PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA
Pós-graduação *Lato Sensu* em Ciência de Dados e Big Data

Evandro Ferreira de Avellar

PREVISÃO DE FLUXO DE CAIXA PARA INVESTIMENTO DE VALORES DO FGTS

Brasília
2023

Evandro Ferreira de Avellar

PREVISÃO DE FLUXO DE CAIXA PARA INVESTIMENTO DE VALORES DO FGTS

Trabalho de Conclusão de Curso apresentado ao Curso de Especialização em Ciência de Dados e Big Data como requisito parcial à obtenção do título de especialista.

Brasília
2023

SUMÁRIO

1. Introdução

1.1. Contextualização

A CAIXA ECONÔMICA FEDERAL é o agente operador do FGTS, como agente operador é responsável na prestação de serviços financeiros para controle de saldo, entradas e saídas, possibilitando a aplicação dos recursos do fundo em várias carteiras de investimento de curto, médio e longo prazo.

O resultado do FGTS consiste na diferença entre as receitas (rendas/rendimentos com operações de crédito, com títulos públicos federais e demais títulos e valores mobiliários, entre outras) e as despesas (remuneração das contas vinculadas, de TR + 3% ao ano, taxa de administração e outras).[1]

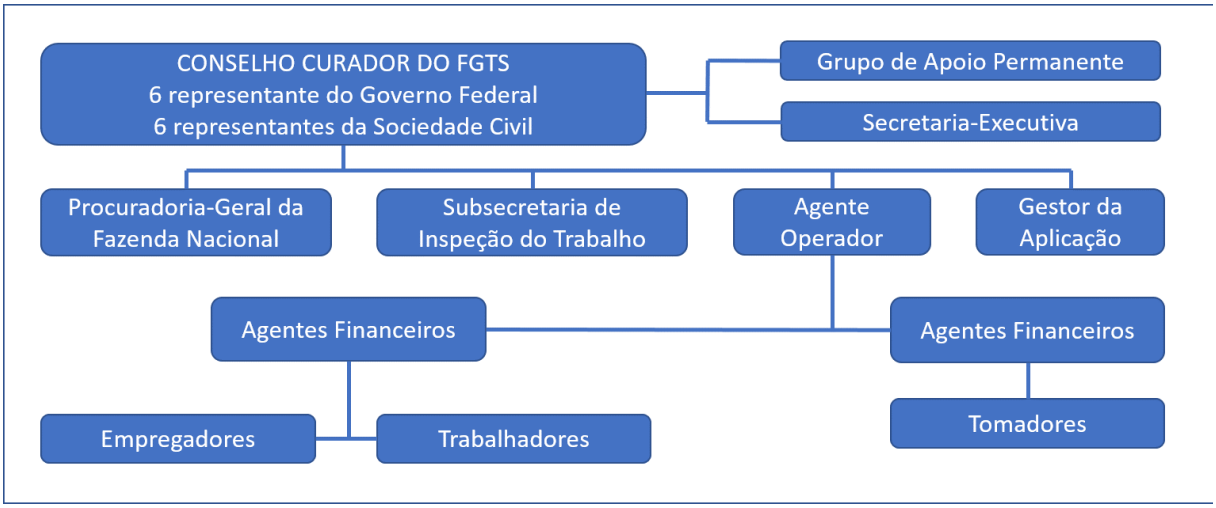


Figura 1: Administração do FGTS

1.2. O problema proposto

É um problema importante para sociedade brasileira pois o recurso do FGTS nasceu com o objetivo de garantir ao trabalhador uma indenização pelo tempo de serviço nos casos de demissão sem justa causa e ainda propiciar a formação de uma reserva a ser utilizada por ele, quando de sua aposentadoria, ou por seus dependentes, quando do seu falecimento.

O FGTS é também uma fonte de recursos para financiamento de programas habitacionais, de saneamento básico e de infraestrutura urbana.

[1]Site do FGTS

Os dados analisados são públicos e estão disponíveis no site do FGTS, o trabalho do TCC estará limitado às informações do site.

O objetivo desse trabalho é encontrar um modelo preditivo utilizando técnicas de modelos estatísticos e de *machine learning* com assertividade e embasamento teórico para apresentar ao Agente Operador do Fundo, permitindo gerar previsões de entradas e saídas de fluxo de caixa.

O período dos dados que serão analisados é de 01/1997 a 12/2022. Durante os trabalhos pode ser necessário restringir um período menor mais recente para maior assertividade nas previsões.

1.3. Objetivos

Encontrar um modelo de previsão de Fluxo de Caixa utilizando técnicas de *Machine Learning* ou modelos estatísticos para permitir aos gestores do fundo planejarem os investimentos de curto, médio e longo prazo com os recursos do FGTS, garantindo a liquidez dos valores necessários para cumprir as obrigações sem perder o melhor retorno com os investimentos.

1.4. Conceitos que serão utilizados

a. **Séries temporais:** Uma série temporal é uma sequência de pontos ordenados de forma cronológica. Normalmente, a série temporal possui uma sequência de dados equidistantes no tempo. A análise de séries temporais é realizada com o intuito de explorar o comportamento passado e também de prever o comportamento futuro em um determinado problema.

b. **Previsão:** a aplicação mais comum quando tratamos de séries temporais é a previsão de valores futuros (*forecast*). A previsão de série temporal é o processo de usar um modelo estatístico para prever valores futuros de uma série temporal com base em resultados anteriores.

c. **Tendência**: a tendência mostra uma direção geral dos dados da série temporal durante um longo período de tempo. Uma tendência pode ser crescente (para cima), decrescente (para baixo) ou horizontal (estacionária).

d. **Sazonalidade**: o componente de sazonalidade exibe uma tendência que se repete em relação ao tempo, direção e magnitude. Alguns exemplos incluem um aumento no consumo de água no verão devido às condições climáticas quentes ou um aumento no número de passageiros de companhias aéreas durante as férias a cada ano.

e. **Variação irregular**: são as flutuações nos dados da série temporal que se tornam evidentes quando a tendência e as variações cíclicas são removidas. Essas variações são imprevisíveis, erráticas e podem ou não ser aleatórias.

f. **Componente cíclico:** Estas são as tendências sem repetição definida em um determinado período de tempo. Esses ciclos não apresentam variação sazonal, mas geralmente ocorrem em um período de 3 a 12 anos, dependendo da natureza da série temporal.

g. **Decomposição ETS A decomposição:** ETS é usada para separar diferentes componentes de uma série temporal. O termo ETS significa erro, tendência e sazonalidade. (SAMISHAWL, 2020).

2. Coleta de Dados

Todos os dados são públicos e foram obtidos no site do FGTS na seção Números do FGTS / Passivo: https://www.fgts.gov.br/Pages/numeros-fgts/passivo-fgts.aspx extraídos no dia 03 de janeiro de 2023.

Dois datasets foram utilizados no estudo:

- ArrecadaçãoConsolidada.xls
- SaquesConsolidados.xls

Originalmente os dados foram coletados com as seguintes características, antes do tratamento dos dados e logo após a importação para o notebook:

2.1 Arquivo ArrecadaçãoConsolidada.xls

Nome da coluna	Descrição	Tipo
Mês/Ano	Mês/Ano de arrecadação no formato mm/aaaa	String
QUANTIDADE(GUIAS)	Quantidade de guias referentes ao Mês/Ano	Número inteiro
VALOR (R$)	Valor total arrecadado no Mês/Ano	Float

2.2 Arquivo SaquesConsolidados.xls

Nome da coluna	Descrição	Tipo
MÊS/ANO	Mês/Ano do saque no formato mm/aaaa	String
QUANTIDADE	Quantidade de saques referentes ao Mês/Ano	Número inteiro
VALOR (R$)	Valor total de saques no Mês/Ano	Float

3. Processamento/Tratamento de Dados

A IDE utilizada para realização das análises é o Jupyter-Notebook.

Para processamento e tratamento dos dados será utilizada a biblioteca Pandas. A importação pode ser feita através do código: `import pandas as pd`.

Os arquivos disponibilizados no site estão no formado xls, na leitura e conversão para o Data Frame com nome *arrec,* foi preciso desconsiderar as primeiras linhas do arquivo excel pois se tratavam de informações explicativas sobre o conteúdo. A exclusão das linhas do cabeçalho foi realizada com o parâmetro **header** do método `pd.read_excel()`.

Os parâmetros `index_col` e `parse_dates` definiu a coluna “Mês/Ano” como índice transformando-o no formato de data.

```
arrec = pd.read_excel('dataset/ArrecadaçãoConsolidada.xls', header=7, index_col='Mês/Ano',parse_dates=True)
arrec.index.freq = 'MS'
```

```
arrec
```

Mês/Ano	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	QUANTIDADE(GUIAS)	Unnamed: 8	VALOR (R$)
01/1997	NaN	NaN	NaN	NaN	NaN	NaN	1910996	NaN	1.355415e+09
02/1997	NaN	NaN	NaN	NaN	NaN	NaN	1864867	NaN	1.012064e+09
03/1997	NaN	NaN	NaN	NaN	NaN	NaN	1955507	NaN	9.969265e+08
04/1997	NaN	NaN	NaN	NaN	NaN	NaN	2028810	NaN	1.013695e+09
05/1997	NaN	NaN	NaN	NaN	NaN	NaN	2010079	NaN	1.021931e+09
...	...	...	...	...	...	...	...	...	...
09/2022	NaN	NaN	NaN	NaN	NaN	NaN	6170799	NaN	1.262671e+10
10/2022	NaN	NaN	NaN	NaN	NaN	NaN	6242096	NaN	1.261092e+10
11/2022	NaN	NaN	NaN	NaN	NaN	NaN	6243593	NaN	1.280695e+10
12/2022	NaN	NaN	NaN	NaN	NaN	NaN	6152530	NaN	1.646719e+10
Total	NaN	NaN	NaN	NaN	NaN	NaN	1362656759	NaN	1.818875e+12

313 rows × 9 columns

A última linha do *dataset* também continha informações de totalização, que não seriam úteis ao estudo. A remoção se deu através do código abaixo:

```
arrec = arrec.head(312)
arrec
```

Mês/Ano	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	QUANTIDADE(GUIAS)	Unnamed: 8	VALOR (R$)
01/1997	NaN	NaN	NaN	NaN	NaN	NaN	1910996	NaN	1.355415e+09
02/1997	NaN	NaN	NaN	NaN	NaN	NaN	1864867	NaN	1.012064e+09
03/1997	NaN	NaN	NaN	NaN	NaN	NaN	1955507	NaN	9.969265e+08
04/1997	NaN	NaN	NaN	NaN	NaN	NaN	2028810	NaN	1.013695e+09
05/1997	NaN	NaN	NaN	NaN	NaN	NaN	2010079	NaN	1.021931e+09
...	...	...	...	...	...	...	...	...	...
08/2022	NaN	NaN	NaN	NaN	NaN	NaN	6285605	NaN	1.290656e+10
09/2022	NaN	NaN	NaN	NaN	NaN	NaN	6170799	NaN	1.262671e+10
10/2022	NaN	NaN	NaN	NaN	NaN	NaN	6242096	NaN	1.261092e+10
11/2022	NaN	NaN	NaN	NaN	NaN	NaN	6243593	NaN	1.280695e+10
12/2022	NaN	NaN	NaN	NaN	NaN	NaN	6152530	NaN	1.646719e+10

312 rows × 9 columns

Foram selecionadas as colunas que continham os dados, descartando as colunas que ficaram vazias no Data Frame, através do código a seguir:

```
df_arrec = arrec[['QUANTIDADE(GUIAS)', 'VALOR (R$)']]
df_arrec.head()
```

Mês/Ano	QUANTIDADE(GUIAS)	VALOR (R$)
01/1997	1910996	1.355415e+09
02/1997	1864867	1.012064e+09
03/1997	1955507	9.969265e+08
04/1997	2028810	1.013695e+09
05/1997	2010079	1.021931e+09

Todos esses passos para as informações dos registros de arrecadação também foram seguidos para tratamento do *dataset* de saques alterando alguns parâmetros, e encontram-se registrados no arquivo do jupyter-notebook:

Dados de saques

```
saque = pd.read_excel('dataset/SaquesConsolidados.xls', header=7, index_col='MÊS/ANO',parse_dates=True)
saque = saque.head(310)
df_saque = saque[['QUANTIDADE', 'VALOR (R$)']]
df_saque.head()
```

MÊS/ANO	QUANTIDADE	VALOR (R$)
1997-01-01	1028047	9.792931e+08
1997-02-01	891569	8.404584e+08
1997-03-01	953599	9.850851e+08
1997-04-01	1046439	1.116987e+09
1997-05-01	967636	1.149939e+09

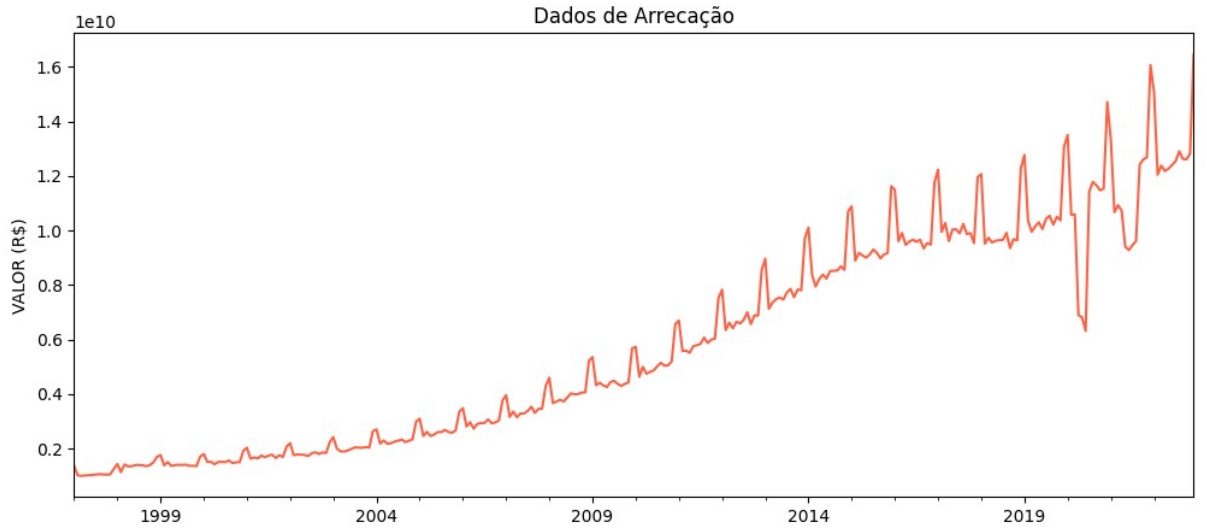
Não foram identificados dados ausentes ou inválidos nas colunas selecionadas, após esses procedimentos.

4. Análise e Exploração dos Dados

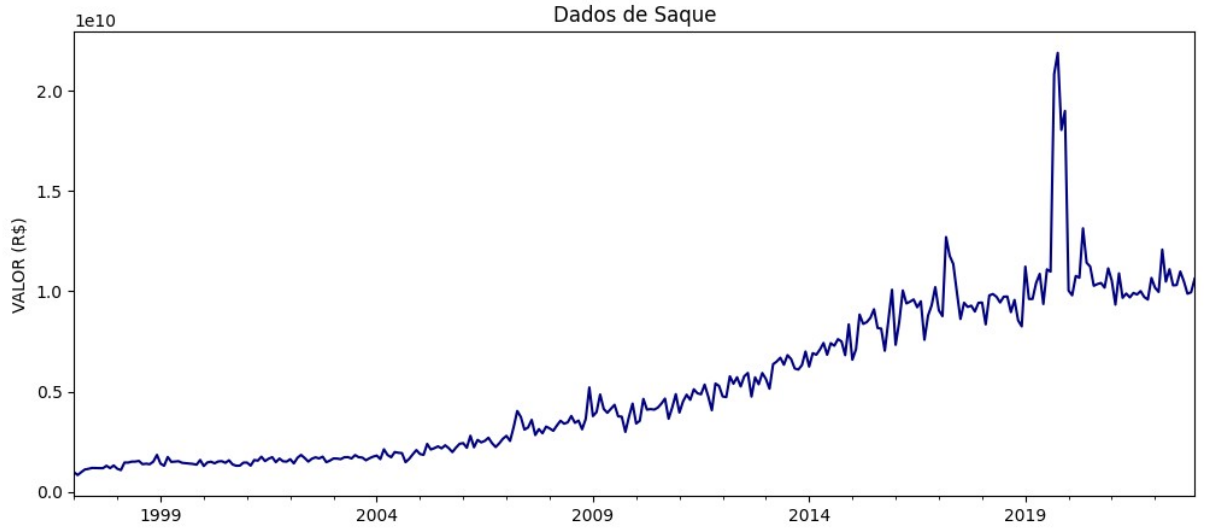
A visualização dos dados auxilia na tomada de decisão para escolher o melhor caminho da análise e exploração. Os dois Data Frames geraram os gráficos abaixo:

4.1 Visualizando os dados dos *datasets* iniciais

```
ax = df_arrec['VALOR (R$)'].plot(figsize=(12,5), title='Dados de Arrecação', color='tomato')
ax.autoscale(axis='x',tight=True)
ax.set(ylabel='VALOR (R$)', xlabel='');
```




```
ax = df_saque['VALOR (R$)'].plot(figsize=(12,5), title='Dados de Saque', color='navy')
ax.autoscale(axis='x',tight=True)
ax.set(ylabel='VALOR (R$)', xlabel='');
```



Através da análise dos gráficos foram descartadas as colunas de quantidade dos dois Data Frames, e as análises serão efetuadas somente da coluna valor.

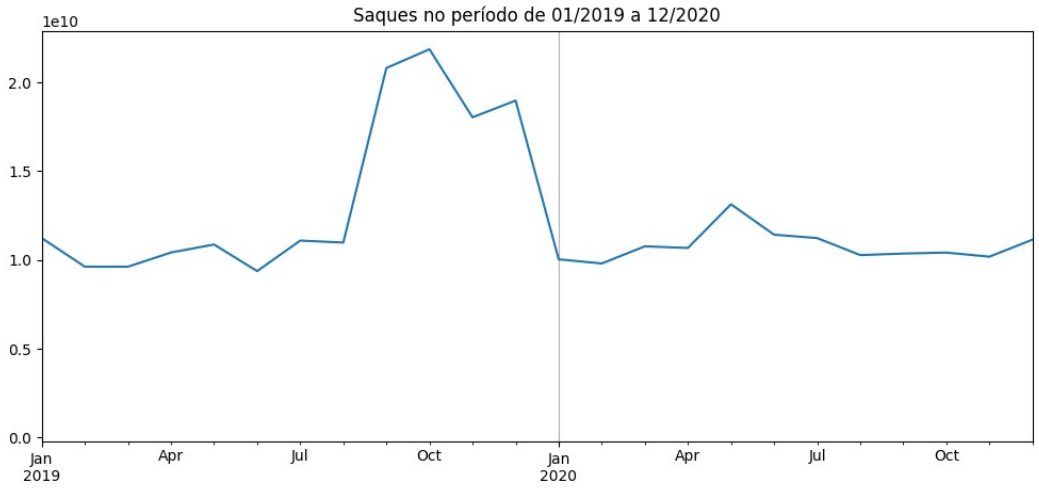
4.2 Identificando comportamentos anômalos

Observando o gráfico de saques é possível verificar um comportamento fora do padrão entre 09/2019 a 01/2020. Ao pesquisar no site do FGTS é possível concluir que esse comportamento foi gerado pelo Saque Imediato (2019/2020).

```
df_saque.idxmax()
```

```
QUANTIDADE   2019-10-01
VALOR (R$)   2019-10-01
dtype: datetime64[ns]
```

```
ax = df_saque['VALOR (R$)'].plot(xlim=['2019-01-01','2020-12-01'], figsize=(12,5))
ax.set(xlabel='Mêses')
ax.set_title('Saques no período de 01/2019 a 12/2020')
ax.xaxis.grid(True)
```

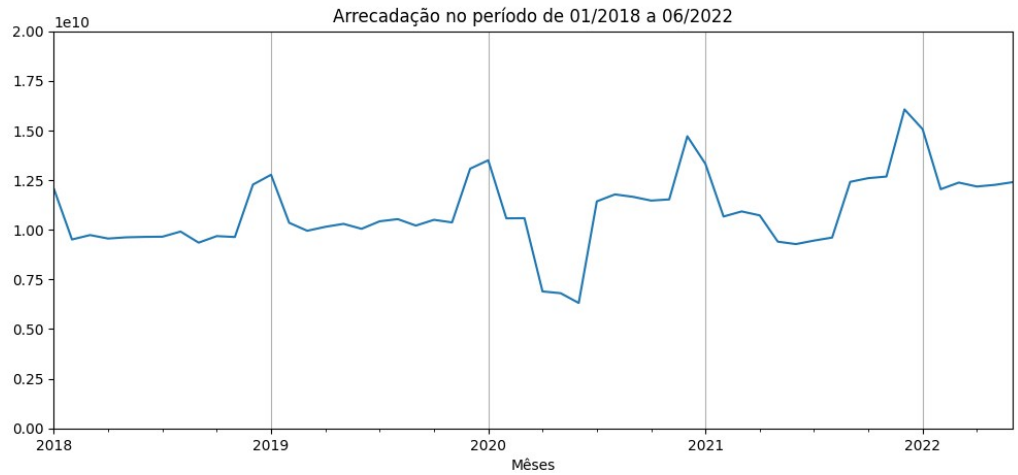


O comportamento da arrecadação também apresenta um comportamento anormal no ano de 2020 com queda expressiva no mês de junho de 2020, o que pode ter sido gerado pelo impacto da pandemia COVID-19 na economia do país.

```
df_arrec['VALOR (R$)']['2018-01-01':'2022-06-01'].idxmin()
```

```
Timestamp('2020-06-01 00:00:00')
```

```
ax = df_arrec['VALOR (R$)'].plot(xlim=['2018-01-01','2022-06-01'], figsize=(12,5), ylim=(0,2e10))
ax.set(xlabel='Mêses')
ax.set_title('Arrecadação no período de 01/2018 a 06/2022')
ax.xaxis.grid(True)
```



4.2.1 Informações sobre o Saque Imediato (2019/2020) que estão no site do FGTS

Os trabalhadores puderam solicitar o Saque Emergencial no período regular de setembro de 2019 a março de 2020.

Para viabilizar os saques, ocorreram os débitos nas contas vinculadas dos trabalhadores, conforme tabela descritiva abaixo:

Mês/ano	Quantidade*	Valor **
Set/19	64.735,6	10.164
Out/19	65.359,9	10.469
Nov/19	51.892,2	8.447
Dez/19	49.666,5	8.568
Jan/20	326,2	76
Fev/20	231,6	54
Mar/20	360,0	91
Abr/20	184,3	45
Total	232.757	37.915

* Quantidade em mil
** Valor em R$ milhão

4.3 Modelos de erro/tendência/sazonalidade

Ao começarmos a trabalhar com dados endógenos e desenvolver modelos de previsão, é útil identificar e isolar fatores que atuam dentro do sistema e influenciam o comportamento. Aqui, o nome "endógeno" considera fatores internos, enquanto "exógeno" se relaciona a forças externas. Esses modelos se enquadram na categoria de modelos de espaço de estado e incluem a decomposição e o alisamento exponencial (descrito em uma seção futura) (MORETTIN; TOLOI, 2006).

A decomposição de uma série temporal tenta isolar componentes individuais, como erro, tendência e sazonalidade (ETS).

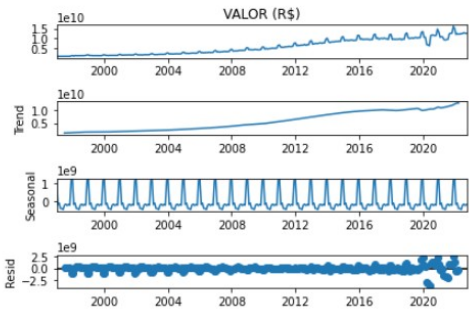
A função `seasonal_decompose` do módulo `statsmodels.tsa.seasonal` é útil para analisar as diferentes componentes de uma série temporal, incluindo a tendência, sazonalidade e os resíduos (ou erro). Através dessa análise, podemos ter uma melhor compreensão do comportamento da série temporal e identificar padrões que possam ser úteis na criação de modelos preditivos. Além disso, a decomposição pode ser usada para remover a sazonalidade e a tendência de uma série, tornando os dados mais estacionários e, assim, permitindo que técnicas estatísticas mais avançadas sejam aplicadas. No entanto, é importante lembrar que a decomposição não é uma técnica de previsão por si só e deve ser usada em conjunto com outros métodos para fazer previsões precisas.

É possível observar que as duas séries temporais possuem tendência e sazonalidade.

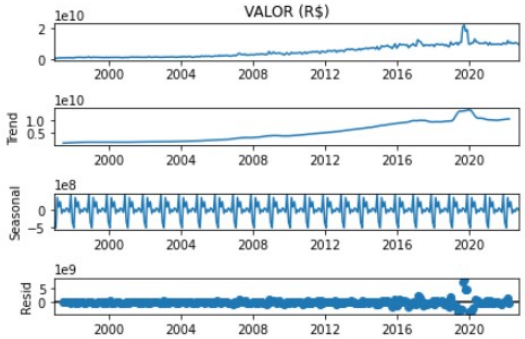
Decomposição ETS

```
from statsmodels.tsa.seasonal import seasonal_decompose
```

```
result = seasonal_decompose(df_arrec['VALOR (R$)'], model='add')
result.plot();
```




```
result = seasonal_decompose(df_saque['VALOR (R$)'], model='add')
result.plot();
```



4.4 Compreendendo a Estacionariedade dos Dados

Muitos modelos estatísticos tradicionais de séries temporais dependem de uma série temporal estacionária. Grosso modo, uma série temporal estacionária é aquela que tem propriedades estatísticas razoavelmente estáveis ao longo do tempo, sobretudo no que diz média e à variância.

Mas a estacionariedade é um conceito enganoso, principalmente quando aplicado a dados de séries temporais reais. Como é intuitivo e nos enganamos com facilidade, confiamos naturalmente nele.

Uma série temporal estacionária é aquela cuja medição de sua série temporal reflete um sistema em um estado estável. As vezes, é difícil afirmar exatamente o que uma série temporal estacionária significa e pode ser mais fácil considerá-la como algo que não fica parado em vez de dizer que é algo estacionário. (NIELSEN, Aileen. 2021)

Para determinar se uma série é estacionária, podemos usar o teste Dickey-Fuller Aumentado. Neste teste, a hipótese nula afirma que ϕ=1 (isso também é chamado de teste de unidade). O teste retorna várias estatísticas. Nosso foco está no *p-value*. Um pequeno valor p(p<0,05) indica forte evidência contra a hipótese nula.

Vamos utilizar a biblioteca `from statsmodels.tsa.stattools import` **`adfuller`** para verificar se as séries são estacionárias.

```
# Dataset Arrecadação

print('Teste Dickey-Fuller - Arrecadação')

df_arrec_test = adfuller(df_arrec['VALOR (R$)'],autolag='AIC')

dfout = pd.Series(df_arrec_test[0:4],index=['ADF teste estatístico','p-value','# lags usados','# observações'])

for key,val in df_arrec_test[4].items():
    dfout[f'valor crítico ({key})']=val
print(dfout)
```

```
Teste Dickey-Fuller - Arrecadação
ADF teste estatístico      1.631739
p-value                    0.997950
# lags usados             15.000000
# observações            296.000000
valor crítico (1%)        -3.452637
valor crítico (5%)        -2.871354
valor crítico (10%)       -2.571999
dtype: float64
```

```
# Dataset de Saques

print('Teste Dickey-Fuller - Saques')

df_saque_test = adfuller(df_saque['VALOR (R$)'],autolag='AIC')

dfout = pd.Series(df_saque_test[0:4],index=['ADF teste estatístico','p-value','# lags usados','# observações'])

for key,val in df_saque_test[4].items():
    dfout[f'valor crítico ({key})']=val
print(dfout)
```

```
Teste Dickey-Fuller - Saques
ADF teste estatístico     -0.446793
p-value                    0.901988
# lags usados             14.000000
# observações            297.000000
valor crítico (1%)        -3.452561
valor crítico (5%)        -2.871321
valor crítico (10%)       -2.571982
dtype: float64
```

Aqui temos dois *p-value* muito altos em 0,998 e 0,902, o que fornece evidências fracas contra as hipóteses nulas e, portanto, falhamos em rejeitar as hipóteses nulas e decidimos que nossos conjuntos de dados não são estacionários.

5. Criação de Modelos

Para comparação, além dos gráficos, será utilizado o método para avaliação de desempenho dos modelos, RMSE.

RMSE (*Root Mean Squared Error*) é a raiz quadrada do MSE e é utilizado para expressar o erro médio de previsão em uma mesma unidade da variável de interesse. É uma medida de dispersão dos erros em relação ao valor real. Quanto menor o valor do RMSE, mais preciso é o modelo de previsão.

MSE (*Mean Squared Error*) é um método de avaliação de desempenho de modelos de previsão que calcula a média dos erros quadrados das previsões em relação aos valores reais. Quanto menor o valor do MSE, melhor é a qualidade das previsões do modelo.

Para compreender o conceito de performance de modelos vamos submeter nossa base a alguns modelos de média móvel e suavização exponencial.

5.1 Média Móvel Simples (SMA)

A aplicação da média móvel simples, ou **SMA** (*Simple Moving Average*), sobre uma sequência $\left(p_i\right)_{i=1}^{m}$ resulta na sequência das médias das subsequências de n elementos da sequência, tal que $n<m,$, e $n,m \in \mathbb{N}$. Desta maneira, dado uma sequência de m elementos $P=\left(p_1,\ldots,p_m\right)$, o cálculo de um termo qualquer da sequência resultante pela média móvel é dado por

$$\bar{p}_i=\frac{p_{i+1}+\ldots+p_{i+n}}{n}=\frac{1}{n}\sum_{j=1}^{n} p_{i+1}$$

$$e\ \bar{p}_{i+1}=\bar{p}_i+\frac{p_{n+i+1}}{n}-\frac{p_{i+1}}{n}.$$

É preciso expandir os extremos de uma sequência de dados por $n\text{-}1$ elementos para aplicar a média móvel a todos os seus elementos; a maneira mais simples de fazer essa expansão é adicionar $n\text{-}1$ zeros aos seus extremos.

A média móvel simples possui 3 formas de ser aplicada: considerando-se os n elementos posteriores, futuros como foi feito acima; os n elementos anteriores, passados (como em séries temporais); ou considerando-se ainda elementos anteriores e posteriores. Neste último caso, apenas $n/2$ elementos devem ser adicionados aos dois extremos. (WIKIPÉDIA)

É possível observar que a média móvel simples já consegue prever com alguma assertividade.

Importante observar que a média móvel é uma técnica simples e pode não ser adequada para todos os tipos de séries temporais. Além disso, é importante lembrar que a média móvel é uma técnica de suavização e não necessariamente faz previsões precisas para o futuro.

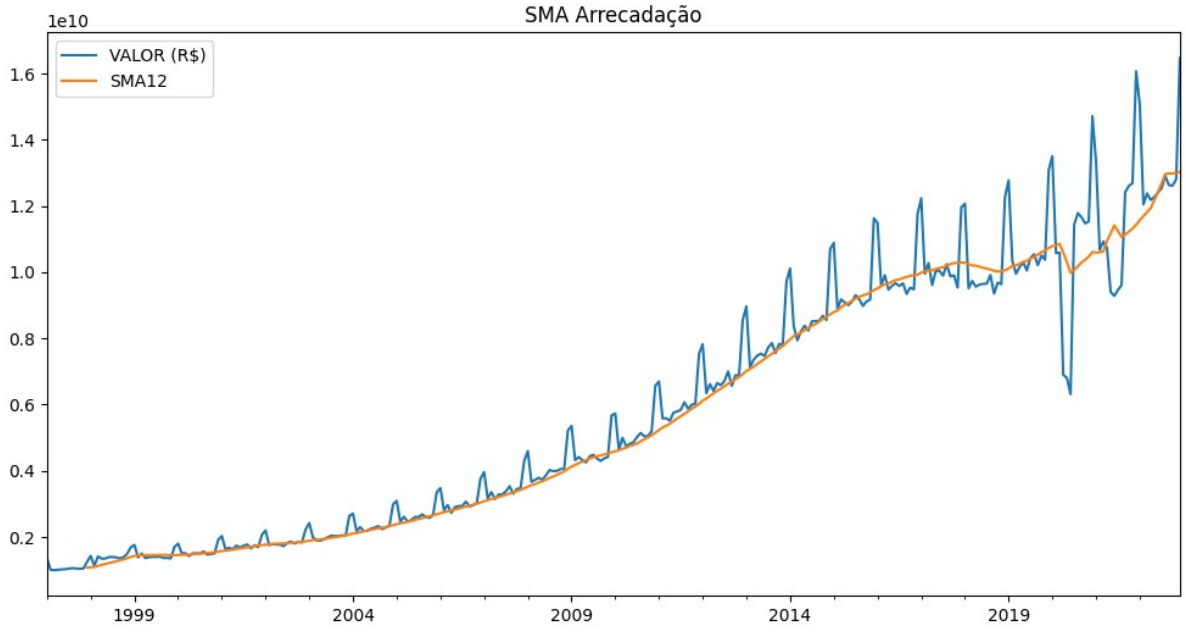
Média Móvel Simples (SMA)

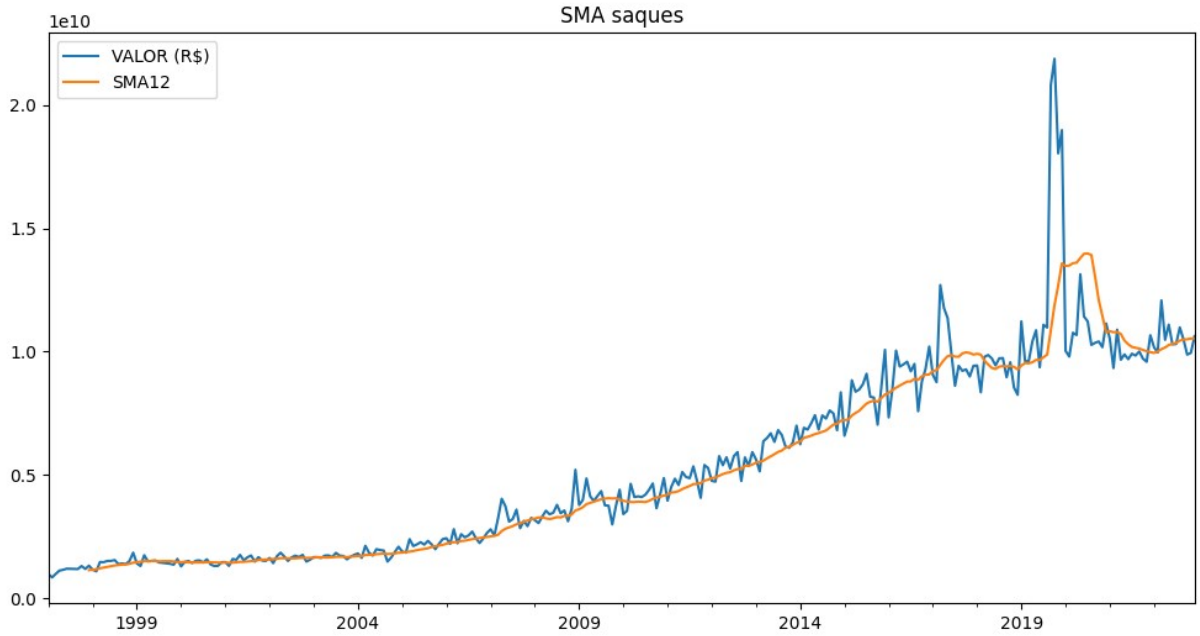
Arrecadação

```
df_arrec['SMA12'] = df_arrec['VALOR (R$)'].rolling(window=12).mean()
```

```
df_arrec
```

Mês/Ano	QUANTIDADE(GUIAS)	VALOR (R$)	SMA12
1997-01-01	1910996	1.355415e+09	NaN
1997-02-01	1864867	1.012064e+09	NaN
1997-03-01	1955507	9.969265e+08	NaN
1997-04-01	2028810	1.013695e+09	NaN
1997-05-01	2010079	1.021931e+09	NaN
...	...	...	...
2022-08-01	6285605	1.290656e+10	1.296263e+10
2022-09-01	6170799	1.262671e+10	1.298019e+10
2022-10-01	6242096	1.261092e+10	1.298088e+10
2022-11-01	6243593	1.280695e+10	1.299129e+10
2022-12-01	6152530	1.646719e+10	1.302479e+10





5.2 Modelo EWMA (Média móvel ponderada exponencialmente)

No item anterior foi mostrado como calcular a SMA com base em uma janela. No entanto, a SMA básica tem algumas fraquezas:

- Janelas menores levarão a mais ruído, em vez de sinal.
- Sempre atrasará pelo tamanho da janela.
- Nunca alcançará o pico ou vale completo dos dados devido à média.
- Realmente não informa sobre possíveis comportamentos futuros, tudo o que realmente faz é descrever tendências em seus dados.
- Valores históricos extremos podem distorcer significativamente sua SMA.

Para ajudar a corrigir algumas dessas questões, podemos usar uma EWMA (Média móvel ponderada exponencialmente).

A Média Móvel Ponderada Exponencialmente (EWMA, do inglês *Exponentially Weighted Moving Average*) é um método utilizado na análise técnica de dados financeiros para calcular uma média móvel ponderada que dá mais peso aos valores mais recentes dos dados analisados. Esse método foi desenvolvido como uma alternativa à Média Móvel Simples (SMA, do inglês *Simple Moving Average*), que pode apresentar algumas limitações em determinadas situações.

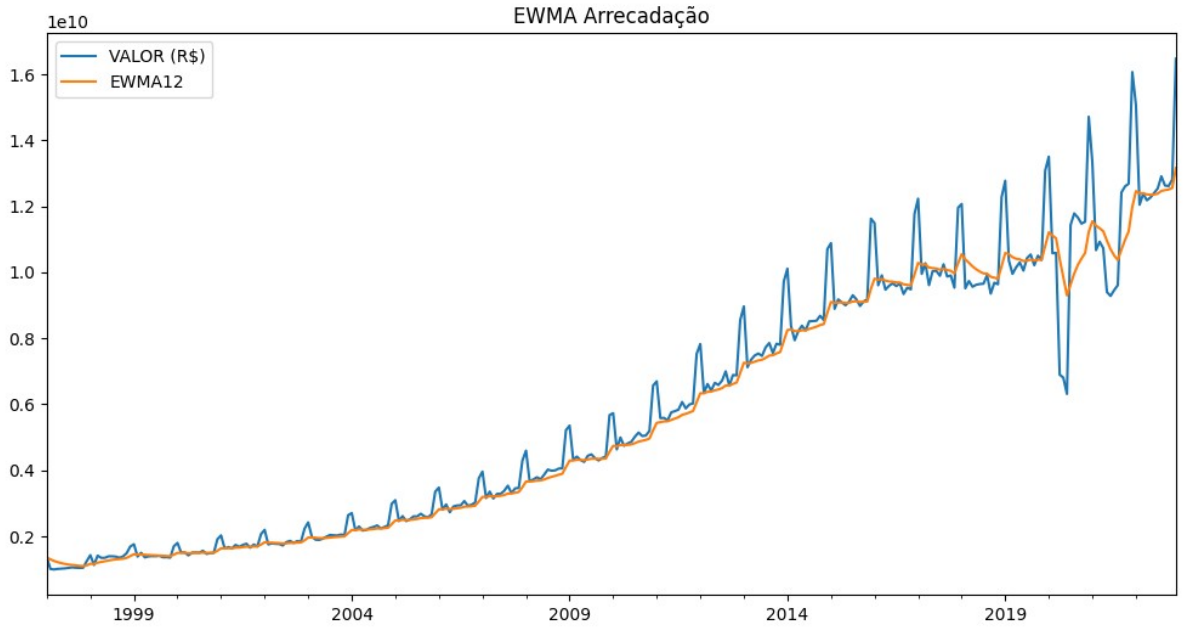
O cálculo da EWMA é feito através de uma fórmula que aplica pesos exponenciais crescentes aos valores mais recentes dos dados, dando mais importância a esses valores em detrimento dos mais antigos. Dessa forma, a EWMA é capaz de suavizar a curva de uma série de dados, reduzindo o ruído e o efeito de atraso que podem ser observados na SMA.

A EMWA enquanto técnica de suavização tem utilidade para previsão de tendências e detecção de mudanças em séries temporais, mas assim como a SMA, não necessariamente faz previsões precisas para o futuro.

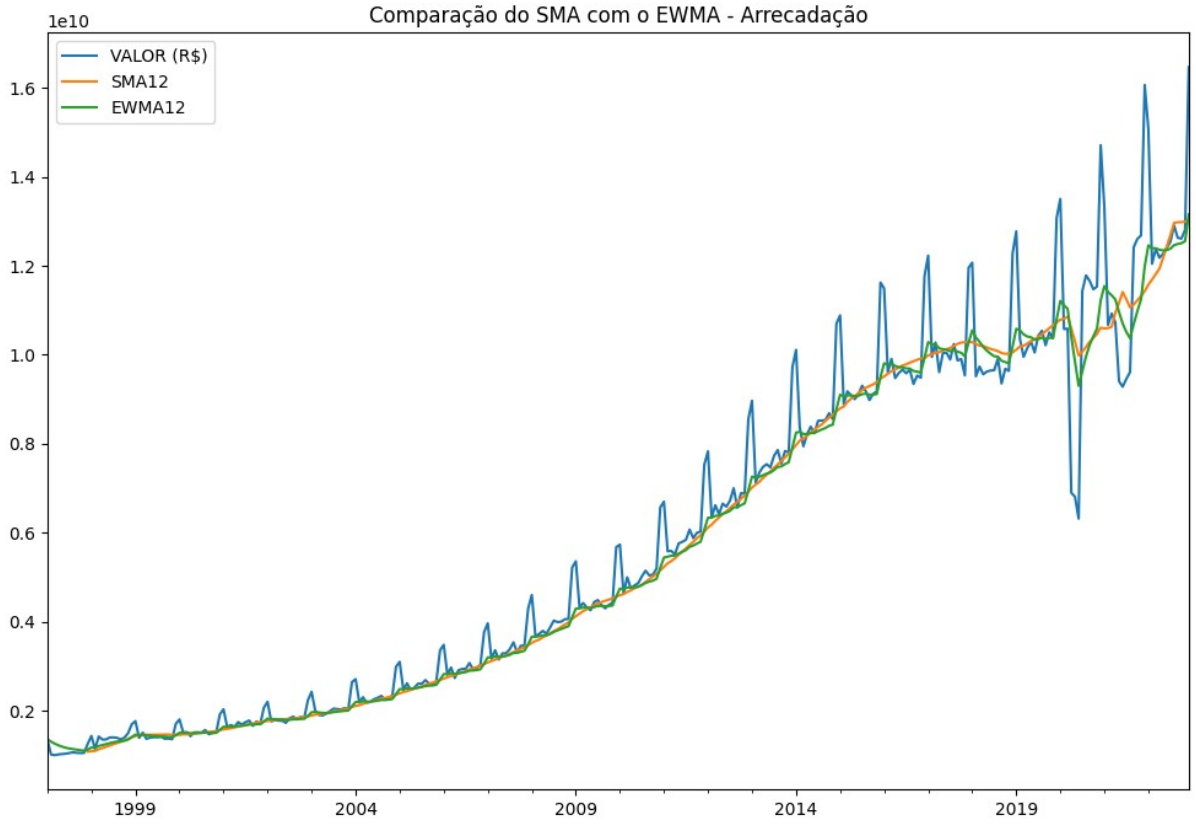
```
df_arrec['EWMA12'] = df_arrec['VALOR (R$)'].ewm(span=12,adjust=False).mean()
```

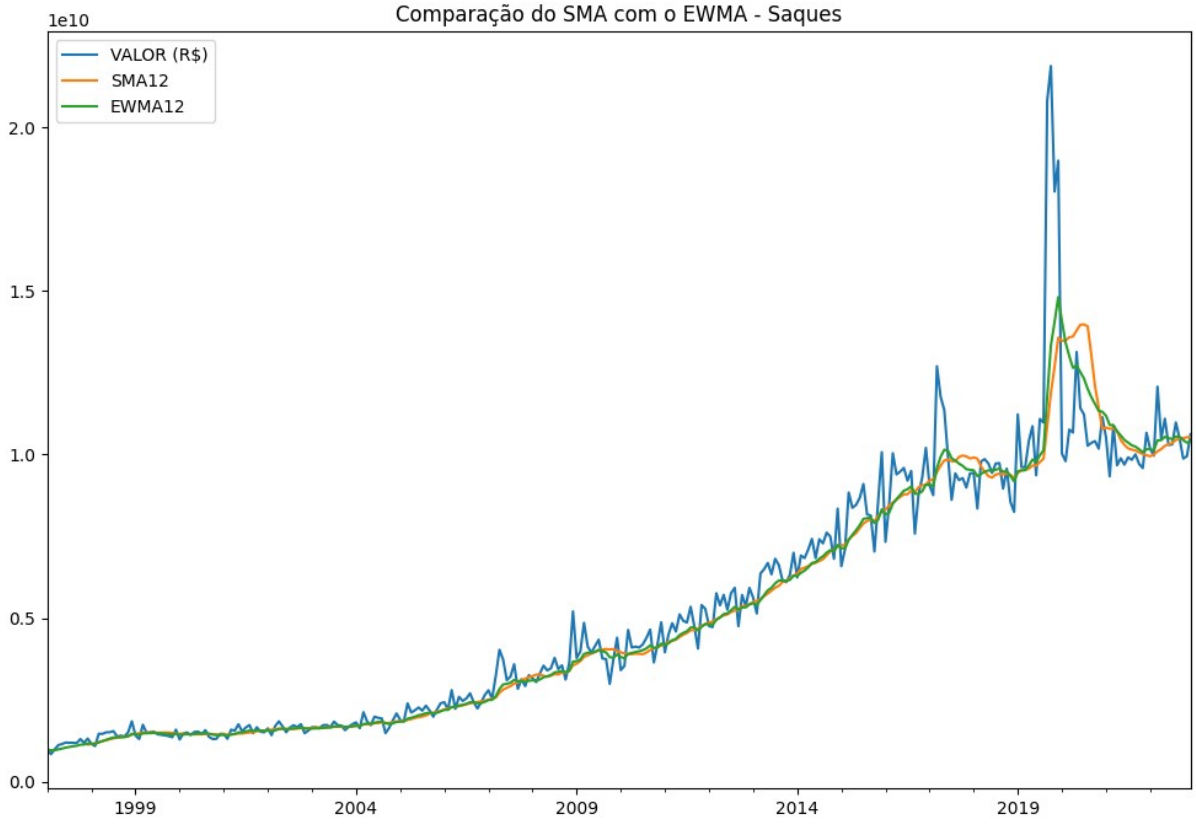
```
df_arrec
```

Mês/Ano	VALOR (R$)	SMA12	EWMA12
1997-01-01	1.355415e+09	NaN	1.355415e+09
1997-02-01	1.012064e+09	NaN	1.302592e+09
1997-03-01	9.969265e+08	NaN	1.255566e+09
1997-04-01	1.013695e+09	NaN	1.218355e+09
1997-05-01	1.021931e+09	NaN	1.188136e+09
...	...	...	...
2022-08-01	1.290656e+10	1.296263e+10	1.246323e+10
2022-09-01	1.262671e+10	1.298019e+10	1.248838e+10
2022-10-01	1.261092e+10	1.298088e+10	1.250723e+10
2022-11-01	1.280695e+10	1.299129e+10	1.255334e+10
2022-12-01	1.646719e+10	1.302479e+10	1.315547e+10



Através do gráfico é possível observar que houve um ganho no desempenho quanto utilizamos o EWSA.


Comparação do SMA com o EWMA - Arrecadação
1e10
VALOR (R$)
SMA12
EWMA12
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
1999
2004
2009
2014
2019


Comparação do SMA com o EWMA - Saques
1e10
VALOR (R$)
SMA12
EWMA12
2.0
1.5
1.0
0.5
0.0
1999
2004
2009
2014
2019

5.3 Suavização Exponencial Dupla (DES)

A Suavização Exponencial Dupla (ou Double Exponential Smoothing, em inglês) é uma técnica de previsão de séries temporais que incorpora o componente de tendência do conjunto de dados. Ela é uma extensão da Suavização Exponencial Simples (SES) que, além do fator de suavização α, adiciona um segundo fator de suavização β para capturar a tendência.

A Suavização Exponencial Dupla se tornou uma das técnicas mais populares para previsão de séries temporais devido à sua simplicidade e eficácia. Ela é amplamente utilizada em áreas como finanças, vendas, operações de negócios, entre outras. Além disso, a técnica é capaz de lidar com tendências lineares e exponenciais, permitindo ajustes aditivos ou multiplicativos, respectivamente.

A Suavização Exponencial Dupla é implementada utilizando uma combinação de médias móveis ponderadas e suavização exponencial simples. O primeiro passo é calcular a média móvel ponderada (ou weighted moving average) dos valores da série temporal. Em seguida, é aplicada a suavização exponencial simples para essa média móvel ponderada, que gera uma previsão suavizada da série temporal.

Vamos explorar melhor uma variação da suavização exponencial que leva em conta a sazonalidade, a Suavização Exponencial Tripla.

5.4 Suavização Exponencial Tripla (TES)

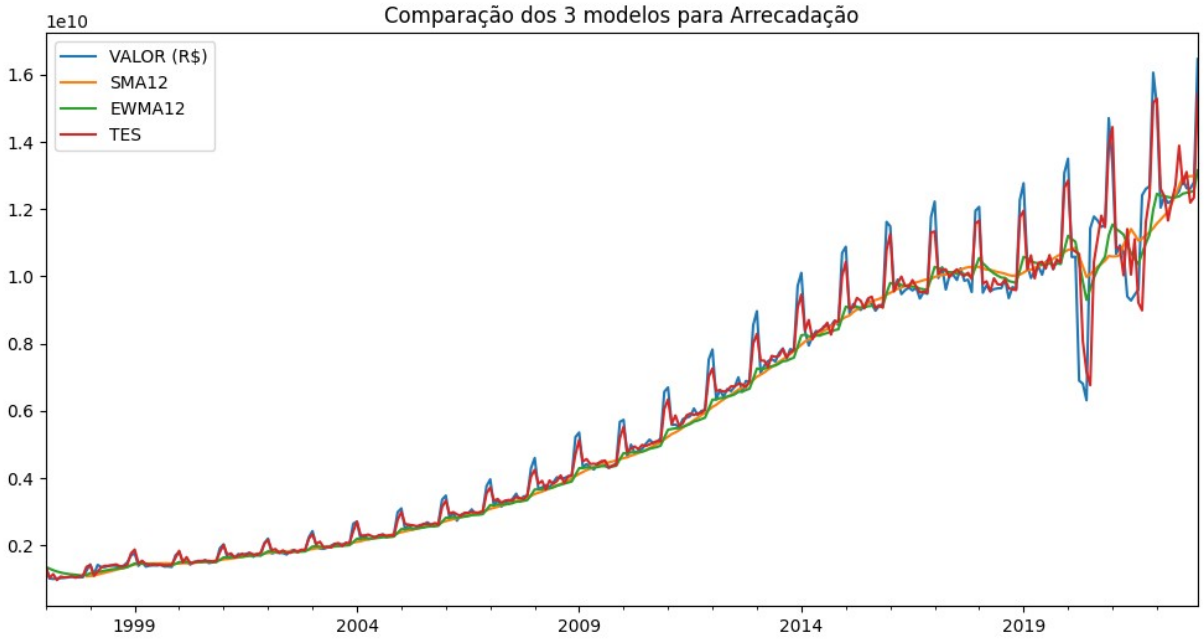
A suavização exponencial tripla, é uma extensão da suavização exponencial dupla que leva em consideração tanto a tendência quanto a sazonalidade nos dados de séries temporais. Envolve a adição de um terceiro fator de suavização, conhecido como gamma, ao modelo. O fator gamma ajusta a sazonalidade dos dados e ajuda a prever os padrões sazonais futuros. O método é particularmente útil para prever dados de vendas, preços de ações e outros dados de séries temporais que exibem tanto tendências quanto padrões sazonais. A suavização exponencial tripla também pode ser adaptada para lidar com vários períodos sazonais, tornando-se uma ferramenta poderosa para analisar e prever dados de séries temporais complexos. (HYNDMAN & ATHANASOPOULOS, 2018)

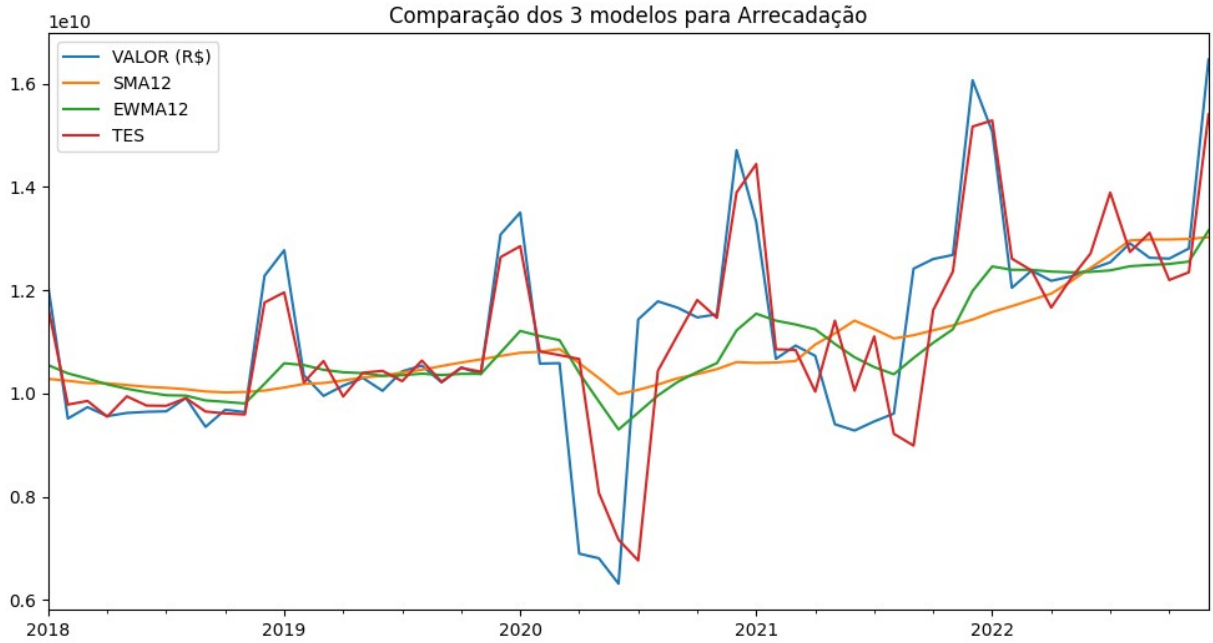
```
from statsmodels.tsa.holtwinters import ExponentialSmoothing
```

Arrecadação

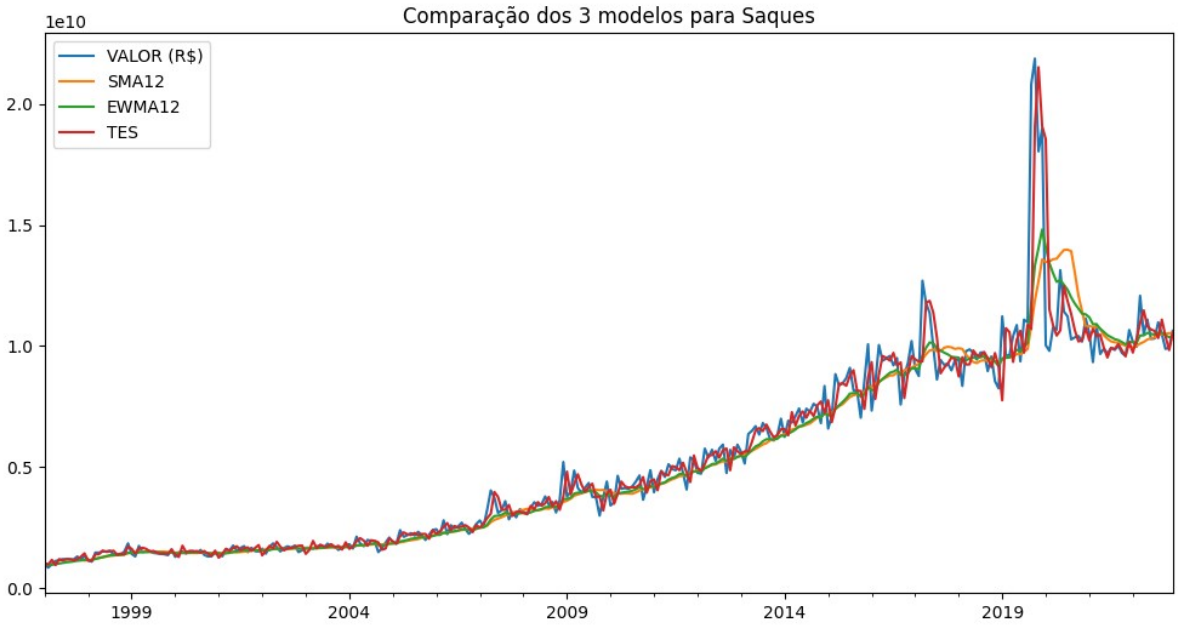
```
df_arrec['TES'] = ExponentialSmoothing(df_arrec['VALOR (R$)'],trend='add',seasonal='add',seasonal_periods=12).fit().fittedvalues
df_arrec.head()
```

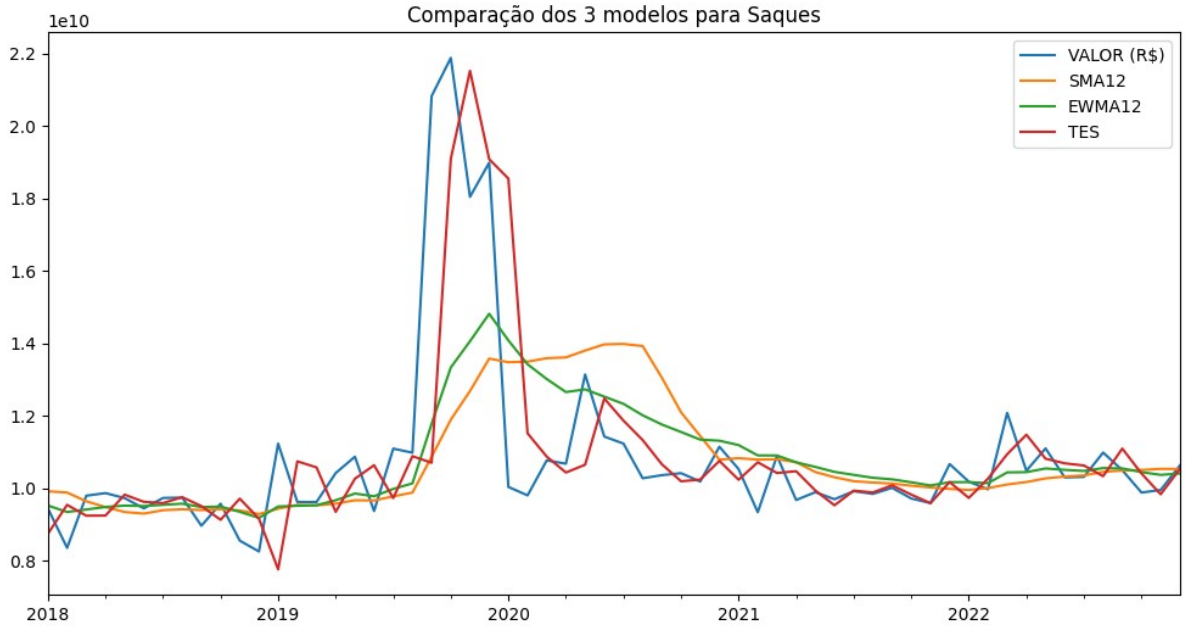
```
ax = df_arrec[['VALOR (R$)', 'SMA12', 'EWMA12', 'TES']].plot(figsize=(12,6))
ax.set(xlabel='')
ax.set_title('Comparação dos 3 modelos para Arrecadação');
```





Os mesmos procedimentos realizados para os saques, chegando nos resultados dos gráficos a seguir e com o detalhamento do código no jupyter-notebook disponibilizado.





Comparando os três modelos através das métricas de desempenho, o modelo de Suavização Exponencial Tripla (TES) apresentou o menor erro.

Arrecadação

```
------------------------------
RMSE (Root Mean Squared Error):

RMSE SMA          867810651.21
RMSE EWMA         772594145.83
RMSE TES          516699461.18
```

Saques

```
------------------------------
RMSE (Root Mean Squared Error):

RMSE SMA         1200702787.59
RMSE EWMA        1026127560.27
RMSE TES          999422042.3
```

6. Previsões (Forecasting)

Foram ajustados vários modelos de suavização aos dados existentes. O objetivo por trás disso é prever o que acontece a seguir.

Qual é a nossa melhor estimativa para o valor do próximo mês? Para os próximos seis meses?

Vamos procurar estender nossos modelos para o futuro. Primeiro, dividiremos os dados conhecidos em conjuntos de treinamento e teste e avaliaremos o desempenho de um modelo treinado nos dados de teste conhecidos.

6.1 Base de treinamento e base de testes

Para garantir que a base de teste seja representativa do conjunto de dados original vamos iniciar a divisão com 25% para base de teste e 75% para treinamento.

Arrecadação

```
len(df_arrec) * .75
```

```
234.0
```

Separar base de teste e base de treino

```
train_data_A = df_arrec['VALOR (R$)'].iloc[:234]
test_data_A = df_arrec['VALOR (R$)'].iloc[234:]
```

```
print(f'Base de teste: {len(test_data_A)}, representa {round(100*len(test_data_A)/len(df_arrec),2)}%')
print(f'Base de treino: {len(train_data_A)}, representa {round(100*len(train_data_A)/len(df_arrec),2)}%')
print(f'Base')
```

```
Base de teste: 78, representa 25.0%
Base de treino: 234, representa 75.0%
Base
```

6.2 Modelo de Suavização Exponencial Tripla

Após separar as bases de teste e treinamento, é gerado o modelo com a base de treino e a previsão com a mesma quantidade de meses da base de teste para comparação.

Ajustar o Modelo

```
from statsmodels.tsa.holtwinters import ExponentialSmoothing
```

```
train_data_A_hw = train_data_A
test_data_A_hw = test_data_A
```

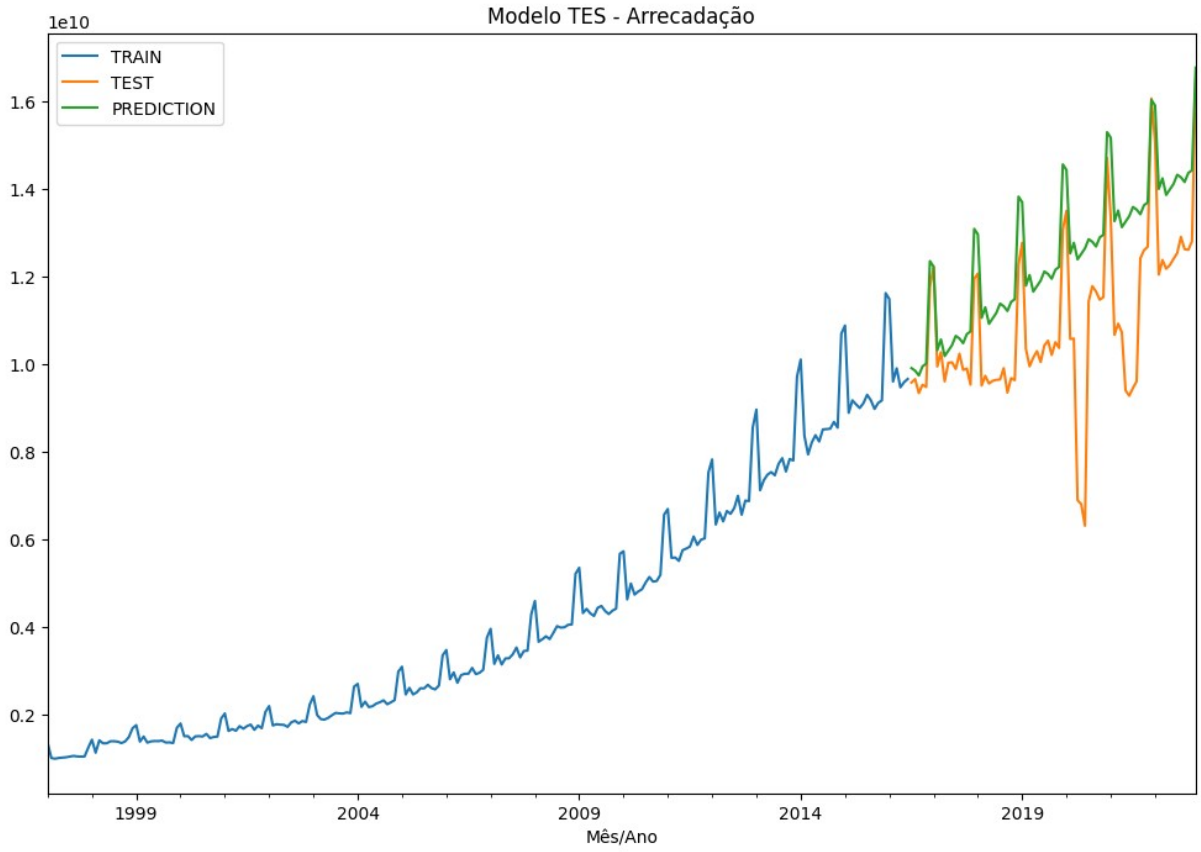
```
fitted_model = ExponentialSmoothing(train_data_A,trend='add',seasonal='add',seasonal_periods=12).fit()
```

Avaliando o Modelo em relação ao Conjunto de Teste

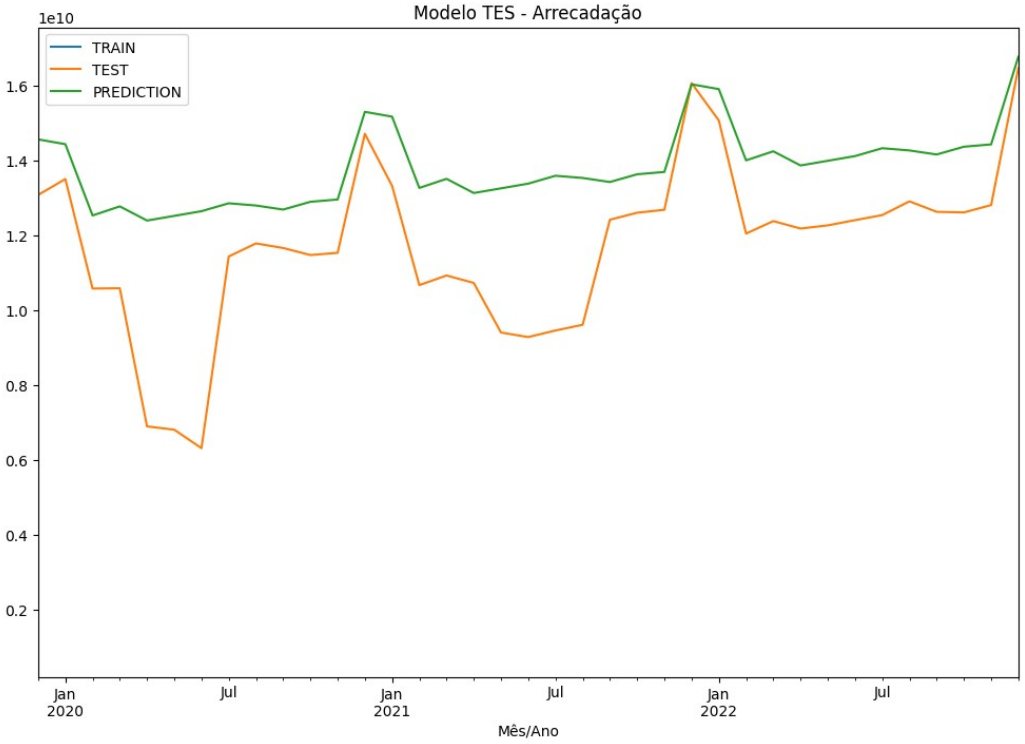
```
predict_HW = fitted_model.forecast(78).rename('HW Forecast')
```

```
train_data_A_hw.plot(legend=True,label='TRAIN')
test_data_A_hw.plot(legend=True,label='TEST',figsize=(12,8), title='Modelo TES')
predict_HW.plot(legend=True,label='PREDICTION');
```

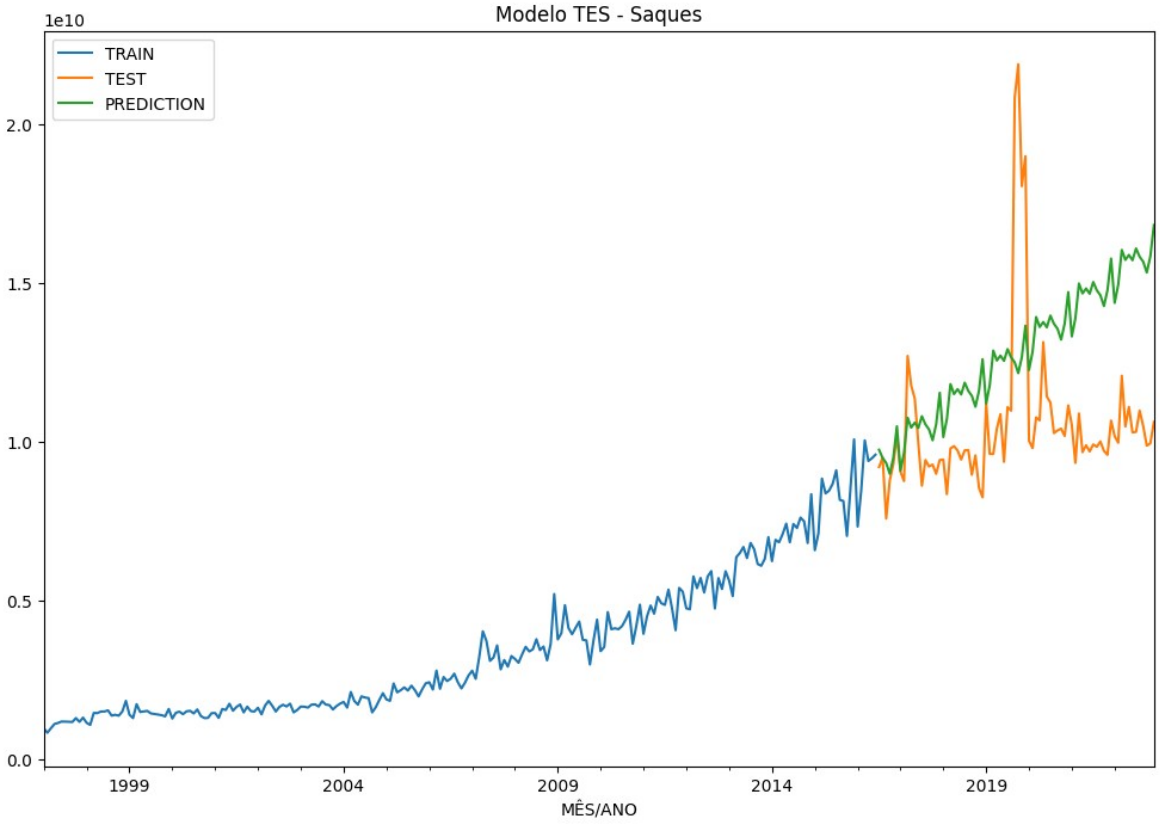
Para visualizar o resultado é preciso construir o gráfico da base de treino, base de teste e predições.




```
train_data_A_hw.plot(legend=True,label='TRAIN')
test_data_A_hw.plot(legend=True,label='TEST',figsize=(12,8), title='Modelo TES - Arrecadação')
predict_A_HW.plot(legend=True,label='PREDICTION',xlim=['2019-12-01','2022-12-01']);
```



Os mesmos procedimentos foram efetuados com a base de Saques. Para não replicar todo o código, para os saques serão exemplificados somente com os gráficos, sendo que todo o código se encontra no jupyter-notebook disponibilizado no repositório do github.

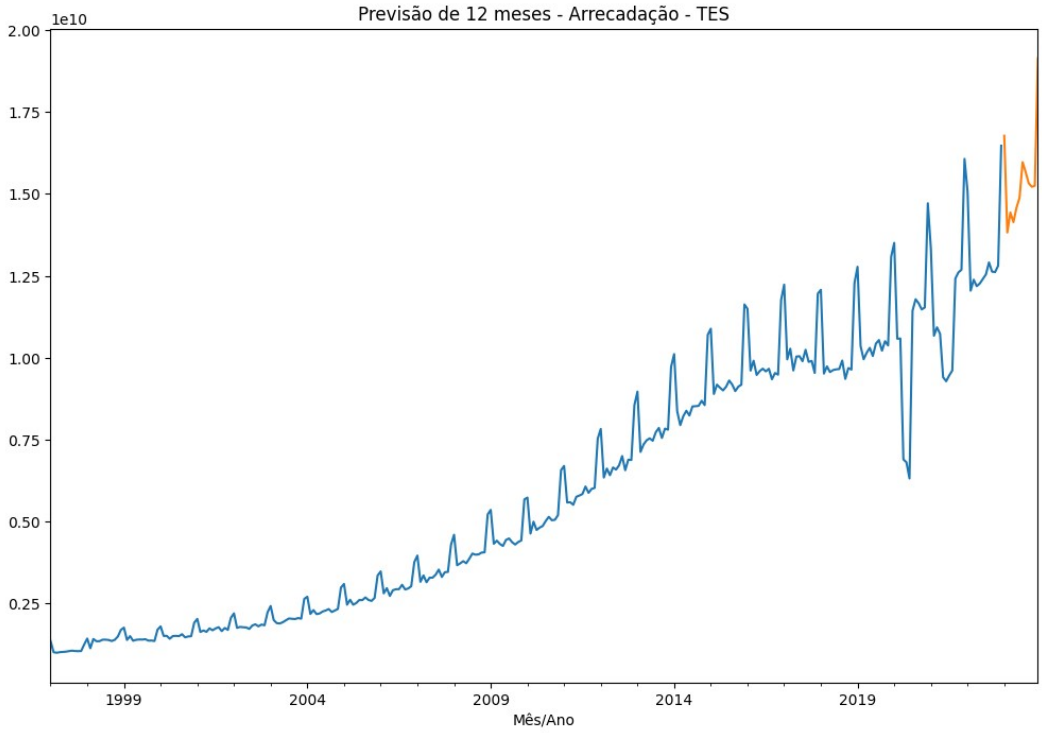


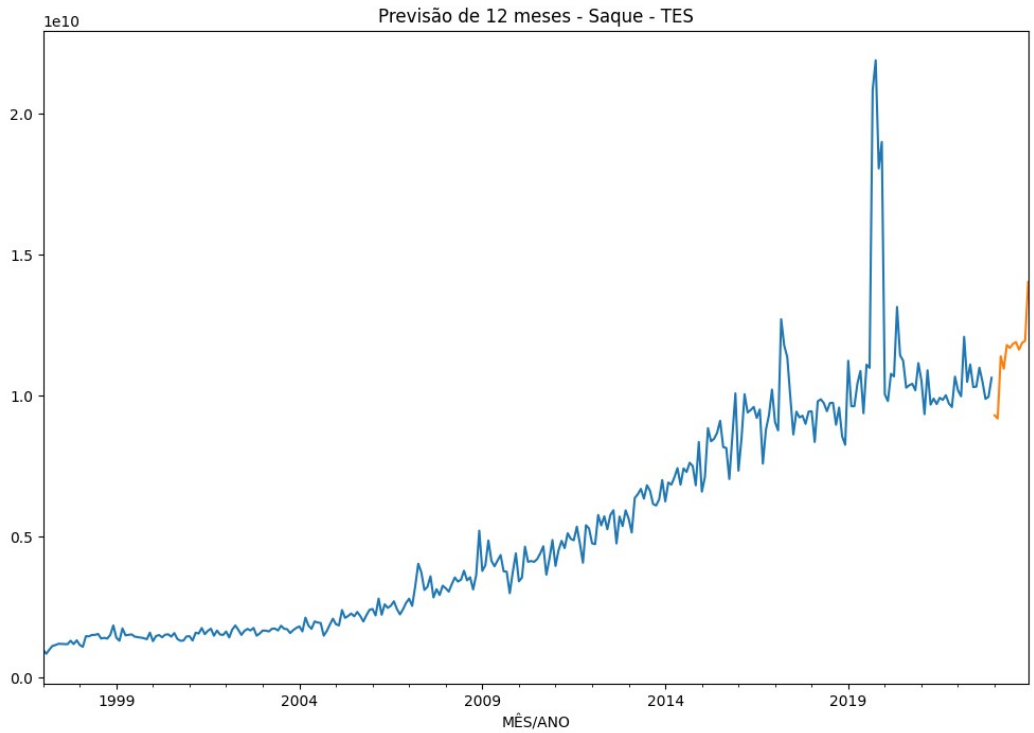
6.2.1 Realizando as previsões para 12 meses

```
final_model_ARfit_A = AutoReg(df_arrec['VALOR (R$)'], lags = 12).fit()
```

```
forecast_predictions_AR_A = final_model_ARfit_A.forecast(12)
```

```
df_arrec['VALOR (R$)'].plot(figsize=(12,8), title='Previsão de 12 meses - Arrecadação - AR')
forecast_predictions_AR_A.plot();
```



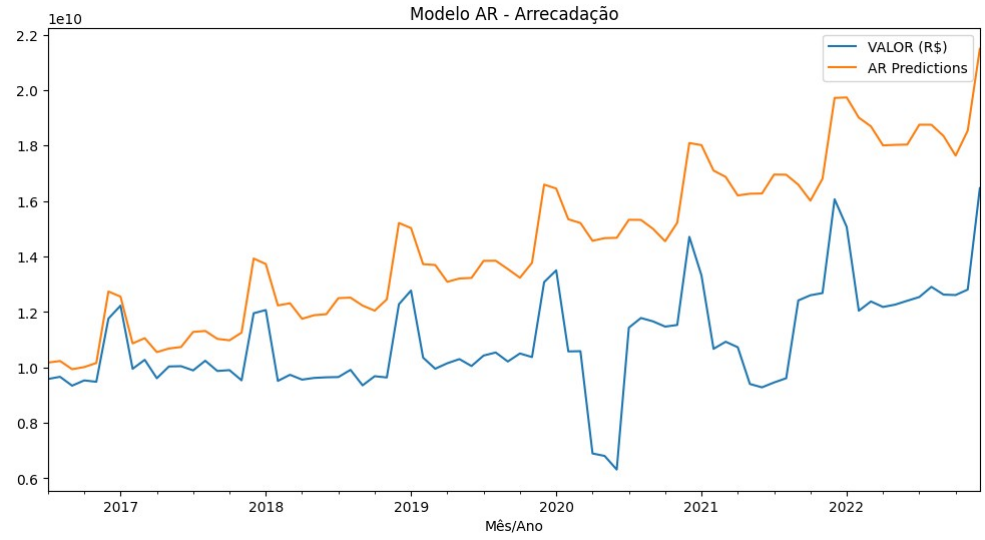


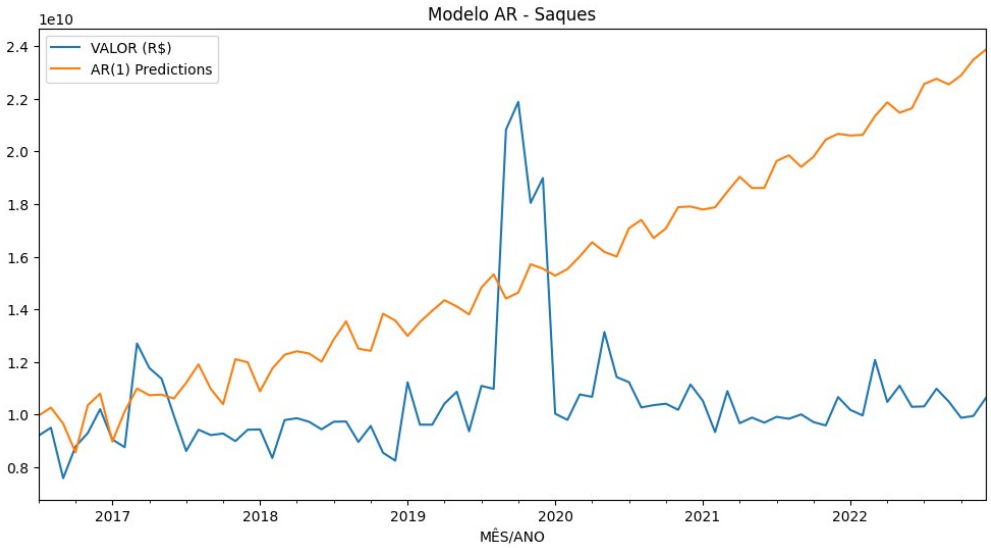
6.3 Modelo de AR (Auto-regressão)

O `statsmodels` é um pacote de Python que fornece diversas ferramentas para a análise estatística, incluindo modelos de séries temporais. Um dos modelos de séries temporais disponíveis no `statsmodels` é o modelo de autorregressão (AR).

Um modelo de autorregressão é um modelo estatístico que tenta prever o valor de uma variável em um determinado período, baseado em seus valores passados. O modelo AR é um modelo linear, ou seja, assume que a relação entre a variável dependente e as variáveis independentes é linear. No caso do modelo AR, a variável dependente é o valor da série temporal em um determinado período e as variáveis independentes são os valores da série temporal em períodos anteriores.

Para utilizar o modelo AR no `statsmodels`, é necessário importar o pacote de séries temporais e instanciar a classe AR. Em seguida, é necessário passar a série temporal como argumento para o método `fit()` e chamar o método `summary()` para obter as estatísticas do modelo.



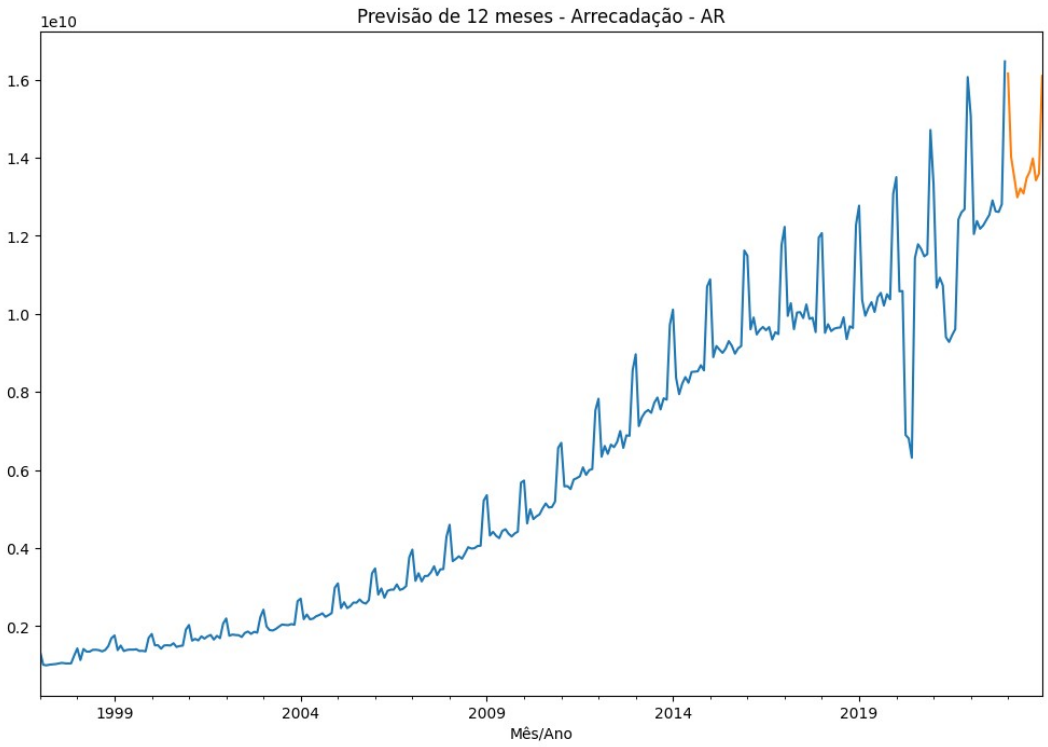


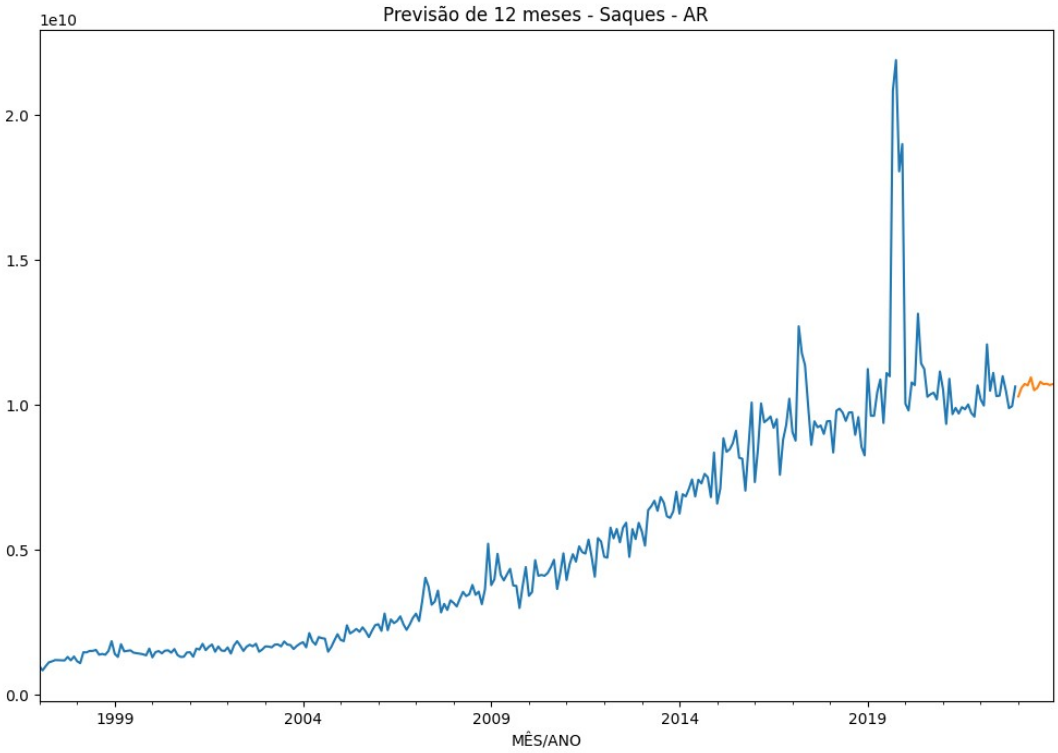
6.3.1 Realizando as previsões para 12 meses

```
final_model_ARfit_A = AutoReg(df_arrec['VALOR (R$)'], lags = 12).fit()
```

```
forecast_predictions_AR_A = final_model_ARfit_A.forecast(12)
```

```
df_arrec['VALOR (R$)'].plot(figsize=(12,8), title='Previsão de 12 meses - Arrecadação - AR')
forecast_predictions_AR_A.plot();
```





6.4 Modelo de ARIMA

ARIMA (*Auto-Regressive Integrated Moving Average*) é um modelo estatístico amplamente utilizado na análise de séries temporais (HYNDMAN & ATHANASOPOULOS, 2018). Ele combina três componentes: a autoregressão (AR), a diferenciação (I) e a média móvel (MA). A autoregressão captura a relação entre uma observação e as observações anteriores, a diferenciação é utilizada para tornar a série estacionária e a média móvel captura a tendência da série temporal (Box & Jenkins, 1976). O objetivo do modelo é prever valores futuros baseados nos valores passados.

O uso do modelo ARIMA é amplamente recomendado para séries temporais estacionárias, pois é capaz de capturar as características de tendência e sazonalidade, e também pode ser utilizado para lidar com dados não estacionários através da diferenciação (Enders, 2014).

O Modelo ARIMA é especificado por três parâmetros de ordem: *(p, d, q).*

- **AR** (*p*) Auto Regressão – um modelo de regressão que utiliza a relação dependente entre uma observação atual e as observações de um período anterior. Um componente autorregressivo (*AR (p)*) refere-se ao uso de valores passados na equação de regressão para a série temporal;
- **I** (*d*) Integração – usa diferenciação de observações (subtraindo uma observação da observação no intervalo de tempo anterior) para tornar a série temporal estacionária. A diferença envolve a subtração dos valores atuais de uma série com seus valores anteriores *d* número de vezes;
- **MA** (*q*) Média móvel – um modelo que usa a dependência entre uma observação e um erro residual de um modelo de média móvel aplicado a observações defasadas. Um componente de média móvel representa o erro do modelo como uma combinação de termos de erro anteriores. A ordem *q* representa o número de termos a serem incluídos no modelo. (ACERVO LIMA, 2023).

Para descobrir os melhores parâmetros *(p, d, q)* vamos utilizar o método `auto_arima` da biblioteca `pmdarima`.

O Auto_ARIMA é uma técnica utilizada em séries temporais para encontrar automaticamente o melhor modelo ARIMA para prever valores futuros com base em dados históricos. É um algoritmo que automatiza o processo de seleção do modelo ARIMA adequado, ajustando automaticamente os parâmetros do modelo para obter a melhor previsão possível.

O Auto_ARIMA permite economizar tempo na seleção manual de modelos e oferece um modelo otimizado e mais preciso para prever valores futuros.

```
pm.auto_arima(arrec['VALOR (R$)'], seasonal=True, m=12).summary()
```

SARIMAX Results

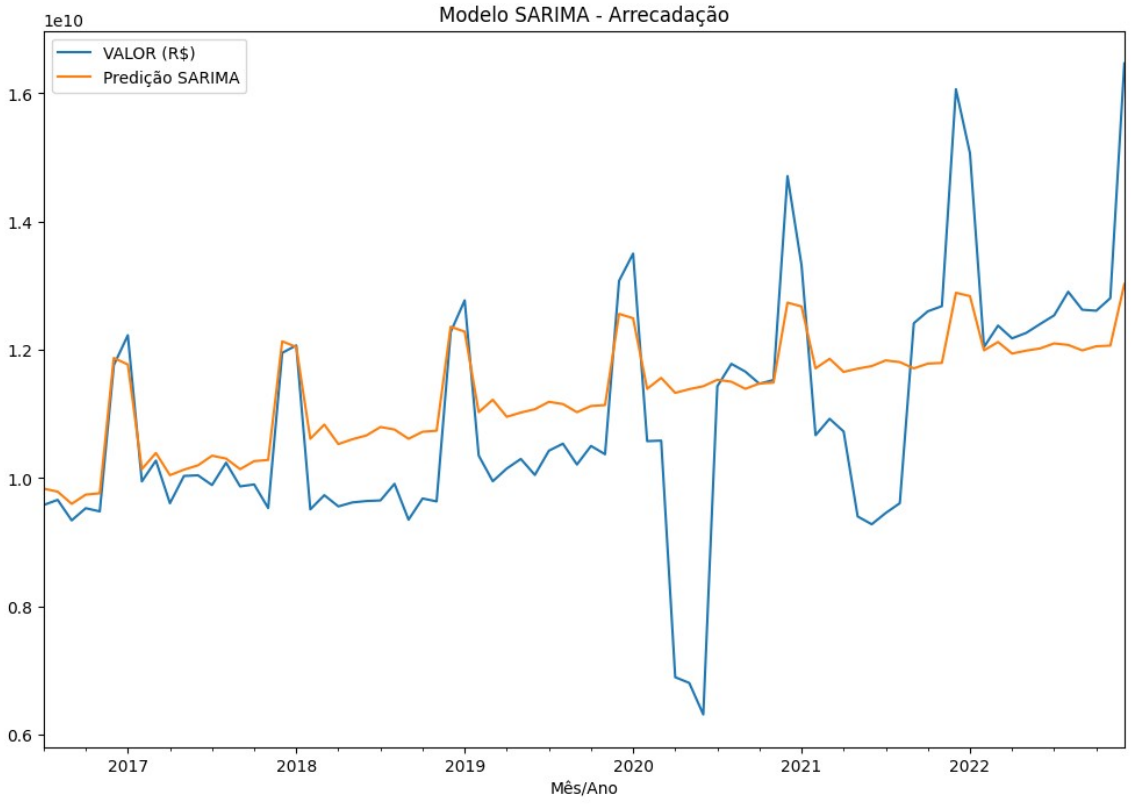
Dep. Variable:	y	**No. Observations:**	312
Model:	SARIMAX(2, 1, 2)x(1, 0, [1], 12)	**Log Likelihood**	-6714.657
Date:	Fri, 07 Apr 2023	**AIC**	13443.314
Time:	16:47:52	**BIC**	13469.493
Sample:	01-01-1997	**HQIC**	13453.778
	- 12-01-2022		
Covariance Type:	opg		

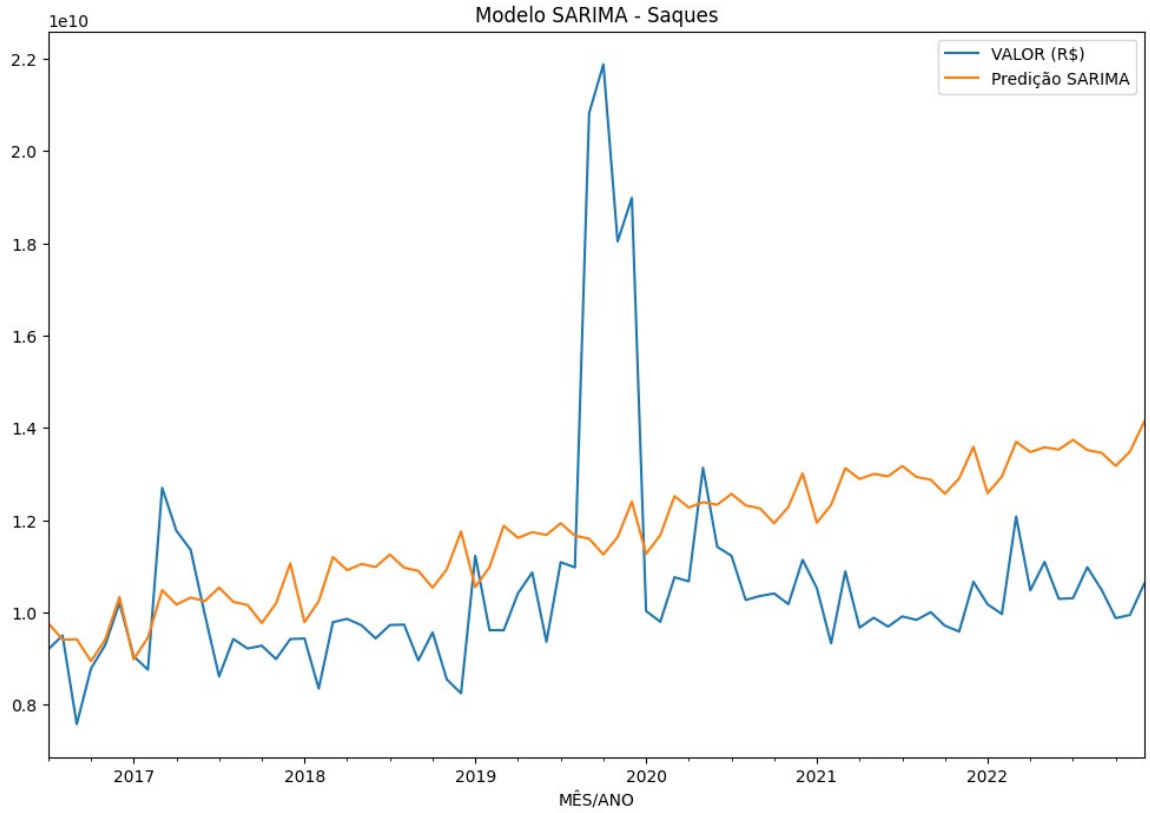
Com esse resultado já sabemos que o modelo utilizado será o SARIMAX, com parâmetros `order = (2, 1, 2)` e `seasonal_order = (1, 0, 1, 12)`.

O modelo SARIMA (Seasonal Autoregressive Integrated Moving Average) é uma extensão do modelo ARIMA que inclui um componente sazonal para capturar a sazonalidade em séries temporais. O modelo SARIMA é especialmente útil em situações em que a série temporal exibe padrões sazonais claros que se repetem ao longo do tempo.

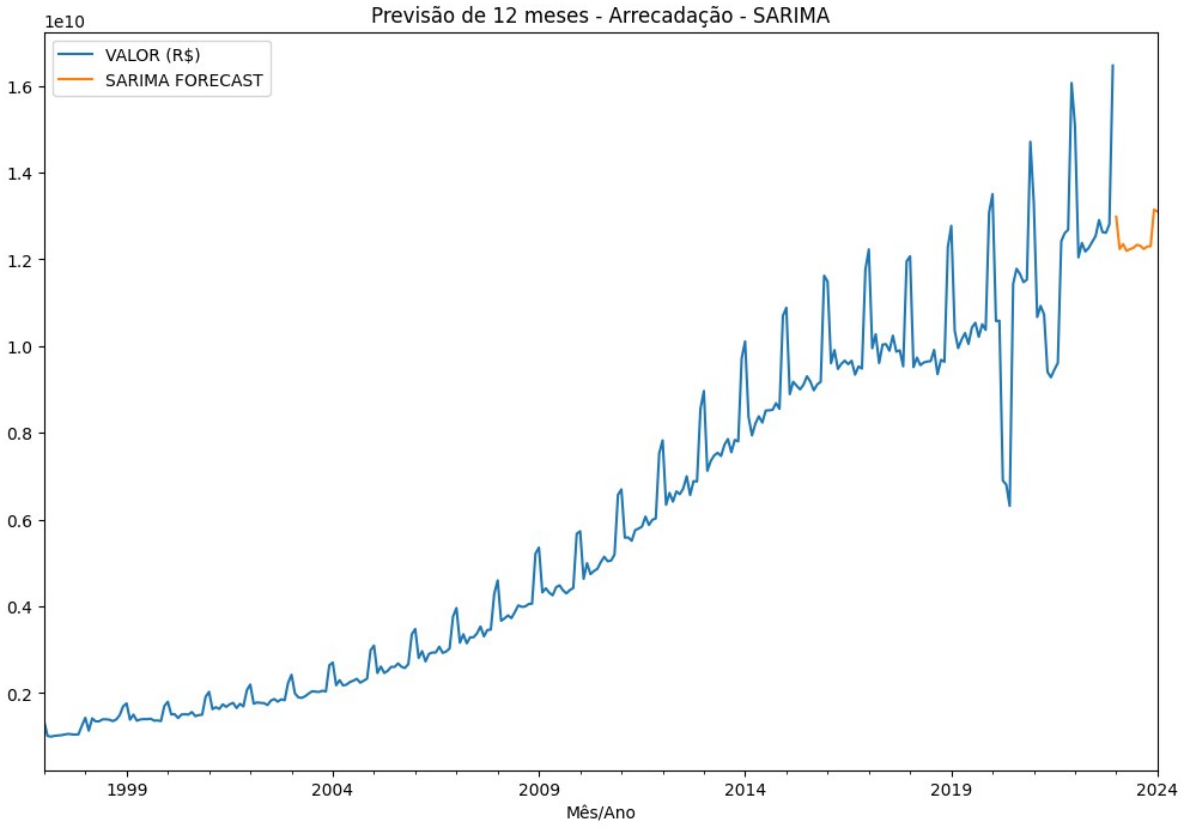
Assim como o modelo ARIMA, o modelo SARIMA envolve a escolha de ordens de diferenciação, ordens de autoregressão e ordens de média móvel. No entanto, o modelo SARIMA inclui termos adicionais para modelar a sazonalidade, que podem ser representados como termos sazonais de autoregressão (SAR) e média móvel sazonal (SMA).

O resultado do modelo é o gráfico abaixo:





6.4.1 Realizando as previsões para 12 meses



Em relação ao modelo do saque será necessário identificar o período que parece significativamente diferente do restante dos dados. Isso poderá ser feito visualmente com a ajuda de gráficos, como um gráfico de linha ou um gráfico de dispersão, ou por meio de cálculos estatísticos, como a média e o desvio padrão.

Escolhendo uma técnica para remover o período outlier da série temporal, incluindo a eliminação simples dos dados ou substituindo os valores outliers pela média ou pela mediana dos valores não outliers ou utilizando métodos estatísticos mais avançados, como o método de Dixon ou o método de Grubbs.

No entanto, vamos avaliar os resultados até aqui para verificar se já é possível comparar a performance dos modelos.

7. Interpretação dos Resultados

Comparando os modelos

```
# from sklearn.metrics import mean_squared_error
from statsmodels.tools.eval_measures import rmse
```

```
def metricas(nome, teste, previsao):
    print(f'RMSE {nome} \t{round(rmse(teste, previsao), 2)}')
```

```
print('------------------------------\nRMSE (Root Mean Squared Error) - Arrecadação:\n')
metricas('HW', test_data_A_hw, predict_A_HW)
metricas('AR', test_data_AR_A, predict_AR_A)
metricas('SARIMA', teste_arima_a, predict_sarima_a)
```

```
------------------------------
RMSE (Root Mean Squared Error) - Arrecadação:

RMSE HW       1986584911.07
RMSE AR       4125401750.14
RMSE SARIMA   1389010175.56
```

```
print('------------------------------\nRMSE (Root Mean Squared Error) - Saques:\n')
metricas('HW', test_data_S_hw, predict_S_hw)
metricas('AR', test_data_AR_S, predict_AR_S)
metricas('SARIMA', teste_arima_s, predict_sarima_s)
```

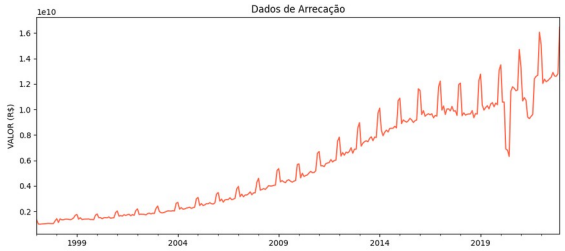
```
------------------------------
RMSE (Root Mean Squared Error) - Saques:

RMSE HW       3652898645.48
RMSE AR       6704612423.77
RMSE SARIMA   2771793455.89
```

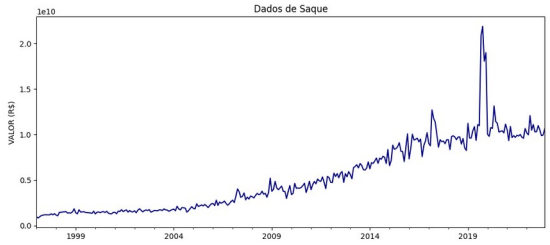
Inicialmente verificamos que o modelo de Suavização Exponencial Tripla havia obtido um resultado melhor que os modelos de média móvel simples e média móvel ponderada exponencialmente.

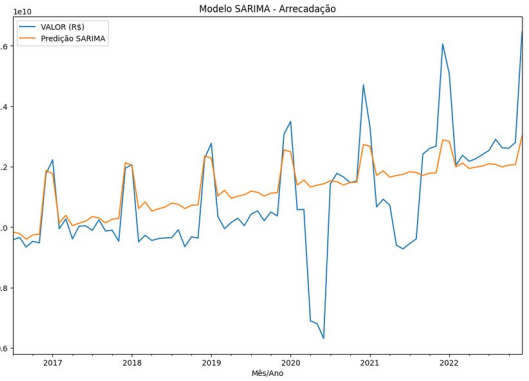
E por fim. mesmo com os períodos *outliers* foi possível observar que o modelo SARIMA teve um resultado melhor em comparação aos outros dois modelos: Holt-winters (Suaviação Exponencial Tripla) e o modelo de Autor-regressão.

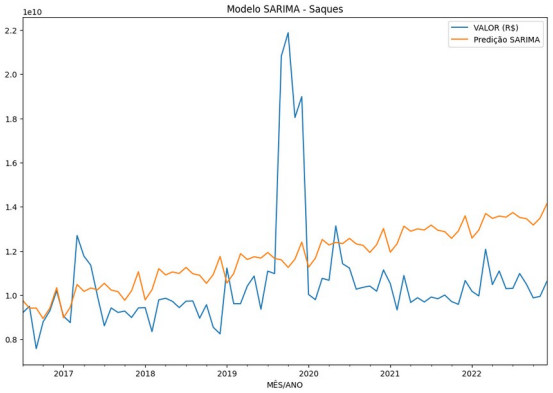
Logo chegamos a conclusão que o modelo SARIMA com a utilização da identificação dos parâmetros com o auxílio do método `auto_arima` da biblioteca `pmdarima` pode ser um bom modelo para fazer as predições de entradas e saídas do fluxo de caixa do FGTS.

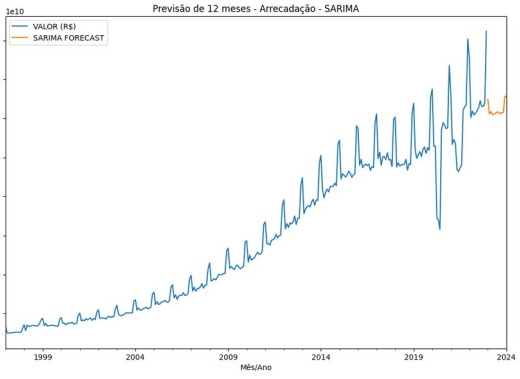
8. Apresentação dos Resultados

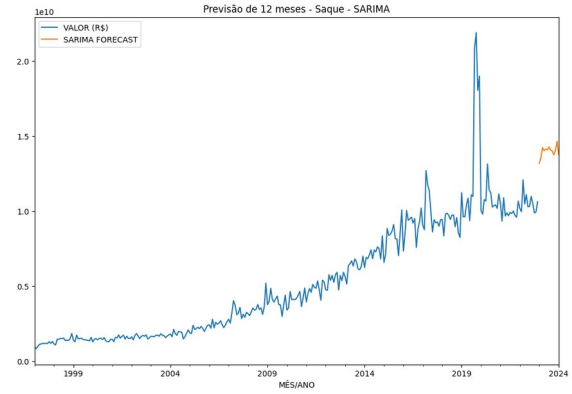













```
--------------------------------
RMSE (Root Mean Squared Error) - Arrecadação:

RMSE HW        1986584911.07
RMSE AR        4125401750.14
RMSE SARIMA    1389010175.56
```

```
--------------------------------
RMSE (Root Mean Squared Error) - Saques:

RMSE HW        3652898645.48
RMSE AR        6704612423.77
RMSE SARIMA    2771793455.89
```

9. Links

Aqui você deve disponibilizar os links para o vídeo com sua apresentação de 5 minutos e para o repositório contendo os dados utilizados no projeto, scripts criados, etc.

Link para o vídeo: youtube.com/...
Link para o repositório: github.com/...

REFERÊNCIAS

ACERVO LIMA (2022). **Python | Modelo ARIMA para previsão de série temporal**. Recuperado em 20 de dezembro de 2022, de https://acervolima.com/python-modelo-arima-para-previsao-de-serie-temporal/

BOX, G. E. P., JENKINS, G. M., REINSEL, G.C. **Time Series Analysis: Forecasting and Control**. New Jersey: John Wiley & Sons 2008.

ENDERS, W. **Applied Econometric Time Series (4th ed.)**. Tuscaloosa, Alabama: John Wiley & Sons 2014.

HYNDMAN, R. J., & ATHANASOPOULOS, G. **Forecasting: principles and practice.** (https://otexts.com/fpp3/) 2018.

WIKIPÉDIA, a enciclopédia livre. **Média Móvel**. Flórida: Wikimedia Foundation, 2022. Disponível em: <https://pt.wikipedia.org/wiki/M%C3%A9dia_m%C3%B3vel#M%C3%A9dia_m%C3%B3vel_simples>. Acesso em: 9 dez. 2022.

MORETTIN, Pedro A.; TOLOI, Clélia M. C. **Análise de séries temporais**. São Paulo: Blucher, 2006.

NIELSEN, Aileen. **Análise Prática de Séries Temporais**. Rio de Janeiro: Alta Books, 2016.

SAMISHAWL. **Python | ARIMA Model for Time Series Forecasting**, publicado em 19 de fevereiro de 2020. Disponível em: https://www.geeksforgeeks.org/python-arima-model-for-time-series-forecasting/. Acesso em: 20 de março de 2023.

APÊNDICE

Programação/Scripts

Cole aqui seus scripts em Python e/ou R.