# Calina - Prova Técnica

Daniel David de Oliveira

25 de Dezembro de 2020

## *CASE*

Em uma agência de *Marketing Digital* uma das épocas mais importante para o ramo de *ecommerce* é a *"Black Friday"*, período sazonal em que muitos dos clientes se planejam com promoções e ações através das mídias pagas para chamar a atenção dos usuários. Pensando nessa data muito especial , um cliente da Calina solicitou uma análise para prevermos qual será a receita da *Black Friday* de 2020.

O banco de dados enviado contém dados de 3 mídias em que o cliente investe (Mídia A, B e C) e o total da receita gerada no site por semana, desde a primeira semana de 2018 até a última semana de outubro de 2020.

Para responder ao cliente análise o banco de dados, crie um modelo teste e um modelo final que deve prever as próximas 4 semanas, respectivas ao mês de novembro de 2020 (a última semana é a semana da *Black Friday*).

***Resposta:*** Para a solução do problema, focarei na variável Total da receita gerada por semana, chamando-a de Receita Total ou Receita e usarei linguagem de cunho técnica/acadêmica para expor meus raciocínios.

## *Lendo o Banco de dados:*

```
library(readxl)
library(magrittr)
library(ggplot2)
library(fpp2)
library(forecast)
  library(GGally)
library(lubridate)
library(MLmetrics)
options(scipen = 10000000) #Tirar a notação científica dos gráficos

dds <- read_excel("C:/Users/danie/OneDrive/Área de Trabalho/Calina - R/Ciência de Dados.
                  col_types = c("date", "numeric", "numeric",
                                "numeric", "numeric"))

dados <- dds[-1]
```

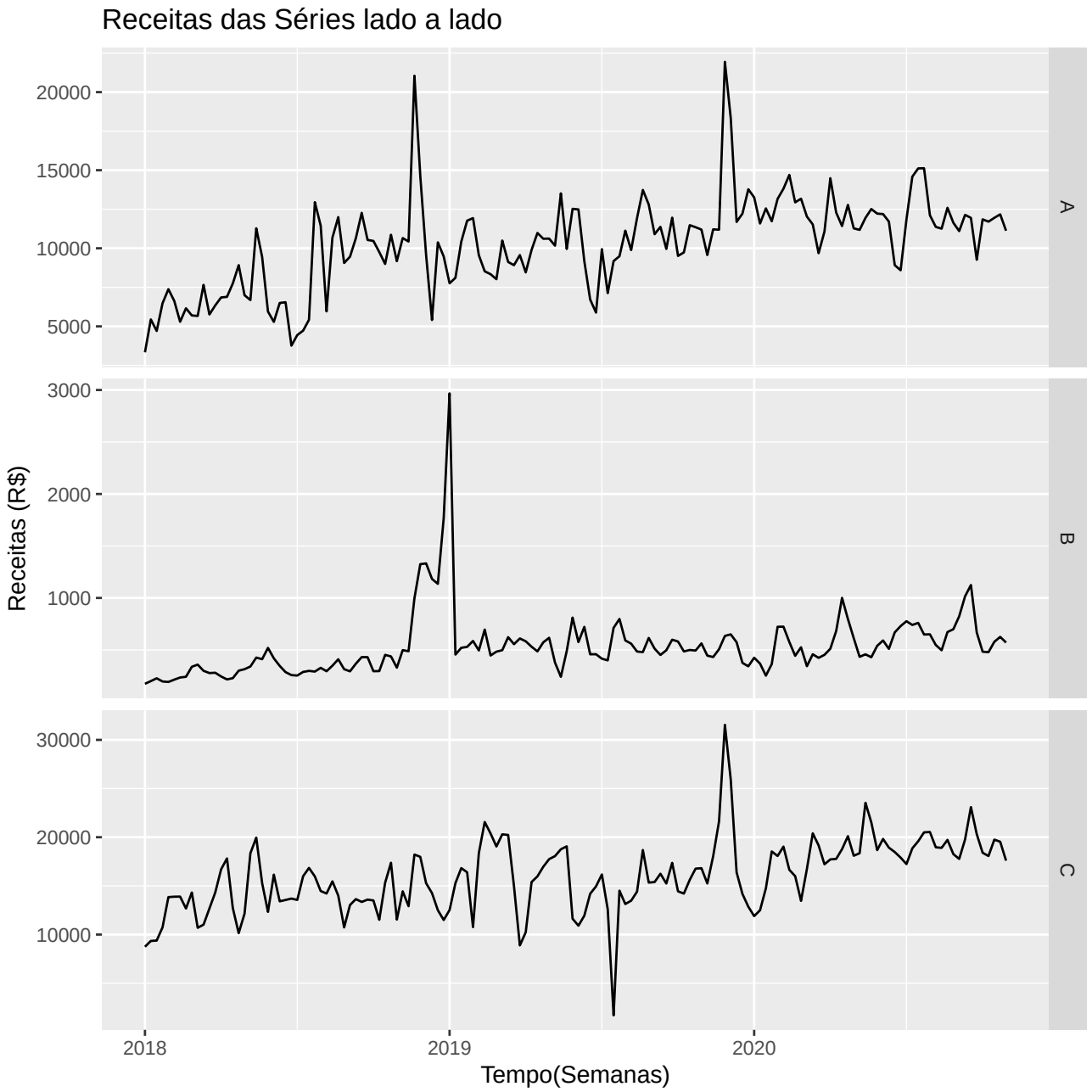
Criando Séries usando o banco de dados

```
#Tirando a variável Week
dados <- dds[-1]

#Criando as Séries
ts = ts(dados,
        start=c(2018, 1),
        frequency = 52)

R.ts = ts(dados[,4],
          start = 2018,
          frequency = 52)
```

## *Análise Descritiva Simples das Séries*

Plot das Séries Temporais das Receitas A, B e C lado a lado

```
autoplot(ts[,-4], facets=TRUE,
         main="Receitas das Séries lado a lado") +
  ylab("Receitas (R$)") +
  xlab("Tempo(Semanas)")
```

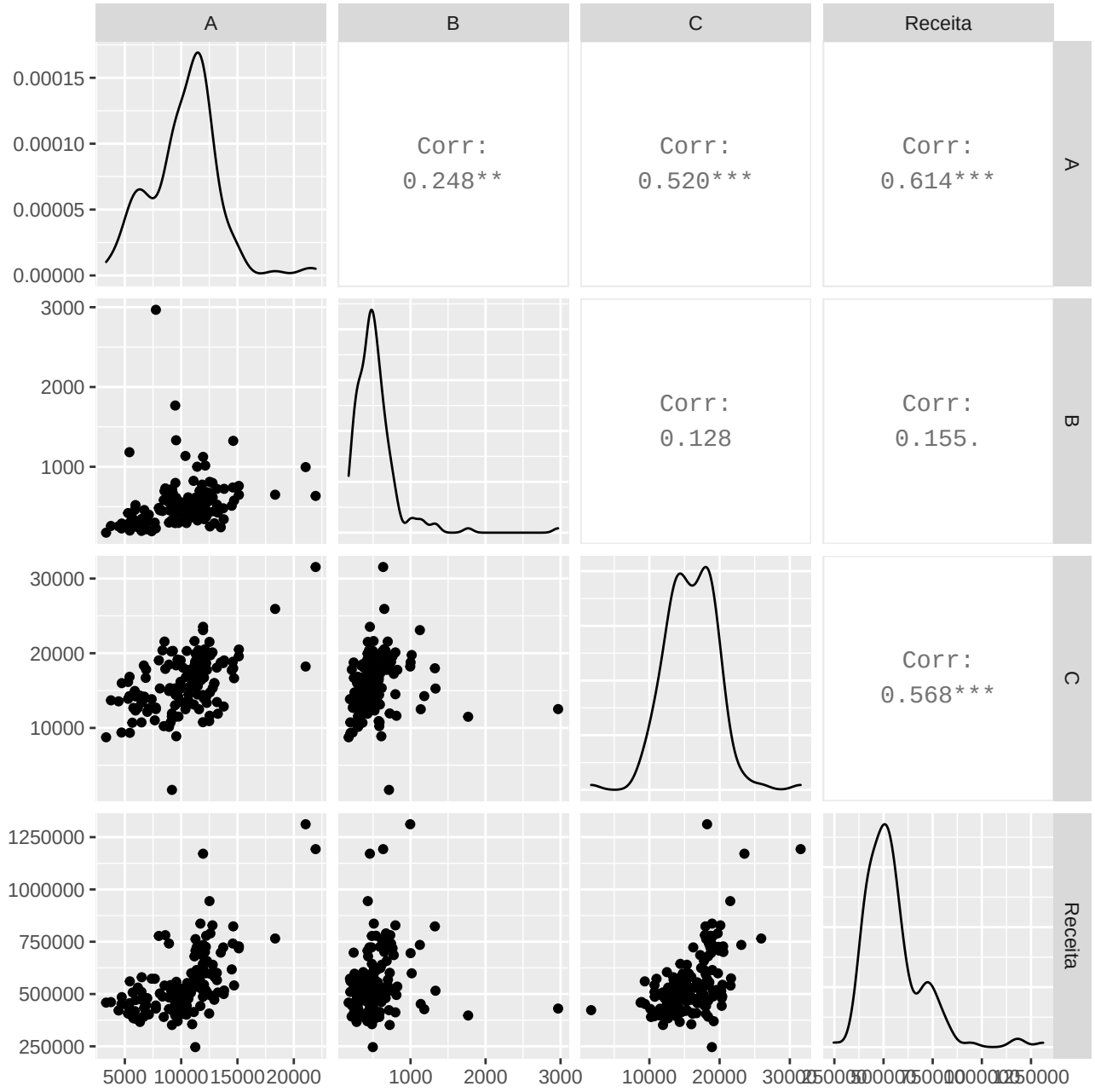



Mostra-se preditores adicionais possívelmente úteis. Importantes para previsão da Receita Total na qual o cliente irá investir nas mídias semanalmente.

Construindo um modelo de regressão linear múltipla pode-se gerar previsões mais precisas utilizando as variáveis de receita das mídias A, B e C. Espera-se que uma, duas ou todas sejam dependentes da variável Receita Total.

Correlação entre as variáveis de Receita e seus gráficos de dispersão

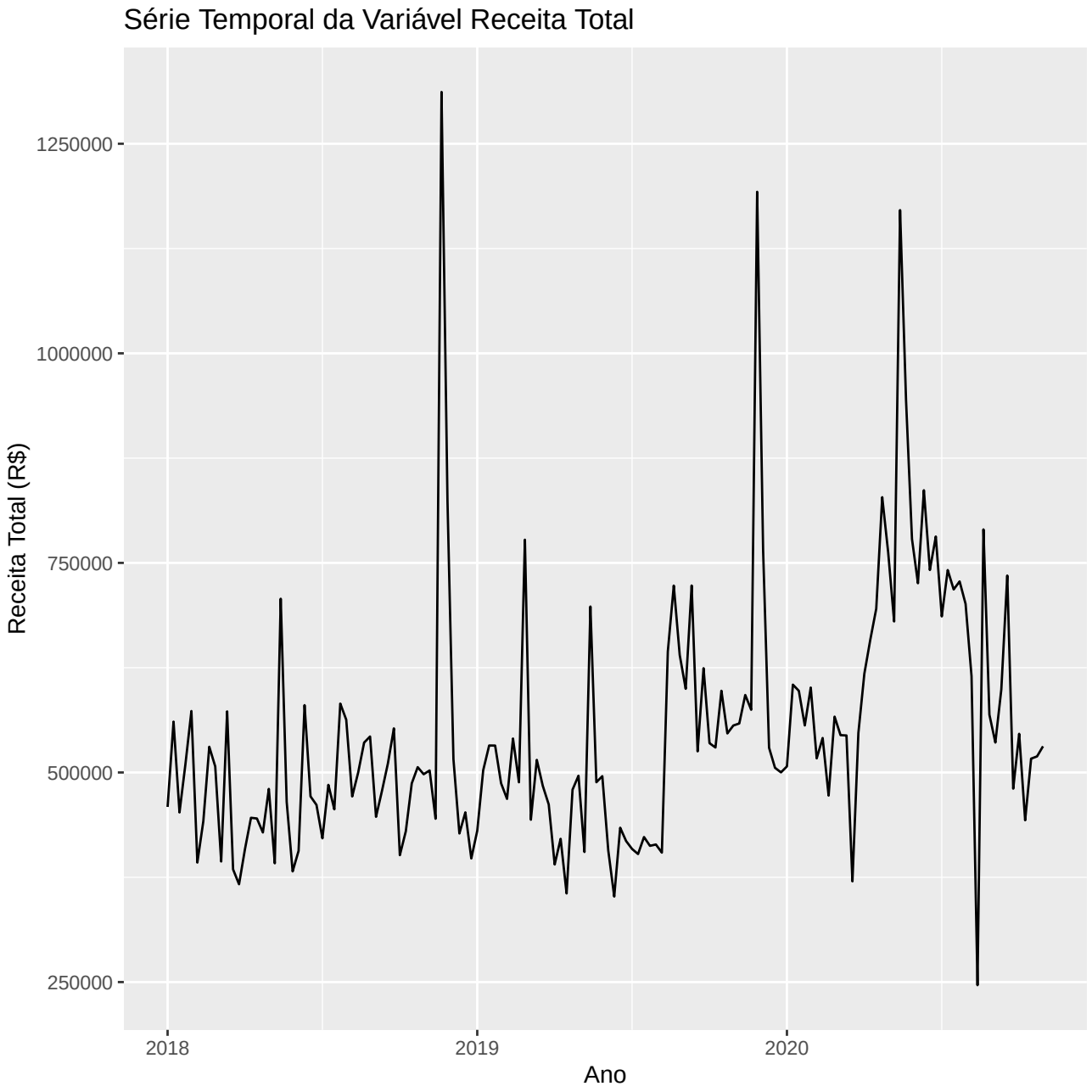
```
ts %>%
  as.data.frame() %>%
  ggpairs()
```




A última linha mostra a relação entre a variável de previsão e cada um dos preditores (Receitas A, B e C). Os gráficos de dispersão mostram relações positivas entre as variáveis Receita e Mídia A, e Receita e Mídia B. A força dessas correlações são mostradas pela correlação dos coeficientes através da última coluna.

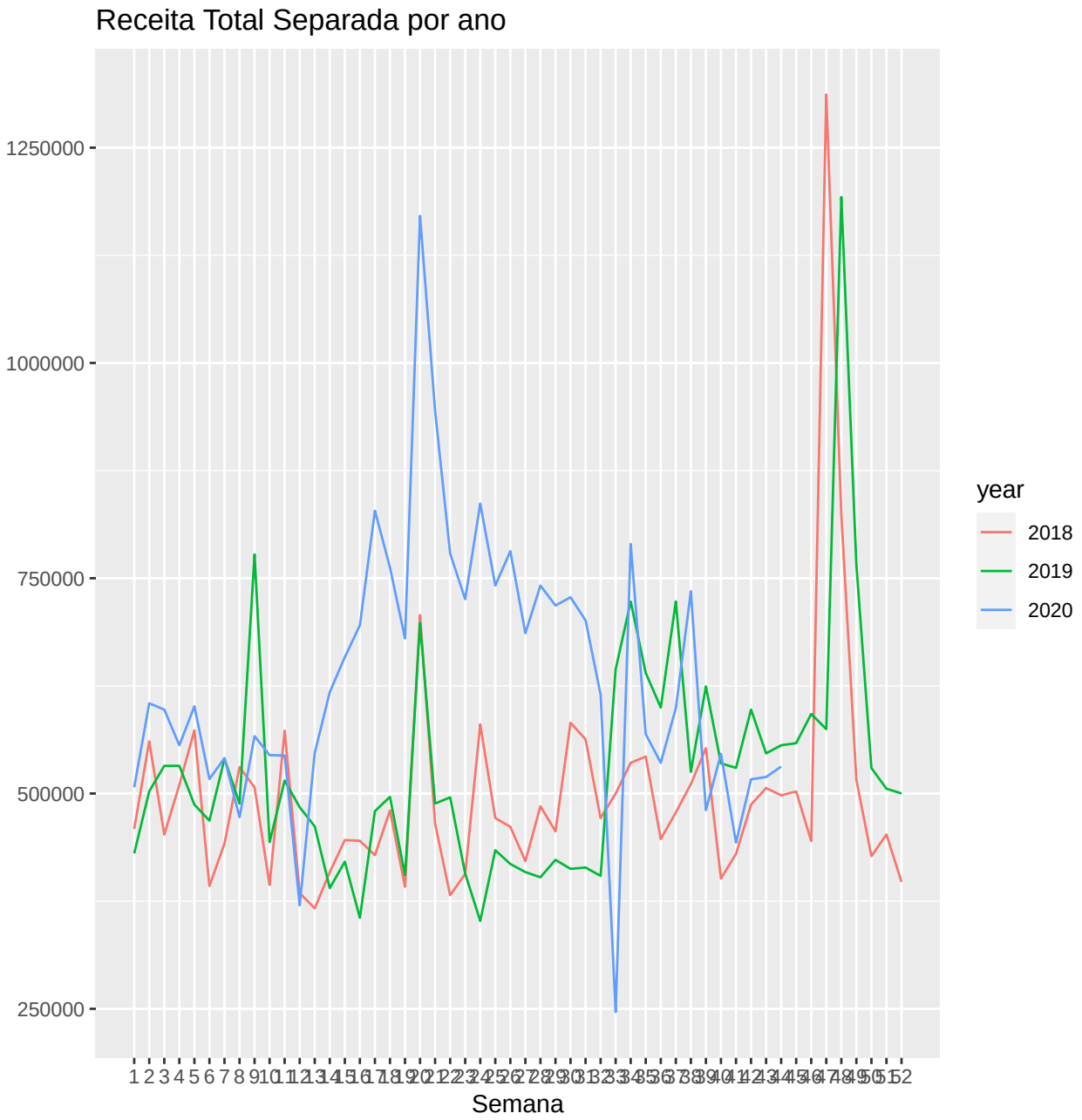
Verificando o Comportamento da Série Temporal (ST) Receita

```
autoplot(R.ts, xlab = "Ano",
         ylab = "Receita Total (R$)",
         main="Série Temporal da Variável Receita Total")
```




Perecebe-se alguns picos nos últimos trimestres dos anos de 2018 e 2019. Podemos ver com mais clareza o comportamento da Série temporal Receita Total separada por Ano

```
#Plot das Séries da Receita Total separada por Ano
                #library - forecast
ggseasonplot(R.ts, xlab="Semana",
             main="Receita Total Separada por ano")
```

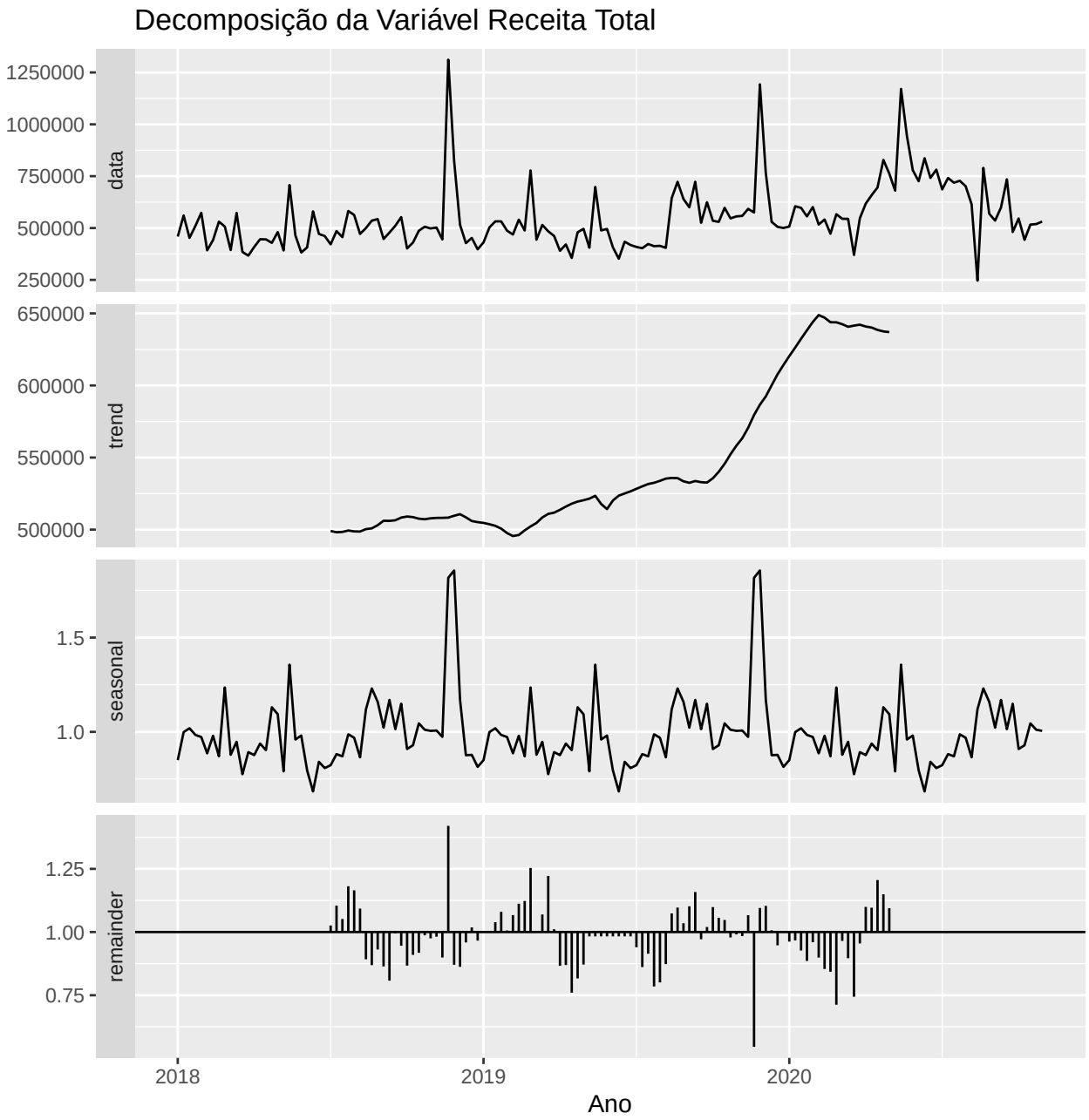



De fato, percebemos um pico de investimento na semana 47 do ano de 2018, outro na semana 48 em 2019, correspondentes aos dias 19/11/2018 e 25/11/2020, datas próximas a *Black Friday* (BF) de cada ano, esse fato pode ser de suma importância futuramente, visto que nossa última semana de previsão terá o evento BF.

*Comentário:* No eixo x o número das semanas estão sobrepostos, no código R podemos ver um pouco mais com clareza qual semana é qual.

Decomposição da ST Receita Total

```
#Decomposição
R.ts %>% decompose(type="multiplicative") %>%
  autoplot() + xlab("Ano") +
  ggtitle("Decomposição da Variável Receita Total")
```




O primeiro gráfico nos mostra o comportamento da Série Temporal da variável Receita. O segundo, mostra-nos a tendencia que a séria possui. O terceiro mostra a sazonalidade e percebemos que no último trimestre há um pico na Receita Total, creio que seja altos investimentos na semana da *Black Friday*.

## *Escolha do Modelo*

Munido dos gráficos de correlação e decomposição, podemos supor alguns modelos.

Das correlações, imagina-se dois modelos:

A primeira suposição, um modelo de Regressão linear com todas as mídias:

```
#Receita ~ A + B + C
fit  <- tslm(Receita ~ A + B + C,
             data = ts)
```

A Segunda retirando do modelo o investimento da mídia B:

```
#Não contando com a mídia B
#Receita ~ A  + C
fit2 <- tslm(Receita ~ A + C,
             data = ts)
```

Terceira ajustando o modelo contando com a sazonalidade e tendência da série temporal. Pode ser um ótimo modelo pela sazonalidade, por outro lado, a tedência pode nos atrapalhar nas previsões das próximas semanas. Apoiando-me no gráfico de Trend mostrado no *plot* de Decomposição:

```
#Receita ~ Tendência + Temporada
fit3  <- tslm(Receita ~  trend + season,
              data = ts)
```

Por fim o modelo usando somente a sazonalidade:

```
#Receita ~ Temporada
fit4 <- tslm(Receita ~ season,
             data = ts)
```

Iremos calcular a estatística de validação cruzada e verificar qual terá o BIC maior.

```
CV(fit)
```

```
##                  CV                 AIC                AICc                 BIC
## 13882909433.7199230        3454.6748382        3455.0973734        3469.6608995
##               AdjR2
##           0.4502268
```

```
CV(fit2)
```

```
##                  CV                 AIC                AICc                 BIC
## 13808837242.2737045        3452.6771602        3452.9568805        3464.6660093
##               AdjR2
##           0.4540098
```

```
CV(fit3)
```

```
##                  CV                AIC               AICc                BIC
## 28219113293.228031        3519.774890        3583.645858        3681.624353
##               AdjR2
##           0.332763
```

```
CV(fit4)
```

```
##                  CV                 AIC                AICc                 BIC
## 35824767315.8067093        3561.9752317        3622.8688487        3720.8274822
##               AdjR2
##           0.1099079
```

Seguimos em frente com o modelo 'fit4' por possuir o maior BIC, e atento ao enunciado, faz sentido que o modelo adote sazonalidade.

# *Modelo de Teste*

A principal pergunta do modelo é: "Quanto o modelo erra em suas previsões?"

A lógica do Modelo de Teste é: Terá 4 etapas, onde teremos 4 modelos de treinos, onde o primeiro contará com as semanas de 1 a 147 e vamos prever a semana 148, comparar os valores da semana prevista com a semana de dado bruto usando a função MAPE() - Perda média de regressão de erro percentual absoluto.

O segundo modelo contará com as semanas de 1 a 146, vamos prever as semanas 147 e 148, comparar os valores das semanas previstas com as semanas com os dados brutos correspondentes usando a função MAPE().

Seguimos essa lógica ate o quarto e último modelo de treino comparando as semanas 145 a 148 e calcularemos seus MAPEs.

O erro percentual absoluto médio (MAPE) é uma medida estatística de quão preciso é um sistema de previsão. Ele mede essa precisão como uma porcentagem, e pode ser calculado como o erro percentual absoluto médio para cada período de tempo menos valores reais divididos por valores reais.

**Etapa 1** - Prevendo a semana 148 e comparando com o valor real do banco de dados

```
#Prevendo a Última Semana
ts.train <-ts(dados[1:147,],
              start=c(2018,1,1),
              frequency = 52)

train <- tslm(Receita ~ season,
              data = ts.train)

forecast(train, h=1)
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2020.827       526964.7 294837.8 759091.6 169877.7 884051.7
```

```
e1 = MAPE(526964.7, 531238.9) # Semana 4 - 26/10/2020
```

**Etapa 2** - Prevendo as semanas 147 e 148 e comparando com o valores reais do banco de dados

```
#Prevendo as 2 últimas Semanas
ts.train <-ts(dados[1:146,],
              start=c(2018,1,1),
              frequency = 52)

train <- tslm(Receita ~ season,
              data = ts.train)

forecast(train, h=2)
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2020.808       526478.3 293104.6 759852.0 167450.8 885505.8
## 2020.827       526964.7 293591.0 760338.4 167937.2 885992.2
```

```r
e2 = MAPE(526478.3, 519165.6) #Semana 3 -19/10/2020

e3 = MAPE(526964.7, 531238.9) #Semana 4 - 26/10/2020
```

**Etapa 3** - Prevendo as semanas 146 a 148 e comparando com os valores reais do banco de dados

```r
#Prevendo as 3 últimas Semanas
ts.train <-ts(dados[1:145,],
              start=c(2018,1,1),
              frequency = 52)

train <- tslm(Receita ~ season,
              data = ts.train)

forecast(train, h=3)
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2020.788       542231.8 307614.1 776849.5 181267.5 903196.1
## 2020.808       526478.3 291860.6 761096.0 165514.0 887442.6
## 2020.827       526964.7 292347.0 761582.4 166000.4 887929.0
```

```r
e4 = MAPE(542231.8, 516503.7) #Semana 2  - 12/10/2020

e5 = MAPE(526478.3, 519165.6) #Semana 3  - 19/10/2020

e6 = MAPE(526964.7, 531238.9) #Semana 4  - 26/10/2020
```

**Etapa 4** - Prevendo as semanas 145 a 148 e comparando com os valores reais do banco de dados

```r
#Prevendo as 4 últimas Semanas
ts.train <-ts(dados[1:144,],
              start=c(2018,1,1),
              frequency = 52)

train <- tslm(Receita ~ season,
              data = ts.train)

forecast(train, h=4)
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2020.769       479803.6 243947.9 715659.4 116910.8 842696.5
## 2020.788       542231.8 306376.1 778087.5 179338.9 905124.7
## 2020.808       526478.3 290622.6 762334.0 163585.4 889371.2
## 2020.827       526964.7 291109.0 762820.4 164071.8 889857.6
```

```r
e7 = MAPE(479803.6 , 443026.8) #Semana 1 - 05/10/2020

e8 = MAPE(542231.8, 516503.7) #Semana 2 - 12/10/2020
```

```r
e9 = MAPE(526478.3, 519165.6) #Semana 3 - 19/10/2020

e10 = MAPE(526964.7, 531238.9) #Semana 4 - 26/10/2020
```

E por fim, quanto o modelo erra, em média, nas suas previsões?

```r
mape <- c(e1,e2,e3,e4,e5,e6,e7,e8,e9,e10)

#Erro médio das minhas previsões é de 2.6%
mean(mape)
```

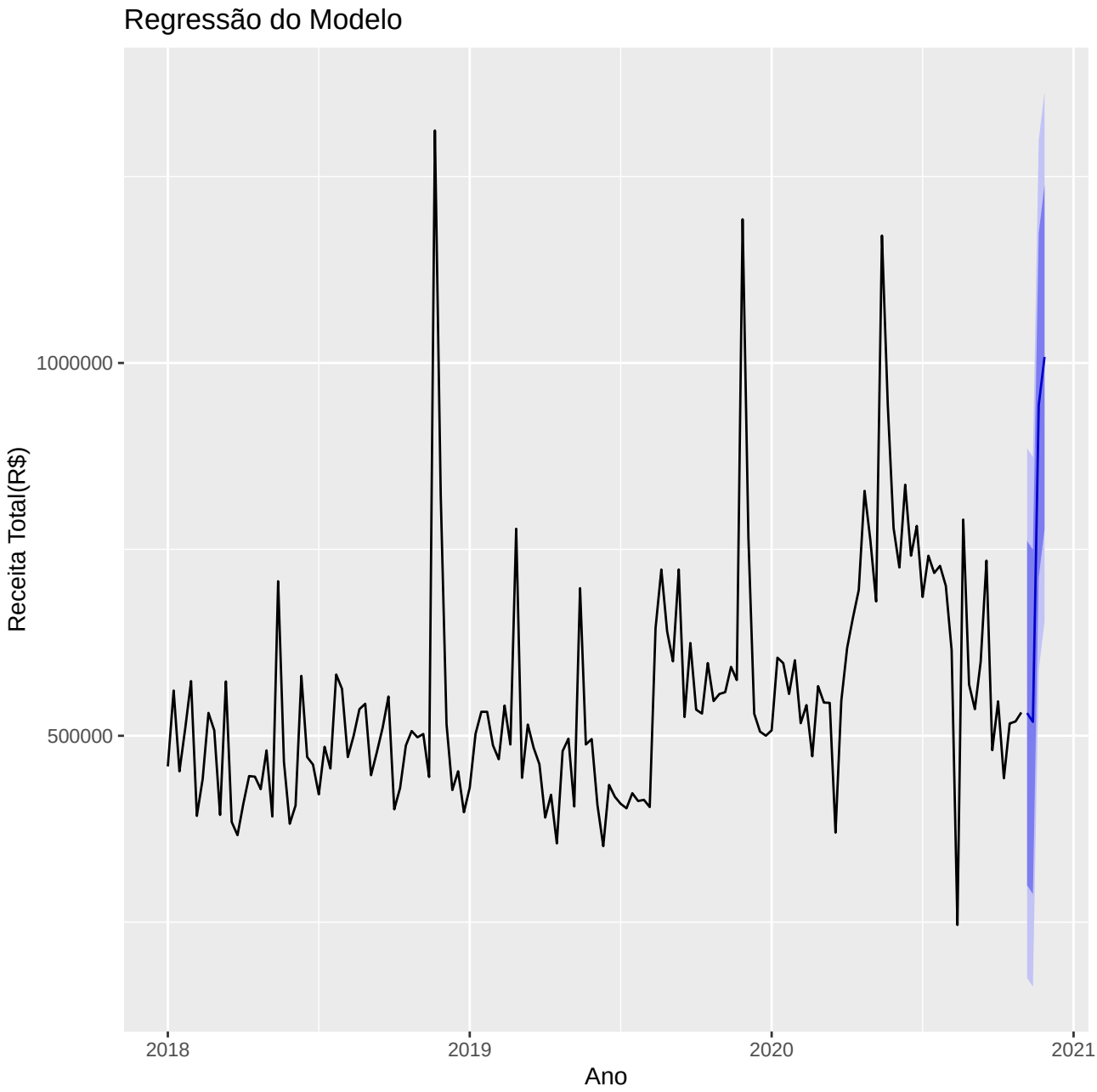
```
## [1] 0.0257076
```

O modelo erra em média 2.6% em suas previsões.

## *Modelo Final*

Vamos ver, graficamente, a previsão do modelo para as 4 próximas semanas

```r
fcst.R <- forecast(fit4, h=4)

autoplot(fcst.R, xlab="Ano",
     ylab="Receita Total(R$)",
     main="Regressão do Modelo")
```




Verificando o valor que está plotado no gráfico acima

```
fcst.R
```

```
##          Point Forecast    Lo 80     Hi 80    Lo 95     Hi 95
## 2020.846       530440.6 299542.0  761339.2 175265.0  885616.2
## 2020.865       518608.5 287709.9  749507.1 163432.9  873784.1
## 2020.885       943263.7 712365.1 1174162.3 588088.1 1298439.3
## 2020.904      1008080.4 777181.8 1238979.0 652904.8 1363256.0
```

Só temos um problema com a sazonalidade, pois a série ta prevendo dois dias de alto investimento de mídia correspondentes as duas últimas semanas do mês de novembro, e sabemos que o cliente fará alto investimento na semana da *Black Friday*.

Desejo continuar com o modelo fit4, mas precisamos fazer algum ajuste.

Há duas possibilidades, a primeira fazer uma variável *dummy* indicando quais dias são *Black Friday* - alternativa sofisticada - podendo criar *dummy* para cada tipo de feriado, como Natal, Páscoa, Dia das Mães.

E outra, apoiando-me no gráfico de Receita Total Separada por Ano, trocar os valores de investimento da semana 47 com a semana 48 do ano de 2018 - visto que para responder nosso Case é uma alternativa viável.

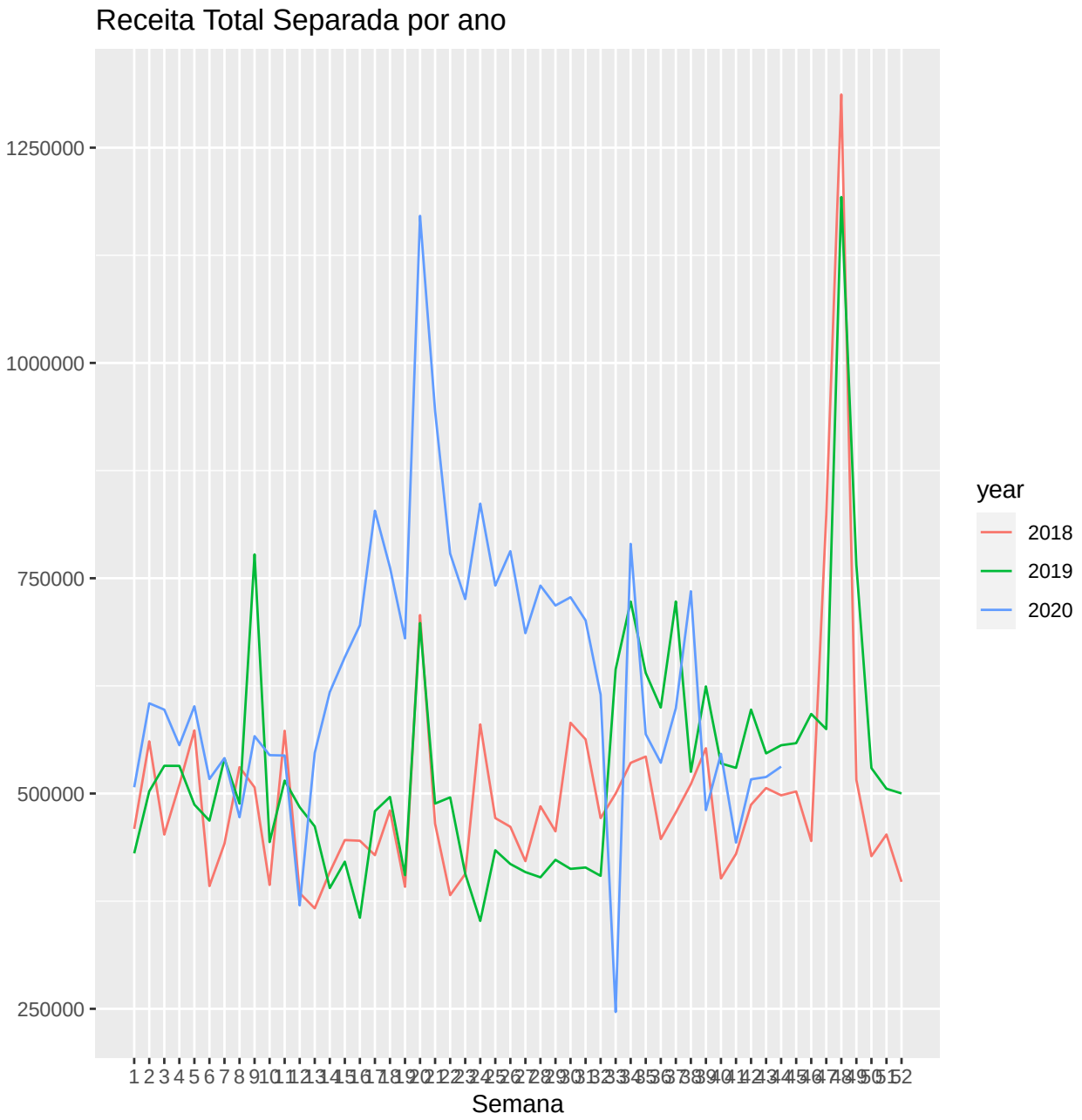
Usarei a segunda opção pois é a ferramenta que estou mais confortável de aplicar. Com isso, o nosso modelo ajustado de Teste manterá o erro médio próximo de 2.6% para cada previsão.

# Modelo Final Ajustado

```
#Trocando os valores
dados1 <- dados
dados1[47,4] <- 823460.9
dados1[48,4] <- 1311737.6


ts = ts(dados1,
        start=c(2018, 1),
        frequency = 52)

R.ts = ts(dados1[,4],
          start = c(2018,1),
          frequency = 52)
```

Perceba que agora possuímos picos de investimento nas semanas 48 dos anos 2018 e 2019.

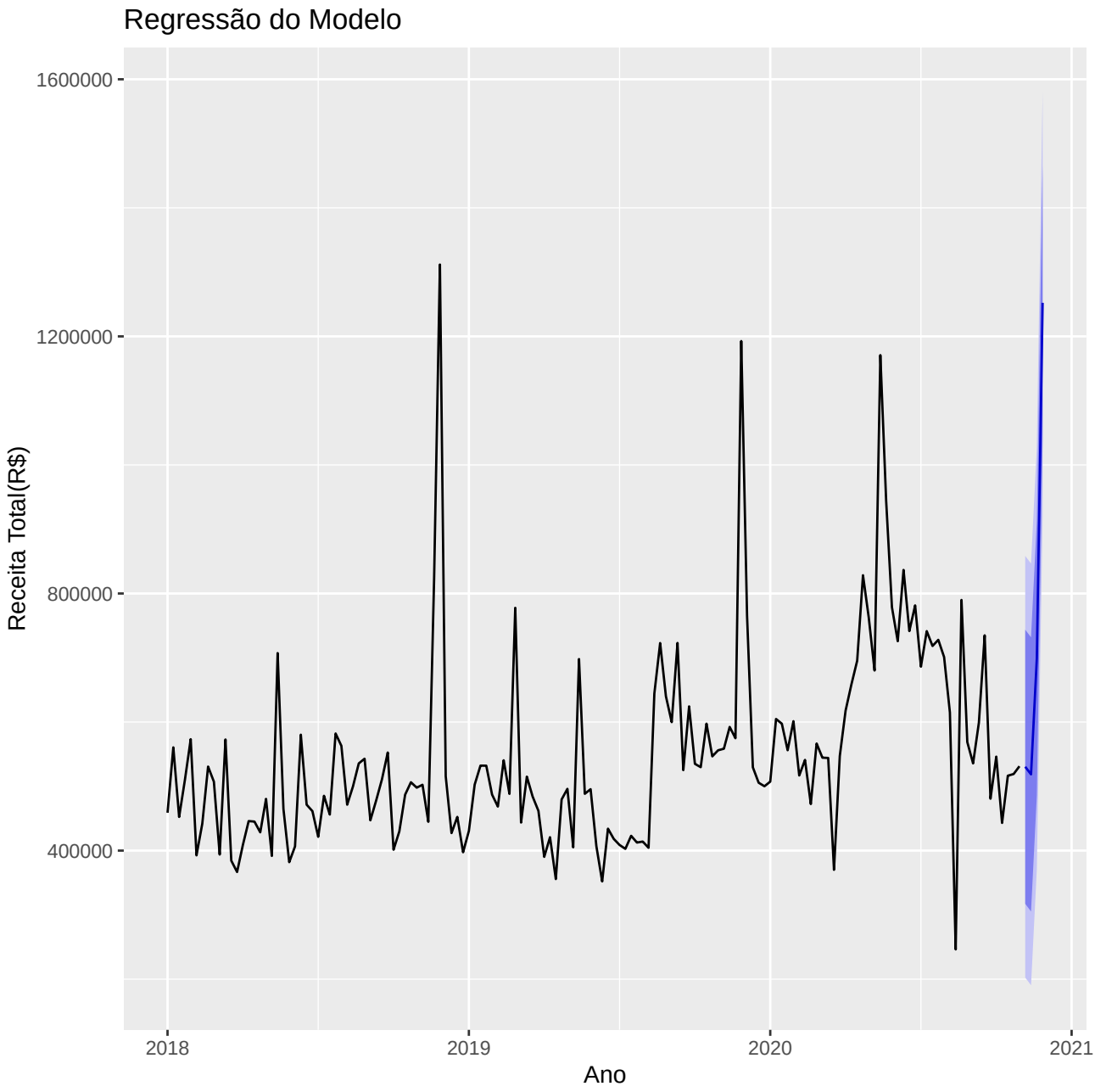
```
#Plot das Séries da Receita Total Separada por Ano

ggseasonplot(R.ts, xlab="Semana",
             main="Receita Total Separada por ano")
```




```
fit4 <- tslm(Receita ~ season,
             data = ts)

fcst.R <- forecast(fit4, h=4)
```

Vamos ver graficamente a previsão do modelo para as 4 próximas semanas

```
autoplot(fcst.R, xlab="Ano",
     ylab="Receita Total(R$)",
     main="Regressão do Modelo")
```




Verificando o valor que está plotado no gráfico acima

```
fcst.R
```

```
##          Point Forecast     Lo 80     Hi 80    Lo 95     Hi 95
## 2020.846       530440.6  317217.8  743663.3 202454.5  858426.6
## 2020.865       518608.5  305385.8  731831.3 190622.5  846594.6
## 2020.885       699125.4  485902.6  912348.1 371139.3 1027111.4
## 2020.904      1252218.8 1038996.0 1465441.5 924232.7 1580204.8
```

```
#plot(fcst.R)
```

Com isso conseguimos prever os investimentos semanal da Receita Total do cliente referente ao mês de novembro de 2020.

Na **semana 1** referente ao dia **02/11/2020** teremos o investimento médio de *R$ 530 440,60*.

Na **semana 2** referente ao dia **09/11/2020** teremos o investimento médio de *R$ 518 608,50*.

Na **semana 3** referente ao dia **16/11/2020** teremos o investimento médio de *R$ 699 125,40*.

Na **semana 4** referente ao dia **23/11/2020** - na semana da *Black Friday* - teremos o investimento médio de *R$ 1 252 218,80*.

***Para o Cliente essa seria a minha resposta final, e apontaria gráficos semelhantes a Receita Total Separada por Ano para mostrar de onde tirei essas estimativas***

## *Adendo*

Houve a tentativa de utilizar uma variável *dummy* indicando os dias das *Black Friday*. Creio que não é viável fazer o método acima frequentemente. (*E.g.* o cliente investe fortemente na *Black Friday*, Natal e Dia dos Pais, o ajuste que fiz anteriormente não é sofisticado. O caminho certo é fazer variáveis *dummy*s para cada feriado).

Acho válido expôr minhas tentativas com a variável *dummy* para que, futuramente, podemos discutir caminhos viáveis e comentar possíveis equívocos.

***Variável Black Friday - BF***

Crio mais um data.frame e adiciono a variável BF com colunas de 0 (zeros) e depois indico quais semanas ocorreram o investimento do cliente com o valor 1 - já indico que esses valores são categóricos.

```
dados2<- dados
dados2["BF"] <- rep('0', 148)
dados2[47,5] <- '1'
dados2[100,5]<- '1'
```

Adicionando as 4 semanas seguintes, o foco é tentar usar a variável BF de alguma forma e dizer pro modelo de predição que a última semana de Novembro de 2020 é *Black Friday*.

```
linha <- data.frame(A=c(0, 0, 0, 0),
B=c(0, 0, 0, 0),
C=c(0, 0, 0, 0),
Receita=c(0, 0, 0, 0),
BF=c('0','0','0','1'))


dados2 <- rbind(dados2, linha)
```

Série Temporal com as variável Receita e BF

```
#ST com a variável BF
ts = ts(dados2[,c(4,5)],
        start=c(2018, 1),
        frequency = 52)
```

Nesse passo paira a seguinte pergunta: "Como que digo pro modelo que a previsão da Quarta semana é a Semana do *Black Friday*?" Usando o código abaixo digo que quero usar os dados das semanas 1 a 148. Mas como digo para manter as informações da variável BF?

Como em dados2 temos 152 semanas, digo para ir ate a semana 148, sendo a semana 44 de 2020. É por aqui que meu racicíciono para, pois eu quero usar a informação da variável BF de algum jeito e não vejo como.

```
#janela de dados da semana 1 a semana 148
ts.w <- window(ts, end = c(2020,44))
```

Quero ver o comportamento das previsões somente com a variável Receita e BF

```
#Vamos ver o comportamento da ST para as 4 semanas

fcst<-forecast(ts.w, h=4)
fcst
```

```
## Receita
##          Point Forecast    Lo 80     Hi 80    Lo 95     Hi 95
## 2020.846       570941.8 448815.7  693068.0 384166.0  757717.7
## 2020.865       557225.8 430745.4  683706.2 363790.8  750660.9
## 2020.885       976735.7 846038.2 1107433.2 776851.1 1176620.3
## 2020.904      1043335.4 908545.0 1178125.8 837191.3 1249479.5
##
## BF
##          Point Forecast     Lo 80    Hi 80    Lo 95    Hi 95
## 2020.846       1.000010 0.8946808 1.105339 0.838923 1.161097
## 2020.865       1.000010 0.8946808 1.105339 0.838923 1.161097
## 2020.885       1.496529 1.3911996 1.601858 1.335442 1.657616
## 2020.904       1.503488 1.3981592 1.608817 1.342401 1.664575
```

E depois visualisar o comportamento em fit4

```
#Usando o Modelo fit 4
fit4 <- tslm(Receita ~ season, data = ts.w)
CV(fit4)
```

```
##                  CV                 AIC                AICc                 BIC
## 35824767315.8067093        3561.9752317        3622.8688487        3720.8274822
##               AdjR2
##           0.1099079
```

```
fcst<-forecast(fit4, h=4)
fcst
```

```
##          Point Forecast    Lo 80     Hi 80    Lo 95     Hi 95
## 2020.846       530440.6 299542.0  761339.2 175265.0  885616.2
## 2020.865       518608.5 287709.9  749507.1 163432.9  873784.1
## 2020.885       943263.7 712365.1 1174162.3 588088.1 1298439.3
## 2020.904      1008080.4 777181.8 1238979.0 652904.8 1363256.0
```

Perceba que o problema com a sazonalidade parecida com o Modelo Final "Normal" - para os dois últimos modelos - permanece, ou seja, a série ta prevendo dois dias de alto investimento de mídia correspondentes as duas últimas semanas do mês de novembro. E esse quadro não é o ideal.