Media marketing Analysis

by Sandeep Kumar Bullagondla

2023-05-10

In this task we are given a dataset contains sample of weekly media marketing data(spanning a total of 208 weeks), recording revenues generated from various spending on three media. Other variables such as competitors and newsletter subscription are also included.
Data variables are as follows:
1) X - weeknumber
2) DATE - date of week
3) revenue - Revenue(ROI)
4) media1_S - Spend on Media 1
5) media2_S - Spend on Media 2
6) media3_S - Spend on Media 3
7) competitor_sales - Competitor sales
8) newsletter – Number of newsletter subscription

Part 1: Classical Marketing data modelling

1a: modelling You are to use the data collected between 2012/8/6 and 2016/5/30 (inclusive) to train your model.

- Build a statistical regression model to determine the effectiveness of marketing strategy in generating revenues. You must carefully justify the choice of your model and the variables used.
- Interpret the results obtained from your model.
- Discuss the limitations of the model used, in the context of media marketing.

Ans) Here I want to built Generalized Liner Model with Gaussian distribution and an identity link function based on below factors:
1) Continuous Response Variable: Response variable in our data is **revenue** which is continuous variable. GLM with gaussian distribution is suitable for modelling continuous outcome.
2)Linear Relationship: GLMs assume a linear relationship between independent and response variable. And if there is linear relationship between independent and response variables, then GLM effectively captures this relationship.
3) Independence of Observation: GLMs assume that the observations are independent of each other. As long as the data points represent Independent instances of Predictors and target variable, the assumption of independence is generally satisfied.
4) Homoscedasticity: The GLM assumes constant variance of errors across different levels of predictors. This implies that the spread of residuals is consistent across the range of predicted values.
From the ggpairs plot (refer Appenix), I have noticed there is kinda linear relationship with some variables and strict positive linear relationship with competitor sales, so I thought fitting GLM model would be really provided the factors mentioned above. The choice of variables is depending on assumption that they impact revenue generation.**media1_S**, **media2_S** and **media3_S** represents the amount of money spent on three different types of media advertising so all these 3 would definitely impact revenue. **competitor_sales** indicate the sales of competitors, which in turn would impact the revenue. **newsletter** is regarded as marketing activity with no media spend, however this is marketing activity and might impact in revenue.

So we consider the above mentioned variables for modelling. From the initial data analysis also it was proved that all the variables have siginificant correlation.
Now we will fit GLM model with revenue as response variable and media1_S, media2_S, media3_S, competitor_sales and newsletter as independent variables for dates from 2012/08/06 to 2016/05/30.

$$\eta_i = g(\mu_i) = \beta_0 + \beta_1 * media1S + \beta_2 * media2S + \beta_3 * media3S + \beta_4 * competitorsales + \beta_5 * newsletter$$

we are using identity link so $g(\mu_i) = \mu_i$, Now final equation will be.

$$\mu_i = \beta_0 + \beta_1 * media1S + \beta_2 * media2S + \beta_3 * media3S + \beta_4 * competitorsales + \beta_5 * newsletter$$

Where μ_i is revenue, $\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5$ are coefficients respectively.

```
library(ggplot2)
library(knitr)
# Read the data from a CSV file
data <- read.csv("weekly_media_sample.csv")

#start and end dates of training data
start_date <- as.Date("2012-08-06")
end_date <- as.Date("2016-05-30")
train_data <- data[data$DATE >= start_date & data$DATE <= end_date, ]
# Fit the Generalised linear regression model
model <- glm(revenue ~ media1_S + media2_S + media3_S + competitor_sales + newsletter,
             data = train_data, family = gaussian(link = "identity"))


# Get the coefficients and their p-values
coefficients <- coef(model)
p_values <- summary(model)$coefficients[, 4]
```

```
# Print the summary of the model
summary(model)
```

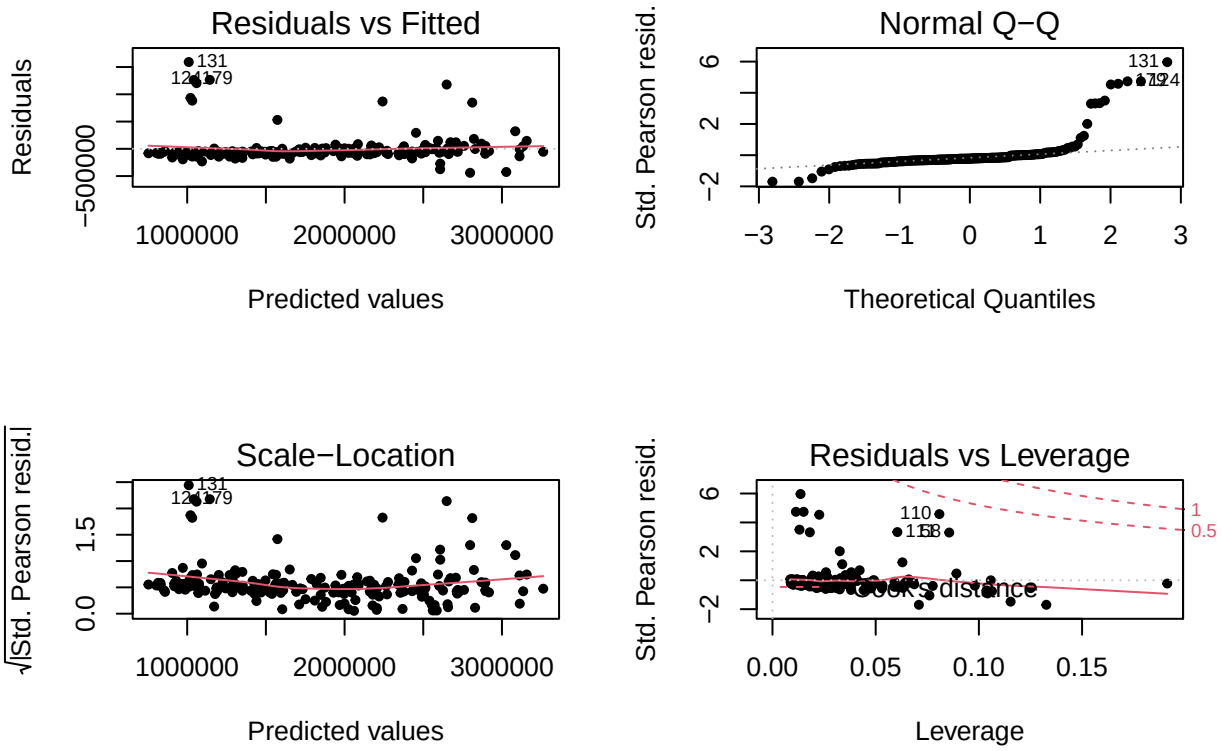
```
##
## Call:
## glm(formula = revenue ~ media1_S + media2_S + media3_S + competitor_sales +
##     newsletter, family = gaussian(link = "identity"), data = train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -439836   -86941   -58602    -2879  1591662
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      9.959e+04  5.427e+04   1.835   0.0680 .
## media1_S         5.041e-01  9.279e-02   5.433 1.65e-07 ***
## media2_S         9.641e-01  4.097e-01   2.353   0.0196 *
## media3_S         3.407e-01  2.231e-01   1.527   0.1283
## competitor_sales 2.885e-01  1.111e-02  25.959  < 2e-16 ***
## newsletter       9.218e-01  1.164e+00   0.792   0.4295
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for gaussian family taken to be 72183887279)
##
##     Null deviance: 1.0234e+14  on 199  degrees of freedom
## Residual deviance: 1.4004e+13  on 194  degrees of freedom
## AIC: 5576
##
## Number of Fisher Scoring iterations: 2
```

- The intercept β_0 is estimated to be 9.959e+04. However, the p-value (0.0680) suggests that the intercept may not be statistically significant at conventional significance levels (e.g., $\alpha = 0.05$).

- The coefficients for media1_S, media2_S, media3_S, competitor_sales, and newsletter are all estimated to be positive. This indicates that an increase in these variables is associated with an increase in revenue.

- Among the predictors, media1_S, media2_S, and competitor_sales have statistically significant coefficients at significance levels ($p < 0.05$), suggesting they have a significant impact on revenue. The coefficients for media3_S and newsletter are not statistically significant ($p > 0.05$), indicating they may not have a significant effect on revenue.
- AIC: The Akaike Information Criterion (AIC) is a measure of the model's goodness-of-fit, considering both the model's complexity and its fit to the data. In this case, the AIC is 5576.



- The first (Top left) plot which gives me the deviance residuals against the predicted values. If the model is good fit, the points should be centered at zero and present a random scatter. From that plot it is clearly understood that model is good model. If the model is not good, then there would be pattern in the line and we should re-model by adding or deleting some covariates.

- The second plot (Top right) is a QQ plot which tests if the distribution of the residuals is normal. If they are then they should lie on the indicated line. We only need to worry if there is systematic deviation from the line, this means that there is a problem in the model. The QQ plot here shows the model we have just fit are deviating quite far from the model so we have trouble estimating high values.
- The third plot (Bottom left) is the scale-location plot and tells you essentially the same thing as the first plot.

- The fourth plot (Bottom right) is the residuals vs. the leverage. Points with high leverage are extreme with respect to the remainder of the data/covariates. From this we do not see any danger in data/data points.

Limitation of model
- In this model we used limited set of independent variables as predictors. However, there may be other important which affect revenue generation in media marketing, such as brand reputation, market conditions, economic factors etc., With these variables omitted model might predict incomplete results. - The model assumes a linear relationship between the independent variables and revenue. However, in media marketing, the relationship between marketing efforts and revenue may not be strictly linear. There might be diminishing returns or interaction effects that are not captured by the model.

1b) Prediction

Use the data between the test period ,2016/6/6 and 2016/7/25 (inclusive), to assess your model.

- Use the model built in 1a to predict the marketing revenues for the test period. Plot your answers on a graph, including 95% intervals.

Sol

Here we have taken data from 2016/06/06 to 2016/07/25. These are from 201th to 208th week. We now predict the values using predict() function and sending media1_S, media2_S, media3_S, competitor_sales, newsletter values of these weeks. Below table shows tha actual values and predicted values of the test data.

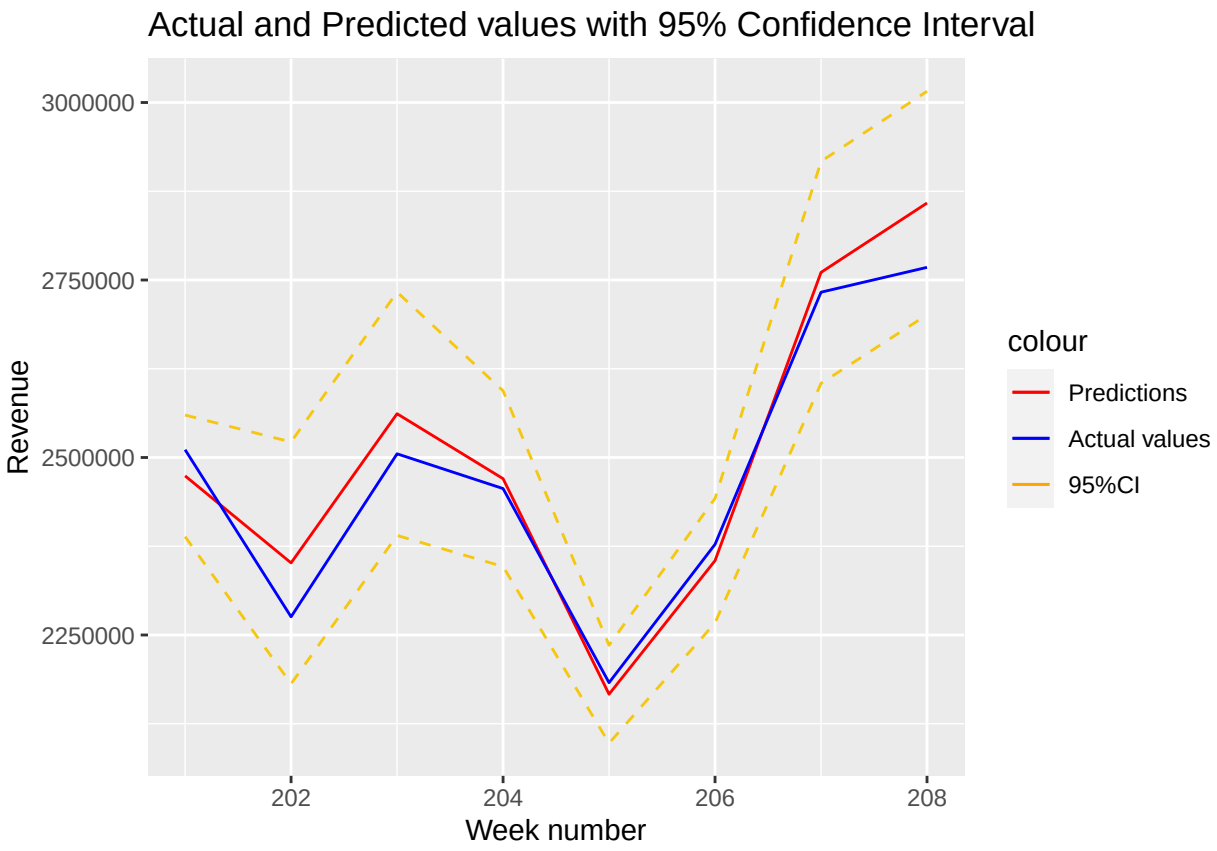
```
#start and end dates for prediction
start_date <- as.Date("2016-06-06")
end_date <- as.Date("2016-07-25")
#test dataset with revenues values
test_data <- data[data$DATE >= start_date & data$DATE <= end_date, ]

#test data with only independent variables
test <- data.frame(media1_S = test_data$media1_S,
                   media2_S = test_data$media2_S,
                   media3_S = test_data$media3_S,
                   competitor_sales= test_data$competitor_sales,
                   newsletter = test_data$newsletter)
```

```
#predictions for GLM model
prediction <- predict(model, newdata = test, type = 'response', se.fit=TRUE)
#95% CI for GLM model
lower <- prediction$fit - 1.96*prediction$se.fit
upper <- prediction$fit + 1.96*prediction$se.fit
```

Table 1: Actual and predcited values of Test data

Date	Actual	Prediction
2016-06-06	2510952	2474107
2016-06-13	2275620	2351558
2016-06-20	2505162	2561615
2016-06-27	2456240	2470159
2016-07-04	2182825	2166562
2016-07-11	2377707	2354747
2016-07-18	2732825	2760714
2016-07-25	2767788	2858430

Below plot is weeknumber vs Revenue with actual, predicted and 95% CI values.



From the above table and table and plot it can be noticed that predicted values are not really far away from actual values and all the predicted and actual values lie in between 95%CI which indicates the model abd predictions are good.

- Assess the predictive performance of your model quantitatively.
  We shall do this by using two metrics

1)RMSE: Root Mean Squared error is the Square root of average squared difference between the predicted and actual values. This metric provides more interpretable measure.

```r
#root mean squred error
rmse<-(sqrt(mean(test_data$revenue - prediction$fit)^2))
rmse
```

```
## [1] 23596.6
```

2)R-squared: It represents the proportion of the variance in the dependent variable that is predictable from the independent variables. A higher R-squared indicates a better fit.

```r
#r-squared vales
total <- sum((test_data$revenue-mean(test_data$revenue))^2)
residual <- sum((prediction$fit - test_data$revenue)^2)
r_squared<- 1 - residual/total
r_squared
```

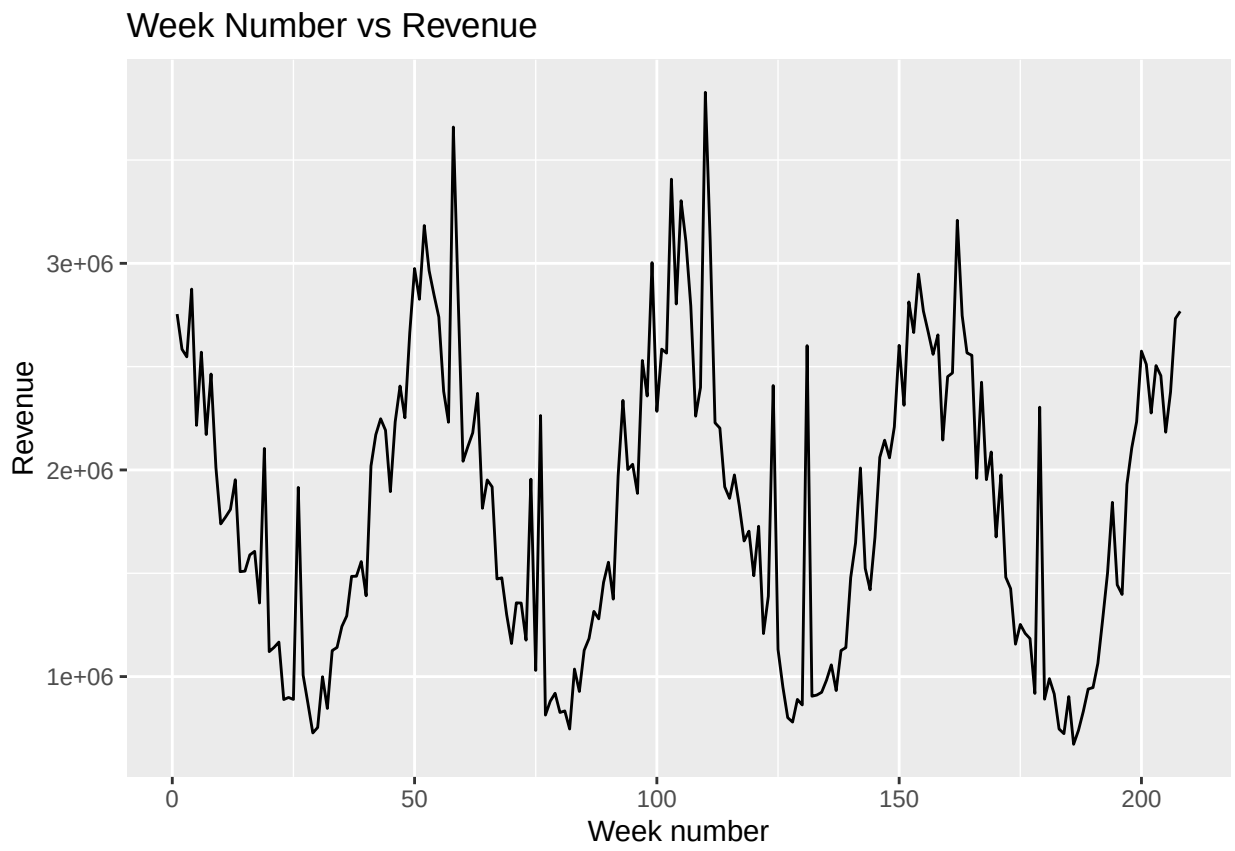
```
## [1] 0.9298715
```

From the above two metrics we can Notice that $RMSE = 23596.6$ which is really good considering the scale of values of actual values. And $R^2 = 0.9298$ which implies model explains data well and good fit.

1c) Temporal Effect

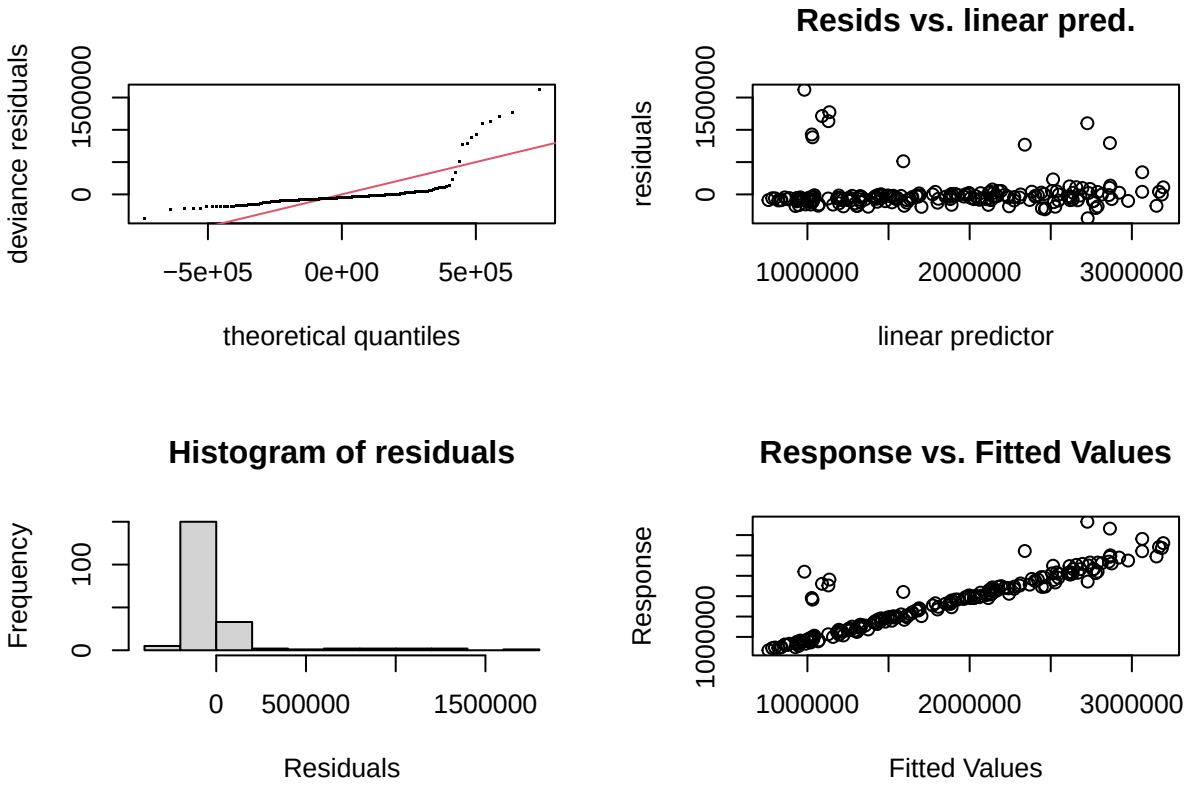
As colleague argued that it is important to consider temporal effects in media marketing, I would support his statement because it is important to consider the seasonality in many industries, media marketing in one among them because this has to be adjusted according to seasonal fluctuations.

Media marketing can be influenced by long-term trends and cycles in the market. Analysing temporal effects will help in identifying and understanding these trends.



From the above plot we can notice there is seasonality and trends in the revenue over time.
Now we shall build GAM model for train_data and predict for test data as above.

```
library(mgcv)
# Fit a GAM time-series model
gam_model <- gam(revenue ~ s(X, k = 50, bs="cs") + s(media1_S) + s(media2_S)+
                 s(media3_S) + s(competitor_sales) + s(newsletter),
                 data = train_data)

# Print the summary of the model
gam.check(gam_model)
```

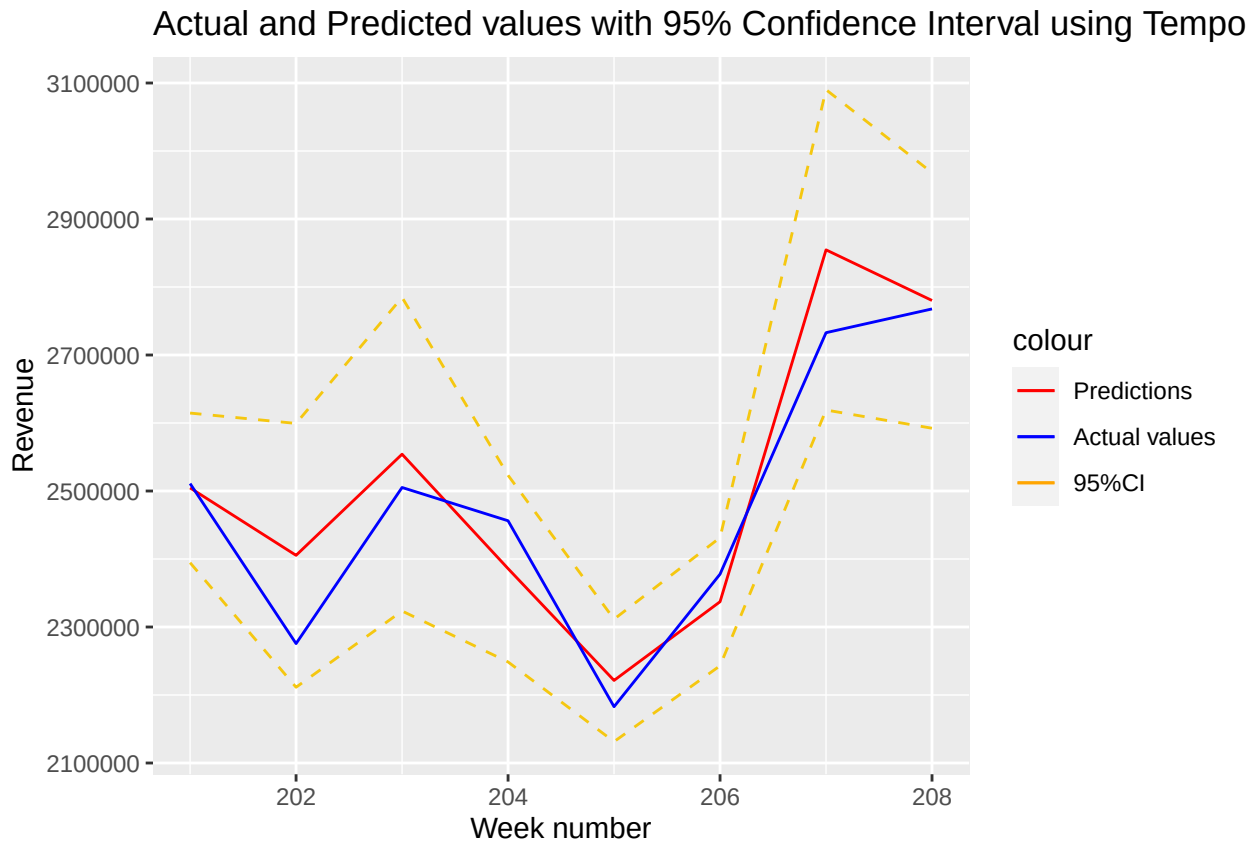



```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 23 iterations.
## The RMS GCV score gradient at convergence was 2883.113 .
## The Hessian was positive definite.
## Model rank =  95 / 95
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                          k'      edf k-index p-value
## s(X)               4.90e+01 2.38e-07    1.00    0.34
## s(media1_S)        9.00e+00 2.09e+00    1.09    0.96
```

```
## s(media2_S)          9.00e+00 1.00e+00    1.04    0.64
## s(media3_S)          9.00e+00 1.72e+00    0.97    0.31
## s(competitor_sales) 9.00e+00 1.00e+00    0.94    0.17
## s(newsletter)        9.00e+00 2.24e+00    1.08    0.90
```

In this case, all the p-values for the smooth terms (s(X), s(media1_S), s(media2_S), s(media3_S), s(competitor_sales), and s(newsletter)) are relatively high, indicating that the basis dimension is appropriate for these terms. Based on the summary, the model seems to be reasonable and well-fit.

```
#test data for GAM model
test <- cbind(X= test_data$X, test)
#prediction for GAM model
gam_predictions <- predict(gam_model, newdata = test, type = "response", se.fit = TRUE)

# 95% CI for GAM
gam_lower <- gam_predictions$fit - 1.96*gam_predictions$se
gam_upper <- gam_predictions$fit + 1.96*gam_predictions$se
```




```
#RMSE for GAM model
gam_rmse<-(sqrt(mean(test_data$revenue - gam_predictions$fit)^2))
gam_rmse
```

```
## [1] 29293.95
```

```r
#R-squared for GAM model
total <- sum((test_data$revenue-mean(test_data$revenue))^2)
residual <- sum((gam_predictions$fit - test_data$revenue)^2)
gam_r_squared <- 1 - residual/total
gam_r_squared
```

```
## [1] 0.8535994
```

From the above plot we notice predictiona nd actual values lie in between 95% Intervals and we also have $RMSE = 29293.9$ & $R^2 = 0.85$ this in turns gives proves that model is suitable for predicting revenue. Other Deep learning techniques can also be used for time-series analysis which are really good compared to above models.

2) Bayesian Modeling

Fit the model in a Bayesian framework using non-informative priors:
- Write down your priors. Why are they non-informative?
- Are your priors conjugate?
- Which Markov chain Monte Carlo (MCMC) algorithm did you use? Why?

From part 1) I have chosen GLM model and fitting Bayesian GLM model using stan_glm with non-informative priors. Non-informative priors I have chosen for both both intercept and co-efficients are normal distribution with location parameter of 0 and scale parameter of 1000.

These priors are considered non-informative because they are chosen to have a broad distribution with a large scale parameter. These non-informative priors do not impose strong assumptions or prior knowledge on the parameters of the model, allowing the data to have a greater influence on the posterior distribution.

The priors used here are non-conjugate priors, because normal prior distribution is used in model, when combined with the likelihood function of the GLM model, the resulting posterior distribution is not a normal distribution. Therefore, the normal prior is not conjugate to the likelihood function of the GLM model. This provide more flexibility and complexity in modeling scenarios.

I have chosen stan_glm() MCMC algorithm, because:
- It allows to fit GLMs with various response distributions and link functions, including Gaussian, Poisson etc., This makes it suitable for wide range of model scenarios.
- It utilizes Bayesian inference, which provides a probabilistic framework for modeling. By estimating posterior distributions, rather than point estimates, Bayesian modeling allows for uncertainty quantification and inference on parameters and predictions.
- It provides diagnostics to assess the convergence of the MCMC chains, such as the potential scale reduction factor (Rhat) and effective sample size (n_eff).

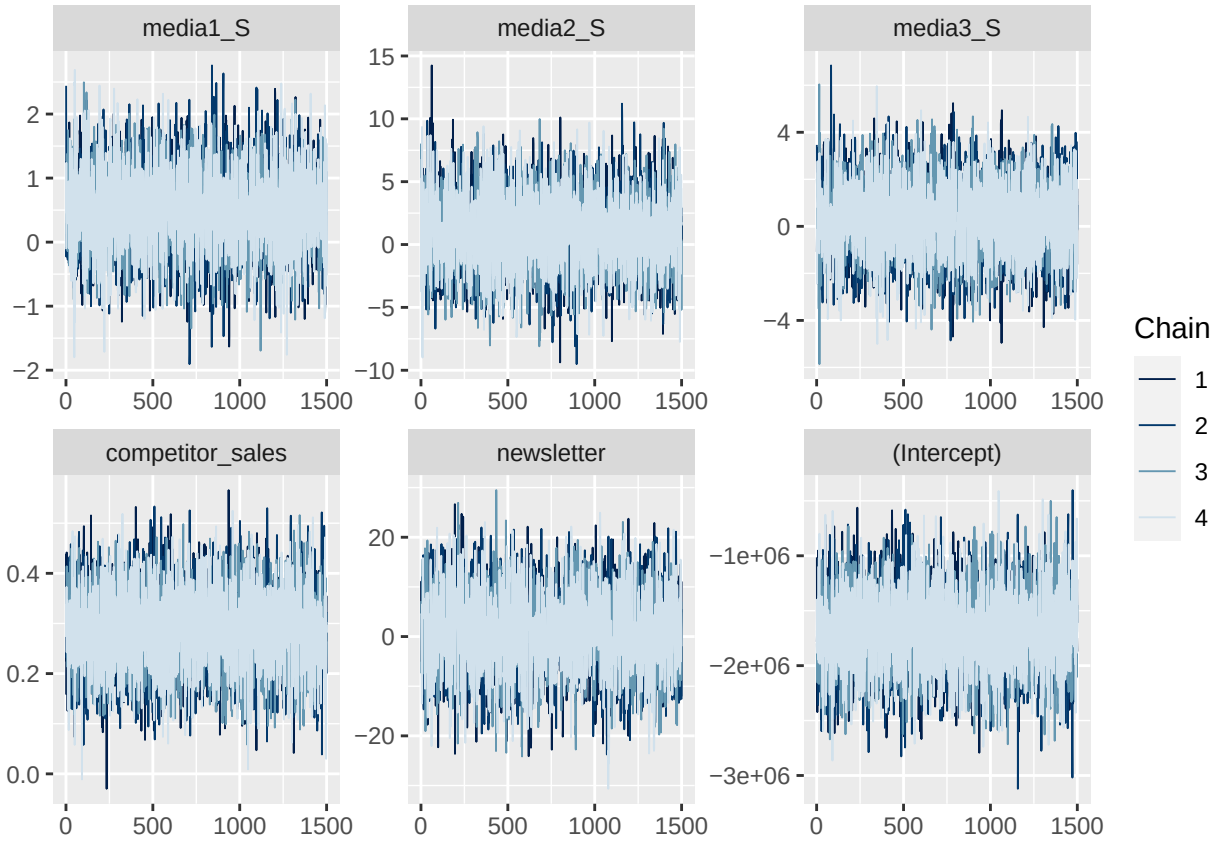
```r
#libraries
library(rstanarm)
library(coda)
library(bayesplot)
# Set non-informative priors
prior_intercept <- normal(location = 0, scale = 1000)
prior <- cauchy(location = 0, scale = 1000)
#formula for model
formula<- revenue ~ media1_S + media2_S + media3_S + competitor_sales + newsletter
# Fit the MCMC glm model
mc_model <- stan_glm(formula,
                data = train_data, family = gaussian(),
                prior_intercept = prior_intercept,
                prior = prior,
                chains = 4, iter = 2000, warmup = 500, seed = 123)
```

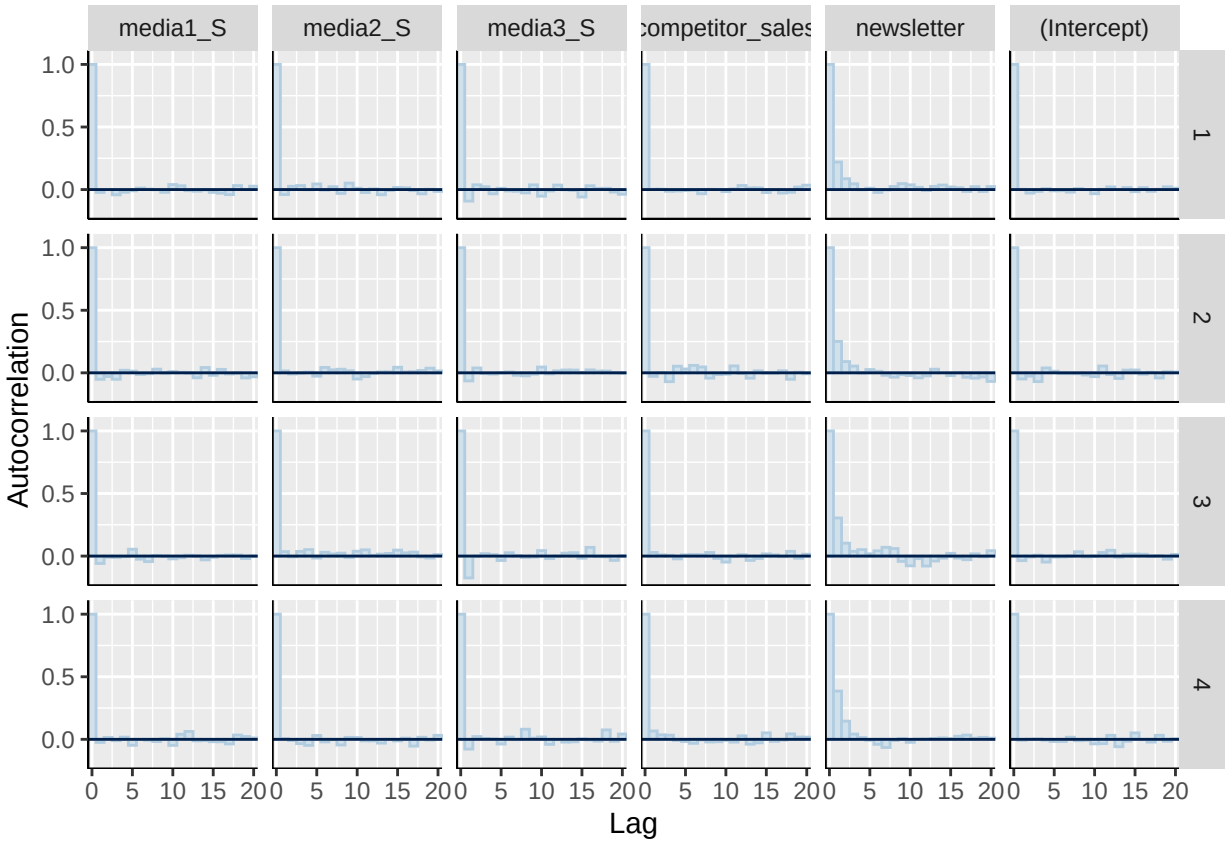
```
##
```

```
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 1: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 1: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 1: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 1: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 1: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 1: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 1: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 10.849 seconds (Warm-up)
## Chain 1:                4.897 seconds (Sampling)
## Chain 1:                15.746 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0.001 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 10 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 2: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 2: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 2: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 2: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 2: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 2: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 2: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 20.644 seconds (Warm-up)
## Chain 2:                4.73 seconds (Sampling)
## Chain 2:                25.374 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
```

```
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 3: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 3: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 3: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 3: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 3: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 3: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 3: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 12.292 seconds (Warm-up)
## Chain 3:                5.945 seconds (Sampling)
## Chain 3:                18.237 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 4: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 4: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 4: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 4: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 4: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 4: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 4: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 16.675 seconds (Warm-up)
## Chain 4:                5.499 seconds (Sampling)
## Chain 4:                22.174 seconds (Total)
## Chain 4:
```

TRACE Plots of chains


media1_S
media2_S
media3_S
competitor_sales
newsletter
(Intercept)
Chain
1
2
3
4



From the above traceplots and ACF barplots it is noted that the MCMC chains have converged really well.

Table 2: Summary of MCMC model

	mean	mcse	sd	10%	50%	90%	n_eff	Rhat
(Intercept)	-1691479.807	4484.951	355465.975	-2143898.197	-1695483.147	-1232220.904	6282	1.000
media1_S	0.500	0.008	0.628	-0.293	0.493	1.308	6512	1.000
media2_S	0.998	0.038	2.876	-2.659	0.981	4.597	5591	1.000
media3_S	0.315	0.018	1.535	-1.646	0.307	2.270	6903	1.000
competitor_sales	0.288	0.001	0.078	0.189	0.290	0.389	5707	1.000
newsletter	0.854	0.144	7.969	-9.453	0.908	11.088	3078	1.000
sigma	1837171.918	2055.393	94158.891	1718616.106	1833486.777	1959126.317	2099	1.000
mean_PPD	-91.944	1706.688	130363.524	-167830.993	601.751	164336.624	5835	1.000
log-posterior	-3183.746	0.044	1.933	-3186.260	-3183.437	-3181.619	1951	1.002

Both Classic GLM model and Bayesian GLM models are similar because, the coefficient estimates and inference results are similar between the Bayesian model and classical GLM model because they both provide estimates of the regression coefficients and associated statistical measures such as standard errors, t-values, and p-values.

In both models:

- The coefficient estimates show the predicted change in the response variable (revenue) associated with a one-unit increase in the related predictor variable, while maintaining other predictors constant.

- The standard errors quantify the uncertainty in the coefficient estimates. Smaller standard errors indicate greater precision in the estimates.
- The t-values determine the significance of the coefficient estimates by comparing them to their standard errors. Higher absolute t-values indicate greater evidence against the null hypothesis (no effect).
- The p-values indicate the probability of observing a coefficient estimate as extreme as the one obtained if the null hypothesis were true. Smaller p-values indicate stronger evidence against the null hypothesis.

Therefore, when comparing the coefficient estimates, standard errors, t-values, and p-values between the models, we can see that they provide similar information about the relationships between the predictors and the response variable. This indicates that the two models are consistent in their estimation and inference regarding the impact of the predictors on the response variable.

2b) Prior choice

To incorporate the expert's belief that Media 3 has no impact on generating marketing revenues while still considering the information from the dataset, we can modify the prior specification for the corresponding coefficient in the Bayesian model.

Initially the stan_glm function was assigned with a non-informative prior to the coefficients, assuming a normal distribution centered around zero. To reflect the expert's belief, we specify a prior that assigns higher probability to values close to zero for the coefficient of media3_S. This can be done by decreasing the prior variance, which will make the prior distribution more concentrated around zero.
This change makes the prior distribution more concentrated around zero, reflecting the expert's belief that Media 3 has no impact. However, it still allows the data to influence the estimation and update the prior beliefs based on the observed evidence. Taking into account the level of certainty the expert has in their belief and the available evidence in the dataset. The modification aims to incorporate the expert's opinion while allowing the data to have a meaningful impact on the parameter estimation.

```
#changing prior for media3_S
prior_media3 <- normal(location = 0, scale = 1)

#fitting Bayesian GLM with new prior for media3_S
mc_model2 <- stan_glm(formula,
                data = train_data,family = gaussian(),
                prior_intercept = prior_intercept,
                prior = c(prior, prior_media3),
                chains = 4,iter = 2000,warmup = 500,
                seed = 123
              )
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 1: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 1: Iteration:  900 / 2000 [ 45%]  (Sampling)
```

```
## Chain 1: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 1: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 1: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 1: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 1: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 12.037 seconds (Warm-up)
## Chain 1:                5.543 seconds (Sampling)
## Chain 1:                17.58 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 2: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 2: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 2: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 2: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 2: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 2: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 2: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 20.414 seconds (Warm-up)
## Chain 2:                4.256 seconds (Sampling)
## Chain 2:                24.67 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 3: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 3: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 3: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 3: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 3: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 3: Iteration: 1700 / 2000 [ 85%]  (Sampling)
```

```
## Chain 3: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 9.276 seconds (Warm-up)
## Chain 3:                4.571 seconds (Sampling)
## Chain 3:                13.847 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0.001 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 10 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 4: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 4: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 4: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 4: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 4: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 4: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 4: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 12.289 seconds (Warm-up)
## Chain 4:                4.249 seconds (Sampling)
## Chain 4:                16.538 seconds (Total)
## Chain 4:
```

Table 3: Summary of MCMC model with changed prior for media3

	mean	mcse	sd	10%	50%	90%	n_eff	Rhat
(Intercept)	-1691479.807	4484.951	355465.975	-2143898.197	-1695483.147	-1232220.904	6282	1.000
media1_S	0.500	0.008	0.628	-0.293	0.493	1.308	6512	1.000
media2_S	0.998	0.038	2.876	-2.659	0.981	4.597	5591	1.000
media3_S	0.315	0.018	1.535	-1.646	0.307	2.270	6903	1.000
competitor_sales	0.288	0.001	0.078	0.189	0.290	0.389	5707	1.000
newsletter	0.854	0.144	7.969	-9.453	0.908	11.088	3078	1.000
sigma	1837171.918	2055.393	94158.891	1718616.106	1833486.777	1959126.317	2099	1.000
mean_PPD	-91.944	1706.688	130363.524	-167830.993	601.751	164336.624	5835	1.000
log-posterior	-3183.746	0.044	1.933	-3186.260	-3183.437	-3181.619	1951	1.002

We notice that there's is no difference in results after changing the priors of media3_S, because the data-driven information from the observed data has a stronger influence on the parameter estimates than the prior beliefs when the data provides strong evidence. If the data clearly indicates a specific value for the coefficient of media3_S, the prior specification will have minimal impact on the final results. The estimates will be mainly determined by the data itself.

In this case, it appears that the data is providing strong evidence for the coefficient of media3_S, and the prior specification is not strong enough to significantly alter the estimates. This is why the results remain similar despite reducing the prior distribution for media3_S.

To incorporate the opinions from the additional expert, we modify the prior specifications for the respective variables in the following way:

1. Newsletter: Since the expert believes there is a strictly positive relationship between newsletter and marketing revenues, we will assign a prior that reflects this belief. A suitable choice could be a exponential distribution with rate = 1.

```
prior_newsletter <- exponential(rate=1)
```

2. Media 1: The expert believes that one unit invested in Media 1 will result in two units of marketing revenues returned. To reflect this belief, we are assigning a prior with a mean of 2 and a relatively narrow distribution around it. A normal distribution with a smaller standard deviation can capture this belief.

```
prior_media1_S<- normal(location = 2, scale = 0.5)
```

3. Media 2: The expert believes that the impact of Media 2 is four times that of Media 1. To incorporate this belief, we adjust the prior specification for Media 2 accordingly. If the prior for Media 1 is N(mean_media1, sd_media1), then the prior for Media 2 can be N(4 * mean_media1, 4 * sd_media1).

```
prior_media2_S <- normal(location = 4 * prior_media1_S$location, scale = 4 * prior_media1_S$scale)
```

4. Competitor_sales: The expert believes that for every unit increase in competitor_sales, the change on revenues returned must range between 0 and 0.3. To incorporate this belief, we can use a bounded prior distribution such as a truncated normal distribution with appropriate bounds.

```
prior_competitor_sales <- function() {
  sample <- rnorm(1, mean = 0.15, sd = 0.05)
  if (sample < 0 || sample > 0.3) sample_competitor_sales() else sample
}
```

By incorporating these prior specifications, we are allowing the prior beliefs of the additional expert to influence the parameter estimation process. Stronger priors will have a larger impact on the results, while weaker priors will allow the data to dominate the estimation process.

Fitting Bayesian model with new priors to predictors.

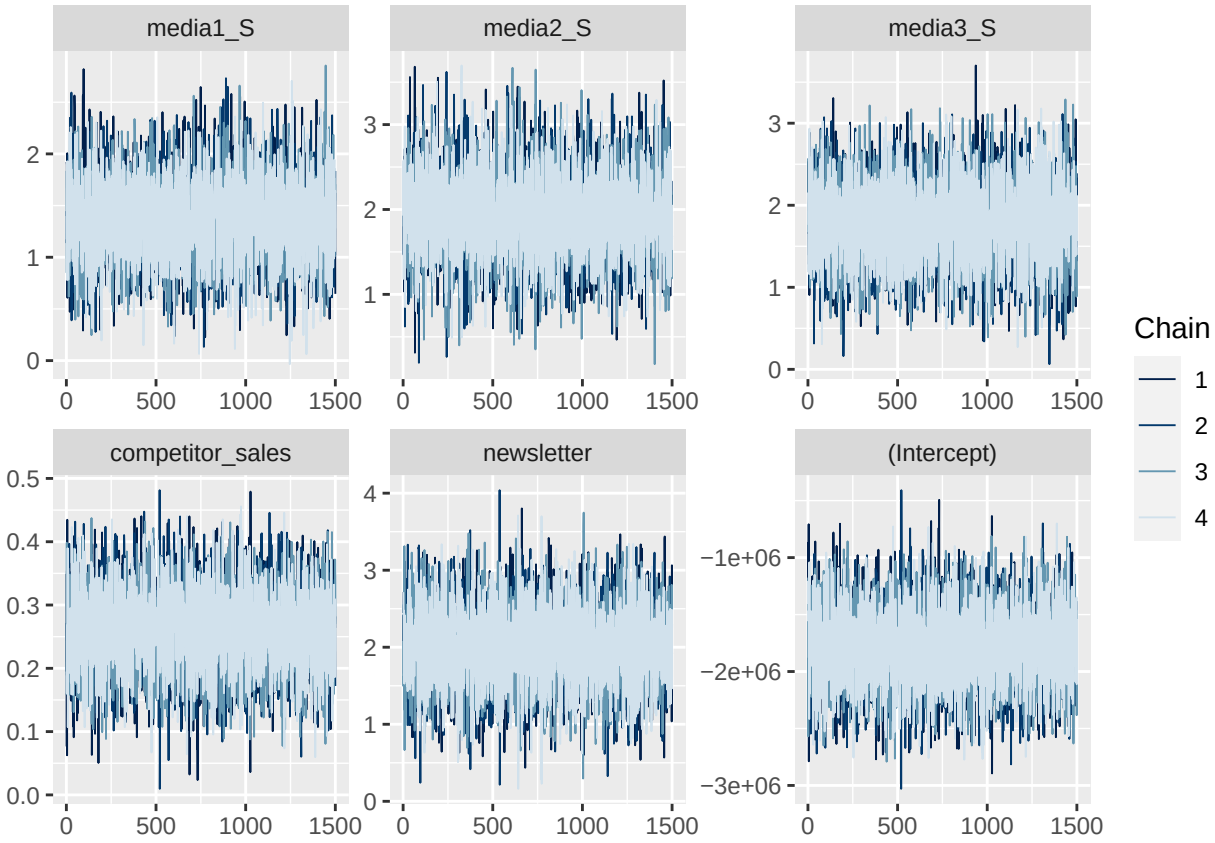
```
prior_intercept <- normal(location = 0, scale = 1000)
#new prior for media1
prior_media1_S <- normal(location = 2, scale = 0.5)
#new prior for media2
prior_media2_S <- normal(location = 4 * prior_media1_S$location, scale = 4 * prior_media1_S$scale)
#new prior for media3
prior_media3_S <- normal(location = 0, scale = 1)
```

```r
#new prior for newsletter
prior_newsletter <- exponential(rate = 1)  # Strictly positive relationship
#new prior for competitor sales
prior_competitor_sales <- function() {
  sample <- rnorm(1, mean = 0.15, sd = 0.05)
  if (sample < 0 || sample > 0.3) sample_competitor_sales() else sample
}
#formula
formula <- revenue ~ media1_S + media2_S + media3_S + competitor_sales + newsletter
#fitting model with new prior for co-efficients of predictors
mc_model3 <- stan_glm(formula, data = train_data,
                family = gaussian(),prior_intercept = prior_intercept,
                prior = c(
                  prior_media1_S,
                  prior_media2_S,
                  prior_media3_S,
                  prior_competitor_sales,
                  prior_newsletter
                ),
                chains = 4,iter = 2000,warmup = 500,seed = 123
              )
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 1: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 1: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 1: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 1: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 1: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 1: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 1: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 1.955 seconds (Warm-up)
## Chain 1:                0.086 seconds (Sampling)
## Chain 1:                2.041 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
```

```
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 2: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 2: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 2: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 2: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 2: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 2: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 2: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 2.083 seconds (Warm-up)
## Chain 2:                0.087 seconds (Sampling)
## Chain 2:                2.17 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 3: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 3: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 3: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 3: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 3: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 3: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 3: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 2.156 seconds (Warm-up)
## Chain 3:                0.09 seconds (Sampling)
## Chain 3:                2.246 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
```

```
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 4: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 4: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 4: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 4: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 4: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 4: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 4: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 1.921 seconds (Warm-up)
## Chain 4:                0.094 seconds (Sampling)
## Chain 4:                2.015 seconds (Total)
## Chain 4:
```



From the above traceplots the chains are converged.

Table 4: Summary of MCMC model with new priors

	mean	mcse	sd	10%	50%	90%	n_eff	Rhat
(Intercept)	-1783608.737	3353.334	338776.587	-2213192.908	-1781977.465	-1344219.449	10206	1
media1_S	1.370	0.005	0.395	0.862	1.375	1.863	7551	1
media2_S	1.957	0.005	0.489	1.341	1.955	2.570	8644	1

	mean	mcse	sd	10%	50%	90%	n_eff	Rhat
media3_S	1.799	0.005	0.474	1.185	1.795	2.414	10204	1
competitor_sales	0.261	0.001	0.064	0.179	0.261	0.341	9583	1
newsletter	1.999	0.006	0.508	1.355	1.997	2.633	7706	1
sigma	1835019.856	996.446	92430.988	1718935.176	1830556.617	1954509.762	8605	1
mean_PPD	-3012.228	1663.328	131256.089	-172086.562	-3296.754	165178.145	6227	1
log-posterior	-3192.342	0.036	1.851	-3194.834	-3192.036	-3190.273	2689	1

Estimates of coefficients:

The intercept term (Intercept) in mc_model has a mean of -1,691,479.8, while in mc_model3 it has a mean of -1,783,608.7. The coefficient for media1_S is estimated to be 0.5 in mc_model and 1.4 in mc_model3. The coefficient for media2_S is estimated to be 1.0 in mc_model and 2.0 in mc_model3. The coefficient for media3_S is estimated to be 0.3 in mc_model and 1.8 in mc_model3. The coefficient for competitor_sales is estimated to be 0.3 in both models. The coefficient for newsletter is estimated to be 0.9 in mc_model and 2.0 in mc_model3.

Standard deviations: The standard deviations for the coefficients vary between the two models, indicating differences in the uncertainty associated with the estimates. Newsletter coefficient:

mc_model3 assigns a higher coefficient to the newsletter variable (2.0) compared to mc_model (0.9), suggesting a stronger relationship between newsletter and marketing revenues in mc_model3.

Mean_PPD: The mean posterior predictive distribution of the outcome variable (mean_PPD) in mc_model is -91.9, while in mc_model2 it is -3,012.2. This represents the average predicted values based on each model.

MCMC diagnostics: The Monte Carlo standard error (mcse), potential scale reduction factor (Rhat), and crude measure of effective sample size (n_eff) values differ between the two models, indicating variations in convergence and sampling efficiency.

APPENDIX

```
library(GGally)
```

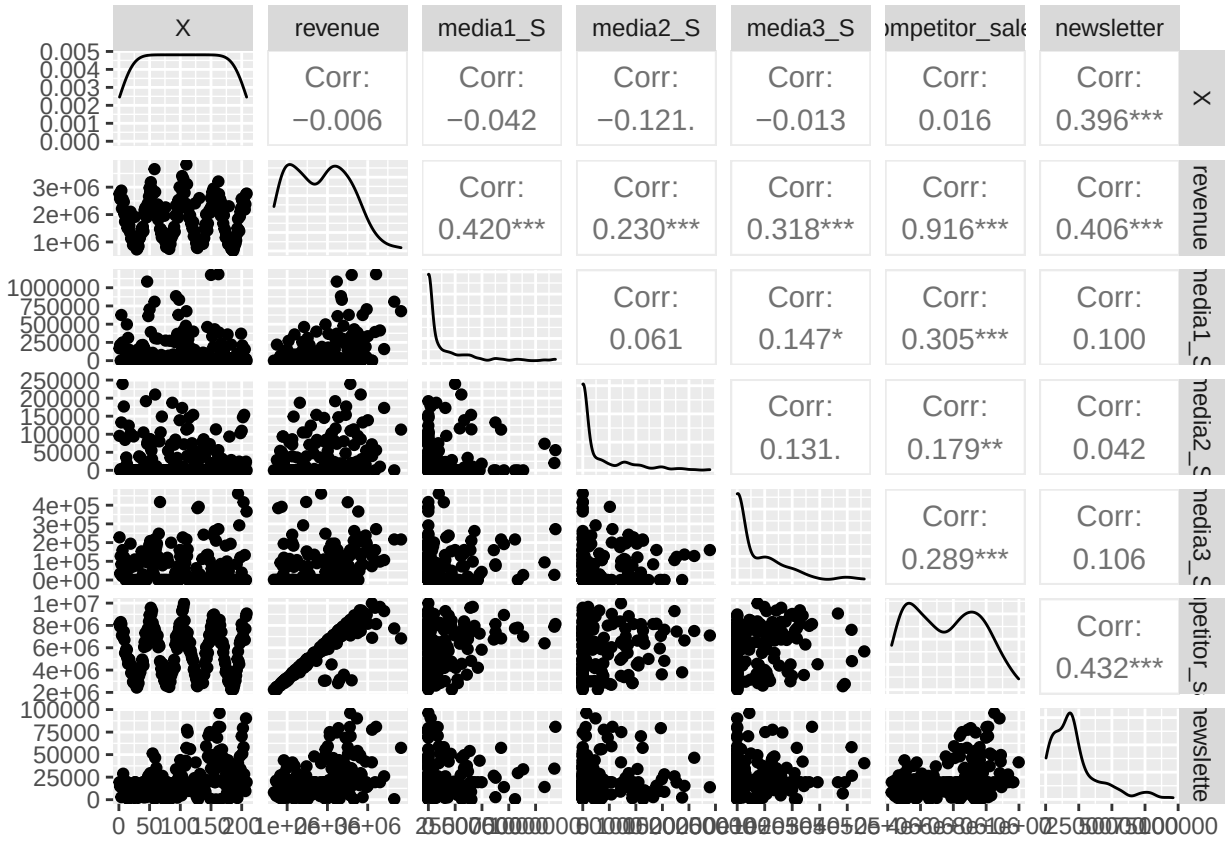
```
## Warning: package 'GGally' was built under R version 4.1.2
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
# Read the data from a CSV file
data <- read.csv("weekly_media_sample.csv")

summary(data)
```

```
##        X              DATE              revenue           media1_S
##  Min.   :  1.00   Length:208         Min.   : 672250   Min.   :      0
##  1st Qu.: 52.75   Class :character   1st Qu.:1165211   1st Qu.:      0
##  Median :104.50   Mode  :character   Median :1874514   Median :      0
##  Mean   :104.50                      Mean   :1822143   Mean   : 111328
##  3rd Qu.:156.25                      3rd Qu.:2378407   3rd Qu.: 138050
##  Max.   :208.00                      Max.   :3827520   Max.   :1185349
##     media2_S         media3_S      competitor_sales    newsletter
##  Min.   :     0   Min.   :     0   Min.   :2240235   Min.   :  301
##  1st Qu.:     0   1st Qu.:     0   1st Qu.:3589581   1st Qu.: 9010
##  Median :     0   Median :     0   Median :5538524   Median :19402
##  Mean   : 27965   Mean   : 64370   Mean   :5538025   Mean   :22387
##  3rd Qu.: 35759   3rd Qu.:108690   3rd Qu.:7311814   3rd Qu.:27547
##  Max.   :239417   Max.   :462012   Max.   :9984742   Max.   :96236
```

```
ggpairs(data[c(-2)])
```



From the above plot we can notice that revenue has positive linear relationship with competitor sales and relationship with other variables seems non-linear. And it is also observed that there's is seasonality for revenue and competitor sales.