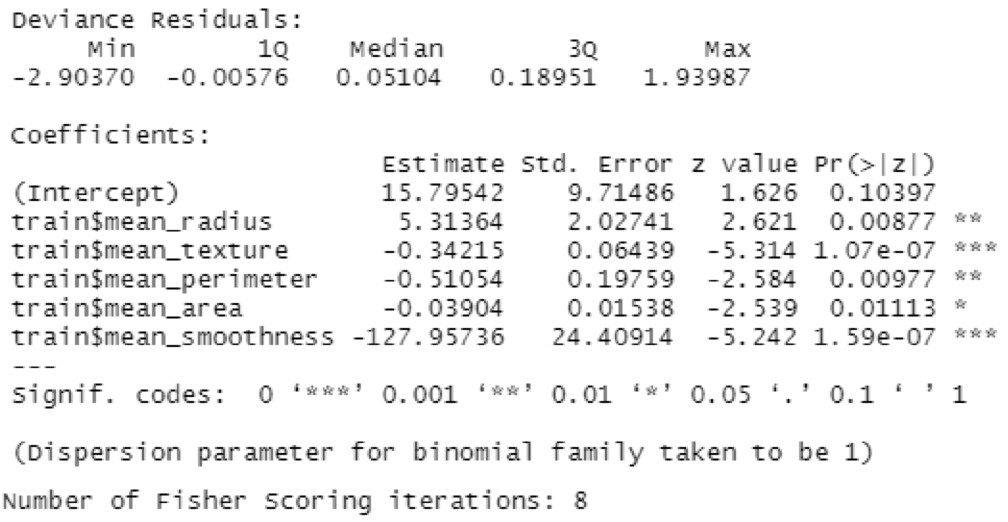
- **Mean Smoothness**: Local variation in radius lengths
- **Diagnosis**

3.3 More details

The dataset used is publicly available and was created by Dr. William H. Wolberg, physician at the U.O. Wisconsin Hospital at Madison, USA. To create the dataset Dr. Wolberg used fluid samples, taken from patients with solid breast masses and an easy-to-use graphical computer program called Xcyt, which is capable of perform the analysis of cytological features based on a digital scan. The program uses a curve-fitting algorithm, to compute features from each one of the cells in the sample, than it calculates the mean value, extreme value and standard error of each feature for the image.(Source: `https://towardsdatascience.com/building-a-simple-machine-learning-model-on-breast-cancer-data-eca4b3b99fa3`)

4 Fitted Models

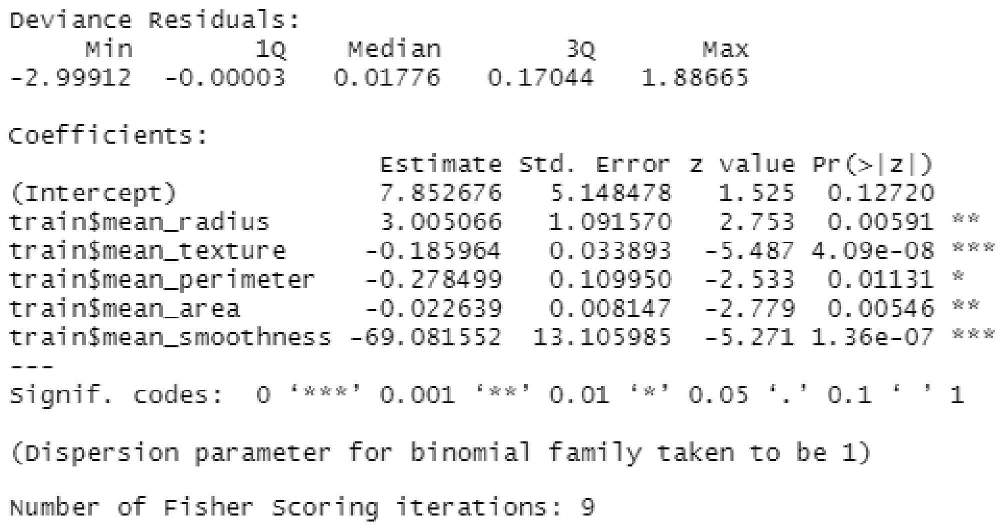
4.1 Logistic Regression Model:



```
Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.90370  -0.00576   0.05104   0.18951   1.93987

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)               15.79542    9.71486   1.626  0.10397
train$mean_radius          5.31364    2.02741   2.621  0.00877 **
train$mean_texture        -0.34215    0.06439  -5.314 1.07e-07 ***
train$mean_perimeter      -0.51054    0.19759  -2.584  0.00977 **
train$mean_area           -0.03904    0.01538  -2.539  0.01113 *
train$mean_smoothness -127.95736   24.40914  -5.242 1.59e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Number of Fisher Scoring iterations: 8
```

Figure 2: Summary of Logit Regression Model

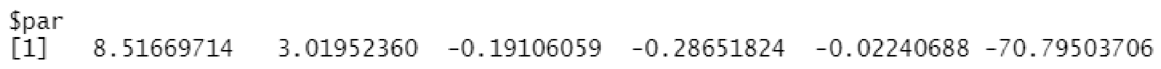
4.2 Probit Regression Model:



```
Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.99912  -0.00003   0.01776   0.17044   1.88665

Coefficients:
                        Estimate Std. Error z value Pr(>|z|)
(Intercept)             7.852676   5.148478   1.525  0.12720
train$mean_radius       3.005066   1.091570   2.753  0.00591 **
train$mean_texture     -0.185964   0.033893  -5.487 4.09e-08 ***
train$mean_perimeter   -0.278499   0.109950  -2.533  0.01131 *
train$mean_area        -0.022639   0.008147  -2.779  0.00546 **
train$mean_smoothness -69.081552  13.105985  -5.271 1.36e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Number of Fisher Scoring iterations: 9
```

Figure 3: Summary of Probit Regression Model

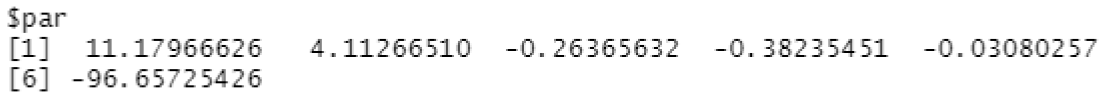
4.3 T-Distribution Regression Model(df=43)[]will be explained later]:



```
$par
[1]   8.51669714   3.01952360  -0.19106059  -0.28651824  -0.02240688 -70.79503706
```

Figure 4: Coefficients of the T-Distribution Regression Model

4.4 Double Exponential Regression Model:



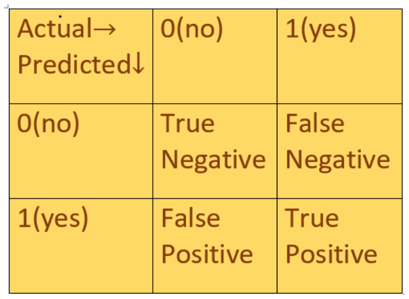
```
$par
[1]  11.17966626   4.11266510  -0.26365632  -0.38235451  -0.03080257
[6] -96.65725426
```

Figure 5: Coefficients of the Double Exponential Regression Model

5 Analysis and Inference

5.1 Confusion Matrix and generally used measures:

For judging how much effective the model is we can try to see the confusion matrix whose elements are true negative(TN), false positive(FP), false negative(FN), true positive(TP). Here is how it looks like:



Actual→ Predicted↓	0(no)	1(yes)
0(no)	True Negative	False Negative
1(yes)	False Positive	True Positive

Figure 6: Confusion Matrix

Here are different mostly used measures w.r.t. which we can judge a model from the confusion matrix:

1. **Accuracy:**$=\frac{TP+TN}{TP+FP+TN+FN}$
2. **Sensitivity:**$=\frac{TP}{TP+FN}$
3. **Specificity:**$=\frac{TN}{TN+FP}$

5.2 Drawbacks:

But, if we choose model considering a certain metric of the matrix, then we may be misguided, i.e. the decision may vary from person to person due to different psychology. For example, one may think that after admitting a non cancer person into a hospital, he/she may die out of tension. Also another may think from a different perspective. So, clearly there is no universal conclusion if we use these type of measures.

5.3 Two pathways:

To get rid of this, we can move into two paths:

1. The **first one** is to use Matthews correlation coefficient (MCC). As proved in `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6941312/`, **"The Matthews correlation coefficient (MCC)**, instead, is a more reliable statistical rate which produces a high score only if the prediction obtained good results in all of the four confusion matrix categories (true positives, false negatives, true negatives, and false positives), proportionally both to the size of positive elements and the size of negative elements in the dataset".

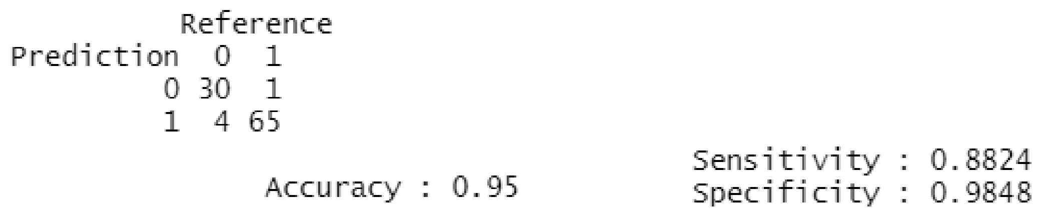
Hence we can judge based on the MCC, which is $\frac{TP.TN-FP.FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$. Intuitively, we can see that, here four balance ratios are considered. As it is not there in the previous metrics, due to that, it is more informative about the matrix.

2. The **second pathway** is to maximize total (economical) revenue in terms of cost, penalty, advantage etc. It depends upon experience, correct information about the field etc. But for our data, as we don't have correct information about the price of treatment etc., so we won't consider this pathway. Rather, we will try to select the **best model(possibly)** based on the first pathway.

So, we will first take the model(i.e., cutoff) which has the highest MCC for each regression and then compare among them again by MCC to reach our conclusion. Cutoff means the probability above which we tell it 1[i.e, say the cutoff is 0.5, then if our output is 0.6, we call it 1 and if 0.3, then 0].

5.4 Confusion Matrices

5.4.1 Logistic Regression Model:

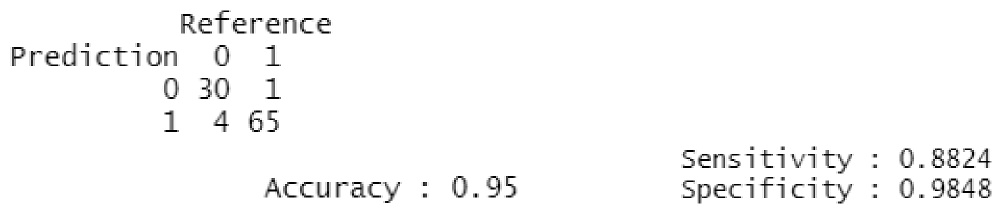


```
          Reference
Prediction  0  1
         0 30  1
         1  4 65
                                    Sensitivity : 0.8824
              Accuracy : 0.95       Specificity : 0.9848
```

Figure 7: Confusion Matrix and related metrics of Logit Regression Model at 0.18 cutoff

The MCC obtained here is maximum(0.8882312) here among all the cutoffs. So, this is the best model for logistic regression(according to MCC metric)

5.4.2 Probit Regression Model:

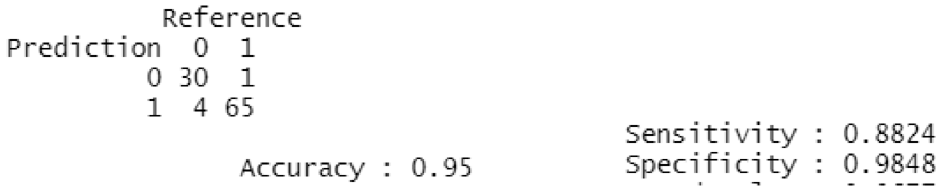


```
          Reference
Prediction  0  1
         0 30  1
         1  4 65
                                    Sensitivity : 0.8824
              Accuracy : 0.95       Specificity : 0.9848
```

Figure 8: Confusion Matrix and related metrics of Probit Regression Model at 0.20 cutoff

The MCC obtained here is maximum(0.8882312) here among all the cutoffs. So, this is the best model for Probit regression(according to MCC metric)

5.4.3 t-Distribution Regression Model:

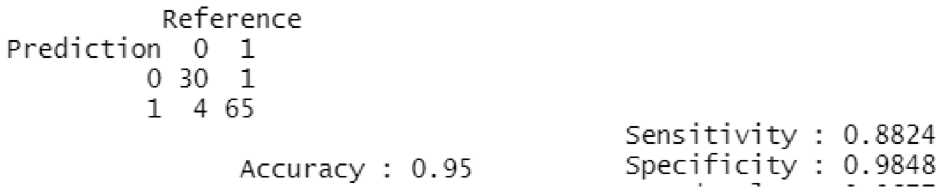


```
            Reference
Prediction  0  1
         0 30  1
         1  4 65
                                   Sensitivity : 0.8824
               Accuracy : 0.95     Specificity : 0.9848
```

Figure 9: Confusion Matrix and related metrics of T-Distribution(df=43) Regression Model at 0.20 cutoff

The MCC obtained here is maximum(0.8882312) here among all the cutoffs. So, this is the best model for t-distribution regression(according to MCC metric)

5.4.4 Double Exponential Regression Model:



```
            Reference
Prediction  0  1
         0 30  1
         1  4 65
                                   Sensitivity : 0.8824
               Accuracy : 0.95     Specificity : 0.9848
```

Figure 10: Confusion Matrix and related metrics of Double Exponential Regression Model at 0.17 cutoff

The MCC obtained here is maximum(0.8882312) here among all the cutoffs. So, this is the best model for double exponential regression(according to MCC metric).

5.5 Which one is the best model then??

The most unbiased model is if we take the cutoff to be 0.5, i.e., it is equally placed to the different errors. So, as we move towards 0.5, our model will have less bias. As four of them are giving same MCC, so we will be taking the one which is less unbiased,i.e., the maximum cutoff, here 0.20. Here are such two, one is T-Distribution(df=43) Regression and another is Probit regression. So, now let's compare these two by random testing datasets.

So, what we did is the following: We did a random sampling from our dataset only, i.e. for i = 100 to 469, we randomly selected i many rows of the dataset and called that to be our testing data, and tested both the models and for each i, we got a cutoff which gives the maximum MCC. So, we can think it in this way that we did a random sampling from the distribution of the optimum cutoffs. So, we have 370 points from the distribution of optimum cutoff for both the models. Similarly, we now have enough datapoints from the distribution of the best mcc for both the models.

So, we now do the following hypothesis testing: We assume that the random variable **'Best Mcc for probit model(X)'** follows $N(\mu_1, \sigma^2)$ and that for t-distrbution(df=43)(Y) follows $N(\mu_2, \sigma^2)$

$$H_0 : \mu_2 = \mu_1 \quad H_1 : \mu_2 \neq \mu_1$$

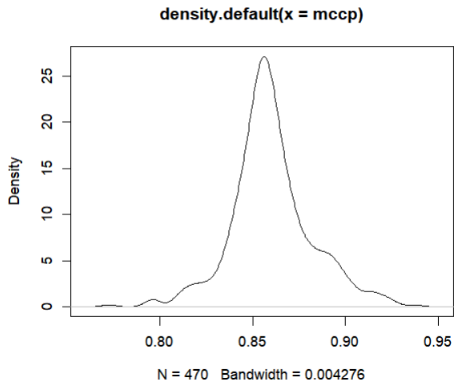


Figure 11: MCC distn Probit

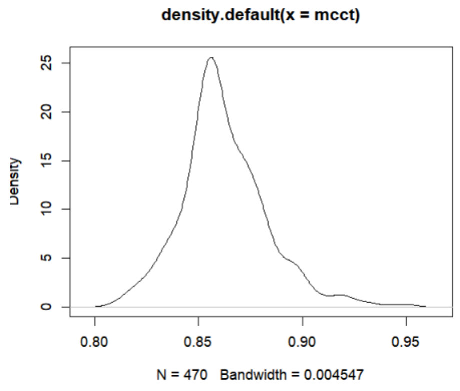


Figure 12: MCC distrn T_{43}

We assume σ to be unknown. So, a size α test will be

$$\mathbb{1}[\frac{|\bar{Y} - \bar{X}|}{S_p\sqrt{\frac{1}{n} + \frac{1}{n}}} \geq t_{2n-2,\frac{\alpha}{2}}]$$

Corresponding p value will be

$$P(X, Y) = 2(1 - F(\frac{|\bar{Y} - \bar{X}|}{S_p\sqrt{\frac{2}{n}}}))$$

, F is the cdf of t_{2n-2}.

In our case, $\bar{Y} = 0.8624161$, $\bar{X} = 0.8600604$, $S_p = 0.4642769$ and the value of corresponding test-statistic is 0.07778159. But $t_{938,0.025}$ is 1.962496. So, we accept(fail to reject) our null hypothesis at 0.05 level(size) of significance and corresponding p value is 0.9380184.

6 Final Conclusion

So, we can see that both the probit and t-distribution models are similarly good in terms of MCC. The average cutoff for probit regression is 0.3389574, and that of t-distribution regression(df=43) is 0.3356596. So, for future prediction, we can use the cutoff as 0.33 and can use any of the two models. So, our final model is these two, because as we have shown that, on an average, they are equally effective. And, from figure 3 it is clear that, all of them are having $P(>|Z|)$ value less than 0.01, i.e., the probability that we can observe in general some value which is as extreme as those are significantly less, infact 2 of them are even lesser than 10^{-06}. So, clearly all the predictors are significant hence we don't wanna remove any predictor from our final model. And, the last few processes of model selection was for removing the bias of the model to some specific data as much as possible. Here our assumption is, the joint distribution followed by all the random variables of our dataset is same as all from where data will be coming in near future. And, due to that assumption and the last few procedures affirm that if we use this model only for future dataset, the model should perform well.

7 R-Codes

```
##Data selection
data=read.csv(file.choose(),header=T)


##install.packages("nimble")
library(nimble)

##Randomize the rows of the data
set.seed(42)
rows=sample(nrow(data))
data1=data[rows,]

##Training and Testing
train=data1[1:469,]
test=data1[470:569,]

###
X_train = cbind(1,train$mean_radius,train$mean_texture,
train$mean_perimeter,train$mean_area,train$mean_smoothness)
```

```r
Y_train = train$diagnosis
K = ncol(X_train)
X_test = cbind(1,test$mean_radius,test$mean_texture,
test$mean_perimeter,test$mean_area,test$mean_smoothness)
Y_test = test$diagnosis
##Train the model
#Logistic
model1=glm(train$diagnosis~train$mean_radius+
train$mean_texture+train$mean_perimeter+train$mean_area+
train$mean_smoothness,
           data=train,
           family=binomial
)
summary(model1)

#Probit
model2=glm(train$diagnosis~train$mean_radius+
train$mean_texture+train$mean_perimeter+train$mean_area+
train$mean_smoothness,
           data=train,
           family = binomial(link = "probit"))
summary(model2)

#t-distribution with 43 degrees of freedom
probit.nll <- function (beta) {
  # linear predictor
  eta <- X_train %*% beta
  # probability
  p <- pt(eta,43)
  # negative log-likelihood
  -sum((1 - Y_train) * log(1 - p) + Y_train * log(p))
}

# requires model matrix `X` and binary response `Y`
probit.gr <- function (beta) {
  # linear predictor
  eta <- X_train %*% beta
  # probability
  p <- pt(eta,43)
  # chain rule
  u <- dt(eta,43) * (Y_train - p) / (p * (1 - p))
  # gradient
  -crossprod(X_train, u)
}

vi = lm(train$diagnosis ~ train$mean_radius
+train$mean_texture+train$mean_perimeter
+train$mean_area+train$mean_smoothness)
```

```r
v=c(3.485330,0.034005,-0.022130,-0.031211,0.000955,-6.906395)
fit1 <- optim(v, probit.nll, gr = probit.gr, method = "BFGS", hessian =  TRUE)
fit1

#double exponential
probit.nll1 <- function (beta) {
  # linear predictor
  eta <- X_train %*% beta
  # probability
  p <- pdexp(eta)
  # negative log-likelihood
  -sum((1 - Y_train) * log(1 - p) + Y_train * log(p))
}

# requires model matrix `X` and binary response `Y`
probit.gr1 <- function (beta) {
  # linear predictor
  eta <- X_train %*% beta
  # probability
  p <- pdexp(eta)
  # chain rule
  u <- ddexp(eta) * (Y_train - p) / (p * (1 - p))
  # gradient
  -crossprod(X_train, u)
}

vi = lm(train$diagnosis ~ train$mean_radius+
train$mean_texture+train$mean_perimeter+train$mean_area+
train$mean_smoothness)
v=c(3.485330,0.034005,-0.022130,-0.031211,0.000955,-6.906395)
fit2 <- optim(v, probit.nll1, gr = probit.gr1,
method = "BFGS", hessian =  TRUE)
fit2

##Prediction
library(caret)
#install.packages("mltools")
library(mltools)
#Logistic
beta1=model1$coefficients
m1=X_test %*% beta1
val1=exp(m1)/(1+exp(m1))
pred1=c(1:100)*0
for(i in 1:100){
  if(val1[i]>0.18) pred1[i]=1
  else pred1[i]=0}
confusionMatrix(as.factor(pred1),as.factor(test$diagnosis))
mcc1=mcc(pred1,test$diagnosis)
```

```
#Probit
beta2=model2$coefficients
m2=X_test %*% beta2
val2=pnorm(m2)
pred2=c(1:100)*0
for(i in 1:100){
  if(val2[i]>0.20) pred2[i]=1
  else pred2[i]=0}
confusionMatrix(as.factor(pred2),as.factor(test$diagnosis))
mcc2=mcc(pred2,test$diagnosis)

#t-distribution
beta3=fit1$par
m3=X_test %*% beta3
val3=pt(m3,43)
pred3=c(1:100)*0
for(i in 1:100){
  if(val3[i]>0.20) pred3[i]=1
  else pred3[i]=0}
confusionMatrix(as.factor(pred3),as.factor(test$diagnosis))
mcc3=mcc(pred3, test$diagnosis)

#Double Exponential
beta4=fit2$par
m4=X_test %*% beta4
val4=pdexp(m4)
pred4=c(1:100)*0
for(i in 1:100){
  if(val4[i]>0.17) pred4[i]=1
  else pred4[i]=0
}
confusionMatrix(as.factor(pred4),as.factor(test$diagnosis))
mcc4=mcc(pred4, test$diagnosis)


##Testing on Random testing sets

#Probit
mccp=c(100:569)*0
cutp=c(100:569)*0+0.15
sigm=c(0.15,0.16,0.17,0.18,0.19,0.20,0.21,0.22,
0.23,0.24,0.25,0.26,0.27,0.28,0.29,0.30,0.31,0.32,
0.33,0.34,0.35,0.36,0.37,0.38,0.39,0.40)
for(i in 100:569){
  test=data[sample(nrow(data),i),]
  X_test = cbind(1,test$mean_radius, test$mean_texture,
  test$mean_perimeter, test$mean_area, test$mean_smoothness)
  beta2 = model2$coefficients
  mp=X_test %*% beta2
  valp = pnorm(mp)
  pred = c(1:i)*0
  mccp[i-99]=0
  for (k in sigm){
    for(j in 1:i) {
      if( valp[j] > k) pred[j] = 1
      else pred[j] =0
    }
    mccp1[i-99]=mcc(pred, test$diagnosis)
    if(mccp1[i-99]>mccp[i-99]) {
      mccp[i-99]=mccp1[i-99];
      cutp[i-99]=k
      }
    else {mccp[i-99]=mccp[i-99]
          cutp[i-99]=cutp[i-99]}
  }}


#T-Distribution with df=43
mcct=c(100:569)*0
mcct1=c(100:569)*0
cutt=c(100:569)*0+0.15
sigm=c(0.15,0.16,0.17,0.18,0.19,0.20,0.21,0.22,0.23,
0.24,0.25,0.26,0.27,0.28,0.29,0.30,0.31,0.32,0.33,
0.34,0.35,0.36,0.37,0.38,0.39,0.40)
for(i in 100:569){
  test=data[sample(nrow(data),i),]
  X_test = cbind(1,test$mean_radius, test$mean_texture,
  test$mean_perimeter, test$mean_area, test$mean_smoothness)
  beta3 = fit1$par
  mt=X_test %*% beta3
  valt = pt(mt,43)
  predt = c(1:i)*0
  mcct[i-99]=0
  for (k in sigm){
    for(j in 1:i) {
      if( valt[j] > k) predt[j] = 1
      else predt[j] =0
    }
    mcct1[i-99]=mcc(predt, test$diagnosis)
    if(mcct1[i-99]>mcct[i-99]) {
      mcct[i-99]=mcct1[i-99];
      cutt[i-99]=k
    }
    else {mcct[i-99]=mcct[i-99]
    cutt[i-99]=cutt[i-99]}
  }}


###Final judgement
plot(density(mcct))
mean(mcct)
mean(mccp)
sum((mccp-mean(mccp))^2)
sum((mcct-mean(mcct))^2)
(0.8624161-0.8600604)/(0.4642769*sqrt(2/470))
qt(0.95,938)
mean(cutp)
mean(cutt)
```

8 Acknowledgement:

We would like to thank Dr. Shyamal Krishna De, Applied Statistics Unit(ASU), ISI Kolkata. We have also taken some resources(data, information etc.) from the following resources:

- https://www.kaggle.com/datasets/merishnasuwal/breast-cancer-prediction-dataset
- https://towardsdatascience.com/building-a-simple-machine-learning-model-on-breast-cancer-data-eca4b3b99fa3
- https://www.wikipedia.org/
- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6941312/