Breast Cancer Prediction using Bayesian Analysis and Generalised Additive Models

Ali Rajabimehr & Craig Manning

STAT802 Advanced Topis in Analytics

Victor Miranda & Parricio Andres Maturana Russel

31 May 2024

Abstract

Breast cancer represents a significant global health challenge, early detection and accurate diagnosis is crucial for successful treatment. This study aims to predict breast cancer diagnosis using features derived from digitized images of Fine Needle Aspirates (FNA) of breast masses. Cell nuclei were analysed for their size, shape and texture, the mean value, largest (or 'worst') value and the standard error of each feature was then identified. Bayesian Logistic Regression and Generalised Additive Models were used to identify significant predictors and the performance of these approaches was compared. The Bayesian model with informative priors achieved an accuracy of 93.15%, while the GAM model slightly outperformed the Bayesian model with an accuracy of 94.90%. Key predictors for malignancy included the cell nucleus radius mean, compactness mean and perimeter standard.

Introduction

Breast cancer is the world's most prevalent cancer. In 2022, there were 2.3 million woman diagnosed with breast cancer and 670,000 deaths worldwide. (World Health Organisation, 2024).

Breast cancer typically originates in the epithelial cells of the breast tissue and can metastasise to other organs in the body, with potentially fatal outcomes. Surgery is often the first line of defence, which involves removing the cancerous tissue to minimize the risk of the cancer spreading. Surgery can also be combined with radiation therapy and medication such as chemotherapy and hormone therapy depending on the needs of the patient. The overall effectiveness of breast cancer treatment significantly depends on early detection and timely

intervention, which can significantly improve survival rates. (New Zealand Cancer Society, 2024).

This study aims to predict breast cancer diagnosis using features derived from the various characteristics of cell nuclei present in the Fine Needle Aspirate (FNA) images. FNA is a minimally invasive diagnostic procedure that involves extracting cells from a breast mass for examination and is often used alongside other diagnostic tools (eg. Mammograms and ultrasounds) to diagnose breast cancer. (Breast Cancer Aotearoa Coalition, 2024).

Objectives

The primary objective of this study is to accurately predict whether a patient has cancer based on features derived from FNA images. The specific goals of this study are:

1. Identify significant predictors of breast cancer
2. Estimating the probability of breast cancer by varying predictor values
3. Evaluating the accuracy of the Bayesian and GAM models
4. Exploring the use of GAMs for capturing non-linear relationships between predictors

Research Design

3.1 Dataset description

The dataset used is this study is the Breast Cancer Wisconsin (Diagnostic) database available from https://archive.ics.uci.edu. The dataset contained 569 cases, with 357 Benign cases and 212 Malignant cases. Each case had 32 attributes, including the patient ID, the diagnosis (M = Malignant, B= Benign) and 30 contextual features computed from the boundary or each cell nucleus. The extracted features are as follows:

1. *Radius:* The average length of radial lines from the center of the cell nucleus to the points on the cell boundary

2. *Perimeter:* The total distance between the points that define the cell nucleus boundary
3. *Area:* The number of pixels within the cell nucleus boundary
4. Compactness: A measure of the compactness of the cell nucleus, calculated as perimeter^2 / area
5. *Smoothness:* The difference between the length of the radial line and the mean length of the surrounding lines
6. *Concavity:* The magnitude of the indentations in the cell nucleus boundary, measured by calculating the distance between each indentation and the cell boundary
7. *Concave points:* The number of concavities in the cell nucleus represented as a count of the number of concavities
8. *Symmetry:* The difference in length between the lines perpendicular to the major axis of the cell nucleus to the nucleus boundary on both sides
9. *Fractal dimensions:* The complexity of the nucleus boundary at different scales
10. *Texture:* The variance of grey scale intensities with the pixels of the cell nucleus within the imaging

For each image, the mean value, the extreme (largest or worst) value and the standard error of each feature was computed, resulting in 30 features. The worst values are considered the most useful for identifying malignant cells, since only a few malignant cells may occur in each sample (Street, W. et al, 1993)

3.2 Feature selection

A combination of correlation analysis, Analysis of Variance (ANOVA) and Variance Inflation Factor (VIF) were used to identify the most relevant features for predicting breast cancer.

Correlation analysis was used to measure the relationship between each predictor and the diagnosis, a threshold of 0.5 was used to identify features which had a strong correlation with the diagnosis. ANOVA was then used to determine whether there were significant differences in means between the different diagnosis. A p-value threshold of 0.05 was used to single out features which were statistically significant. Finally, a VIF threshold of 10 was applied to identify features with high multicollinearity and remove them from the model.

The combination of these methods ensures that the features selected have a strong relationship with the diagnosis and are not highly correlated with each other. The final set of selected features included radius (mean), compactness (mean), perimeter standard error, compactness (worst), concavity (worst), and concave points worst.

3.3 Feature standardisation

The Area (Mean), Perimeter (Mean) and Radius (Mean) have larger ranges than other features. In order to improve the convergence and a better interpretation of the coefficients, the selected features were standardised using the Scale() function in R (by subtracting the mean and dividing by the standard deviation). Without scaling, these features could have a disproportionate impact on the model interpretation.

3.4 Bayesian Logistic regression

A Bayesian logistic regression model with different priors was then fitted using the Just Another Gibbs Sample (JAGS) in R with the feature selection. The model formula is:

$$logit(p_i) = \beta_0 + \beta_1 \times radius\_mean_i + \beta_2 \times compactness\_mean_i + \beta_3 \times perimeter\_se_i + \beta_4 \times compactness\_worst_i + \beta_5 \times concavity\_worst_i + \beta_6 \times concave\_points\_worst_i$$

Where p_i is the probability of a malignant diagnosis, and β_0, $\beta_1 \ldots \beta_6$ are the beta coefficients.

Five different sets of priors were then used:

1. Non-informative: $\beta_0 \sim N(0, 0.01)$, $\beta_j \sim N(0, 0.01)$
2. Informative (1): $\beta_0 \sim N(0, 1)$, $\beta_j \sim N(0, 1)$
3. Informative (2): $\beta_0 \sim N(0, 2)$, $\beta_j \sim N(0, 2)$
4. Informative (3): $\beta_0 \sim N(0, 0.1)$, $\beta_j \sim N(0, 0.1)$
5. Informative (4): $\beta_0 \sim N(0, 0.5)$, $\beta_j \sim N(0, 0.5)$

The analysis was conducted using R and the Just Another Gibbs Sampler (JAGS) package with 4 chains, 5,000 iterations and a burn-in of 1,000. The trace plots and the effective sample sizes were then analysed to determine convergence.

3.5 Generalised Additive models

In addition to the Bayesian logistic regression, A Generalised Additive Model (GAM) was fitted to the selected features. The GAM allows for identification of non-linear relationships between the predictors and the diagnosis. The model was fitted using the MGCV package in R.

ResultsCorrelation analysis

A correlation analysis was performed to identify features highly correlated with the diagnosis (malignant or benign). Features with an absolute correlation greater than 0.5 with the diagnosis were selected, these are shown in the table below.

Table 1

Feature selection based on correlation with Diagnosis

Feature	Correlated with Diagnosis
Radius (mean)	0.730
Perimeter (mean)	0.742
Area (mean)	0.708
Compactness (mean)	0.598
Concavity (mean)	0.685
Concave points (mean)	0.776
Radius (Standard error)	0.567
Perimeter (Standard error)	0.556
Area (Standard error)	0.548
Radius (Worst)	0.776
Perimeter (Worst)	0.782
Area (Worst)	0.733
Compactness (Worst	0.598
Concavity (Worst)	0.682
Concave points (Worst)	0.763

Note: A correlation plot with all features is included in *Appendix A.*

4.2 ANOVA Test

An ANOVA test was then conducted to determine the features significantly associated with the diagnosis. Features with p-values less than 0.05 were selected. The results of ANOVA are shown in the table below:

Table 2

Features selection based on ANOVA

Feature	ANOVA p-value
Radius (Mean)	< 2e-16
Texture (Mean)	< 2e-16
Perimeter (Mean)	< 2e-16
Area (Mean)	< 2e-16
Smoothness (Mean)	1.11e-05
Compactness (Mean)	< 2e-16
Concavity (Mean)	< 2e-16
Concave points (Mean)	< 2e-16
Symmetry (Mean)	1.38e-06
Radius (Standard error)	5.06e-06
perimeter (Standard error)	3.43e-06
area (Standard error)	2.15e-06
compactness (Standard error)	6.63e-05
concavity (Standard error)	4.64e-05
Concave points (Standard error)	3.37e-06
Radius worst (Standard error)	< 2e-16
Texture worst (Standard error)	< 2e-16
Perimeter worst (Standard error)	< 2e-16
Area worst (Standard error)	< 2e-16

Smoothness (Worst)	2.62e-05
Compactness (Worst)	< 2e-16
Concavity (Worst)	< 2e-16
Concave points (Worst)	< 2e-16
Symmetry (Worst)	4.12e-07
Fractal dimension (Worst)	0.0004

4.3 Combined features

The features from both the correlation analysis and the ANOVA test were then intersected to identify the most relevant features. The combined selected features are shown in the table below:

Table 3

Combined feature selection – Correlation and ANOVA

Feature
Radius (Mean)
Perimeter (Mean)
Area (Mean)
Compactness (Mean)
Concavity (Mean)
Concave points (Mean)
Radius (Standard error)
Perimeter (Standard error)
Area (Standard error)
Radius (Worst)
Perimeter (Worst)
Area (Worst)

Compactness (Worst)
Concavity (Worst)
Concave point (Worst)

4.4 Variance Inflation Factor (VIF)

After selecting features based on correlation and ANOVA, multicollinearity was address by calculating the Variance Inflation Factor for each feature. VIF measures the extent to which the variance of the estimated coefficients my be inflated due to multicollinearity. In this study, features with a VIF value greater than 10 were removed iterarly. The final set of selected features based on the VIF analysis are listed in the table below.

Table 4

Features with VIF < 10

Feature
Radius (Mean)
Compactness (Mean)
Perimeter (Standard error)
Area (Standard error)
Radius (Worst)
Perimeter (Worst)
Area (Worst)
Compactness (Worst)

4.5 Bayesian models

Five Bayesian logistic regression models were created with different priors to assess the impact of prior knowledge on the models performance. The priors ranges from non information to informative with varying levels of certainty. Non-informative priors assume little prior domain expertise, allowing the results to have a stronger influence on the posterior distribution. Informative priors, on the other hand incorporate prior knowledge or domain expertise about the parameter values.

4.6 Model evaluation and comparison

To evaluate and compare the Bayesian models the Gelman diagnostic and effective sample sizes were computed for each model to assess the convergence of the Markov Chain Monte Carlo (MCMC) algorithm. The Gelman diagnostic for all parameters in the model was close to 1, indicating that the chains had converged to the same stationary distribution. The trace plots of the parameters showed adequate mixing fluctuating around a constant with stable variance. The sample sizes were all greater than 200 (Appendix B), indicating that the sample obtained are representative of the posterior.

The Bayesian Information Criterion (BIC) was used to compare the models and identify the model with the best fit. The model with informative priors (3) had the lowest BIX value (191.3452) and was thus selected as the best model. The model suggests that incorporating some prior knowledge with a moderate level of certainty improved the model's performance compared to a model with either non informative or highly informative prior beliefs.

Table 5

Comparison of models using Bayesian Information Criterion

Feature	BIC
Non-Informative prior	191.4740
Informative prior (1)	196.3215
Informative prior (2)	202.7823
Informative prior (3)	191.3542
Informative prior (4)	193.0549

4.7 Posterior distribution and interpretation

Table 5

Confusion matrix for informative prior (3)

	Actual 0	Actual (1)
Predicted (0)	340	22
Predicted (1)	170	190

A confusion matrix was run on the posterior distribution, indicating that the model had an accuracy of 93.15%, sensitivity of 91.79% and specificity of 93.92%, indicating that the model has performed well in classifying benign and malignant cases.

4.8 Credible intervals

Credible intervals were used to identify significant features, features with credible intervals not containing zero were considered significant predictors of breast cancer.

Table 6

Credible intervals for informative priors (3) model

Coefficient	Feature	2.5%	97.5%
β_0	(Intercept)	-1.38	-0.43
β_1	Radius (Mean)	1.68	3.65
β_2	Compactness (Mean)	-2.88	-0.75
β_3	Perimeter Standard Error	1.24	3.21
β_6	Concave Points (Worst)	2.83	5.65

1. *Intercept (β_0)*: The negative intercept suggests a low baseline (log-odds) probability of cancer without considering other features.
2. *Radius Mean (β_1)*: A larger radius (mean) significantly increases the probability of breast cancer, aligning with clinical knowledge.
3. *Compactness Mean (β_2)*: Higher compactness (mean) values decrease the probability of cancer, potentially indicating benign conditions.
4. *Perimeter Standard Error (β_3)*: Greater variability in perimeter measurements (perimeter standard error) increases the probability of breast cancer.
5. *Concave Points Worst (β_6)*: A higher number of concave portions in the worst cases (concave points worst) significantly increases the probability of breast cancer, likely due to the irregular shape of malignant tumours.

4.9 Predictive Probabilities

The predictive probabilities for specific Radius (Mean) values (5 and 25) were calculated to illustrate the impact of this feature on the likelihood of breast cancer. The density plots (Appendix A) visualise the distribution of predicted probabilities, showing a clear separation between low and high radius (Mean) values.

Table 7

Predictive probabilities of breast cancer

Feature	Value	Probability of Breast Cancer
Radius (Mean)	5	0.9360
Radius (Mean)	25	0.9403

4.10 Generalised Additive Model (GAM) comparison

A GAM was fitted using the significant features identified from the Bayesian model to capture potential non-linear relationships between the predictors and the diagnosis variable.

In the context of breast cancer prediction, GAMs allow for smooth, non-linear effects of the predictors, potentially improving prediction accuracy.

Advantage of using GAM for prediction:

1. *Flexibility:* Can model non-linear relationships without specifying the exact form of the relationship.
2. *Interpretability:* The smooth functions used in GAMs are easy to interpret.
3. *Performance:* Can achieve high predictive accuracy, as seen in this study.

Disadvantage of using GAMs for prediction:

1. *Computational cost:* Fitting GAMs can be computationally intensive.
2. *Overfitting:* Without proper regularisation, GAMs can possible overfit the data.

The GAM showed a slightly better performance than the Bayesian model, with an accuracy of 94.90%, sensitivity of 94.63%, and specificity of 95.05%. This suggests that incorporating

non-linear relationships through smoothing functions in the Generalised Additive Model, may help improve the model's predictive power.

Discussion

This study demonstrates the effectiveness of Bayesian logistic regression and GAMs in predicting breast cancer diagnosis using features derived from FNA images. The identified significant predictors included the Radius (Mean), Compactness (Mean), perimeter (Standard error), and Concave Points (Worst), The Bayesian model with informative priors (3) proved to be the best fit, indicating the importance of prior knowledge in modelling the data. The GAM showed slightly better performance than the Bayesian model, suggesting its robustness for predicting breast cancer.

Based on the credible intervals obtained from the Bayesian model with informative prior (3), the most important feature appears to be Concave Points (Worst). Concave points (Worst) are the measures of the severity of concave portions of the cell nucleus boundary.

This feature has the highest absolute coefficient value ($\beta 6$) and its credible interval is [2.83, 5.65], which is furthest from zero. This indicates that a higher number of concave points in the worst-case scenario significantly increases the likelihood of malignancy. This finding aligns with the irregular shape often seen in malignant tumours.

In that respect, a one unit increase in the Concave Points (Worst), results in a 99.60% probability of breast cancer, holding all other features as constant.

Table 8

Bayesian model with informative prior (3) - Probabilities of breast cancer

Feature	Log odds	Probability of Breast Cancer
Radius (Mean) β_1	3.65	$\frac{e^{3.65}}{1+e^{3.65}} \approx 0.974$
Compactness (Mean) β_2	-0.75	$\frac{e^{-0.75}}{1+e^{-0.75}} \approx 0.32$
Perimeter (Standard Error) β_3	3.21	$\frac{e^{3.21}}{1+e^{3.21}} \approx 0.96$
Concave Points (Worst) β_6	5.65	$\frac{e^{5.65}}{1+e^{5.65}} \approx 0.996$

The findings of this study are consistent with previous research that has shown the importance of using features derived from FNA images for the early diagnosis and prediction of breast cancer (Yadav & Agarwal, 2019).

One limitation of this study is the relatively small sample size of 569 instances from a single source, which might not be representative of an overall general population. A larger and recent dataset could provide more robust results and improve on the findings.

References

Breast Cancer Aotearoa Coalition (2024). Diagnostic Tests. Retrieved from https://www.breastcancer.org.nz/aboutBC/detection/diagnostic-tests

Mishra, A. K., Roy, P., Bandyopadhyay, S., & Das, S. K. (2022). A multi-task learning based approach for efficient breast cancer detection and classification. Expert Systems, 39(9), 1-18. https://doi.org/10.1111/exsy.13047

Monirujjaman Khan, M., Islam, S., Sarkar, S., Ayaz, F. I., Ananda, M. K., Tazin, T., Almalki, F. A. (2022). Machine Learning Based Comparative Analysis for Breast Cancer Prediction. Journal of Healthcare Engineering, 1-15. https://doi.org/10.1155/2022/4365855

Nick Street,W., Wolberg, W. H. Mangasarian, O. L. "Nuclear feature extraction for breast tumor diagnosis," Proc. SPIE 1905, Biomedical Image Processing and Biomedical Visualization, (29 July 1993); https://doi.org/10.1117/12.148698

New Zealand Cancer Society (2024). A guide for people with breast cancer. Retrieved from: https://www.cancer.org.nz/cancer/types-of-cancer/breast-cancer/

Subramanian, A. A. V., & Venugopal, J. P. (2023). A deep ensemble network model for classifying and predicting breast cancer. Computational Intelligence, 39(2), 258-282. https://doi.org/10.1111/coin.12563

van de Schoot, R., Kaplan, D., Denissen, J., Asendorpf, J. B., Neyer, F. J., & van Aken, M. A. G. (2014). A Gentle Introduction to Bayesian Analysis: Applications to Developmental Research. *Child Development, 85*(3), 842-860.

World Health Organisation (2024). Breast Cancer. Retrieved from https://www.who.int/news-room/fact-sheets/detail/breast-cancer

Zaylaa, A. J., & Kourtian, S. (2024). Advancing Breast Cancer Diagnosis through Breast Mass Images, Machine Learning, and Regression Models. Sensors (14248220), 24(7), 2312-2328. https://doi.org/10.3390/s24072312

Appendix A

Figure 1 – Correlations of all features

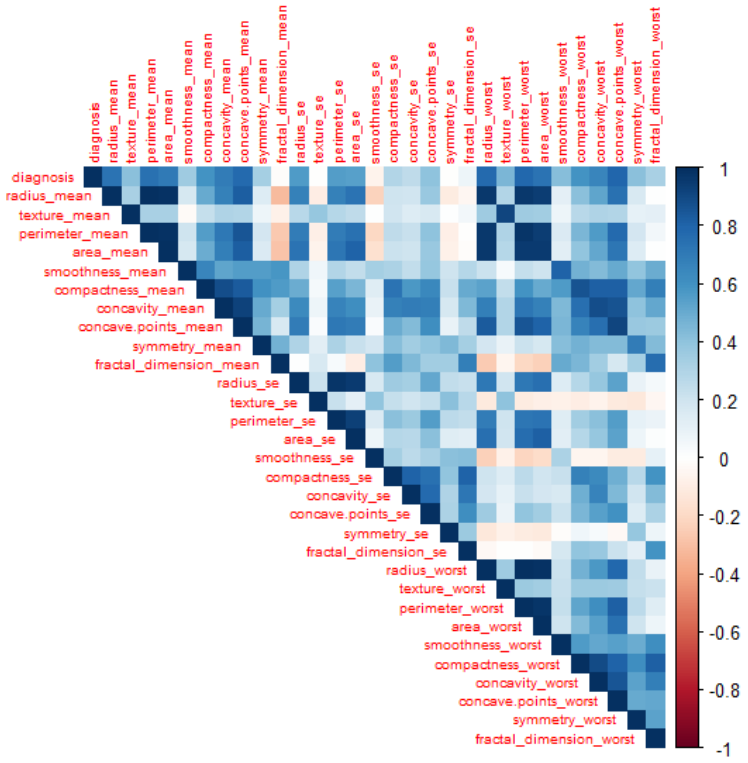


Figure 2 – Histogram of Radius (Mean) = 5

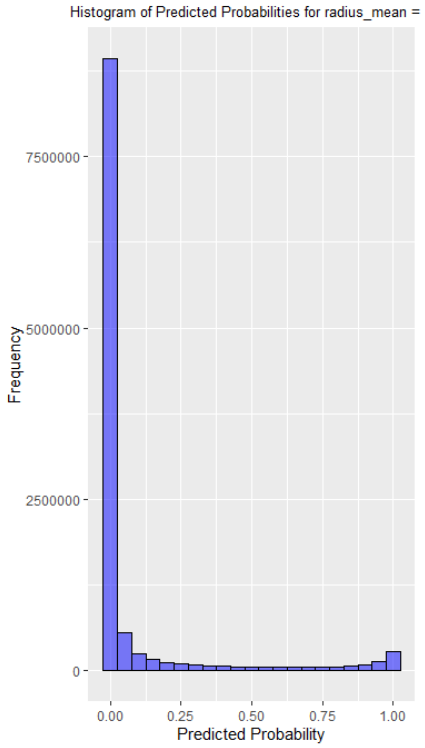


Figure 3– Histogram plot of Radius (Mean) = 25

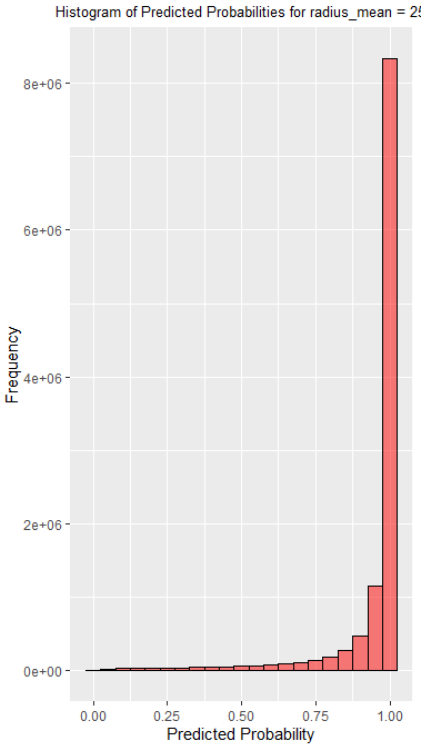


Figure 4 – Confidence intervals – Perimeter (Standard Error)

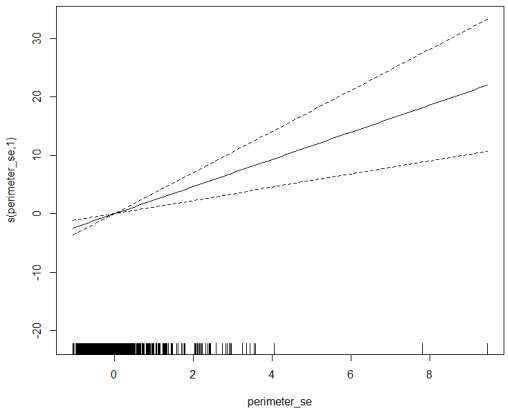


Figure 5 – Confidence intervals – Compactness (Worst)

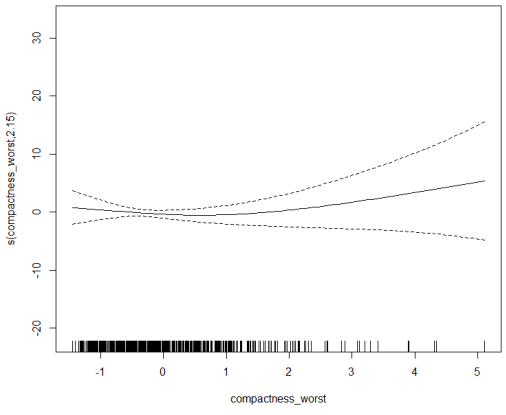


Figure 6 – Confidence intervals – Radius (Mean)

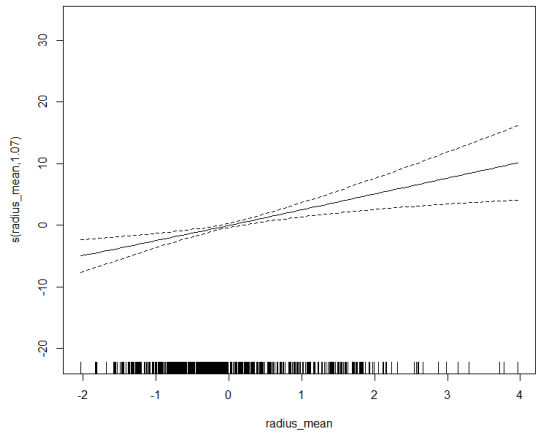


Figure 7 – Confidence intervals – Compactness (Mean)

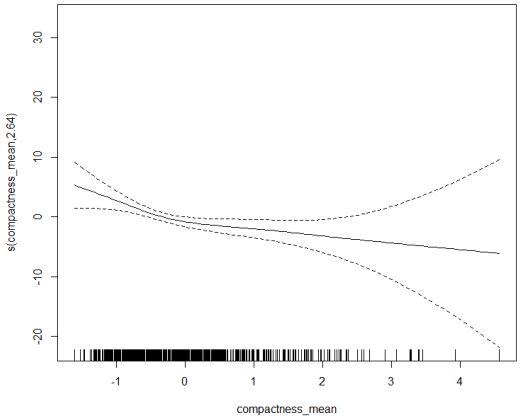


Figure 8 – Confidence intervals – Concavity (Worst)

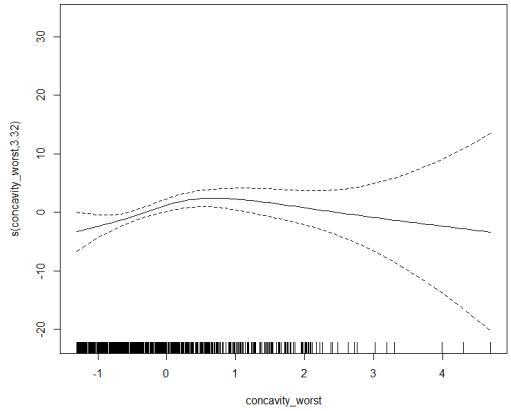
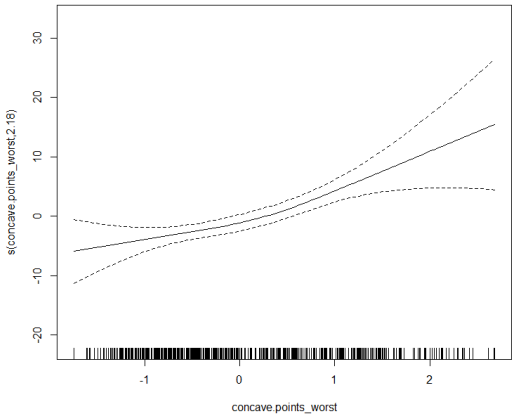
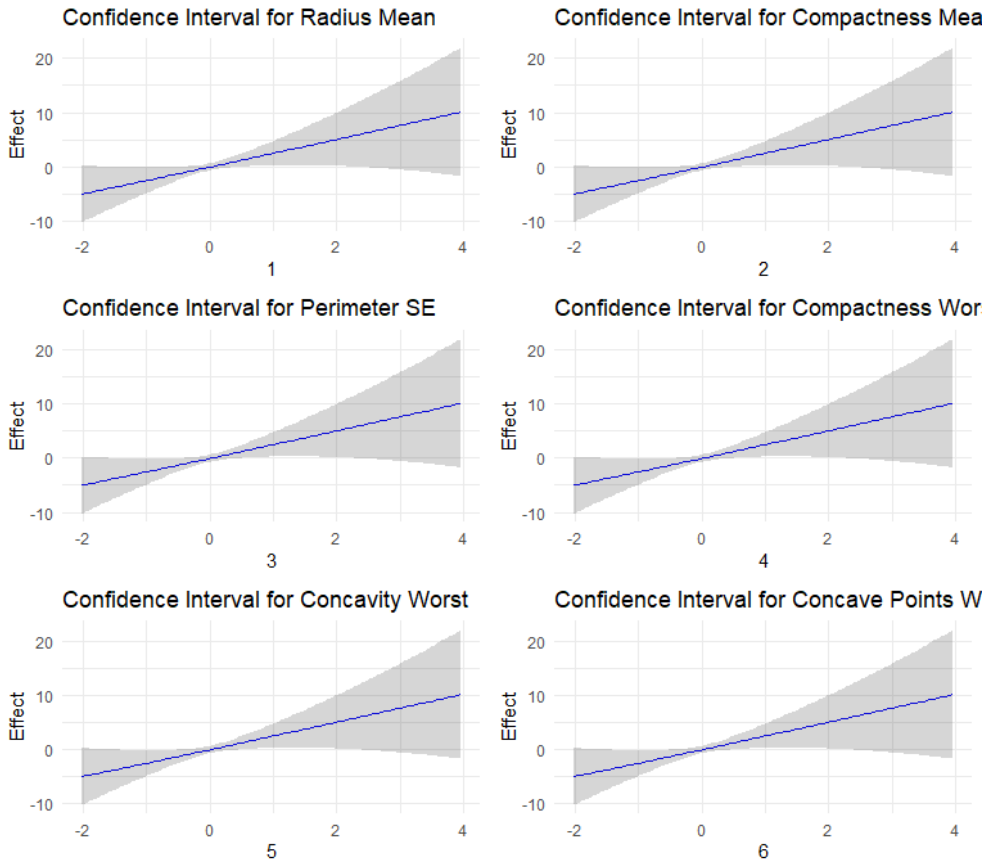


Figure 9 – Confidence intervals – Concave points (Worst)





Summary of Coefficients with 95% HPD Intervals and Sensitivity Analysis

			HPD Interval			Sensitivity Analysis
	Variable	**Coefficient**	**Mean**	**Lower**	**Upper**	**Sensitivity**
mean	Intercept	Intercept	-0.9245995	-1.4054554	**-0.4534119**	-21.597836
b[1]	radius_mean	b[1]	2.6793597	1.7132584	**3.6990844**	43.579767
b[2]	compactness_mean	b[2]	-1.7132360	-2.7679399	**-0.7038549**	-34.725554
b[3]	perimeter_se	b[3]	2.1912948	1.2453518	**3.1882478**	39.946505
b[4]	compactness_worst	b[4]	0.6168453	-0.3508257	**1.6539369**	14.950073
b[5]	concavity_worst	b[5]	0.3042044	-0.5538504	**1.1783761**	7.546999
b[6]	concave.points_worst	b[6]	4.1250484	2.7559730	**5.5419663**	48.409437

Appendix B – R Code and Results

```
> # Load the data
> df <- read.csv("data.csv")
> df <- df %>%
+   select(2:32) %>%
+   mutate(diagnosis = ifelse(diagnosis == "M", 1, 0))
>
> # Correlation heatmap
> correlation_matrix <- cor(df)
> corrplot(correlation_matrix, method = "color", type = "upper", tl.cex = 0.6
)
>
> # Feature selection using correlation
> correlations <- cor(df)
> target_correlation <- abs(correlations[,"diagnosis"])
> correlation_threshold <- 0.5
> correlation_selected_features <- names(target_correlation[target_correlatio
n > correlation_threshold & target_correlation < 1])
> print("Features selected based on correlation:")
[1] "Features selected based on correlation:"
> print(correlation_selected_features)
 [1] "radius_mean"          "perimeter_mean"       "area_mean"            "co
mpactness_mean"     "concavity_mean"
 [6] "concave.points_mean"  "radius_se"            "perimeter_se"         "ar
ea_se"              "radius_worst"
[11] "perimeter_worst"      "area_worst"           "compactness_worst"    "co
ncavity_worst"      "concave.points_worst"
>
> # ANOVA for feature selection
> anova_results <- sapply(df %>% select(-diagnosis), function(x) {
+   summary(aov(x ~ df$diagnosis))[[1]]["Pr(>F)"][1]
+ })
> anova_threshold <- 0.05
> anova_selected_features <- names(anova_results[anova_results < anova_thresh
old])
> print("Features selected based on ANOVA:")
[1] "Features selected based on ANOVA:"
> print(anova_selected_features)
 [1] "radius_mean"             "texture_mean"            "perimeter_mean"
"area_mean"
 [5] "smoothness_mean"         "compactness_mean"        "concavity_mean"
"concave.points_mean"
 [9] "symmetry_mean"           "radius_se"               "perimeter_se"
"area_se"
[13] "compactness_se"          "concavity_se"            "concave.points_se"
"radius_worst"
[17] "texture_worst"           "perimeter_worst"         "area_worst"
"smoothness_worst"
[21] "compactness_worst"       "concavity_worst"         "concave.points_wors
t"     "symmetry_worst"
[25] "fractal_dimension_worst"
>
> # Combine selected features from correlation and ANOVA
> selected_features <- intersect(correlation_selected_features, anova_selecte
d_features)
> print("Features selected after combining correlation and ANOVA:")
[1] "Features selected after combining correlation and ANOVA:"
> print(selected_features)
 [1] "radius_mean"          "perimeter_mean"       "area_mean"            "co
mpactness_mean"     "concavity_mean"
 [6] "concave.points_mean"  "radius_se"            "perimeter_se"         "ar
ea_se"              "radius_worst"
[11] "perimeter_worst"      "area_worst"           "compactness_worst"    "co
ncavity_worst"      "concave.points_worst"
>
```

```
> # Subset the dataframe to keep only the selected features and the target va
riable
> df <- df %>% select(diagnosis, all_of(selected_features))
>
> # Calculate VIF and remove high VIF features
> remove_high_vif <- function(data, threshold = 10) {
+   vif_values <- vif(lm(diagnosis ~ ., data = data))
+   while (max(vif_values) > threshold) {
+     data <- data %>% select(-all_of(names(vif_values)[which.max(vif_values)
]))
+     vif_values <- vif(lm(diagnosis ~ ., data = data))
+   }
+   return(data)
+ }
>
> # Apply the function to remove high VIF features
> df <- remove_high_vif(df)
> print("Features selected after removing high VIF features:")
[1] "Features selected after removing high VIF features:"
> print(colnames(df))
[1] "diagnosis"            "radius_mean"          "compactness_mean"     "per
imeter_se"         "compactness_worst"
[6] "concavity_worst"      "concave.points_worst"
>
> # Standardize the features (excluding the diagnosis variable)
> df_scaled <- df %>%
+   mutate(across(-diagnosis, ~ scale(.) %>% as.vector))
>
> # Verify the binary values for diagnosis
> table(df_scaled$diagnosis)

  0   1
357 212
>
> # Prepare data for JAGS
> jags_data <- list(
+   N = nrow(df_scaled),
+   K = ncol(df_scaled) - 1,
+   X = as.matrix(df_scaled %>% select(-diagnosis)),
+   y = df_scaled$diagnosis
+ )
>
> # Define JAGS models with different priors
> model_strings <- list(
+   "non_informative" = "
+   model {
+     for (i in 1:N) {
+       y[i] ~ dbern(p[i])
+       logit(p[i]) <- b0 + inprod(b[1:K], X[i,])
+     }
+     b0 ~ dnorm(0, 0.01)
+     for (j in 1:K) {
+       b[j] ~ dnorm(0, 0.01)
+     }
+   }",
+   "informative_1" = "
+   model {
+     for (i in 1:N) {
+       y[i] ~ dbern(p[i])
+       logit(p[i]) <- b0 + inprod(b[1:K], X[i,])
+     }
+     b0 ~ dnorm(0, 1)
+     for (j in 1:K) {
+       b[j] ~ dnorm(0, 1)
+     }
+   }",
+   "informative_2" = "
```

```
+   model {
+     for (i in 1:N) {
+       y[i] ~ dbern(p[i])
+       logit(p[i]) <- b0 + inprod(b[1:K], X[i,])
+     }
+     b0 ~ dnorm(0, 2)
+     for (j in 1:K) {
+       b[j] ~ dnorm(0, 2)
+     }
+   }",
+   "informative_3" = "
+   model {
+     for (i in 1:N) {
+       y[i] ~ dbern(p[i])
+       logit(p[i]) <- b0 + inprod(b[1:K], X[i,])
+     }
+     b0 ~ dnorm(0, 0.1)
+     for (j in 1:K) {
+       b[j] ~ dnorm(0, 0.1)
+     }
+   }",
+   "informative_4" = "
+   model {
+     for (i in 1:N) {
+       y[i] ~ dbern(p[i])
+       logit(p[i]) <- b0 + inprod(b[1:K], X[i,])
+     }
+     b0 ~ dnorm(0, 0.5)
+     for (j in 1:K) {
+       b[j] ~ dnorm(0, 0.5)
+     }
+   }"
+ )
>
> # Function to write model string to file and run JAGS model
> run_jags_model <- function(model_string, file_name, jags_data, inits, param
s, n_chains = 4, n_adapt = 1000, n_iter = 5000) {
+   writeLines(model_string, con = file_name)
+   jags_model <- jags.model(file = file_name, data = jags_data, inits = init
s, n.chains = n_chains, n.adapt = n_adapt)
+   update(jags_model, n.iter = n_adapt)
+   samples <- coda.samples(jags_model, variable.names = params, n.iter = n_i
ter)
+   return(samples)
+ }
>
> # Initial values
> inits <- function() {
+   list(b0 = 0, b = rep(0, ncol(df_scaled) - 1))
+ }
>
> # Parameters to monitor
> params <- c("b0", paste0("b[", 1:(ncol(df_scaled) - 1), "]"))
>
> # Run models with different priors
> samples_list <- list()
> for (model_name in names(model_strings)) {
+   samples_list[[model_name]] <- run_jags_model(model_strings[[model_name]],
paste0("logistic_regression_model_", model_name, ".jags"), jags_data, inits,
params)
+ }
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 569
   Unobserved stochastic nodes: 7
```

```
   Total graph size: 6271

Initializing model

  |++++++++++++++++++++++++++++++++++++++++++++++++++| 100%
  |**************************************************| 100%
  |**************************************************| 100%
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 569
   Unobserved stochastic nodes: 7
   Total graph size: 6271

Initializing model

  |++++++++++++++++++++++++++++++++++++++++++++++++++| 100%
  |**************************************************| 100%
  |**************************************************| 100%
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 569
   Unobserved stochastic nodes: 7
   Total graph size: 6271

Initializing model

  |++++++++++++++++++++++++++++++++++++++++++++++++++| 100%
  |**************************************************| 100%
  |**************************************************| 100%
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 569
   Unobserved stochastic nodes: 7
   Total graph size: 6271

Initializing model

  |++++++++++++++++++++++++++++++++++++++++++++++++++| 100%
  |**************************************************| 100%
  |**************************************************| 100%
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 569
   Unobserved stochastic nodes: 7
   Total graph size: 6271

Initializing model

  |++++++++++++++++++++++++++++++++++++++++++++++++++| 100%
  |**************************************************| 100%
  |**************************************************| 100%
> # Define the log_lik function
> log_lik <- function(b0, b, X, y) {
+   eta <- b0 + X %*% b
+   p <- 1 / (1 + exp(-eta))
+   log_lik <- sum(dbinom(y, size = 1, prob = p, log = TRUE))
+   return(log_lik)
+ }
>
```

```
>
>
> # Convergence diagnostics and model comparison
> bic_values <- numeric(length(model_strings))
> names(bic_values) <- names(model_strings)
>
> for (model_name in names(samples_list)) {
+   samples <- samples_list[[model_name]]
+
+   # Convergence diagnostics
+   gelman_diag <- gelman.diag(samples, multivariate = FALSE)
+   print(paste("Gelman diagnostic for", model_name, ":"))
+   print(gelman_diag)
+
+   effective_size <- effectiveSize(samples)
+   print(paste("Effective size for", model_name, ":"))
+   print(effective_size)
+
+   # Log-likelihood calculation
+   posterior_means <- colMeans(as.matrix(samples))
+   b0_mean <- posterior_means["b0"]
+   b_mean <- posterior_means[grep("^b\\[", names(posterior_means))]
+   log_lik_value <- log_lik(b0_mean, b_mean, jags_data$X, jags_data$y)
+
+   # BIC calculation
+   bic_values[model_name] <- -2 * log_lik_value + (ncol(jags_data$X) + 1) *
log(nrow(jags_data$X))
+ }
[1] "Gelman diagnostic for non_informative :"
Potential scale reduction factors:

     Point est. Upper C.I.
b0            1       1.00
b[1]          1       1.00
b[2]          1       1.00
b[3]          1       1.00
b[4]          1       1.01
b[5]          1       1.01
b[6]          1       1.00

[1] "Effective size for non_informative :"
      b0     b[1]     b[2]     b[3]     b[4]     b[5]     b[6]
7351.697 7512.286 2020.423 4105.302 1675.344 2309.766 2338.834
[1] "Gelman diagnostic for informative_1 :"
Potential scale reduction factors:

     Point est. Upper C.I.
b0            1       1.00
b[1]          1       1.00
b[2]          1       1.00
b[3]          1       1.00
b[4]          1       1.01
b[5]          1       1.00
b[6]          1       1.00

[1] "Effective size for informative_1 :"
      b0     b[1]     b[2]     b[3]     b[4]     b[5]     b[6]
9464.039 9045.151 3258.560 7661.968 2408.848 3095.875 3675.646
[1] "Gelman diagnostic for informative_2 :"
Potential scale reduction factors:

     Point est. Upper C.I.
b0            1          1
b[1]          1          1
b[2]          1          1
b[3]          1          1
b[4]          1          1
```

```
b[5]          1          1
b[6]          1          1

[1] "Effective size for informative_2 :"
      b0     b[1]     b[2]     b[3]     b[4]     b[5]     b[6]
8957.282 9445.830 3783.209 7635.490 2679.829 3578.826 4388.795
[1] "Gelman diagnostic for informative_3 :"
Potential scale reduction factors:

     Point est. Upper C.I.
b0            1       1.00
b[1]          1       1.00
b[2]          1       1.00
b[3]          1       1.00
b[4]          1       1.01
b[5]          1       1.01
b[6]          1       1.00

[1] "Effective size for informative_3 :"
      b0     b[1]     b[2]     b[3]     b[4]     b[5]     b[6]
8079.815 8582.263 2220.534 5260.204 1681.562 2495.655 2569.921
[1] "Gelman diagnostic for informative_4 :"
Potential scale reduction factors:

     Point est. Upper C.I.
b0            1       1.00
b[1]          1       1.00
b[2]          1       1.01
b[3]          1       1.00
b[4]          1       1.00
b[5]          1       1.00
b[6]          1       1.00

[1] "Effective size for informative_4 :"
      b0     b[1]     b[2]     b[3]     b[4]     b[5]     b[6]
8299.660 8152.139 2561.666 6605.232 2097.174 2792.475 3339.881
>
> # Model Comparison Table using BIC
> comparison_table <- data.frame(
+   Model = names(bic_values),
+   BIC = bic_values
+ )
>
> print("Model Comparison Table:")
[1] "Model Comparison Table:"
> print(comparison_table)
                          Model      BIC
non_informative non_informative 191.5237
informative_1     informative_1 196.3196
informative_2     informative_2 202.8841
informative_3     informative_3 191.3673
informative_4     informative_4 193.0938
>
> # Select the best model
> best_model_name <- names(bic_values)[which.min(bic_values)]
> best_samples <- samples_list[[best_model_name]]
> cat("The best model is", best_model_name, "with the lowest BIC.\n")
The best model is informative_3 with the lowest BIC.
>
> # Extract b_samples correctly
> b_samples <- as.matrix(best_samples)[, grep("^b\\[", colnames(as.matrix(bes
t_samples)))]
>
> # Posterior predictive checks
> b0_samples <- as.matrix(best_samples)[, "b0"]
>
>
```

```
>
>
> # Variable names for coefficients
> variable_names <- c("Intercept", "radius_mean", "compactness_mean", "perime
ter_se", "compactness_worst", "concavity_worst", "concave.points_worst")
>
> # Calculate mean, HPD intervals, and sensitivity analysis
> calc_summary_hpd_sensitivity <- function(samples) {
+   mean_val <- mean(samples)
+   hpd_interval <- HPDinterval(mcmc(samples), prob = 0.95)
+   # Sensitivity analysis in terms of probability change
+   P0 <- 1 / (1 + exp(0))
+   P1 <- 1 / (1 + exp(-mean_val))
+   sensitivity <- (P1 - P0) * 100  # Sensitivity analysis in percentage
+   return(c(mean = mean_val, lower = hpd_interval[1], upper = hpd_interval[2
], sensitivity = sensitivity))
+ }
>
> # Apply to each coefficient
> b0_summary <- calc_summary_hpd_sensitivity(b0_samples)
> b_summaries <- apply(b_samples, 2, calc_summary_hpd_sensitivity)
>
> # Combine summaries into a data frame
> coefficients <- data.frame(
+   Variable = variable_names,
+   Coefficient = c("Intercept", colnames(b_samples)),
+   Mean = c(b0_summary["mean"], b_summaries["mean", ]),
+   Lower = c(b0_summary["lower"], b_summaries["lower", ]),
+   Upper = c(b0_summary["upper"], b_summaries["upper", ]),
+   Sensitivity = c(b0_summary["sensitivity"], b_summaries["sensitivity", ])
+ )
>
> # Create a publication-ready table
> kable(coefficients, format = "html", caption = "Summary of Coefficients wit
h 95% HPD Intervals and Sensitivity Analysis") %>%
+   kable_styling(bootstrap_options = c("striped", "hover", "condensed"), ful
l_width = F) %>%
+   add_header_above(c(" " = 1, " " = 1, "HPD Interval" = 3, " " = 1, "Sensit
ivity Analysis" = 1)) %>%
+   column_spec(6, bold = TRUE)
>
> # Predict on the entire dataset
> X <- as.matrix(df_scaled %>% select(-diagnosis))
>
> # Matrix multiplication results
> test_product <- b_samples %*% t(X)
>
> # Predict probabilities using explicit logistic function
> logit_preds <- sweep(test_product, 1, b0_samples, '+')
> pred_probs_new <- exp(logit_preds) / (1 + exp(logit_preds))
> pred_probs <- apply(pred_probs_new, 2, mean)
>
> # Threshold to convert probabilities to binary predictions
> threshold <- 0.5
> predictions <- ifelse(pred_probs > threshold, 1, 0)
>
> # Confusion matrix
> confusion_matrix <- table(Predicted = predictions, Actual = df_scaled$diagn
osis)
> print("Confusion Matrix for Bayesian Model:")
[1] "Confusion Matrix for Bayesian Model:"
> print(confusion_matrix)
         Actual
Predicted   0   1
        0 340  22
        1  17 190
>
```

```
> # Calculate credible intervals using HPD method
> credible_intervals <- HPDinterval(best_samples, prob = 0.95)
> print("Credible Intervals for Bayesian Model:")
[1] "Credible Intervals for Bayesian Model:"
> print(credible_intervals)
[[1]]
          lower      upper
b0   -1.3839015 -0.4228450
b[1]  1.7475185  3.6981746
b[2] -2.7158005 -0.6577017
b[3]  1.1670444  3.1440979
b[4] -0.4458890  1.6245336
b[5] -0.5932655  1.1162447
b[6]  2.8211713  5.5556769
attr(,"Probability")
[1] 0.95

[[2]]
          lower      upper
b0   -1.3711123 -0.4438710
b[1]  1.7021686  3.7265830
b[2] -2.7163940 -0.6875672
b[3]  1.2959626  3.2404940
b[4] -0.3563055  1.6579676
b[5] -0.5861728  1.1419497
b[6]  2.8561730  5.6237621
attr(,"Probability")
[1] 0.95

[[3]]
          lower      upper
b0   -1.3991655 -0.4501109
b[1]  1.7264313  3.6997536
b[2] -2.7892060 -0.7537138
b[3]  1.2488172  3.1791740
b[4] -0.2803080  1.6344138
b[5] -0.5850213  1.1337814
b[6]  2.7204252  5.5019284
attr(,"Probability")
[1] 0.95

[[4]]
          lower      upper
b0   -1.4062771 -0.4417347
b[1]  1.6953298  3.6667416
b[2] -2.8007227 -0.6848866
b[3]  1.2343077  3.1201496
b[4] -0.3755522  1.6306726
b[5] -0.5569060  1.1937033
b[6]  2.7120656  5.5346146
attr(,"Probability")
[1] 0.95

>
> # Extract significant coefficients
> credible_intervals_matrix <- as.data.frame(credible_intervals)
> significant_coefficients <- which(credible_intervals_matrix$lower * credibl
e_intervals_matrix$upper > 0)
>
> coefficient_names <- c("b0", paste0("b[", 1:(ncol(df_scaled) - 1), "]"))
> significant_coefficients_names <- coefficient_names[significant_coefficient
s]
>
> feature_names <- c("(Intercept)", colnames(df_scaled)[-1])
> significant_features <- feature_names[significant_coefficients]
>
> significant_table <- data.frame(
```

```
+   Coefficient = significant_coefficients_names,
+   Feature = significant_features
+ )
>
> print("Significant Coefficients and Related Features:")
[1] "Significant Coefficients and Related Features:"
> print(significant_table)
  Coefficient              Feature
1          b0          (Intercept)
2        b[1]          radius_mean
3        b[2]     compactness_mean
4        b[3]         perimeter_se
5        b[6] concave.points_worst
>
> # Function to predict probabilities and calculate HPD intervals for varying
radius_mean
> predict_probability <- function(radius_mean_value) {
+   # Create new data with the specified radius_mean
+   new_data <- df_scaled
+   new_data$radius_mean <- (radius_mean_value - mean(df$radius_mean)) / sd(d
f$radius_mean)  # Standardize the radius_mean value
+
+   # Predict probabilities
+   X_new <- as.matrix(new_data %>% select(-diagnosis))
+   test_product_new <- b_samples %*% t(X_new)
+   logit_preds_new <- sweep(test_product_new, 1, b0_samples, '+')
+   pred_probs_new <- exp(logit_preds_new) / (1 + exp(logit_preds_new))
+
+   list(pred_probs_new = as.vector(pred_probs_new))
+ }
>
> # Predict for radius_mean = 5 and radius_mean = 25
> pred_probs_5 <- predict_probability(5)
> pred_probs_25 <- predict_probability(25)
>
> cat("Probability of having breast cancer with radius_mean=5:", mean(pred_pr
obs_5$pred_probs_new), "\n")
Probability of having breast cancer with radius_mean=5: 0.09284126
> cat("Probability of having breast cancer with radius_mean=25:", mean(pred_p
robs_25$pred_probs_new), "\n")
Probability of having breast cancer with radius_mean=25: 0.9408215
>
> # Histograms for predicted probabilities with radius_mean = 5
> df_pred_5 <- data.frame(pred_probs = pred_probs_5$pred_probs_new)
> hist_5 <- ggplot(df_pred_5, aes(x = pred_probs)) +
+   geom_histogram(binwidth = 0.05, fill = "blue", alpha = 0.5, color = "blac
k") +
+   ggtitle("Histogram of Predicted Probabilities for radius_mean = 5") +
+   xlab("Predicted Probability") +
+   ylab("Frequency") +
+   theme(plot.title = element_text(size = 10, hjust = 0.5))
>
> # Histograms for predicted probabilities with radius_mean = 25
> df_pred_25 <- data.frame(pred_probs = pred_probs_25$pred_probs_new)
> hist_25 <- ggplot(df_pred_25, aes(x = pred_probs)) +
+   geom_histogram(binwidth = 0.05, fill = "red", alpha = 0.5, color = "black
") +
+   ggtitle("Histogram of Predicted Probabilities for radius_mean = 25") +
+   xlab("Predicted Probability") +
+   ylab("Frequency") +
+   theme(plot.title = element_text(size = 10, hjust = 0.5))
>
> # Arrange both histograms in one plot
> grid.arrange(hist_5, hist_25, ncol = 2)
> # Fit a GAM model on reduced model (significant features)
> gam_reduced_model <- gam(diagnosis ~ s(radius_mean) + s(compactness_mean) +
```

```
+                             s(perimeter_se) + s(compactness_worst) + s(conca
vity_worst) +
+                             s(concave.points_worst),
+                           data = df_scaled, family = binomial)
>
> # Predict using GAM reduced model
> gam_reduced_predictions <- predict(gam_reduced_model, type = "response")
> gam_reduced_predicted_classes <- ifelse(gam_reduced_predictions > threshold
, 1, 0)
>
> # Create confidence interval plots for GAM model
> plot_smooth <- function(model, term, title) {
+   plot_data <- plot(model, se = TRUE, select = term, plot = FALSE)
+   term_plot <- data.frame(x = plot_data[[1]]$x,
+                           fit = plot_data[[1]]$fit,
+                           se = plot_data[[1]]$se)
+   ggplot(term_plot, aes(x = x, y = fit)) +
+     geom_line(color = "blue") +
+     geom_ribbon(aes(ymin = fit - 1.96 * se, ymax = fit + 1.96 * se), alpha
= 0.2) +
+     ggtitle(title) +
+     xlab(term) +
+     ylab("Effect") +
+     theme_minimal()
+ }
>
> # Plot confidence intervals for each significant feature
> ci_radius_mean <- plot_smooth(gam_reduced_model, 1, "Confidence Interval fo
r Radius Mean")
> ci_compactness_mean <- plot_smooth(gam_reduced_model, 2, "Confidence Interv
al for Compactness Mean")
> ci_perimeter_se <- plot_smooth(gam_reduced_model, 3, "Confidence Interval f
or Perimeter SE")
> ci_compactness_worst <- plot_smooth(gam_reduced_model, 4, "Confidence Inter
val for Compactness Worst")
> ci_concavity_worst <- plot_smooth(gam_reduced_model, 5, "Confidence Interva
l for Concavity Worst")
> ci_concave_points_worst <- plot_smooth(gam_reduced_model, 6, "Confidence In
terval for Concave Points Worst")
>
> # Display all plots (in a publication, you may want to arrange these neatly
)
> library(gridExtra)
> grid.arrange(ci_radius_mean, ci_compactness_mean, ci_perimeter_se,
+              ci_compactness_worst, ci_concavity_worst, ci_concave_points_wo
rst,
+              ncol = 2)
>
> # Function to calculate accuracy, sensitivity, and specificity
> calculate_metrics <- function(predictions, actual) {
+   confusion <- table(Predicted = predictions, Actual = actual)
+   accuracy <- sum(diag(confusion)) / sum(confusion)
+   sensitivity <- confusion[2, 2] / (confusion[2, 2] + confusion[2, 1])
+   specificity <- confusion[1, 1] / (confusion[1, 1] + confusion[1, 2])
+   list(accuracy = accuracy, sensitivity = sensitivity, specificity = specif
icity)
+ }
>
> # Calculate metrics for Bayesian model
> bayesian_metrics <- calculate_metrics(predictions, df_scaled$diagnosis)
> print("Bayesian Model Metrics:")
[1] "Bayesian Model Metrics:"
> print(bayesian_metrics)
$accuracy
[1] 0.9314587

$sensitivity
```

```
[1] 0.9178744

$specificity
[1] 0.9392265

>
> # Calculate metrics for GAM reduced model
> gam_reduced_metrics <- calculate_metrics(gam_reduced_predicted_classes, df_
scaled$diagnosis)
> print("GAM Reduced Model Metrics:")
[1] "GAM Reduced Model Metrics:"
> print(gam_reduced_metrics)
$accuracy
[1] 0.9490334

$sensitivity
[1] 0.9463415

$specificity
[1] 0.9505495

>
> # Compare Bayesian and GAM models
> comparison_table <- data.frame(
+   Model = c("Bayesian", "GAM Reduced"),
+   Accuracy = c(bayesian_metrics$accuracy, gam_reduced_metrics$accuracy),
+   Sensitivity = c(bayesian_metrics$sensitivity, gam_reduced_metrics$sensiti
vity),
+   Specificity = c(bayesian_metrics$specificity, gam_reduced_metrics$specifi
city)
+ )
>
> print("Comparison of Bayesian and GAM Models:")
[1] "Comparison of Bayesian and GAM Models:"
> print(comparison_table)
        Model  Accuracy Sensitivity Specificity
1    Bayesian 0.9314587   0.9178744   0.9392265
2 GAM Reduced 0.9490334   0.9463415   0.9505495
```