RMIT UNIVERSITY

MATH2269

APPLIED BAYESIAN STATISTICS ASSIGNMENT 2

Bayesian Analysis of Property Prices in Melbourne

Introduction

In this study, we aim to model the Sale prices of properties across Melbourne. We attempt to model it using the other features such as Area, bedrooms, bathrooms, cark parks and Property Type. We use the expert knowledge given at hand and attempt to incorporate prior distributions into a Bayesian linear regression model. MCMC Sampling is carried out using JAGS and the results are assessed.

Executive Summary

This study lays focus on the Bayesian analysis of the sale price of properties in Melbourne using a Bayesian linear regression model. The key objective is to estimate the sale price of properties dependent on major factors such as area, number of bedrooms and bathrooms, car parks, and property type. MCMC sampling was conducted via JAGS. A Student t-distribution was used because of potential outliers.

The exploratory data analysis showed skewness in sale prices, indicating the highly volatile nature of the real estate market in Melbourne. Prior knowledge was integrated into the Bayesian model wherever possible. We used vague priors when expert knowledge about the priors was poor and strong priors when expert knowledge was strong. MCMC Diagnostics were assessed to check the convergence and posteriors are displayed for each parameter. Finally, predictions for 5 properties were performed.

Data Description

The dataset consists of property sale prices in Melbourne, represented as below.

Sale Price: Sale price of the property in AUD.
Area: Land size in m^2 of the property.
Bedrooms: The number of bedrooms.
Bathrooms: The number of bathrooms.
CarParks: The number of car parks.
PropertyType: Categorical, with 0 indicating a House and 1 indicating a Unit.

Exploratory Analysis

Histogram of Sale Price:

Arka Dasgupta
S3980896

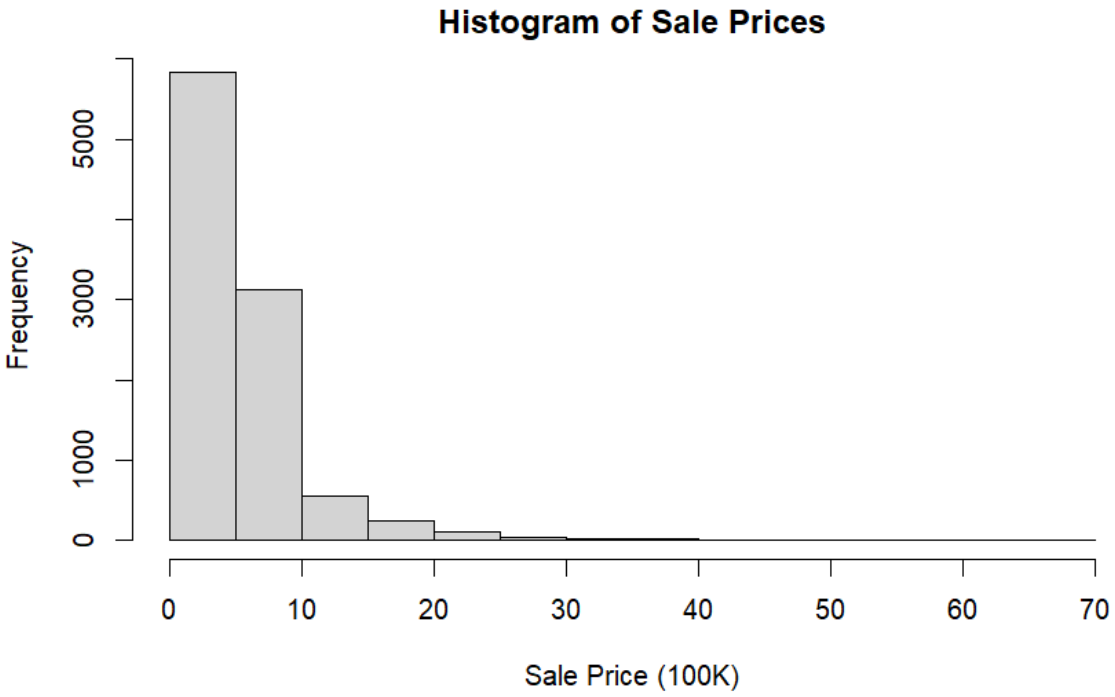


Figure 1 - Histogram of Sale Prices

It's clear from Figure 1 that the distribution of Sale Prices is heavily skewed to the right. This makes some sense as real estate has a lot of high priced properties.

Density Plot:

In addition to the histogram, a density plot of sale prices was generated. The plot below (Figure 2) shows a highly skewed distribution with majority of sales occurring below 10,000,000 AUD, further highlighting the need for robust modelling, especially with a potential for outliers.

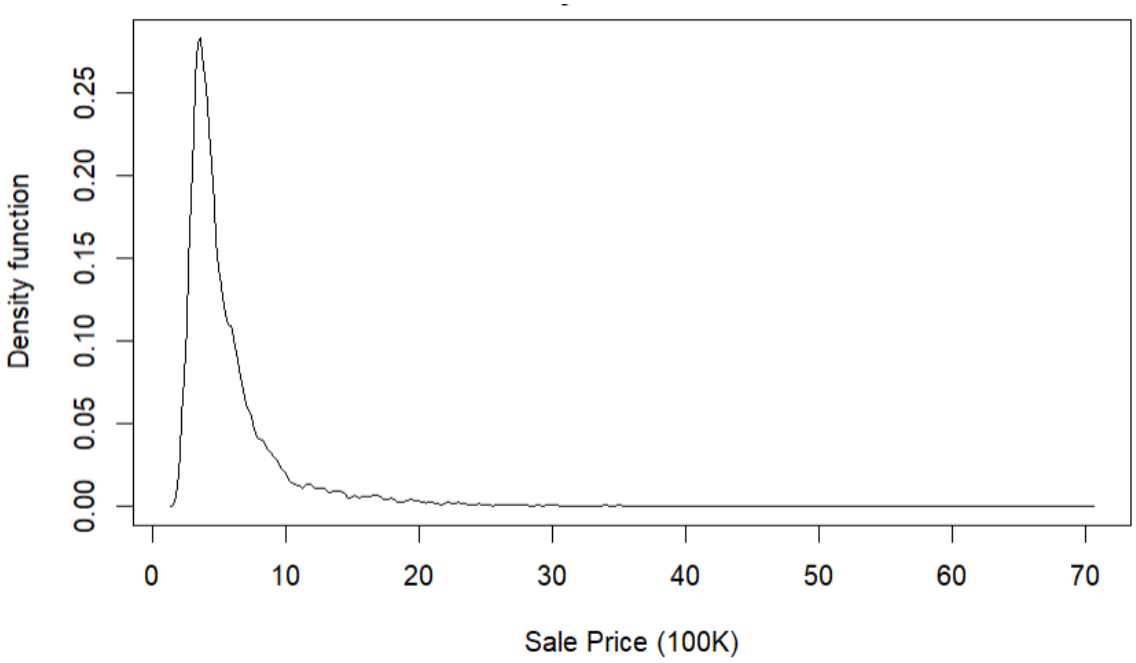


Figure 2 - Desnity Plot for Sale Price

Scatter Plots:

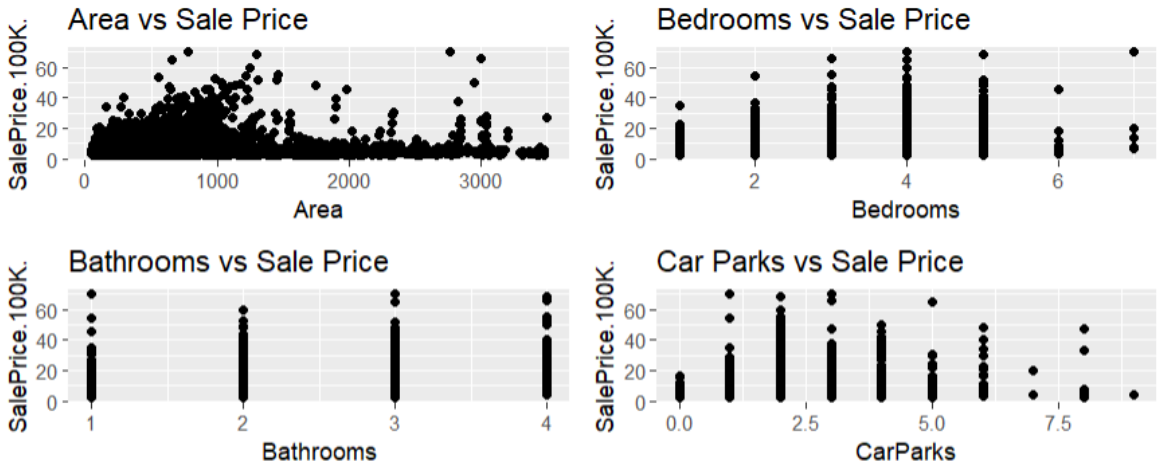


Figure 3 - Scatterplots for each predictor against Saleprice

From the scatter plots in Figure 3 above, it is clear that all the predictors have pretty much a positive relationship with Sale Price. For instance, larger properties tend to be more expensive. There is a lot of variation too as factors like location or quality can affect the price significantly. The sale price seems to cluster more consistently for properties with 2–4 bedrooms, but properties with more bedrooms (e.g., 6) show more outliers with extreme prices. Bathrooms have the weakest relationship compared to the others, showing least contribution to increase in sale prices.Cark parks exhibit a poisitve relationship as well, albeit with some variation in each group. There could also be other factors such as open/closed cark park and location that could cause these variations.

RMIT UNIVERSITY

JAGS Model Implementation:

The model we are working with could be expressed as:

$Y = \beta0 + \beta1 * Area + \beta2 * Bedrooms + \beta3 * Bathrooms + \beta4 * CarParks + \beta5 * PropertyType + \varepsilon$

where:

- **Y** is theSale Price.
- **β0** is the intercept.
- **β1**, **β2**, **β3**, **β4**, and **β5** represent the regression coefficients for **Area**, **Bedrooms**, **Bathrooms**, **CarParks**, and **Property Type**.
- **ε** is the error term.

We will be using a Student t distribution, from our aforementioned inference from the histogram (Figure 1). This will also help in dealing with outliers. Since it's evident that there are potentially some extreme values in sale prices(since it is real estate), it is a good fit for us.

The Expression for likelihood of sale prices is: **$Yi \sim t(\mu i, \sigma, v)$**

where:

- **Yi** - Sale price
- **μi -** predicted value of the Sale price
- **σ** i- Scale of the t distribution
- **v** - degrees of freedom in the t distribution.

For regression coefficients **βi,** we use a normal distribution as our prior distribution:

- **β0** (Intercept): We use a vague prior, possibly having a large variance.
- **β1** (Area): As per expert knowledge provided to us, we are aware that for every increase by 1 meter squared of land, the sale price increases by 90 AUD.
- **β2** (Bedrooms): In this case, expert knowledge says that for every additional bedroom sale price goes up by 100,000 AUD. Since this information is weak, we use a weak prior with large variance.
- **β3** (Bathrooms): We have no expert knowledge about bathrooms.
- **β4** (CarParks): A strong prior is used to reflect this knowledge. We have strong expert knoeldge that with each additional carpark the sale price goes up by 120,000 AUD.
- **β5** (Property Type): We have strong prior knowledge that Sale price goes down by 150,000 AUD if the property is an unit instead of a house.

σ is set to an unform prior to represent the uncertainty.

For the degrees of freedom (v), we will be using an exponential prior.

Arka Dasgupta
S3980896

[**v ~ Exp(λ)**]

JAGS Model:

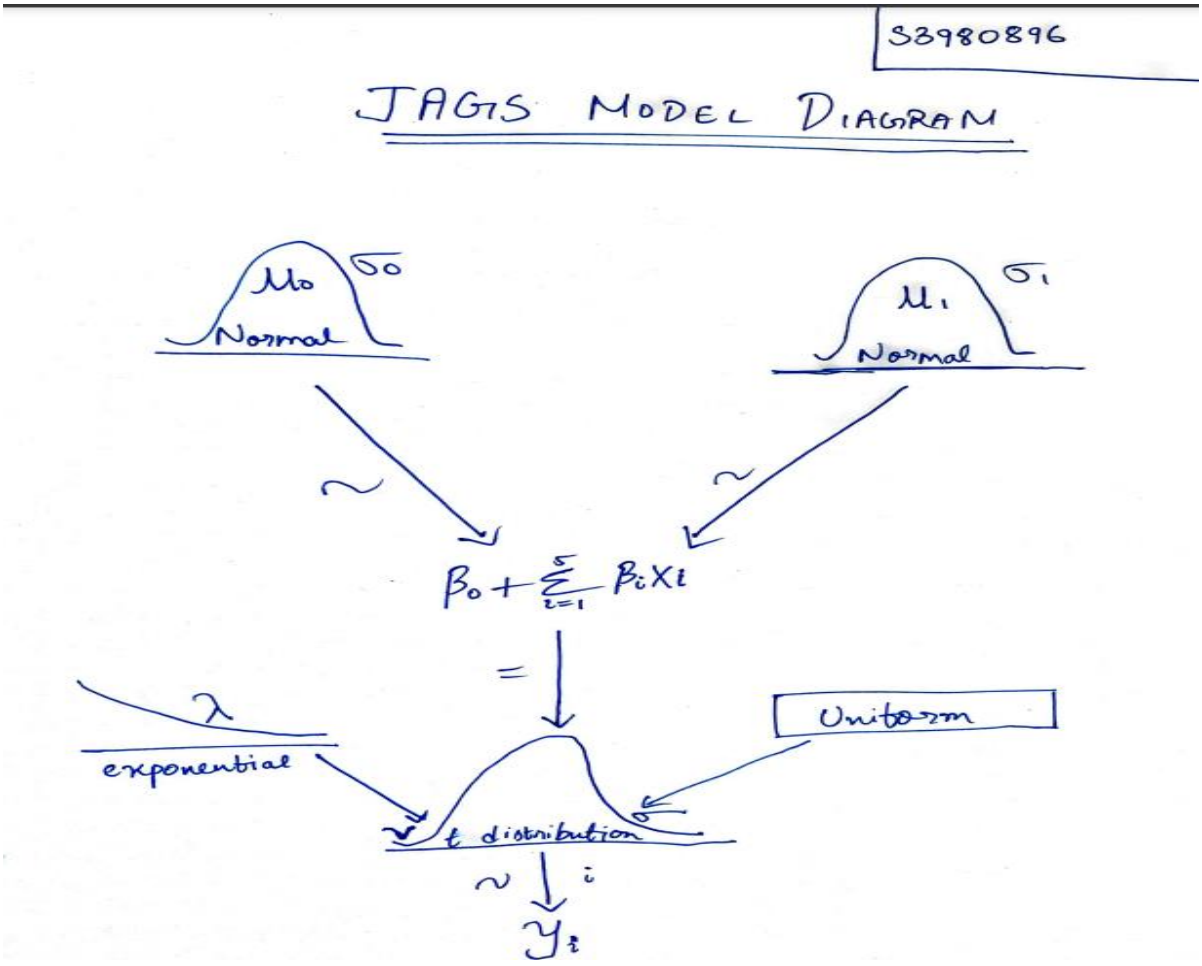


Figure 4 - JAGS Model Diagram

MCMC Settings:

Chains: 3 chains were run in parallel. Running multiple chains could help us verify the convergence of the MCMC process.

numSavedSteps: 5000 steps, to balance computational time while trying to preserve accuracy in predictions.

Thinsteps: 5. By setting this, we are ensuring every 5th sample in the iteration is saved. This can help in avoiding correlations between consecutives samples.

saveName: We save this parameter to a filename “PropertyPrices”. This is because my computational time for sample generation was extremely high, and to eliminate the risk of losing all the progress of the generated chains, I attempted to save the samples in an MCMC

Object that could be loaded and used at a later point without having to re run the entire sampling process.

MCMC Diagnostics:

From the 95% HDIS as per Figure 5 below, it is safe to say that we have achieved decent convergence.

Let us assess the findings:

1. Trace Plots:

From the generated traceplots for betas, sigma and nu, it is clear that there is an overlap of all the chains and no chains are isolated. This is a clear indication that our sampling was good and the chains are mixed well.

2. Effective Sample Size (ESS) and MCSE:

- The ESS values we get are all well above 1000, beta1 having an ESS of 10,192 and beta4 having n ESS of 5,991.
- The MCSE values are all low across all the parameters. For instance, beta0 has an MCSE of 0.00238 while beta1 has an MCSE value of 3.23 e-07. These values are low enough for use to display meaningful posterior distributions for our next step of analysis.

3. Autocorrelation Plots:

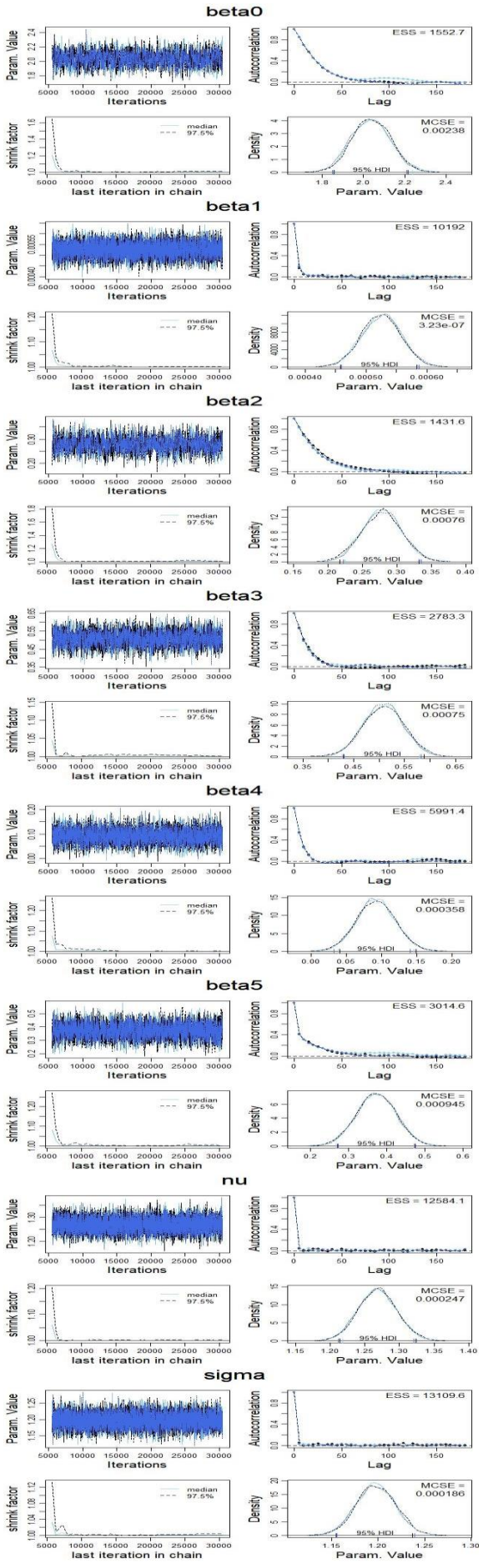
We can observe a significant drop at the second lag for the autocorrelation plots. This is an indicator that all the samples are pretty much independent after a small lag. This is again a good sign as it shows that our sampling process was effective and meaningful.

4. Density Plots:

The density plots for beta2 and beta5 are smooth and exhibit good convergence. However, beta0 shows a broad posterior compared to the others(along with having a large MCSE). This could mean that the density plot for beta0 is suggesting high uncertainty, which is obvious from the variability seen in the data.

In conclusion, the MCMC diagnostic plots show that the chains have converged well, with no significant issues detected in the sampling process. The large ESS values and the small MCSE values indicate good posterior prediction. Even though beta0 shows the largest MCSE and a broadest density plot, it is is consistent with our expectations given the variability in the intercept.

RMIT UNIVERSITY



Arka Dasgupta
S3980896

Figure 4 - MCMC diagnostics for Generated samples

Posterior Distributions:

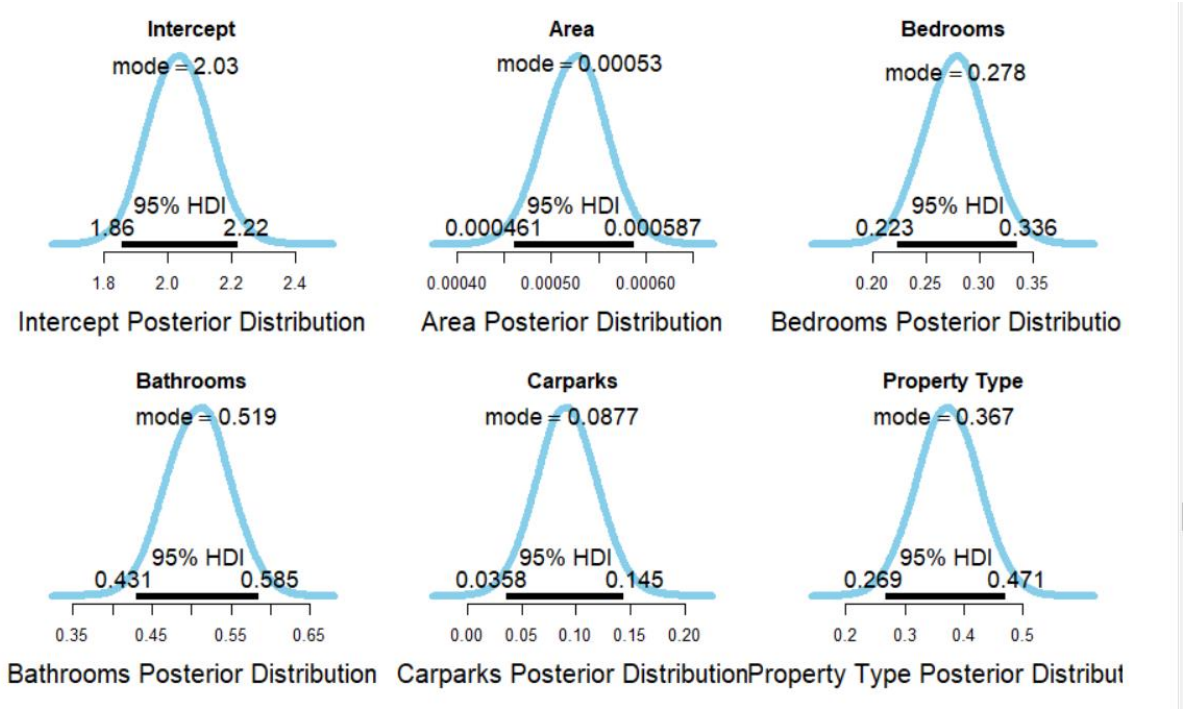


Figure 5 - Posterior distributions

Intercept: The mode is 2.03 and the 95% HDI ranges from 1.85 to 2.22.

Area: The mode is 0.00053 and the 95% HDI ranges from 0.000461 to 0.000587. This aligns with the strong expert knowledge that every extra square meter increases property price by 90 AUD.

Bedrooms: The mode is 0.278 while the HDI range lies between 0.223 to0.336. A slightly wider HDI indicates the uncertainty in this estimate compared to other parameters having strong priors.

Bathrooms: The mode is 0.519 and the 95% HDI range lies between 0.431 and 0.585. This indicates that number of bathrooms is an important parameter in sale price, given it will increase by 51,900 AUD per bathroom.

Carparks: The mode is 0.0877 and the 95% HDI ranges from 0.0358 to 0.145. There is some variability around this estimate compared to our expert knowledge. Also, the interval is relatively wider.

Property Type: The observed mode is 0.367 and the 95% HDI ranges from 0.269 to 0.471. This indicates that units sell for 36,700 AUD lesser than houses.

Overall, the generated plots suggest that the MCMC process has converged well for all parameters, with no significant inconsistencies. The 95% HDI intervals provide reasonable bounds for each parameter, supporting the expert knowledge while allowing the data to refine these estimates.

Arka Dasgupta
S3980896

Predictions

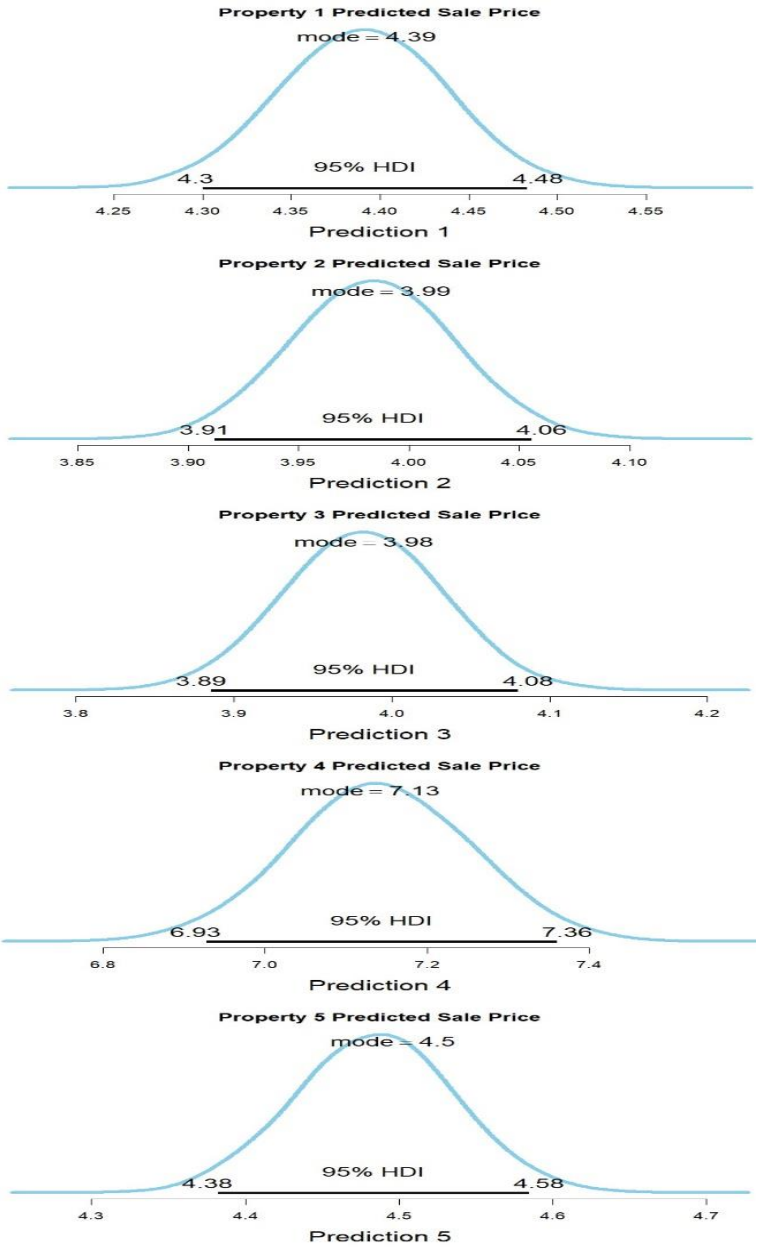


Figure 6 - Property Price Predictions

From Figure 6, we can conclude our predictions for the Sale Prices for 5 properties.

Prediction 1: 439,000 AUD

Prediction 2: 399,000 AUD

Prediction 3: 398,000 AUD

Arka Dasgupta
S3980896

Prediction 4: 713,000 AUD

Prediction 5: 450,000 AUD

The predicted sale prices align quite well with the actual sale prices. The t distribution is quite well suited for a scenario like this where we potentially might have outliers or heavy tails, as observed in Figure 1 during our exploratory analysis earlier.

Property 4 has a higher mode with a wider 95% HDI, which makes sense given that larger properties can have more variability. On the other hand, we observe narrower HDIs for the properties like Property 1, 2, and 3. It clearly shows that when there are more normal sales, heavy tails of the t-distribution make barely any difference and we can be more certain about our prediction. We have thus ensured that our model is efficient enough to handle the variability in the data by using t distribution, especially for properties having extreme or unusual sale prices.

Conclusion

Our Bayesian linear regression model has successfully captured the relationship between sale prices and property features in Melbourne. Prior expert knowledge provided to us has been leveraged wherever possible. The MCMC diagnostic checks confirm the robustness of the model, given the low MCSE scores and satisfactory convergence. Using a t distribution also immensely helped in making the model effective against potential outlies.

The predictions carried out align quite well with the actual sale prices, which is a further indicator that despite having a skewed distribution and potential heavy outliers a good model can make meaningful predictions.

References

Dr. Haydar Demirhan, RMIT University – MATH2269 Applied Bayesian Statistics Lecture Notes and R Codes, 2024

Bob Oppie, GitHub Repository – Kruschke Doing Bayesian Data Analysis, 2024. Retrieved from (https://github.com/boboppie/kruschke-doing_bayesian_data_analysis/blob/master/2e/DBDA2E-utilities.R)

Arka Dasgupta
S3980896

RMIT UNIVERSITY

Appendix

Souce Code:

```
library(ggplot2)
library(ggpubr)
library(rjags)
library(runjags)

# Set working directory and load data
setwd("C:/Users/ARKA/Sem 4/ABS")
data = read.csv("C:/Users/ARKA/Sem 4/ABS/Assignment2PropertyPrices.csv")

# Check the structure and first few rows of the dataset
head(data)
summary(data)
colnames(data)
# Exploratory Data Analysis
# Defining predictor and response variable names
yName = "SalePrice.100K."
xName = c("Area", "Bedrooms", "Bathrooms", "CarParks", "PropertyType")

head(data)
class(data$SalePrice.100K.)
# Histogram of SalePrice.100K.
hist(data$SalePrice.100K., main="Histogram of Sale Prices", xlab="Sale Price (100K)")

# Density plot of SalePrice.100K.
plot(kde(data$SalePrice.100K.), main="Density of Sale Prices", xlab="Sale Price (100K)")
# Scatter plots for each predictor against SalePrice.100K.
p1 <- ggplot(data, aes(x=Area, y=SalePrice.100K.)) + geom_point() + ggtitle("Area vs Sale Price")
```

Arka Dasgupta
S3980896

```
p2 <- ggplot(data, aes(x=Bedrooms, y=SalePrice.100K.)) + geom_point() +
ggtitle("Bedrooms vs Sale Price")

p3 <- ggplot(data, aes(x=Bathrooms, y=SalePrice.100K.)) + geom_point() +
ggtitle("Bathrooms vs Sale Price")

p4 <- ggplot(data, aes(x=CarParks, y=SalePrice.100K.)) + geom_point() + ggtitle("Car Parks
vs Sale Price")

p5 <- ggplot(data, aes(x=PropertyType, y=SalePrice.100K.)) + geom_point() +
ggtitle("Property Type vs Sale Price")


# Combine scatter plots into one figure

figure <- ggarrange(p1, p2, p3, p4, p5, nrow=3, ncol=2)

print(figure)

# Summary statistics of the dataset

summary(data)

# JAGS model

modelString = "

  model {

    for (i in 1:Ntotal) {

      SalePrice[i] ~ dt(mu[i], tau, nu)

      mu[i] <- beta0 + beta1 * Area[i] + beta2 * Bedrooms[i] +

               beta3 * Bathrooms[i] + beta4 * CarParks[i] +

               beta5 * PropertyType[i]

    }


    # Priors for coefficients

    beta0 ~ dnorm(0, 1.0E-5)    # Vague prior for intercept

    beta1 ~ dnorm(90, 1/(10^2)) # Strong prior for Area

    beta2 ~ dnorm(100000, 1/(50000^2)) # Weak prior for Bedrooms

    beta3 ~ dnorm(0, 1/(50000^2)) # Vague prior for Bathrooms

    beta4 ~ dnorm(120000, 1/(5000^2)) # Strong prior for CarParks

    beta5 ~ dnorm(-150000, 1/(10000^2)) # Strong prior for PropertyType


    # Priors for error terms and degrees of freedom
```

```
  tau <- pow(sigma, -2)
  sigma ~ dunif(0, 100000)   # Uniform prior for scale parameter
  nu ~ dexp(1)               # Exponential prior for degrees of freedom (robustness)
 }
"
writeLines(modelString, con="PropertyPricesModel.txt")
# Function to generate MCMC samples using JAGS
library(runjags)
library(coda)

genMCMC = function(data, numSavedSteps=10000, thinSteps=1, nChains=3,
saveName="PropertyPrices") {

 dataList = list(
  SalePrice = data$SalePrice.100K.,  # Update to use correct column name
  Area = data$Area,
  Bedrooms = data$Bedrooms,
  Bathrooms = data$Bathrooms,
  CarParks = data$CarParks,
  PropertyType = data$PropertyType,
  Ntotal = nrow(data)
 )

 parameters = c("beta0", "beta1", "beta2", "beta3", "beta4", "beta5", "sigma", "nu")

 runJagsOut <- run.jags(
  method="parallel",
  model="PropertyPricesModel.txt",
  monitor=parameters,
  data=dataList,
  n.chains=nChains,
  adapt=500, burnin=5000,
```

```
    sample=numSavedSteps, thin=thinSteps,
    summarise=FALSE, plots=FALSE
  )

  codaSamples = as.mcmc.list(runJagsOut)

  if (!is.null(saveName)) {
    save(codaSamples, file=paste0(saveName, "_MCMC.Rdata"))
  }

  return(codaSamples)
}

colnames(data)
# Generate MCMC samples
mcmcCoda = genMCMC(data=data, numSavedSteps=5000, thinSteps=5, nChains=3,
saveName="PropertyPrices")
# Loading MCMC results
load("PropertyPrices_MCMC.Rdata")
# For diagMCMC function
source("C:/Users/ARKA/Sem 4/ABS/DBDA2E-utilities.R")
parameterNames <- varnames(codaSamples)
for (parName in parameterNames) {
  diagMCMC(
    codaObject = codaSamples,
    parName = parName,
    saveName = NULL,
    saveType = NULL
  )
}
plotMCMC <- function(mcmcCoda, saveName = NULL, saveType = NULL) {
  parameterLabels <- c(
```

```
    "Intercept",     # beta0
    "Area",          # beta1
    "Bedrooms",      # beta2
    "Bathrooms",     # beta3
    "Carparks",      # beta4
    "Property Type"  # beta5
  )

  # Extracting parameter names from MCMC samples
  parameterNames <- varnames(mcmcCoda)
  numParams <- length(parameterNames)
  par(mfrow = c(2, 3))

  for (i in 1:numParams) {
    parName <- parameterNames[i]
    parLabel <- parameterLabels[i]

    # Displaying posterior distribution for each parameter
    plotPost(
      paramSampleVec = mcmcCoda[, parName],
      main = paste(parLabel),
      xlab = paste0(parLabel, " Posterior Distribution"),
      showCurve = TRUE
    )
  }

  par(mfrow = c(1, 1))

  if (!is.null(saveName) && !is.null(saveType)) {
    saveGraph(file = saveName, type = saveType)
  }
}
```

Arka Dasgupta
S3980896

```
plotMCMC(
  mcmcCoda = codaSamples,
  saveName = NULL,
  saveType = NULL
)
# Diagnostic Plots for each parameter
fileNameRoot <- "ABS2"
graphFileType <- "jpeg"
parameterNames <- varnames(codaSamples)

for (parName in parameterNames) {
  diagMCMC(
    codaObject = codaSamples,
    parName = parName,
    saveName = fileNameRoot,
    saveType = graphFileType
  )
}

hdi <- function(sampleVec, credMass = 0.95) {
  sortedPts <- sort(sampleVec)
  ciIdxInc <- floor(credMass * length(sortedPts))
  nCIs <- length(sortedPts) - ciIdxInc
  ciWidth <- rep(0, nCIs)

  for (i in 1:nCIs) {
    ciWidth[i] <- sortedPts[i + ciIdxInc] - sortedPts[i]
  }

  HDImin <- sortedPts[which.min(ciWidth)]
  HDImax <- sortedPts[which.min(ciWidth) + ciIdxInc]
```

```
  return(c(HDImin, HDImax))
}

# Converting MCMC samples to matrix
pred_matrix <- as.matrix(codaSamples)

# Extracting posterior samples of the parameters
beta0 <- pred_matrix[, "beta0"]
beta1 <- pred_matrix[, "beta1"]
beta2 <- pred_matrix[, "beta2"]
beta3 <- pred_matrix[, "beta3"]
beta4 <- pred_matrix[, "beta4"]
beta5 <- pred_matrix[, "beta5"]

# Defining the new properties to predict
xPred <- array(NA, dim = c(5, 5))
xPred[1, ] <- c(600, 2, 2, 1, 1)
xPred[2, ] <- c(800, 3, 1, 2, 0)
xPred[3, ] <- c(1500, 2, 1, 1, 0)
xPred[4, ] <- c(2500, 5, 4, 4, 0)
xPred[5, ] <- c(250, 3, 2, 1, 1)



pred1 <- beta0 + beta1 * xPred[1, 1] + beta2 * xPred[1, 2] + beta3 * xPred[1, 3] + beta4 *
xPred[1, 4] + beta5 * xPred[1, 5]
pred2 <- beta0 + beta1 * xPred[2, 1] + beta2 * xPred[2, 2] + beta3 * xPred[2, 3] + beta4 *
xPred[2, 4] + beta5 * xPred[2, 5]
pred3 <- beta0 + beta1 * xPred[3, 1] + beta2 * xPred[3, 2] + beta3 * xPred[3, 3] + beta4 *
xPred[3, 4] + beta5 * xPred[3, 5]
pred4 <- beta0 + beta1 * xPred[4, 1] + beta2 * xPred[4, 2] + beta3 * xPred[4, 3] + beta4 *
xPred[4, 4] + beta5 * xPred[4, 5]
pred5 <- beta0 + beta1 * xPred[5, 1] + beta2 * xPred[5, 2] + beta3 * xPred[5, 3] + beta4 *
xPred[5, 4] + beta5 * xPred[5, 5]
```

Arka Dasgupta
S3980896

```
# Calculating 95% HDI for each prediction
hdi_pred1 <- hdi(pred1)
hdi_pred2 <- hdi(pred2)
hdi_pred3 <- hdi(pred3)
hdi_pred4 <- hdi(pred4)
hdi_pred5 <- hdi(pred5)

# Summarizing Bayesian estimates (median) and HDI for each property
summary_preds <- data.frame(
  Property = c(1, 2, 3, 4, 5),
  Estimate = c(median(pred1), median(pred2), median(pred3), median(pred4),
median(pred5)),
  HDI_Lower = c(hdi_pred1[1], hdi_pred2[1], hdi_pred3[1], hdi_pred4[1], hdi_pred5[1]),
  HDI_Upper = c(hdi_pred1[2], hdi_pred2[2], hdi_pred3[2], hdi_pred4[2], hdi_pred5[2])
)

print(summary_preds)

# Plots
plotPost(pred1, showCurve = TRUE, xlab = "Prediction 1", main = "Property 1 Predicted
Sale Price")
plotPost(pred2, showCurve = TRUE, xlab = "Prediction 2", main = "Property 2 Predicted
Sale Price")
plotPost(pred3, showCurve = TRUE, xlab = "Prediction 3", main = "Property 3 Predicted
Sale Price")
plotPost(pred4, showCurve = TRUE, xlab = "Prediction 4", main = "Property 4 Predicted
Sale Price")
plotPost(pred5, showCurve = TRUE, xlab = "Prediction 5", main = "Property 5 Predicted
Sale Price")
```

Arka Dasgupta
S3980896

RMIT UNIVERSITY

Arka Dasgupta
S3980896