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

Marco Sérgio Almeida Veludo Gouveia

PREVISÃO DA ARRECADAÇÃO FEDERAL

Belo Horizonte
2021

Marco Sérgio Almeida Veludo Gouveia

PREVISÃO DA ARRECADAÇÃO FEDERAL

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.

Belo Horizonte
2021

SUMÁRIO

1. Introdução 4
1.1. Contextualização 4
1.2. O problema proposto 4
2. Coleta e Tratamento de Dados 6
2.1. Série histórica da arrecadação federal 6
2.2. Série histórica do PIB 9
2.3. Série histórica do IPCA 11
3. Análise e Exploração dos Dados 14
3.1. Decomposição STL e Tratamento de Outliers 16
3.2. Estacionariedade da Série 22
3.3. Sazonalidade da Série 25
4. Criação de Modelos de Machine Learning 26
4.1. Facebook Prophet 28
4.2. Modelos ARIMA 34
5. Apresentação dos Resultados 41
6. Links 43
REFERÊNCIAS 44
APÊNDICE 45
Apresentação 45

1. Introdução

1.1. Contextualização

O modelo orçamentário[1] para a gestão do dinheiro público no Brasil tem por base a elaboração de 3 leis, quais sejam, O PLANO PLURIANUAL (PPA), LEI DE DIRETRIZES ORÇAMENTÁRIAS – LDO e a LEI ORÇAMENTÁRIA ANUAL (LOA). Note-se que esse modelo orçamentário é aplicado a todas as esferas de governo, seja federal, estadual ou municipal.

Anualmente o Congresso Nacional precisa enviar, até 31 de agosto, o projeto da LOA (PLOA), uma lei que **estima as receitas** e fixa as despesas públicas para o período de um exercício financeiro.

Torna-se, portanto, fundamental o aperfeiçoamento da previsão de receitas para um adequado planejamento das políticas públicas e investimentos públicos, afetando a vida de todos os brasileiros.

Apesar de aplicável a todas as esferas de governo, nesse trabalho vamos nos concentrar na estimativa de receitas fazendárias no âmbito federal.

1.2. O problema proposto

Será aplicada a ferramenta 5W1H para possibilitar uma melhor visão do problema proposto e da solução.

What? (O que?)

Propõe-se utilizar técnicas de ciência de dados para analisar a série histórica da arrecadação fazendária federal total (sem detalhar cada tributo), realizar ajustes de atipicidades (*outliers*) e projetar a receita prevista num momento futuro, avaliando a performance de cada metodologia empregada.

[1] Fonte: Legislação Orçamentária :: Portal do Orçamento (senado.leg.br) - https://www12.senado.leg.br/orcamento/legislacao-orcamentaria. Acesso em 22 abr. 2021.

Serão aliados à série histórica dois outros conjuntos de dados (*datasets*), referente ao Produto Interno Bruto – PIB e ao Índice de Preços ao Consumidor Amplo – IPCA, para verificação se a combinação dessas variáveis pode melhorar a qualidade da predição.

Why? (Por quê?)

Durante o processo de elaboração do orçamento, a receita precisa ser estimada antecipadamente, para que então o governo tenha uma previsão de quanto poderá gastar, com impactos positivos na definição das políticas públicas, bem como no efetivo controle de gastos.

Who? (De quem?)

Os dados analisados estão disponíveis publicamente e são gerados pela Secretaria da Receita Federal do Brasil - RFB e pelo Instituto Brasileiro de Geografia e Estatística - IBGE.

Where? (De onde?)

Serão utilizados dados da arrecadação fazendária federal, de abrangência nacional, dados do PIB Brasil e da série histórica do IPCA.

When? (Qual período?)

O período que está sendo analisado compreende os anos de 2004 a 2019, evitando as fortes variações ocorridas com a pandemia da Covid-19 já no início de 2020.

How? (Como?)

O ambiente de programação computacional será baseado primordialmente na linguagem Python com pacotes voltados a ciência de dados, executada no ambiente Jupyter. Também será utilizada a linguagem R e o ambiente “R Studio” para situações específicas.

Os modelos de predição serão baseados no ARIMA e suas variações, como SARIMA e SARIMAX. Também será testada a performance do pacote Facebook PROPHET.

2. Coleta e Tratamento de Dados

Conforme mencionado na seção anterior, foram utilizados 3 *datasets* na elaboração deste estudo: série histórica da arrecadação federal, série histórica do PIB Brasil e série histórica do IPCA.

Nos próximos tópicos serão detalhados os procedimentos utilizados para a coleta desses dados e o tratamento realizado em cada etapa.

2.1. Série histórica da arrecadação federal

Para a coleta dos dados da arrecadação federal, serão utilizados os dados de Arrecadação por Estado[2], disponíveis publicamente no sítio da RFB na internet em formato de planilhas eletrônicas, nos padrões XLS e ODS.

Figura 1 – Portal da Receita Federal do Brasil - Dados Abertos - Arrecadação por Estado

O notebook "**01. Download da Arrecadação.ipynb**" foi desenvolvido para realizar *web scraping* das planilhas disponíveis no sítio da RFB, utilizando principalmente o pacote *Beautiful Soup*[3].

Partindo da URL inicial, são localizados todos os anos com publicação da arrecadação e se inicia um enlace identificando: os arquivos (meses) disponíveis, o

[2] Arrecadação por Estado – Receita Federal do Brasil. Disponível em: https://receita.economia.gov.br/dados/receitadata/arrecadacao/arrecadacao-por-estado. Acesso em: 01 abr. 2021.
[3] Beautiful Soup: Disponível em: https://pypi.org/project/beautifulsoup4/. Acesso em: 01 abr. 2021.

link completo para cada arquivo e os respectivos tipos (XLS/ODS). Em seguida o download dos arquivos é realizado para a pasta “planilhas”, criada no projeto.

Com a execução do notebook 01, foram recuperadas 192 planilhas, uma vez que foram considerados apenas os anos definidos no escopo do trabalho, 2004 a 2019.

Tendo em vista que os padrões de nomenclatura adotados pela Receita Federal para os nomes dos arquivos variam ao longo do tempo, foi necessário criar uma função para padronizá-los no formato ‘AAAA-MM-arrecadação-uf’ em preparação à próxima etapa.

2.1.1. Tratamento dos dados da arrecadação e construção do dataset

Da análise de cada planilha, observam-se pequenas diferenças em formatação e nomenclatura, além de ter sido adotado o formato .ODS (LibreOffice) a partir de março de 2011.

O notebook “**02. Gera dataset de arrecadação a partir das planilhas.ipynb**” foi desenvolvido para percorrer o conteúdo de todas as planilhas, armazenando o resultado de cada uma delas em um dataframe Pandas individual, os quais foram, posteriormente, concatenados com os demais dataframes.

As planilhas originalmente foram elaboradas tendo os tributos nas linhas e os estados nas colunas. Para viabilizar as etapas de aplicação dos modelos preditivos foi necessário realizar a transposição do dataframe, conforme ilustrado na figura seguinte.

RECEITA	AC	AL	AP	AM	BA	CE
IMPOSTO SOBRE IMPORTAÇÃO	2.992	175.830	476.880	12.396.707	15.577.437	4.270.052
IMPOSTO SOBRE EXPORTAÇÃO	0	0	0	0	154.564	2.556
IPI - TOTAL	299.179	2.601.116	406.071	8.826.165	49.686.824	13.034.467
IPI - FUMO	287.722	1.317.950	234.512	1.291.484	5.083.655	4.060.081
IPI - BEBIDAS	1.189	702.806	0	2.972.949	13.931.673	3.920.213
IPI - AUTOMÓVEIS	0	0	2.348	1.465	7.514.550	30.399
IPI - VINCULADO À IMPORTACAO	0	5.132	148.730	3.307.354	10.847.992	2.887.167
IPI - OUTROS	10.268	575.228	20.482	1.252.913	12.308.955	2.136.607
IMPOSTO SOBRE A RENDA - TOTAL	3.303.814	21.535.324	4.438.255	64.522.334	119.524.932	81.700.242
IRPF	133.103	826.895	151.975	1.325.242	3.775.278	2.358.916
IRPJ	1.029.968	4.710.124	1.145.378	28.340.908	60.807.591	42.717.143
ENTIDADES FINANCEIRAS	0	68.763	0	106.837	4.974.949	12.598.185
DEMAIS EMPRESAS	1.029.968	4.641.361	1.145.378	28.234.071	55.832.642	30.118.958
IMPOSTO S/ RENDA RETIDO NA FONTE	2.140.744	15.998.304	3.140.902	34.856.184	54.942.066	36.624.183
IRRF - RENDIMENTOS DO TRABALHO	1.621.587	9.442.645	3.011.515	21.611.044	35.193.769	25.708.310
IRRF - RENDIMENTOS DO CAPITAL	492.713	1.098.741	14.787	4.177.770	11.168.153	7.769.452
IRRF - REMESSAS P/ EXTERIOR	0	4.726.181	14.511	6.813.305	4.805.548	1.105.330
IRRF - OUTROS RENDIMENTOS	26.444	730.736	100.089	2.254.065	3.774.597	2.041.093

estado	IMPOSTO SOBRE IMPORTAÇÃO	IMPOSTO SOBRE EXPORTAÇÃO	IPI - TOTAL	IPI - FUMO	IPI - BEBIDAS	IPI - AUTOMÓVEIS	IPI - VINCULADO À IMPORTACAO	IPI - OUTROS	IMPOSTO SOBRE A RENDA - TOTAL	...
AC	2992	0	299179	287722	1189	0	0	10268	3303814	...
AL	175830	0	2601116	1317950	702806	0	5132	575228	21535324	...
AP	476880	0	406071	234512	0	2348	148730	20482	4438255	...
AM	12396707	0	8826165	1291484	2972949	1465	3307354	1252913	64522334	...
BA	15577437	154564	49686824	5083655	13931673	7514550	10847992	12308955	119524932	...
CE	4270052.0	2556.0	13034467.0	4060081.0	3920213.0	30399.0	2887167.0	2136607.0	81700242.0	...
DF	358455.0	0.0	9052703.0	2967279.0	3013464.0	0.0	973119.0	2098841.0	1181653650.0	...
ES	61102313	0	56702711	3645033	1810944	842611	45407052	4997071	57930935	...
GO	312022	0	17779835	4145424	7363865	1458808	63629	4748108	74561609	...
MA	1606199.0	13.0	7175643.0	801428.0	4362053.0	0.0	1299088.0	713074.0	16835283.0	...
MT	10557.0	0.0	6255792.0	1115776.0	4486383.0	1851.0	19652.0	632130.0	31498454.0	...

Figura 2 - Planilha do mês Jan-2004 carregada em dataframe com transposição das linhas/colunas.

A primeira coluna em 2004 recebeu a denominação de “RECEITA ”, porém em outros anos foi “RECEITAS ” (com e sem um espaço ao final). Dessa forma, todas foram padronizadas como “tributo” antes de transpor o dataframe. (Figura 3)

Conforme a definição de escopo, o interesse desse estudo está na **arrecadação fazendária total**, em outras palavras, o total das receitas administradas pela RFB, com exceção das receitas previdenciárias. No entanto, o rótulo da linha que contém essa informação muda de denominação ao longo do tempo.

```
# Padroniza as diversas denominações atribuídas às colunas com o nome dos tributos
df.rename(columns={'RECEITA ':'tributo', 'RECEITAS':'tributo',
                   'RECEITAS ':'tributo'}, inplace=True)

# Padroniza o nome da linha que contém a métrica de interesse como 'RECEITA FAZENDARIA TOTAL'
d_rotulos = {'RECEITA ADMINISTRADA PELA RFB': 'RECEITA FAZENDARIA TOTAL',
     'RECEITAS ADMINISTRADAS PELA RFB': 'RECEITA FAZENDARIA TOTAL',
     'RECEITA ADMINISTRADA PELA SRF': 'RECEITA FAZENDARIA TOTAL',
     'RECEITA ADMINISTRADA': 'RECEITA FAZENDARIA TOTAL',
     'SUBTOTAL [A]': 'RECEITA FAZENDARIA TOTAL'}
df['tributo'].replace(d_rotulos, inplace=True)
```

Figura 3 - Ajuste nos rótulos da variável de interesse – notebook “02. Gera dataset de arrecadação a partir das planilhas.ipynb”

A solução adotada foi identificar todos os rótulos utilizados para localizar a linha de interesse ao longo dos anos e atribuir um novo nome padronizado a todas elas como 'RECEITA FAZENDARIA TOTAL', antes de realizar a transposição, conforme Figura 3.

Os ajustes após a transposição consistiram em utilizar a primeira linha de dados como nome das colunas, descartando-a em seguida juntamente com a linha de totais, que apenas representa a soma dos estados, e por fim atribuído o nome ‘estado’ à primeira coluna.

Foi acrescentada ainda uma coluna com a data, para identificar o ano e o mês de ocorrência da arrecadação, que futuramente será usada para compor o índice da série temporal. Para a consolidação do dataframe com todos os meses, foram mantidas apenas as colunas que atenderam ao escopo do presente estudo.

```
# Utiliza a primeira linha de dados para rótulo das colunas
df_t.columns = df_t.iloc[0]
df_t.drop([0,28], inplace=True)
df_t.rename(columns={'tributo':'estado'}, inplace=True)

# Cria uma coluna com o data (mês) de referência
df_t['data'] = pd.to_datetime(ano + '-' + mes)

#Agrega as colunas de interesse do dataframe no dicionário com a chave AAAAMM
d_df[ano+mes] = df_t[['data', 'estado', 'RECEITA FAZENDARIA TOTAL']]
```

Figura 4 - Ajuste no nome das colunas, nova coluna com a data e agregação em dicionário

Antes de exportar o dataset para um arquivo CSV, os valores da arrecadação foram colocados em base R$1 milhão. Destaque-se que não há valores ausentes (*missing values*) e a coluna "estado" tem, como valores únicos, as siglas dos 26 estados mais o Distrito Federal.

O resultado foi um dataset com **5.184 linhas e 3 colunas**, tendo sido armazenado no arquivo "arrecadação_uf.csv", com as seguintes características:

Nome da coluna	Descrição	Tipo
data	Data de apropriação da arrecadação (mensal)	Data
estado	Refere-se aos estados brasileiros mais o DF.	Texto
arrecadacao	Valor da arrecadação fazendária total, na base R$1 milhão	Número decimal

2.2. Série histórica do PIB

O Produto Interno Bruto - PIB é a soma de todos os bens e serviços finais produzidos por um país, estado ou cidade, geralmente em um ano[4]. O PIB do Brasil em 2020 foi de R$7,4 trilhões.

[4] Produto Interno Bruto – PIB. Disponível em: https://www.ibge.gov.br/explica/pib.php. Acesso em: 01 abr. 2021.

Por inferência, essa parece ser uma medida que pode influenciar na arrecadação tributária de um país, estado ou município. Como o escopo desse trabalho é a arrecadação federal, serão utilizados os dados do PIB Brasil.

As informações a respeito do PIB são divulgadas pelo IBGE e estão disponíveis na Internet, no Sistema de Contas Nacionais Trimestrais - SCNT[5] que, no menu lateral, disponibiliza o item Tabelas.

Em 05 de abril de 2021, estava disponível o arquivo “Tabelas Completas – 4º trimestre 2020”, acessível para download no endereço https://ftp.ibge.gov.br/Contas_Nacionais/Contas_Nacionais_Trimestrais/Tabelas_Completas/Tab_Compl_CNT.zip

Descompactado, há um único arquivo “Tab_Compl_CNT_4T20.xls” contendo várias planilhas. Da análise das informações, optou-se por utilizar a planilha “Valores Correntes”, que tem na sua coluna “R” o valor corrente do PIB Brasil.

2.2.1. Tratamento dos dados do PIB e construção do dataset

Para transformar essa planilha num dataset, de forma que possa ser combinado com os demais, foi elaborado o notebook “**03. Gera dataset do PIB a partir da planilha SCNT.ipynb**”.

Observa-se que os dados estão disponíveis de forma trimestral, tendo a cada 04 linhas uma quinta que realiza a soma anual; essas linhas de consolidação precisaram ser localizadas e apagadas.

Período	PIB
1995	**705.992**
1996.I	**189.323**
1996.II	**204.611**
1996.III	**221.513**
1996.IV	**239.316**
1996	**854.764**
1997.I	**219.117**

periodo	PIB
1996.I	189323.299148
1996.II	204610.728455
1996.III	221513.234375
1996.IV	239316.345834
1997.I	219117.049382
1997.II	232889.544198

Figura 5 - Planilha "Valores Correntes" do PIB carregada em um dataframe sem as linhas anuais

[5] Sistema de Contas Nacionais Trimestrais – SCNT. Disponível em: https://www.ibge.gov.br/estatisticas/economicas/contas-nacionais/9300-contas-nacionais-trimestrais.html. Acesso em 22 abr. 2021.

A próxima etapa consistiu em criar a coluna ‘data’, do tipo datetime, com a transformação da identificação dos períodos de '1996.I' para '1996-01-01', possibilitando o descarte da coluna ‘periodo’.

```
# Função criada para transformar '1996.I' para formato datetime '1996-01-01'
def converte_trimestre(periodo):
    ano, num_trimestre = periodo.split('.')
    d_conversao = {'I': '-01', 'II':'-04', 'III': '-07', 'IV': '-10'}
    trimestre = pd.to_datetime(f'{ano}{d_conversao[num_trimestre]}')
    return trimestre

# Cria a coluna data
df_pib['data'] = df_pib[['periodo']].applymap(converte_trimestre)
```

	periodo	PIB	data
1	1996.I	189323.299148	1996-01-01
2	1996.II	204610.728455	1996-04-01
3	1996.III	221513.234375	1996-07-01
4	1996.IV	239316.345834	1996-10-01
6	1997.I	219117.049382	1997-01-01
7	1997.II	232889.544198	1997-04-01

Figura 6 - Conversão dos períodos para formato datetime

Antes de exportar o dataset para um arquivo CSV, os dados foram truncados para o período definido no escopo (2004 a 2019). Destaque-se que não há valores ausentes (*missing values*).

O resultado do dataset final foi armazenado no arquivo “pib.csv”, com as seguintes características:

Nome da coluna	Descrição	Tipo
data	Data de aferição da média trimestral do PIB (mensal)	Data
PIB	Valor corrente do PIB no trimestre (em R$ 1 milhão)	Número decimal

2.3. Série histórica do IPCA

Produzido continuamente no âmbito do Sistema Nacional de Índices de Preços ao Consumidor – SNIPC, o Índice Nacional de Preços ao Consumidor Amplo – IPCA[6] tem por objetivo medir a inflação de um conjunto de produtos e serviços comercializados no varejo, referentes ao consumo pessoal das famílias. Uma vez que há vários tributos cuja base de cálculo é o valor da nota fiscal de produto e serviços, assume-se que seja uma segunda medida interessante para auxiliar na melhoria da qualidade de predições futuras da arrecadação.

[6] Índice Nacional de Preços ao Consumidor Amplo – IPCA. Disponível em: https://www.ibge.gov.br/estatisticas/economicas/precos-e-custos/9256-indice-nacional-de-precos-ao-consumidor-amplo.html. Acessado em 02 abr. 2021.

Diferentemente da coleta do PIB, foi utilizado dessa vez o menu “Downloads” na lateral esquerda, pasta IPCA -> Serie_Historica e em seguida escolhido o arquivo “ipca_SerieHist.zip”[7].

Em acesso realizado em 13/04/2021 estava disponível o arquivo “ipca_202103SerieHist.xls”, com a série histórica do IPCA desde janeiro 1994 até março de 2021.

2.3.1. Tratamento dos dados do IPCA e construção do dataset

Para possibilitar a transformação da planilha com o IPCA em um dataset adequado para as próximas etapas do presente trabalho, foi desenvolvido o notebook “**04. Gera dataset do IPCA a partir da planilha série histórica.ipynb**”

A planilha original está formatada de maneira a facilitar a impressão, paginada a cada 5 anos, com repetição dos cabeçalhos, colunas sem preenchimento e linhas em branco.

A carga da planilha se iniciou da linha 9, com o conteúdo das colunas de interesse “A:H” e foi armazenado num dataframe Pandas. Conforme esperado foram identificados vários problemas relacionados à formatação da planilha, deixando várias linhas e colunas sem valores (NaN), conforme Figura 7.

SÉRIE HISTÓRICA DO IPCA

(continua)

ANO	MÊS	NÚMERO ÍNDICE (DEZ 93 = 100)	VARIAÇÃO (%)				
			NO MÊS	3 MESES	6 MESES	NO ANO	12 MESES
1994	JAN	141,31	41,31	162,13	533,33	41,31	2.693,84
	FEV	198,22	40,27	171,24	568,17	98,22	3.035,71
	MAR	282,96	42,75	182,96	602,93	182,96	3.417,39
	ABR	403,73	42,68	185,71	648,92	303,73	3.828,49
	MAI	581,49	44,03	193,36	695,71	481,49	4.331,19
	JUN	857,29	47,43	202,97	757,29	757,29	4.922,60
	JUL	915,93	6,84	126,87	548,17	815,93	4.005,08
	AGO	932,97	1,86	60,44	370,67	832,97	3.044,89
	SET	947,24	1,53	10,49	234,76	847,24	2.253,15
	OUT	972,06	2,62	6,13	140,77	872,06	1.703,17
	NOV	999,37	2,81	7,12	71,86	899,37	1.267,54
	DEZ	1016,46	1,71	7,31	18,57	916,46	916,46
1995	JAN	1033,74	1,70	6,35	12,86	1,70	631,54
	FEV	1044,28	1,02	4,49	11,93	2,74	426,83
	MAR	1060,47	1,55	4,33	11,95	4,33	274,78

ano	mes	indice_dez93	no_mes	3_meses	6_meses	no_ano	12_meses
1994	JAN	141.31	41.31	162.13	533.33	41.31	2693.84
NaN	FEV	198.22	40.27	171.24	568.17	98.22	3035.71
NaN	MAR	282.96	42.75	182.96	602.93	182.96	3417.39
NaN	ABR	403.73	42.68	185.71	648.92	303.73	3828.49
NaN	MAI	581.49	44.03	193.36	695.71	481.49	4331.19
NaN	JUN	857.29	47.43	202.97	757.29	757.29	4922.6
NaN	JUL	915.93	6.84	126.87	548.17	815.93	4005.08
NaN	AGO	932.97	1.86	60.44	370.67	832.97	3044.89
NaN	SET	947.24	1.53	10.49	234.76	847.24	2253.15
NaN	OUT	972.06	2.62	6.13	140.77	872.06	1703.17
NaN	NOV	999.37	2.81	7.12	71.86	899.37	1267.54
NaN	DEZ	1016.46	1.71	7.31	18.57	916.46	916.46
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1995	JAN	1033.74	1.7	6.35	12.86	1.7	631.54
NaN	FEV	1044.28	1.02	4.49	11.93	2.74	426.83

Figura 7 - Planilha da série histórica do IPCA carregada em um dataframe

[7] IPCA Série Histórica. Disponível em: https://ftp.ibge.gov.br/Precos_Indices_de_Precos_ao_Consumidor/IPCA/Serie_Historica/ipca_SerieHist.zip. Acesso em 14 abr. 2021.

Para eliminar as linhas que estavam com cabeçalhos duplicados ou representavam espaços em branco, foi utilizada a coluna “índice_dez93” como referência. Uma vez que é um índice cumulativo, deve ter sempre um valor numérico. Foi atribuído “NaN” em todas as linhas em que não houvesse números válidos, o que possibilitou a eliminação de todas as linhas que, nessa coluna, tivessem valores ausentes.

	ano	mes	indice_dez93	no_mes	3_meses	6_meses	no_ano	12_meses
62	NaN	NOV	1453.4	-0.12	-0.32	-0.93	1.32	1.76
63	NaN	DEZ	1458.2	0.33	0.23	-0.62	1.65	1.65
64	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
65		NaN	NaN	NaN	NaN	NaN	NaN	NaN
66	SÉRIE HISTÓRICA DO IPCA	NaN	NaN	NaN	NaN	NaN	NaN	NaN
67	NaN	NaN	NaN	NaN	NaN	NaN	NaN	(continuação)
68	NaN	NaN	NaN	NaN	NaN	VARIAÇÃO	NaN	NaN
69	ANO	MÊS	NÚMERO ÍNDICE	(%)	NaN	NaN	NaN	NaN
70	NaN	NaN	(DEZ 93 = 100)	NO	3	6	NO	12
71	NaN	NaN	NaN	MÊS	MESES	MESES	ANO	MESES
72	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
73	1999	JAN	1468.41	0.7	0.91	0.2	0.7	1.65

	ano	mes	indice_dez93
62	NaN	NOV	1453.40
63	NaN	DEZ	1458.20
64	NaN	NaN	NaN
65		NaN	NaN
66	SÉRIE HISTÓRICA DO IPCA	NaN	NaN
67	NaN	NaN	NaN
68	NaN	NaN	NaN
69	ANO	MÊS	NaN
70	NaN	NaN	NaN
71	NaN	NaN	NaN
72	NaN	NaN	NaN
73	1999	JAN	1468.41

Figura 8 - Atribuição de NaN em linhas não numéricas da coluna indice_dez93

Restou para ajuste o preenchimento dos anos nos meses de FEV a DEZ, uma vez que na planilha original só estão registrados no mês de JAN (Figura 7). Em seguida foi realizado um mapeamento para transformar as abreviações dos meses JAN a DEZ em números de 1 a 12.

O correto preenchimento das colunas “ano” e “mes” possibilitou a criação de uma coluna ‘data’ no formato AAAA-MM-DD do tipo datetime e a eliminação das colunas ‘ano’ e ‘mes’.

Antes de exportar o dataset para um arquivo CSV, os dados foram truncados para o período definido no escopo (2004 a 2019). Destaque-se que não há valores ausentes (*missing values*). O dataset resultante foi armazenado no arquivo “ipca.csv”, com as seguintes características:

Nome da coluna	Descrição	Tipo
Data	Data de aferição do IPCA (mensal)	Data
Índice_dez93	Número índice tomando por base o valor 100 para o mês de dezembro de 1993.	Número decimal

no_mes	Variação (%) do IPCA observado no mês	Número decimal
3_meses	Variação (%) do IPCA em 3 meses	Número decimal
6_meses	Variação (%) do IPCA em 6 meses	Número decimal
no_ano	Variação (%) do IPCA no ano	Número decimal
12_meses	Variação (%) do IPCA em 12 meses	Número decimal

3. Análise e Exploração dos Dados

A presente análise exploratória foi realizada com o auxílio do notebook Python "**05. Análise e exploração dos dados de arrecadação.ipynb**" e do script em R "**05.1. Detecção de Outliers com R - forecast - tsoutliers.R**".

Iniciou-se a análise pelo carregamento para um dataframe Pandas do arquivo contendo o dataset gerado no item 2.1 "arrecadacao_uf.csv", utilizando a coluna data como índice.

```
# Verifica o tipo das colunas, do índice e ocorrência de NaN
df_uf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 5184 entries, 2004-01-01 to 2019-12-01
Data columns (total 2 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   estado       5184 non-null   object
 1   arrecadacao  5184 non-null   float64
dtypes: float64(1), object(1)
memory usage: 121.5+ KB
```

```
#Verifica estatísticas descritivas
df_uf.describe()
```

	arrecadacao
count	5184.00
mean	2025.12
std	4954.98
min	5.01
25%	102.75
50%	362.19
75%	1361.51
max	52581.76

Figura 9 - Informações da estrutura e estatística descritiva do dataframe

Do lado esquerdo observamos que o índice é do tipo *DatetimeIndex* e está abrangendo o período escopo do trabalho, a coluna numérica 'arrecadacao' é do tipo *float64* e a coluna 'estado' é do tipo *object*. O dataframe não tem ocorrência de valores ausentes.

Observa-se nas estatísticas descritivas que o desvio padrão é quase 2,5x a média, com valores mínimos e máximos muito díspares, denotando que há uma irregularidade bastante grande entre as arrecadações dos estados ao longo do tempo.

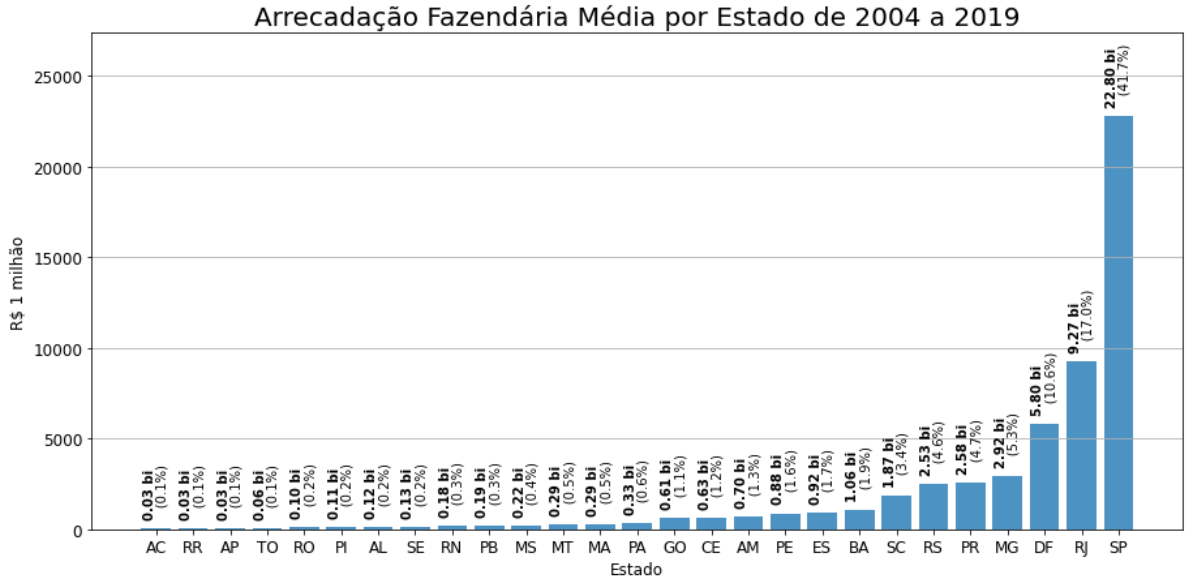


Figura 10 - Média aritmética da arrecadação fazendária federal por estado, período 2004 a 2019

O gráfico de barras da Figura 10 comprova a disparidade de arrecadação entre os estados, com a média da arrecadação variando de R$ 0,03 bilhões (0,01%) no Acre, Roraima e Amapá, até R$22,8 bilhões (41,7%) em São Paulo.

A região Sudeste ocupa a primeira posição com quase 66% de representatividade e aproximadamente R$36 bilhões de arrecadação fazendária média. No outro extremo está a região Norte, com R$1,3 bilhões e 2,3% de participação na arrecadação fazendária nacional.

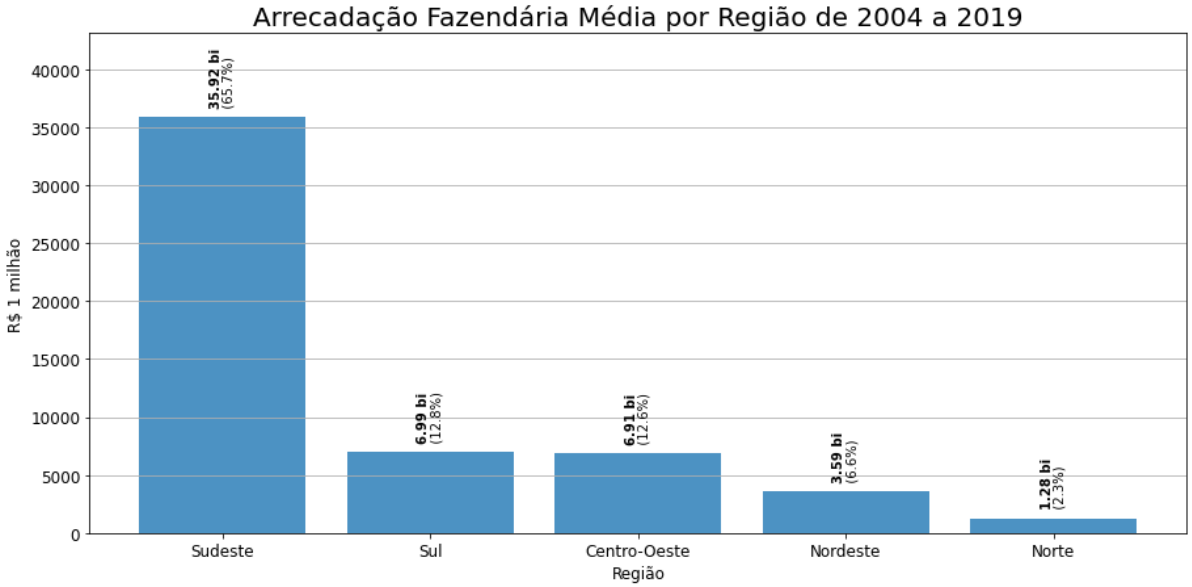


Figura 11 - Média aritmética da arrecadação fazendária federal por região, período 2004 a 2019

3.1. Decomposição STL e Tratamento de Outliers

Inicialmente será traçado um gráfico do comportamento da coluna “arrecadacao” ao longo do tempo.

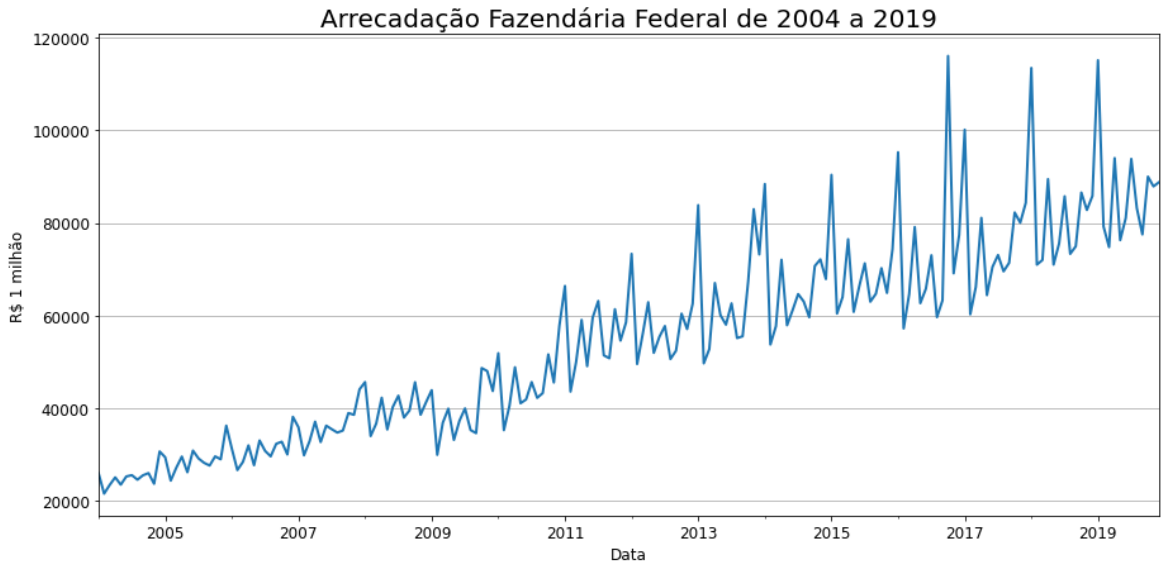


Figura 12 - Arrecadação Fazendária Federal de 2004 a 2019

Percebe-se na figura 12 uma tendência bem definida de crescimento ao longo do tempo, com forte sazonalidade. Passou-se então à decomposição da série temporal da arrecadação, utilizando STL - *Season-Trend decomposition using LOESS*, para identificar as características da série temporal de tendência (*Trend*), sazonalidade (*Season*) e os resíduos (*Resid*) da decomposição.

Do primeiro quadro (*Trend*) da figura 13 se confirma a forte tendência da série temporal da arrecadação, sempre crescente, tendo um ponto de inflexão em 2008/2009, provavelmente ligado à crise econômica[8] de 2009, iniciada no mercado imobiliário dos Estados Unidos e que se refletiu em vários setores da economia brasileira.

[8] Última recessão da economia brasileira ocorreu durante crise de 2009. Disponível em: https://www.correiobraziliense.com.br/app/noticia/economia/2014/08/29/internas_economia,444682/ultima-recessao-da-economia-brasieira-ocorreu-durante-crise-de-2009.shtml. Acesso em 03 abr. 2021.

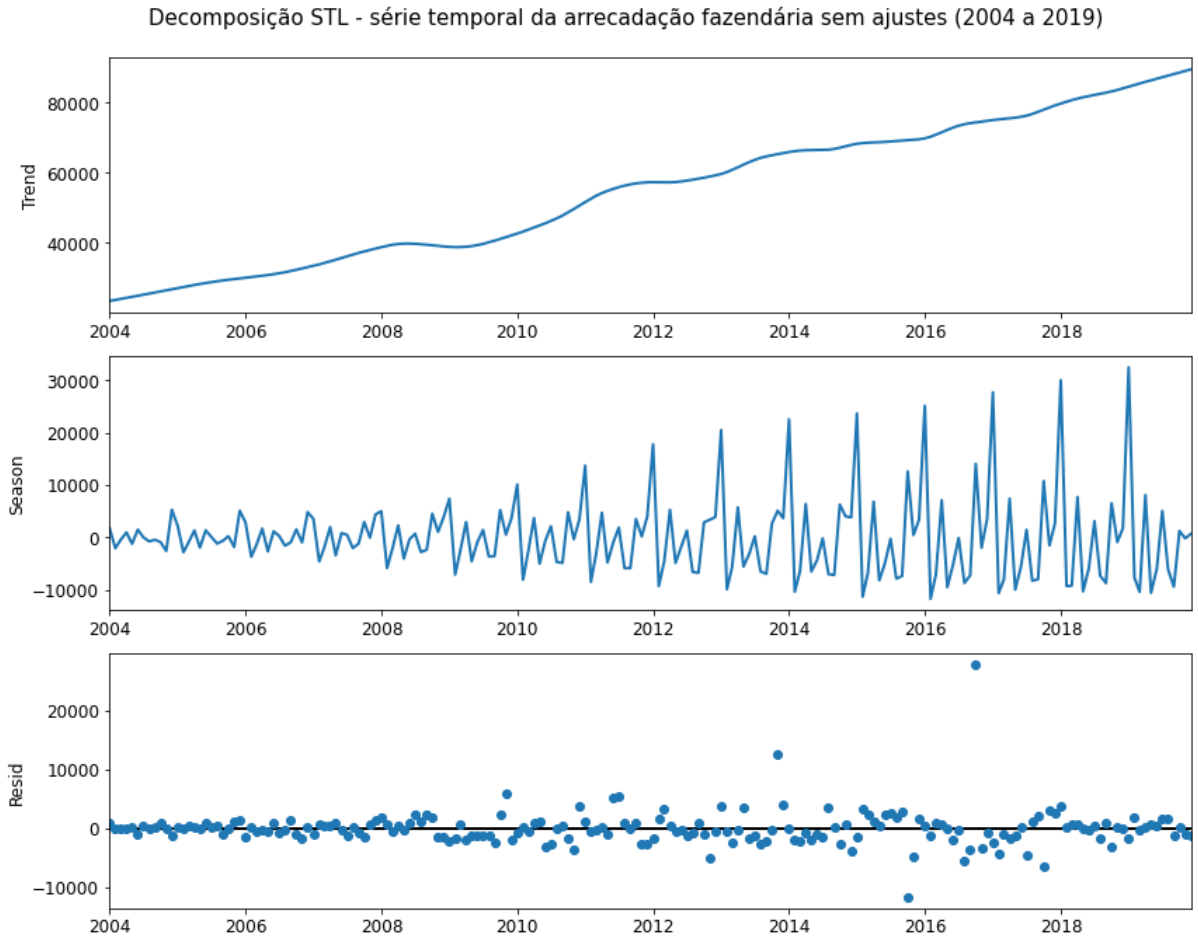


Figura 13 - Decomposição STL da série temporal da arrecadação

No segundo gráfico (*Season*) da mesma figura, fica muito evidente a componente sazonal da série. Esse comportamento é esperado em virtude da diversidade de vencimentos dos tributos, com periodicidade mensal, trimestral, anual, dentre outros. Mais detalhes podem ser verificados na Agenda Tributária, disponível no sítio da Receita Federal na Internet.

No entanto, o que chama mais atenção é o gráfico de dispersão dos resíduos (*Resid*), com dois outliers muito discrepantes, um próximo de 2014 e outro próximo a 2017. Para identificar estatisticamente os outliers da série temporal, foi necessário recorrer ao pacote "tsoutliers"[9] que, até o momento de elaboração desse trabalho, só estava disponível na plataforma R.

[9] Package 'tsoutliers'. Disponível em: https://cran.r-project.org/web/packages/tsoutliers/tsoutliers.pdf. Acesso em: 03 abr. 2021.

Para realizar a análise dos outliers foi desenvolvido um script em R, denominado "**05.1. Detecção de Outliers com R - forecast - tsoutliers.R**", no qual foram realizados os mesmos passos do Python e, a seguir, a detecção dos outliers.

O resultado da execução retorna os índices da série considerados outliers, bem como propõe os valores que deveriam ser substituídos:

```
> outliers$index
[1]   1  13  25  37 119 154 169 181

> outliers$replacements
[1]  44040.39  45290.19  49262.09  51899.12  64904.65  73774.69  96208.84 101070.02
```

Voltamos ao notebook 05 no Python e identificamos esses valores no dataframe atual, tomando o cuidado de subtrair 1 (um) ao valor dos índices identificados no R que começam a partir de 1 e no Python a partir do 0 (zero) conforme figura 14.

```
# Índices dos outliers detectados pelo tsoutliers no R
lista = [1, 13, 25, 37, 119, 154, 169, 181]
# Os índices no Python começam em zero e no R em 1, necessário subtrair 1 dos índices
lista = [x-1 for x in lista]
df.iloc[lista][['arrecadacao']]
```

data	arrecadacao
2004-01-01	25927.01
2005-01-01	29382.55
2006-01-01	31256.47
2007-01-01	35858.85
2013-11-01	82983.56
2016-10-01	116084.25
2018-01-01	113487.90
2019-01-01	115156.06

Figura 14 - Outliers identificados na série temporal da arrecadação usando tsoutliers no R

Nota-se que o *tsoutliers* apontou seis das oito ocorrências de outliers em janeiro, o que parecem representar falsos positivos. Apesar dessa aparente distorção, foram indicados os dois outliers visualizados no gráfico dos resíduos, ocorridos nos meses **2013-11** e **2016-10**.

No intuito de melhorar a identificação dos outliers na série temporal, aplicou-se uma transformação Box-Cox à série, deixando que o próprio *tsoutliers* chegasse à conclusão do valor ótimo para o λ (lambda).

```
> outliers = tsoutliers(ts_arrecadacao, lambda = "auto")

> outliers$index
[1]  12  18  24  25  37  62  71 119 154

> outliers$replacements
[1] 25451.31 28161.57 30946.52 35790.90 42823.68 32788.80 42863.25 64272.33 74378.36
```

A nova relação de outliers parece mais bem distribuída e nota-se a repetição apenas dos pontos 119 e 154, referente aos meses **2013-11** e **2016-10**. Considerando esses dois pontos terem sido os únicos coincidentes nas duas execuções, eles serão considerados outliers e ajustados na série temporal da arrecadação.

```
# Índices dos outliers detectados pelo tsoutliers no R utilizando a transformação Box-Cox
lista = [12, 18, 24, 25, 37, 62, 71, 119, 154]
# Os índices no Python começam em zero e no R em 1, necessário subtrair 1 dos índices
lista = [x-1 for x in lista]
df.iloc[lista][['arrecadacao']]
```

data	arrecadacao
2004-12-01	30632.51
2005-06-01	30823.35
2005-12-01	36212.19
2006-01-01	31256.47
2007-01-01	35858.85
2009-02-01	29907.37
2009-11-01	48006.71
2013-11-01	82983.56
2016-10-01	116084.25

Figura 15 - Outliers apontados pelo *tsoutliers* na série temporal da arrecadação com a transformação Box-Cox

Os valores substitutos indicados pelo *tsoutliers* estão relativamente próximos nas duas execuções e poderiam ser adotadas as médias aritméticas de cada par, porém a Secretaria da Receita Federal do Brasil realiza análises e disponibiliza

relatórios[10] mensais com o resultado da arrecadação e, nesse caso particular, os exatos valores das atipicidades estão disponíveis.

Segundo o relatório de análise da arrecadação mensal RFB de **outubro-2016**, o desempenho da arrecadação, tanto no mês de outubro quanto no período acumulado, foi bastante influenciado pelo regime especial de regularização cambial e tributária – RERCT[11].

Duas rubricas tiveram um aumento expressivo quando comparadas com o mesmo mês do ano anterior: Outras receitas administradas pela RFB (R$ 24.069 milhões/905,58%): resultado explicado pelo recolhimento, em outubro/16, de aproximadamente R$22,5 bilhões, a título de recolhimento de multa do regime especial de regularização cambial e tributária – RERCT; IRPJ (R$ 34.744 milhões/+174,91%): esse resultado deveu-se, basicamente ao recolhimento, em outubro, de R$22,5 bilhões, a título de IRPJ, relativo ao RERCT. Totalizando a relevante monta de **R$45 bilhões** nesse mês, conforme confirmado na Figura 16.

ARRECADAÇÃO DAS RECEITAS FEDERAIS
PERÍODO: OUTUBRO - 2016/2015

UNIDADE: R$ MILHÕES

RECEITAS	OUTUBRO				JANEIRO A OUTUBRO			
	ARRECADAÇÃO (PREÇOS CORRENTES)		VARIAÇÃO [A]/[B]%		ARRECADAÇÃO (PREÇOS CORRENTES)		VARIAÇÃO [C]/[D]%	
	2016 [A]	2015 [B]	NOMINAL	REAL (IPCA)	2016 [C]	2015 [D]	NOMINAL	REAL (IPCA)
ADMINISTRADAS PELA RFB	**146.369**	**99.248**	**47,48**	**36,71**	**1.039.744**	**977.977**	**6,32**	**(2,73)**
. RERCT	45.069	-	-	-	46.823	-	-	-
. DEMAIS	101.300	99.248	2,07	(5,38)	992.921	977.977	1,53	(7,04)
ADMINISTRADAS POR OUTROS ÓRGÃOS	**2.432**	**4.282**	**(43,22)**	**(47,36)**	**20.307**	**26.606**	**(23,67)**	**(30,16)**
TOTAL	**148.801**	**103.530**	**43,73**	**33,24**	**1.060.052**	**1.004.583**	**5,52**	**(3,46)**

Figura 16 - Quadro[12] da arrecadação de outubro de 2016 e de 2015, destacando a influência do RERCT no desempenho da arrecadação.

[10] Relatórios do Resultado da Arrecadação. Disponível em: https://receita.economia.gov.br/dados/receitadata/arrecadacao/relatorios-do-resultado-da-arrecadacao. Acesso em 03 abr. 2021.
[11] Regime especial de regularização cambial e tributária. Disponível em https://receita.economia.gov.br/acesso-rapido/legislacao/legislacao-por-assunto/rerct. Acesso em 03 abr. 2021.
[12] Fonte: Secretaria da Receita Federal do Brasil.

Retrocedendo a **novembro-2013**, segundo o relatório de análise da arrecadação mensal elaborado pela RFB, o resultado é explicado, principalmente, pela adesão de contribuintes aos parcelamentos instituídos pela Lei 12.865/13, e consequente pagamento de **R$20 bilhões**, conforme quadro da Figura 17.

ARRECADAÇÃO DA LEI Nº 12.865/13
PERÍODO: NOVEMBRO DE 2013
(A PREÇOS CORRENTES)

UNIDADE: R$ MILHÕES

DISCRIMINAÇÃO	R$ MILHÕES
REABERTURA LEI Nº 11.941/09 (ART. 17)	93,6
PIS/COFINS - ENTIDADES FINANCEIRAS (ART. 39)	12.076,4
EXCLUSÃO DO ICMS DA BASE DE CÁLCULO DO PIS/COFINS (ART. 39, § 1º)	614,9
IRPJ/CSLL - TBU (ART. 40)	7.571,8
TOTAL	**20.356,7**

Figura 17 – Tabela[13] com a arrecadação proveniente da adesão dos contribuintes ao parcelamento especial da Lei 12.865/13.

Os **valores atípicos** identificados nos relatórios da RFB nos meses 2013-11 e 2016-10, confirmados nas duas execuções do *tsoutliers*, foram subtraídos dos valores originais da série de arrecadação nos respectivos meses. A série ajustada foi decomposta e comparada com a anterior para verificação do impacto dos ajustes.

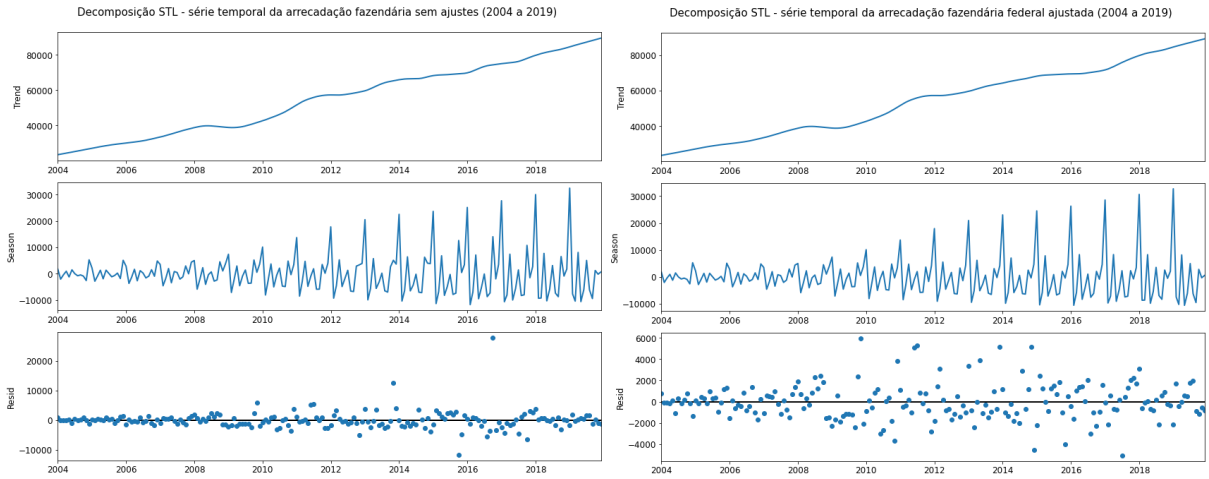


Figura 18 - Decomposição STL da série temporal antes (esquerda) e depois (direita) do ajuste dos outliers

[13] Fonte: Secretaria da Receita Federal do Brasil.

Não se identifica alterações relevantes na componente de tendência (*Trend*). A componente sazonal apresenta uma característica mais uniforme, evidenciada a partir de 2013. (Figura 18)

Destaca-se que, em virtude da magnitude dos outliers, a escala dos resíduos (*Resid*) do lado esquerdo está “achatando” completamente as demais observações. Após realizados os ajustes, o gráfico de dispersão do lado direito permite visualizar de forma mais adequada os resíduos.

Na figura 19, são comparadas as séries ajustada e original, traçadas no mesmo gráfico. Percebe-se claramente (em azul) que os dois picos de arrecadação estavam alheios à tendência e à sazonalidade dos demais anos, tendo a linha com a arrecadação ajustada (em laranja) ficado com um aspecto mais uniforme.

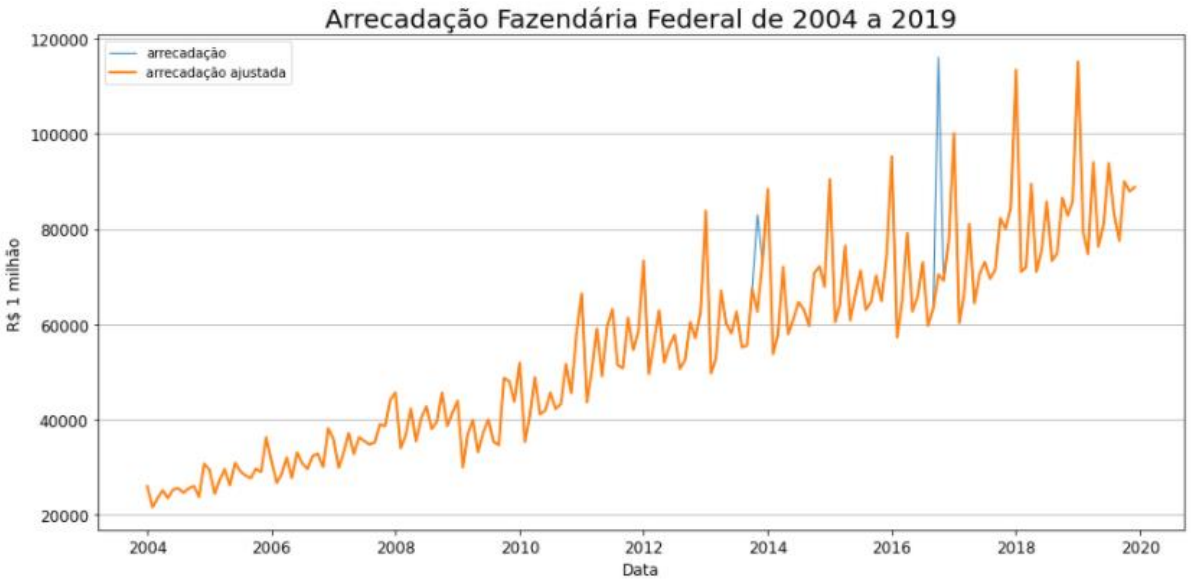


Figura 19 - Comparação entre a série temporal original e com ajuste dos outliers

3.2. Estacionariedade da Série

Uma suposição comum em muitas técnicas de predição em séries temporais é que os dados sejam estacionários, ou seja, que não apresentem tendências e, portanto, tenham suas características estatísticas, como média e variância, constantes ao longo do tempo.

Durante a análise da decomposição da série da arrecadação no subcapítulo anterior, ficou evidente que a série tem uma clara tendência de crescimento ao longo

do tempo. A característica de estacionariedade pode ser avaliada através do teste de Dickey-Fuller aumentado *ADF (Augmented Dickey-Fuller Test)*, conforme figura 20.

```
#Para varificar a estacionariedade da série, vamos usar o teste ADF - 'Augmented Dickey-Fuller Test'
from statsmodels.tsa.stattools import adfuller

# Função para realizar o teste ADF e introduzir labels aos resultados
def adfuller_test(x):
    result=adfuller(x)
    labels = ['ADF Test Statistic','p-value']
    for value,label in zip(result,labels):
        print(f'{label} : {value:.4f}')
    if result[1] <= 0.05:
        print('''Forte evidência para rejeitar a hipótese nula (Ho)
        Indicativo de que a série é estacionária.''')
    else:
        print('''Não é possível rejeitar a hipótese nula (Ho):
        Existe pelo menos uma raiz unitária, indicando série NÃO estacionária.''')

# Teste ADF sobre a série temporal da arrecadação
adfuller_test(df_ajustado['arrecadacao'])
```

```
ADF Test Statistic : 0.0929
p-value : 0.9656
Não é possível rejeitar a hipótese nula (Ho):
        Existe pelo menos uma raiz unitária, indicando série NÃO estacionária.
```

Figura 20 - Teste ADF aplicado sobre a série da arrecadação ajustada

Para realizar o teste utilizou-se a função adfuller do módulo statsmodels, statsmodels.tsa.stattools.adfuller. A hipótese nula do teste é de que existe pelo menos uma raiz unitária, indicando que a série é **não** estacionária. O teste concluiu com p-value de 0,9656, bem acima de 0,05, indicando que a série é de fato **não** estacionária.

```
# Calcula a diferença de 1 período
df_ajustado['arrecadacao_d1']=df_ajustado['arrecadacao'].diff(periods=1)

df_ajustado[['arrecadacao', 'arrecadacao_d1']].head()
```

data	arrecadacao	arrecadacao_d1
2004-01-01	25927.01	NaN
2004-02-01	21519.05	-4407.95
2004-03-01	23391.92	1872.87
2004-04-01	25041.10	1649.17
2004-05-01	23455.60	-1585.50

Figura 21 - Resultado da diferenciação com um período na série temporal da arrecadação ajustada

Uma técnica muito utilizada para transformar uma série não estacionária em estacionária consiste na diferenciação do período atual com o período anterior. Foi

então criada uma coluna, chamada ‘arrecadacao_d1’, para armazenar o resultado da diferenciação entre o período atual e o imediatamente anterior (lag 1), conforme Figura 21.

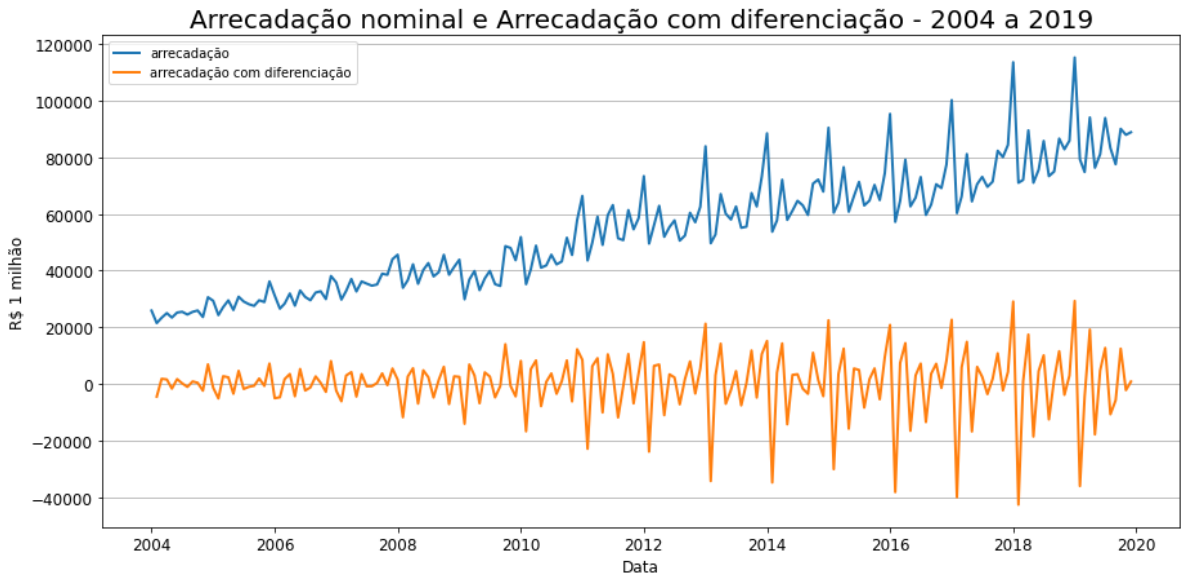


Figura 22 - Arrecadação ajustada nominal e com diferenciação em um período

Comparando-se os gráficos da arrecadação ajustada em valores nominais (em azul) com a sua diferenciação (em laranja), visualmente a tendência de crescimento parece ter sido eliminada.

Para averiguação, foi executado novamente o teste ADF, agora sobre a nova coluna ‘arrecadacao_d1’; observa-se um p-value 0,0024, indicando que a série se tornou **estacionária**.

```
# Vamos testar agora sobre a primeira diferenca

adfuller_test(df_ajustado['arrecadacao_d1'].dropna())
```

```
ADF Test Statistic : -3.8568
p-value : 0.0024
Forte evidência para rejeitar a hipótese nula (Ho)
        Indicativo de que a série é estacionária.
```

Figura 23 - Teste ADF aplicado sobre a primeira diferença da série da arrecadação ajustada

Essa informação será importante para as etapas seguintes do presente estudo, principalmente na aplicação dos modelos ARIMA detalhados no próximo capítulo.

3.3. Sazonalidade da Série

Na busca de *insigths* a respeito do comportamento sazonal da série, traçamos um gráfico (figura 24) onde cada linha representa um ano do período sob análise; no eixo ‘x’ estão os meses de ocorrência da arrecadação e no eixo ‘y’ o valor da arrecadação em milhões de Reais.

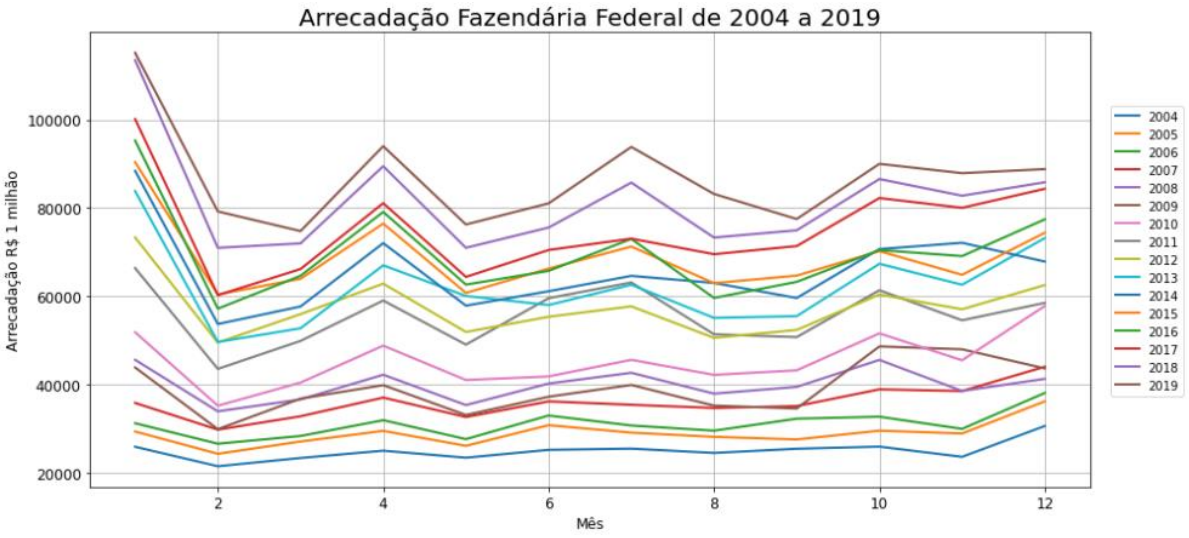


Figura 24 - Arrecadação de cada ano do período, traçados individualmente ao longo dos meses

O comportamento sazonal parece se repetir a cada ano, tendo como meses de maior arrecadação janeiro, abril, julho e outubro, o que pode ser confirmado calculando a arrecadação média de todos os anos em cada mês. (Figura 25)

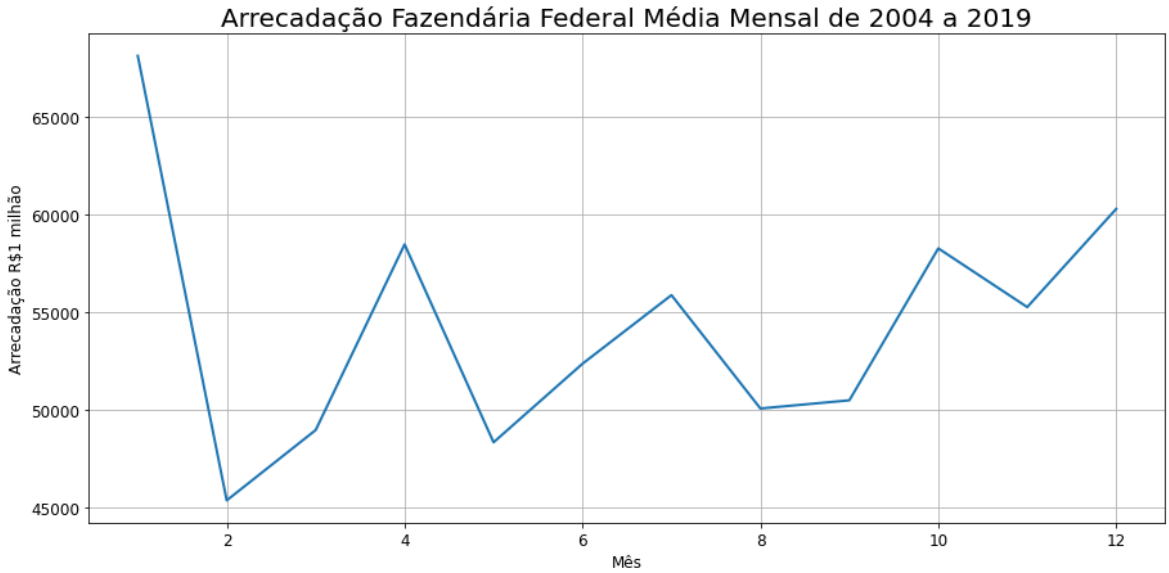


Figura 25 - Média mensal da arrecadação de 2004 a 2019

Realizando um comparativo da figura anterior com o comportamento do último ano da série (2019), percebe-se na figura 26 certa semelhança entre ambas, indicando que o valor de abril/2020 pode ser melhor previsto considerando o valor de abril/2019 e não do período imediatamente anterior (março/2020).

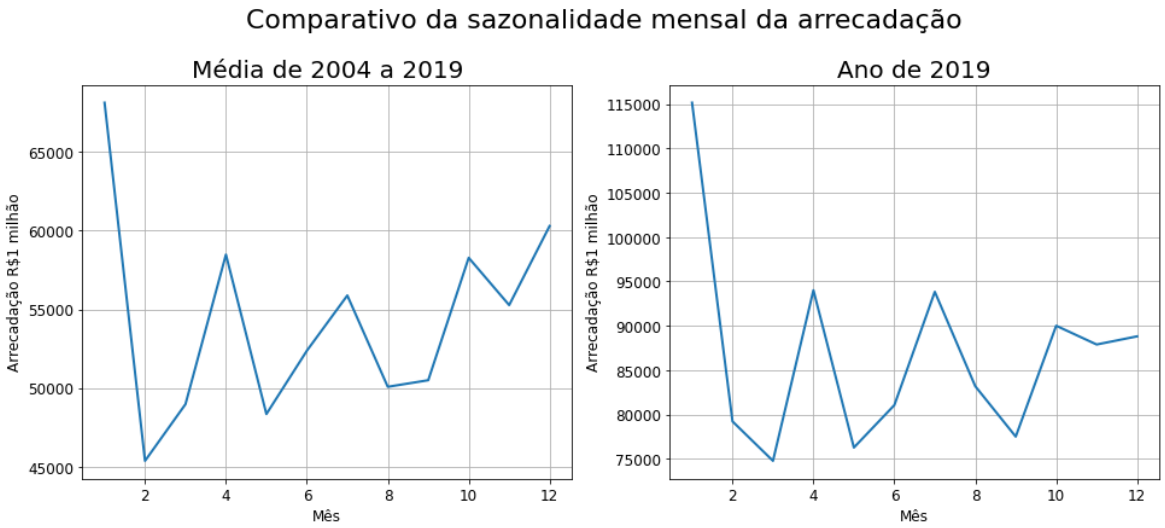


Figura 26 - Comparativo da média mensal da arrecadação de 2004 a 2019 com a arrecadação de 2019

4. Criação de Modelos de Machine Learning

Com o fito de identificar o melhor modelo preditivo aplicado à série temporal da arrecadação fazendária federal, serão criados modelos utilizando a família ARIMA, incluindo as variações SARIMA e SARIMAX, bem como será utilizado o pacote de previsão de séries temporais do Facebook, chamado Prophet[14].

Serão realizados experimentos utilizando apenas a variável contendo a arrecadação federal, série temporal **univariada**, assim como as combinações entre arrecadação, PIB e IPCA, série temporal **multivariada**.

O primeiro passo foi colocar em um único dataframe todas as colunas de interesse. Os arquivos 'arrecadacao_Brasil_ajustada.csv', 'pib.csv' e 'ipca.csv', gerados nos capítulos anteriores, foram carregados em datrafames individuais e, em

[14] Facebook Prophet. Disponível em: https://facebook.github.io/prophet/. Acesso em: 15 abr. 2021.

seguida, foram agregados em duas etapas, usando a coluna data como parâmetro de junção.

```
# Construir um Dataset agregando Arrecadação, PIB e IPCA

# Arrecadação + IPCA
df = pd.merge(df_arrecadacao, df_ipca, on="data")

# (Arrecadação + IPCA) + PIB
df = pd.merge(df, df_pib, on='data')
```

Figura 27 - Agregação dos dataframes da arrecadação, PIB e IPCA

O dataset resultante, depois de descartadas as colunas sem interesse, está representado na amostra impressa na figura 28.

data	arrecadacao	IPCA_dez93	IPCA	PIB
2004-01-01	25927.005459	2246.43	0.76	148261.162720
2004-02-01	21519.053577	2260.13	0.61	148261.162720
2004-03-01	23391.923997	2270.75	0.47	148261.162720
2004-04-01	25041.096858	2279.15	0.37	160598.320168
2004-05-01	23455.595482	2290.77	0.51	160598.320168

Figura 28 - Amostra do dataset utilizado nos modelos preditivos

Abaixo identificamos o formato e conteúdo das colunas do respectivo dataset.

Coluna	Descrição	Tipo
data	Data usada como índice mensal do dataset	Data
arrecadacao	Valor da arrecadação fazendária total (em R$1 milhão)	Número decimal
IPCA_dez93	Número índice tomando por base o valor 100 no mês de dezembro de 1993.	Número decimal
IPCA	Variação (%) do IPCA observada no mês	Número decimal
PIB	Valor corrente do PIB (em R$1 milhão)	Número decimal

Para validação das predições, o conjunto de dados foi segmentado em um dataset de treino e outro de teste. O dataset de treino ficou com o período de 2004 a 2015 (12 anos), enquanto o de teste com os últimos 4 anos (2016 a 2019), conforme figura 29.

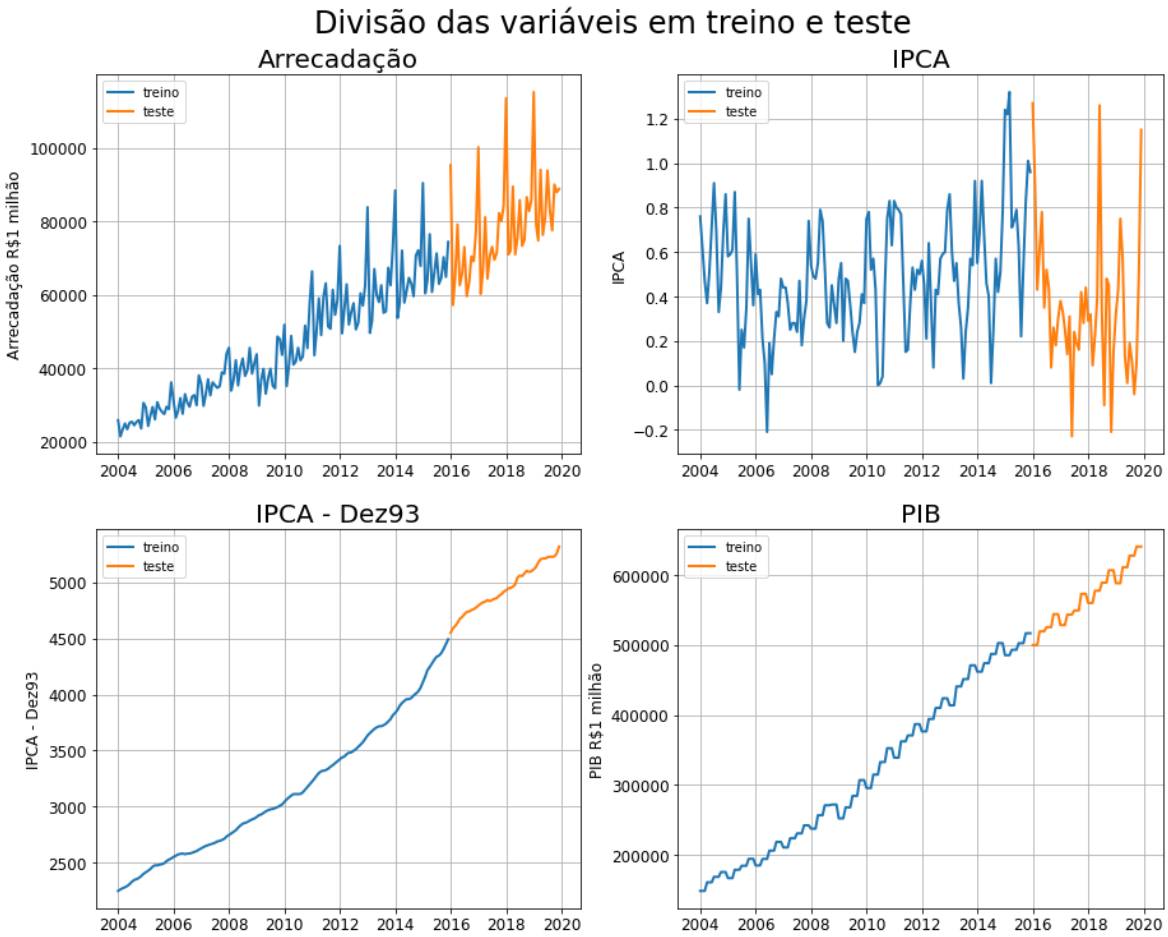


Figura 29 - Gráfico de cada variável na divisão do dataset em treino e teste[15]

4.1. Facebook Prophet

Segundo a documentação oficial do Facebook, disponível em https://facebook.github.io/prophet/, o Prophet é um software de código aberto disponível para Python e R, cuja finalidade é realizar previsões em séries temporais. O seu melhor desempenho ocorre em séries temporais com forte componente sazonal e com muitos períodos, sendo robusto para detecção de valores ausentes, mudanças de tendência e lidando bem com outliers.

Para implementação do modelo, o Prophet espera receber como entrada um dataframe padronizado, contendo apenas duas colunas: 'ds' com o período das

[15] Notebook: "07. Previsão da arrecadação com Auto ARIMA.ipynb"

observações (*datestamp*) e ‘y’ que representa o valor numérico que se deseja prever. Trata-se, portanto, de um modelo de previsão de séries temporais univariadas.

Utilizou-se os datasets de treino e de teste criados no item anterior, no entanto aproveitando-se apenas a coluna ‘arrecadacao’, renomeada para ‘y’. Destaca-se que o índice ‘data’ precisou ser transformado em uma coluna do dataframe com a denominação ‘ds’.

A execução é bastante direta e o desenvolvimento está detalhado no notebook ‘**06. Previsão da arrecadação com Facebook Prophet.ipynb**’. Inicia-se instanciando um novo objeto Prophet e chamando o método *fit()* com o dataframe que contém a série temporal. Tudo funciona de forma automática e o Prophet já identifica sazonalidade anual (*yearly*) e, por padrão, o modo sazonal aditivo (*additive*).

```
# Instancia objeto Prophet e cria o modelo
forecaster = Prophet()
model = forecaster.fit(train)

model.seasonalities
```

```
OrderedDict([('yearly',
              {'period': 365.25,
               'fourier_order': 10,
               'prior_scale': 10.0,
               'mode': 'additive',
               'condition_name': None})])
```

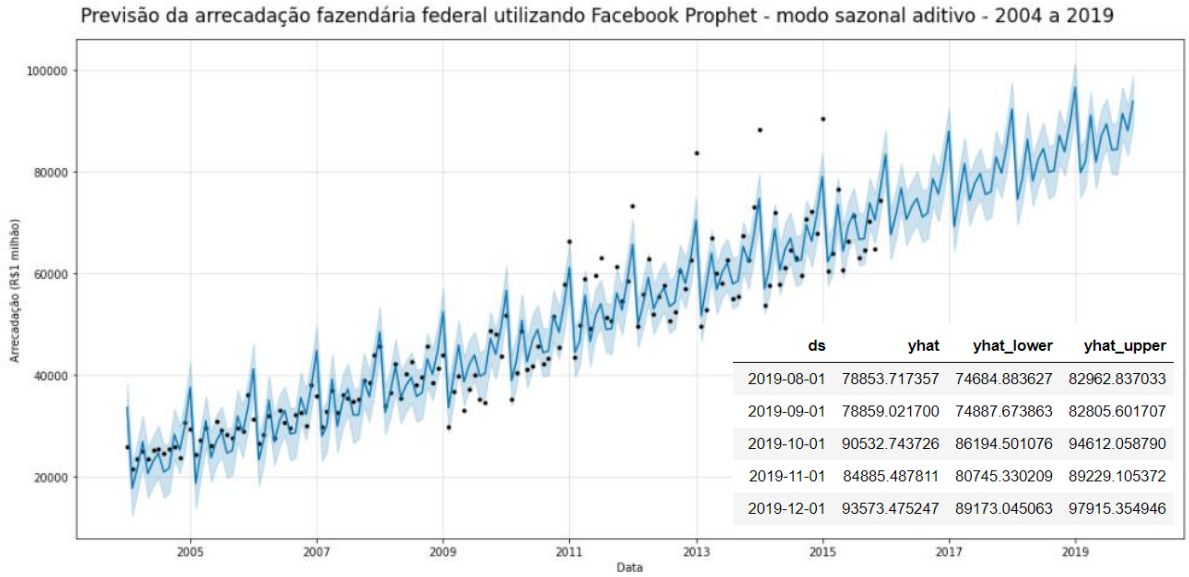
Figura 30 - Parâmetros iniciais para o modelo Facebook Prophet

Para realizar as previsões é necessário passar como parâmetro um dataframe apenas com a coluna ‘ds’, contendo os períodos desejados no futuro. Como já dividimos os dados em treino e teste, pode ser utilizada diretamente a respectiva coluna do dataframe de teste.

Todavia, o modelo disponibiliza o método ‘*make_future_dataframe’*, que gera a quantidade de períodos desejados e, por padrão, inclui os períodos de treino também. Isso torna possível visualizar o gráfico do ajuste do modelo no período completo, conforme figura 31. Uma amostra dos dados de previsão foi sobreposta ao gráfico.

A faixa azul clara representa o intervalo de confiança do modelo, definido por padrão em 95%, que vai do ‘yhat_lower’ ao ‘yhat_upper’. Os pontos pretos são as

ocorrências da arrecadação efetiva (treino) e a linha azul escura os valores previstos pelo modelo (yhat).



ds	yhat	yhat_lower	yhat_upper
2019-08-01	78853.717357	74684.883627	82962.837033
2019-09-01	78859.021700	74887.673863	82805.601707
2019-10-01	90532.743726	86194.501076	94612.058790
2019-11-01	84885.487811	80745.330209	89229.105372
2019-12-01	93573.475247	89173.045063	97915.354946

Figura 31 - Previsão da arrecadação fazendária federal utilizando Facebook Prophet - modo sazonal aditivo - 2004 a 2019

Observa-se que a grande maioria das ocorrências da arrecadação efetiva se encontra dentro do intervalo de confiança, com exceção dos meses de janeiro de 2012 a 2015. Esse parece ser um comportamento de tendência crescente, com impacto nos períodos futuros de predição; para esses casos se apresenta mais interessante utilizar o modo sazonal multiplicativo.

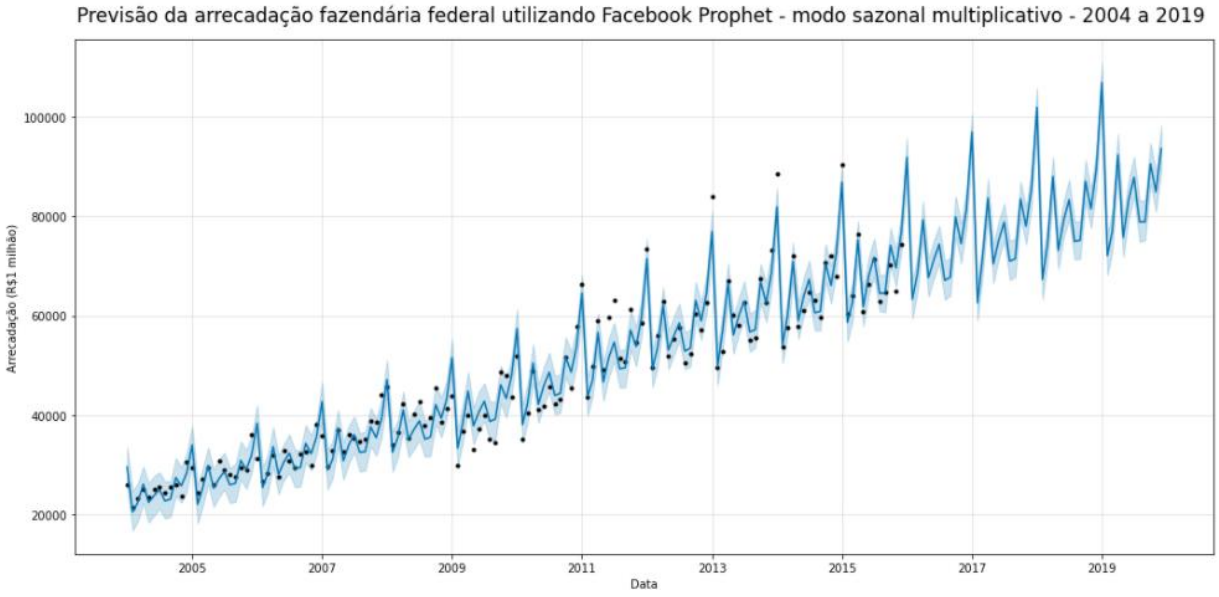


Figura 32 - Previsão da arrecadação fazendária federal utilizando Facebook Prophet - modo sazonal multiplicativo - 2004 a 2019.

De forma visual, aparentemente essa alteração de modelo melhorou a qualidade das previsões em períodos mais recentes, melhor capturando os picos ocorridos nos meses de janeiro, a partir de 2012. (Figura 32)

Para melhor identificar graficamente a qualidade das previsões, foram traçados conjuntamente os dados da arrecadação efetivamente ocorrida e das previsões realizadas em um conjunto de dados no futuro, não conhecidos por ambos os modelos ajustados.

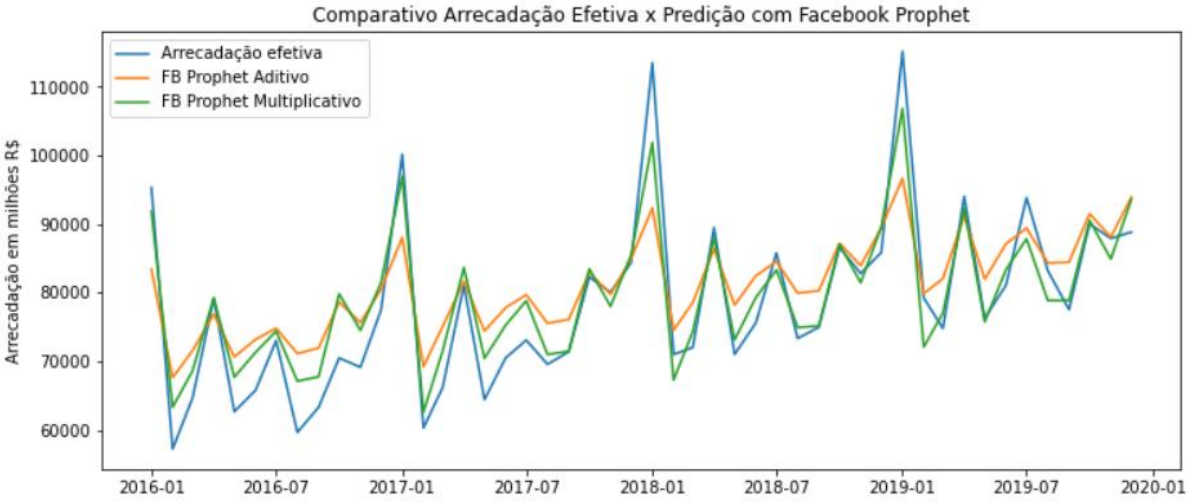


Figura 33 – Comparativo da série de arrecadação efetiva com as previsões do Facebook Prophet na base de teste, em modo sazonal aditivo e multiplicativo.

Percebe-se, na figura 33, que o modelo multiplicativo melhorou bastante a qualidade das previsões do dataset de teste, tanto nos picos como nos vales, o que indica ser um modelo preditivo mais adequado. Objetivando a certificação dessa afirmativa, passou-se a analisar os resíduos de ambos os modelos durante o treino.

Figura 34 – Comparativo da dispersão dos resíduos padronizados dos modelos aditivo e multiplicativo

Analisando o gráfico de dispersão dos resíduos padronizados da figura 34, ambos parecem flutuar de forma aleatória ao longo de zero. No entanto, o modelo aditivo teve maior espalhamento e, particularmente nos anos finais, o modelo multiplicativo apresentou resíduos bastante menores.

Na sequência comparou-se o histograma dos residuais de ambos os modelos, conforme a figura 35. Novamente o modelo multiplicativo parece superior, tendo os seus resíduos apresentado uma distribuição desejável, mais próxima da normal.

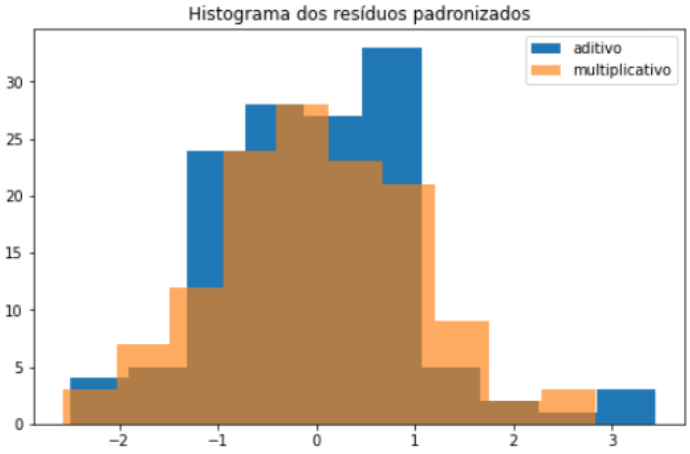


Figura 35 - Comparativo do histograma dos resíduos padronizados dos modelos aditivo e multiplicativo

Para confirmação, foi realizado o teste de Shapiro-Wilk nos resíduos de ambos os modelos, apontando um valor-p para o modelo aditivo de 0,0020, bem abaixo de 0,05, devendo ser rejeitada a hipótese de normalidade da distribuição. Já no modelo multiplicativo ficou em 0,8444, bem acima de 0,05, com indicação de que a hipótese de normalidade da distribuição não deve ser rejeitada.

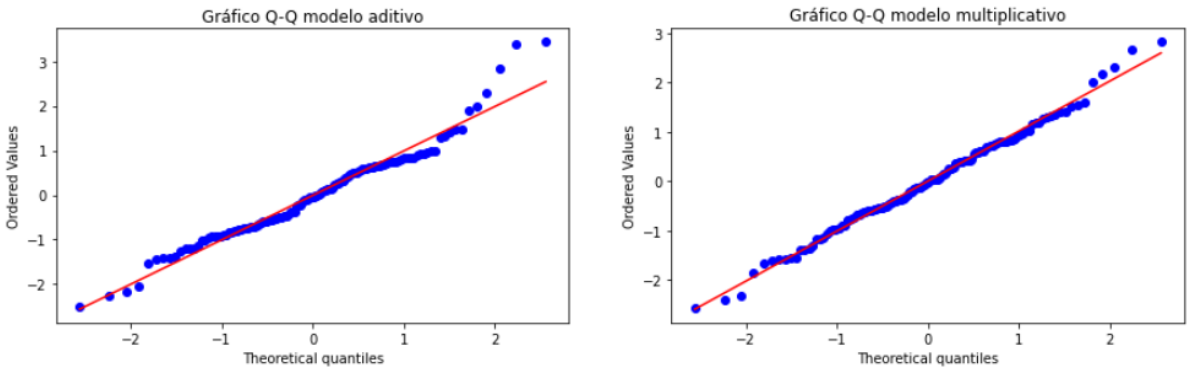


Figura 36 - Comparativo do gráfico Q-Q para os resíduos padronizados dos modelos aditivo e multiplicativo

Traçou-se também o gráfico Q-Q de probabilidades (figura 36), para comparar graficamente uma distribuição normal com as duas distribuições dos resíduos dos modelos. Novamente o modelo multiplicativo obteve melhor performance, tendo os pontos azuis, em geral, ficado mais alinhados com a linha vermelha.

Por fim, verificou-se o correlograma dos resíduos dos modelos (figura 37). Embora ambos tenham pontos de autocorrelação acima da faixa limite de significância, o que não é desejável, o modelo multiplicativo apresentou redução maior ao longo das desfasagens, demonstrando um melhor comportamento.

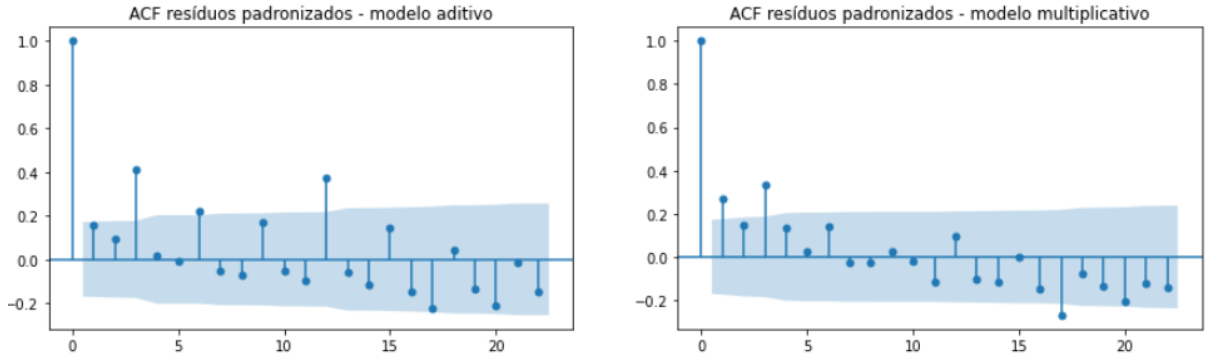


Figura 37 – Comparativo dos correlogramas dos resíduos padronizados dos modelos aditivo e multiplicativo

As métricas MAPE - *Mean Absolute Percentage Error*, RMSE - *Root Mean Squared Error* e o coeficiente de determinação - R^2, foram calculadas sobre as previsões no período de teste, ou seja, com os dados que o modelo não conhecia durante o treinamento. Um resumo está apresentado na tabela a seguir.

Modelo	**MAPE**	**RMSE**	**R^2**
FB Prophet aditivo	7.66%	7327.24	0,66
FB Prophet multiplicativo	4.62%	4335.84	0.88

Tabela 1 - Coeficiente R2 e medidas de erro MAPE e RMSE dos modelos baseados no Facebook Prophet

Por qualquer aspecto que seja analisado, o modelo com modo sazonal multiplicativo apresentou performance superior para o conjunto de dados da arrecadação federal fazendária utilizado.

4.2. Modelos ARIMA

Modelos ARIMA são muito utilizados para realizar análises e previsões em séries temporais. A sigla ARIMA significa modelo auto-regressivo (AR) integrado (I) de médias móveis (MA) (autoregressive integrated moving average, em inglês).

Considerados generalizações do modelo ARMA, junção dos modelos autorregressivo (AR) e de médias móveis (MA), que só podem ser aplicados em séries estacionárias, os modelos ARIMA superam essa limitação através da parte integrada (I), que pode aplicar uma ou mais diferenciações na série original para torná-la estacionária, como observado no item 3.2 do presente trabalho.

Os modelos são representados na forma ARIMA(*p,d,q*), sendo:
p: a ordem do modelo autorregressivo (AR);
d: o grau de diferenciação;
q: a ordem do modelo de médias móveis (MA);

Dada a forte componente sazonal da série temporal da arrecadação federal, conforme detalhado no item 3.1 e 3.3 do presente trabalho, cabe ressaltar uma importante limitação dos modelos ARIMA que é o fato de não suportarem sazonalidade. Para transpor essa dificuldade será utilizado uma extensão do modelo ARIMA chamado de Seasonal ARIMA (SARIMA) que consegue modelar a componente sazonal em séries univariadas.

Os modelos SARIMA são representados como ARIMA(*p, d, q)(P, D, Q*)*m*, onde *p, d* e *q*, são os mesmos já mencionados acima e *P, D* e *Q* são os termos equivalentes da parte sazonal do modelo; *m* representa o número de períodos por sazonalidade.

4.2.1. Aplicação do Modelo ARIMA

O primeiro experimento, na tentativa de modelar a série temporal da arrecadação fazendária, utilizou o modelo ARIMA convencional, disponível na biblioteca statsmodels[16]. Os datasets utilizados para treino e teste são os descritos no

[16] Statsmodels. Disponível em: https://www.statsmodels.org/stable/index.html. Acesso em: 03. Abr. 2021.

item 4 do presente trabalho e os procedimentos no notebook “**07.1. Previsão da arrecadação com ARIMA**”

No entanto, conforme a breve explicação no item anterior, combinada com o tópico 3.3 deste trabalho, de fato o modelo não conseguiu se ajustar de forma adequada ao comportamento da série, dada a sua forte componente sazonal.

Uma primeira abordagem para identificação de (p, d, q) utilizando os gráficos ACF e PACF, levou a indicação inicial de um modelo ARIMA(3, 1, 3), porém o mesmo se mostrou extremamente insatisfatório.

Na tentativa de identificação dos melhores hiperparâmetros para ajuste do modelo, foi então realizado um enlace com diversas ordens de modelos ARIMA, variando os valores de ‘p’ e ‘q’ de 0 a 5, e o ‘d’ entre 1 e 2. A cada etapa foi calculado e armazenado o erro RMSE entre as previsões e a base de teste, bem como registrado o AIC de cada modelo (p, d ,q).

O modelo com menor erro RMSE foi o ARIMA(3, 2, 5), com AIC de 2.866, bem próximo dos 2.871, menor valor registrado durante os ajustes. Apesar disso, novamente, em que pese a tendência de o novo modelo acompanhar melhor o crescimento da série temporal, o resultado está longe de representar um bom ajuste, conforme figura 38.

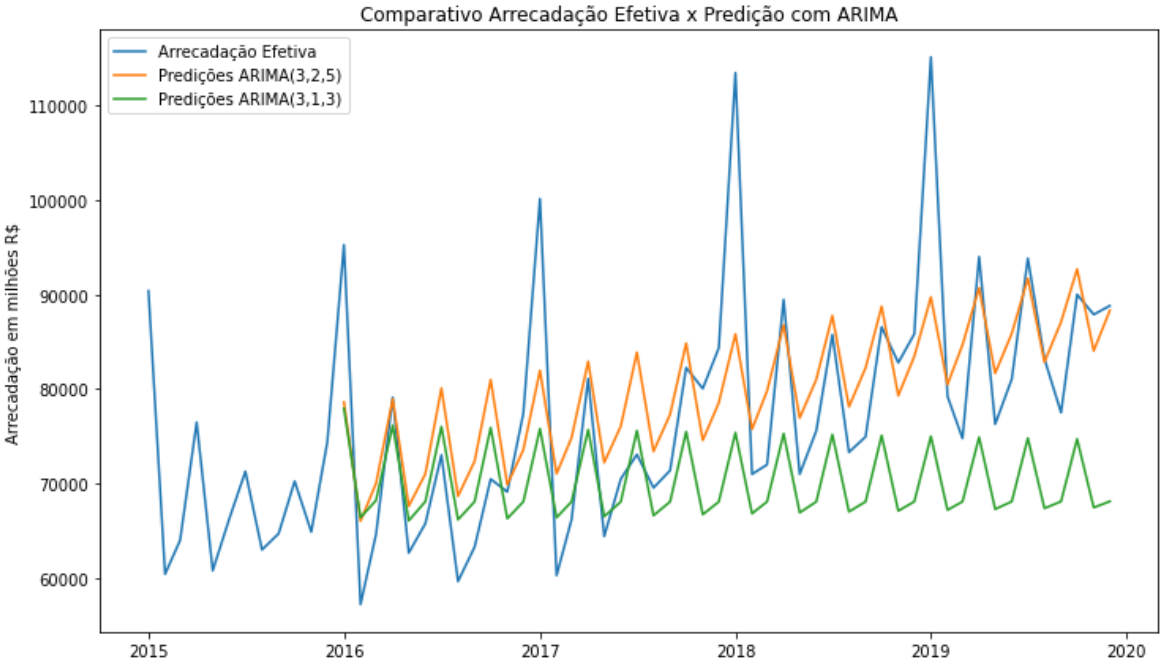


Figura 38 - Comparativo Arrecadação Efetiva x Predições dos modelos ARIMA (3,1,3) e ARIMA (3,2,5).

Dessarte, essa abordagem foi descartada em detrimento da disponibilidade dos modelos SARIMA, explorados no item seguinte.

4.2.2. Aplicação do Modelo SARIMA utilizando Auto ARIMA

Para a realização dessa etapa, optou-se pela biblioteca pmdarima[17] que, segundo a descrição do projeto, foi concebida para preencher a lacuna no Python para análise de séries temporais, fazendo um paralelo às funcionalidades equivalentes ao 'auto.arima' na linguagem R.

Todos os procedimentos adotados nesse item estão contidos no notebook '**07. Previsão da arrecadação com Auto ARIMA.ipynb'**. Os datasets de treino e de teste utilizados são os definidos e caracterizados no item 4 deste trabalho.

Carregada a biblioteca e os datasets, passou-se à definição da parametrização da função auto_arima() para execução sobre a base de treino, deixando que o próprio modelo escolhesse os melhores parâmetros (p,d,q)(P,D,Q). O estudo de sazonalidade da arrecadação, contido no item 3.3, indicou a utilização de *m*=12.

Destaca-se que o auto-ARIMA está configurado, por padrão, a usar o critério *Akaike Information Criterion* – AIC para escolher o modelo mais adequado. Utilizando o AIC, espera-se que sejam escolhidos os parâmetros que vão gerar um modelo mais simples e com menor quantidade de informações desperdiçadas.

Foram realizados experimentos utilizando apenas os dados da arrecadação (série univariada) e outros com a arrecadação em conjunto com todas as combinações possíveis das demais variáveis IPCA, IPCA_dez93 e PIB (multivariada).

Para cada um desses modelos ajustados foram realizadas as previsões para o período de teste (2016 a 2019) e calculadas as medidas de acurácia para cada situação. Os resultados e os modelos são então agregados em um dataframe que tem todas as informações necessárias para permitir comparabilidade entre as abordagens univariada e as diversas multivariadas.

[17] pmdarima: ARIMA estimators for Python. Disponível em: http://alkaline-ml.com/pmdarima/index.html. Acesso em 03 abr. 2021..

Ressalta-se que as métricas MAPE, RMSE e R^2 foram calculadas sobre os períodos futuros da base de teste, ou seja, com os dados que o modelo não conhecia durante o treinamento. Um resumo das informações dos modelos testados é apresentado na tabela 2.

Variáveis	Ordem do Modelo	AIC do Modelo	RMSE	R2	R2 Ajustado	MAPE (%)
univariado	SARIMA(1, 0, 3)(0, 1, 1, 12)	2543.92	4022.06	0.90	0.89	4.52
[PIB]	SARIMA(3, 0, 0)(1, 1, 2, 12)	2511.10	3953.02	0.90	0.90	3.74
[IPCA]	SARIMA(1, 0, 0)(0, 1, 1, 12)	2552.41	4234.44	0.88	0.88	4.27
[IPCA_dez93]	SARIMA(3, 0, 2)(2, 1, 0, 12)	2526.29	4448.91	0.87	0.87	4.27
[IPCA, IPCA_dez93]	SARIMA(1, 0, 1)(1, 1, 0, 12)	2532.06	4323.17	0.88	0.88	4.60
[IPCA, PIB]	SARIMA(3, 0, 0)(1, 1, 2, 12)	2514.96	4158.41	0.89	0.89	3.99
[IPCA_dez93, PIB]	SARIMA(3, 0, 0)(1, 1, 2, 12)	2512.48	3215.48	0.93	0.93	3.32
[IPCA, IPCA_dez93, PIB]	SARIMA(3, 0, 0)(1, 1, 2, 12)	2516.42	3299.85	0.93	0.93	3.42

Tabela 2 – Modelos SARIMA gerados com as combinações das variáveis arrecadação, PIB e IPCA

Para realizar um diagnóstico mais apurado de suas qualidades de previsão de valores futuros, bem como dos seus resíduos de treino, foram escolhidos dois modelos: o modelo univariado e o modelo multivariado de melhor performance.

Destaca-se que o modelo multivariado SARIMA(3,0,0)(1,1,2)12, utilizando as variáveis ‘IPCA_dez93’ e ‘PIB’, foi o que apresentou as melhores métricas RMSE, R2 e MAPE na previsão da arrecadação, bem como ficou empatado com o menor valor AIC, juntamente com o modelo que usou apenas o PIB como variável exógena. Esse é um forte indicativo de que o modelo, além de ter apresentado a melhor performance sobre a base de teste, é um dos mais simples dentre os testados.

Para manter a homogeneidade das análises, repetiremos o mesmo caminho percorrido no item 4.1. relativamente ao Facebook Prophet.

Objetivando melhor identificar a qualidade das previsões de forma gráfica, foi realizado um comparativo da arrecadação efetiva e da prevista em um conjunto de dados que os modelos não conheciam. (Figura 39)

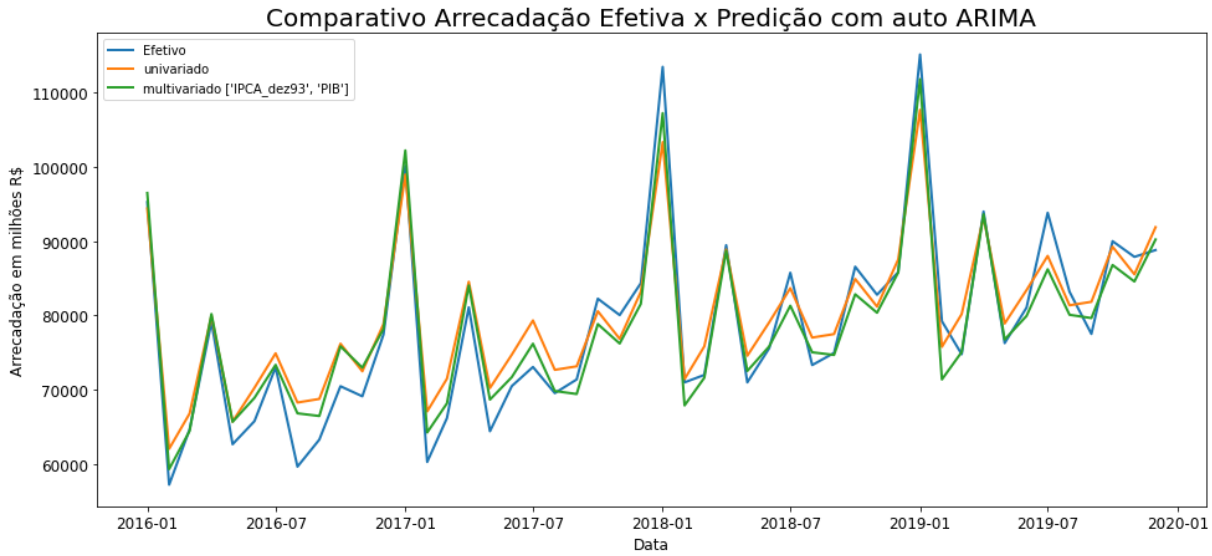


Figura 39 – Comparativo da série da arrecadação federal efetiva com as previsões utilizando SARIMA univariado e multivariado - 2016 a 2019

Nota-se que a previsão do modelo multivariado acompanha melhor a arrecadação efetiva, tanto nos picos como nos vales, o que indica ser um modelo preditivo mais adequado. Para certificação dessa afirmativa, passou-se a analisar os resíduos de ambos os modelos durante o treino. (Figura 40)

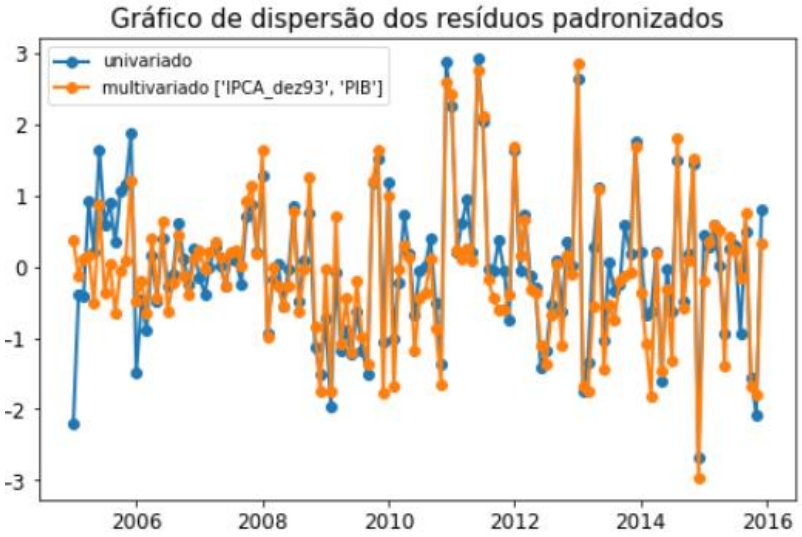


Figura 40 – Comparativo da dispersão dos resíduos padronizados dos modelos univariado e multivariado

Analisando o gráfico de dispersão dos resíduos padronizados, ambos parecem flutuar de forma aleatória ao longo de zero. Porém, o modelo multivariado obteve menor espalhamento em geral, com destaque nos anos iniciais.

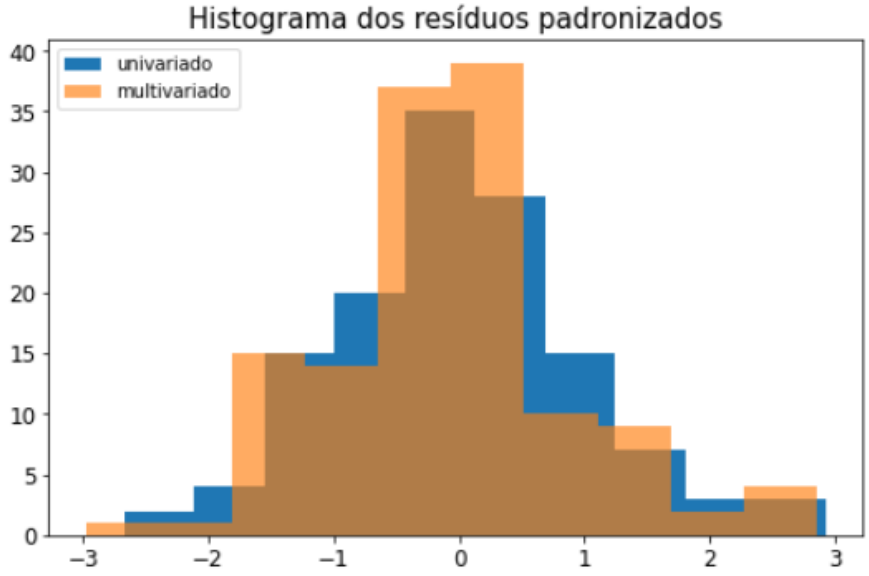


Figura 41 - Comparativo do histograma dos resíduos padronizados dos modelos univariado e multivariado

Visualmente não se consegue definir na figura 41, qual dos dois histogramas dos valores residuais apresentou uma distribuição mais próxima da normal. No entanto, o teste de Shapiro-Wilk apontou um valor-p de 0,1318 para o modelo univariado e apenas 0,0037 para o multivariado, devendo ser rejeitada a hipótese de normalidade da distribuição residual desse último.

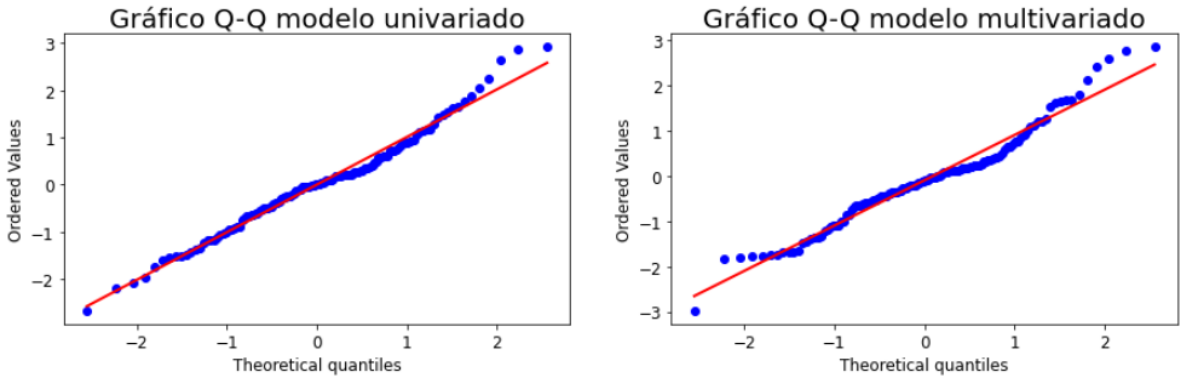


Figura 42 - Comparativo do gráfico Q-Q para os resíduos padronizados dos modelos univariado e multivariado

No gráfico Q-Q de probabilidades (figura 42) o modelo univariado obteve melhor performance, tendo os pontos azuis, em geral, ficado mais alinhados com a linha vermelha, conforme era esperado dado o resultado do teste Shapiro-Wilk.

Ambos os modelos apresentaram correlograma dos resíduos padronizados muito satisfatórios, conforme figura 43.

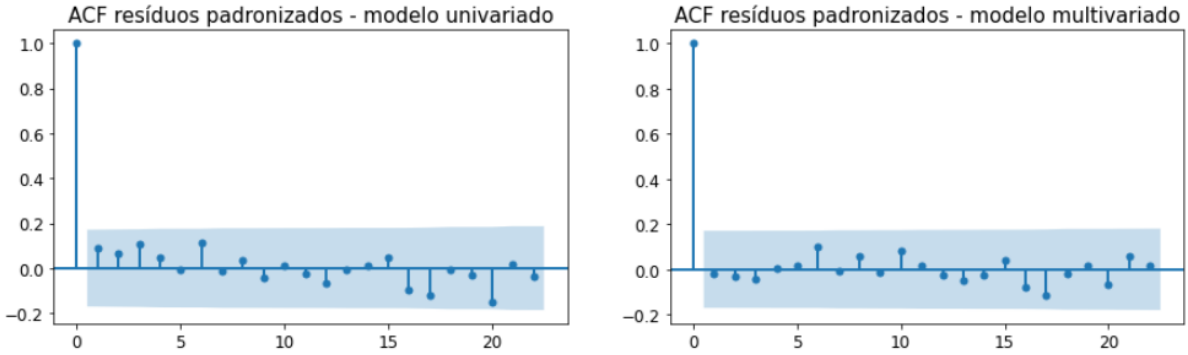


Figura 43 - Comparativo dos correlogramas dos resíduos padronizados dos modelos univariado e multivariado

As métricas MAPE, RMSE e o coeficiente R^2 foram calculadas sobre as previsões no período da base de teste, ou seja, com os dados que o modelo não conhecia durante o treinamento. Um resumo está apresentado na tabela abaixo.

Modelo	**MAPE**	**RMSE**	**R2**
SARIMA(1, 0, 3)(0, 1, 1, 12) univariado	4.52%	4022.06	0.90
SARIMAX(3, 0, 0)(1, 1, 2, 12) (PIB e IPCA_dez93)	3.32%	3215.48	0.93

Tabela 3 - Coeficiente R2 e medidas de erro MAPE e RMSE dos modelos SARIMA

Analisando pelo aspecto das métricas MAPE, RMSE e coeficiente R^2, o modelo univariado apresentou resultado inferior ao modelo multivariado. No entanto, sob o aspecto de sua implementação, o modelo multivariado apresenta maior complexidade, uma vez que será necessário prever também as variáveis exógenas nos períodos futuros.

Exemplificativamente, para realizar a previsão da arrecadação fazendária federal para os próximos 12 meses, seria necessário estimar o PIB e o IPCA_dez93 para igual quantidade de períodos. Os eventuais erros dessas estimativas podem levar a erros maiores do que os aqui mensurados com a base de teste.

5. Apresentação dos Resultados

A partir do problema proposto – apresentar soluções baseadas em ciência de dados para o aperfeiçoamento da estimativa da receita fazendária federal, administrada pela Secretaria Receita Federal do Brasil, tão importante para que o governo tenha uma previsão de quanto poderá gastar, com impactos na definição das políticas públicas do governo e no efetivo controle de gastos – foram realizadas as seguintes etapas:

Coleta e tratamento dos dados: desenvolvimento de notebooks em Jupyter Python para download automático das planilhas da arrecadação federal e posterior carga para um Pandas dataframe, juntamente com as planilhas do IPCA e do PIB Brasil. O tratamento das informações superou as dificuldades com formatos bastante distintos e valores ausentes (*missing values*).

Análise e exploração dos dados: essa etapa demandou bastante tempo, mas foi de uma grande riqueza de aprendizado sobre os dados da arrecadação, identificando o seu comportamento de tendência, de estacionarieade e sazonalidade. Durante esse tópico recorreu-se à linguagem R para auxílio na detecção de importantes valores atípicos (*outliers*)

Modelos de Machine Learning: com 3 notebooks desenvolvidos apenas nesse tópico, também houve um grande investimento de tempo na identificação dos melhores modelos preditivos e da melhor combinação de parâmetros e variáveis de entrada buscando as melhores performances.

Avaliação dos Resultados: Todos os ajustes dos modelos foram realizados com a base de treinamento, abrangendo o período de 2004 a 2015, enquanto as previsões de períodos futuros e respectivos cálculos dos erros dessas estimativas foram realizados sobre a base de teste, ou seja, a partir de um conjunto de dados que os modelos não conheciam, abarcando o período de 2016 a 2019.

O primeiro modelo preditivo testado foi o Facebook Prophet, apresentando facilidade de uso, realizou rapidamente o ajuste e as previsões e obteve uma performance de previsão bastante interessante, com erros MAPE e RMSE em linha com modelos mais difíceis de implementar.

No tocante aos modelos da família ARIMA, o modelo convencional foi testado e descartado, uma vez que não se apresentou adequado à forte componente sazonal da série temporal da arrecadação federal fazendária (vide item 3.3 e 4.2.1).

Por outro lado, os modelos SARIMA que introduzem o termo sazonal performaram muito bem, assim como os SARIMAX com componente sazonal e variáveis exógenas.

Com a utilização do Auto-ARIMA, a busca da melhor parametrização para a ordem (p,d,q)(P,D,Q) dos modelos foi bastante simplificada, assemelhando-se à facilidade de uso do Facebook Prophet. Porém, em termos de velocidade de execução, o Prophet foi muito superior.

Todos os modelos eleitos como melhores em sua categoria tiveram os seus respectivos resíduos analisados com o uso de testes estatísticos e de ferramentas gráficas, tendo apresentado bons resultados.

Um resumo dos resultados obtidos para o coeficiente de determinação R^2, bem como das medidas MAPE e RMSE, são apresentados na tabela a seguir.

Modelo	MAPE	RMSE	R2
Facebook Prophet multiplicativo	4.62%	4335.84	0.88
SARIMA(1,0,3) (0,1,1,12) univariado	4.52%	4022.06	0.90
SARIMAX(3,0,0) (1,1,2,12) [PIB e IPCA_dez93]	3.32%	3215.48	0.93

Tabela 4 - Coeficiente R2 e medidas de erro MAPE e RMSE dos modelos Facebook Prophet e SARIMA (X)

Por fim, foi elaborado o gráfico da figura 44, contendo as previsões realizadas pelos melhores modelos Facebook Prophet, SARIMA e SARIMAX, no período de teste (2016 a 2019), em comparação com a arrecadação ocorrida de fato no mesmo período. **Percebe-se uma aderência bastante satisfatória entre as previsões e a arrecadação efetiva.**

Como sugestão de trabalho futuro, poderiam ser avaliados modelos preditivos do IPCA e do PIB Brasil para acoplar ao modelo SARIMAX apresentado, verificando se ele alcançará a excepcional performance apontada durante os testes realizados neste trabalho.

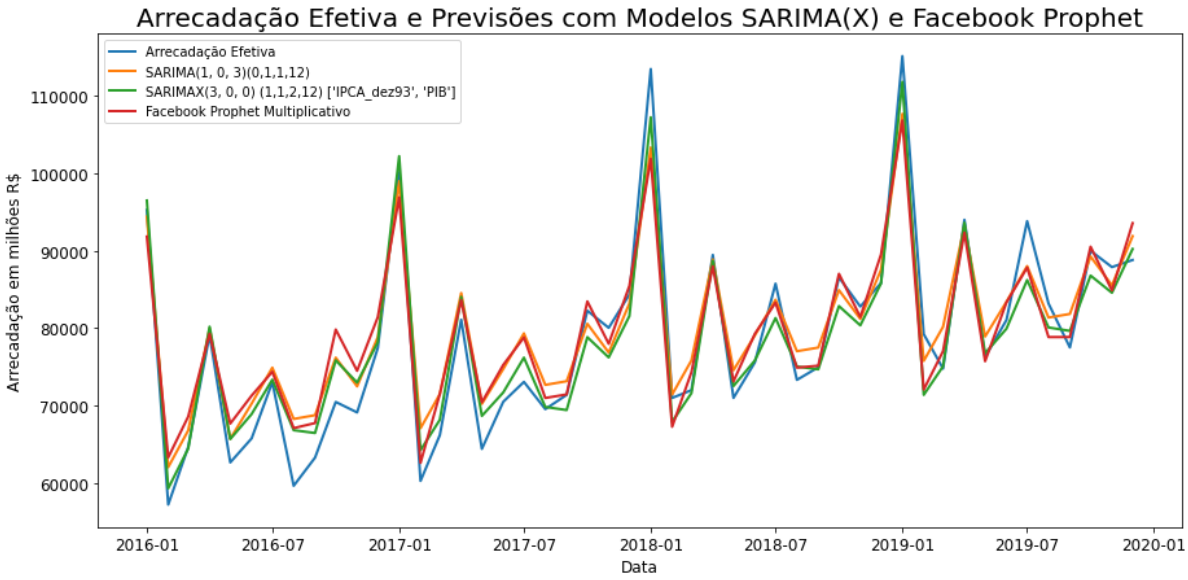


Figura 44 – Comparação da arrecadação efetiva com os melhores modelos Facebook Prophet e SARIMA(X)

6. Links

Abaixo estão os links para o vídeo resumo desse trabalho, bem como para o repositório de arquivos na plataforma Github.

O repositório contém as pastas com os datasets, planilhas, notebooks Jupyter Python e scripts em linguagem R, utilizados nesse projeto.

Link para o vídeo: https://youtu.be/SWsxUq3o-d8

Link para o repositório: https://github.com/mveludo/tcc-puc-mg

REFERÊNCIAS

Legislação Orçamentária: Portal do Orçamento (senado.leg.br). Disponível em: https://www12.senado.leg.br/orcamento/legislacao-orcamentaria. Acesso em 22 abr. 2021.

Arrecadação por Estado – Secretaria Receita Federal do Brasil. Disponível em: https://receita.economia.gov.br/dados/receitadata/arrecadacao/arrecadacao-por-estado. Acesso em: 01 abr. 2021.

Beautiful Soup. Disponível em: https://pypi.org/project/beautifulsoup4/. Acesso em: 01 abr. 2021.

Produto Interno Bruto - PIB. Disponível em: https://www.ibge.gov.br/explica/pib.php Acesso em: 01 abr. 2021.

Sistema de Contas Nacionais Trimestrais – SCNT. Disponível em: https://www.ibge.gov.br/estatisticas/economicas/contas-nacionais/9300-contas-nacionais-trimestrais.html. Acesso em 02 abr. 2021.

Índice Nacional de Preços ao Consumidor Amplo – IPCA. Disponível em: https://www.ibge.gov.br/estatisticas/economicas/precos-e-custos/9256-indice-nacional-de-precos-ao-consumidor-amplo.html. Acessado em 02 abr. 2021.

Package 'tsoutliers'. Disponível em: https://cran.r-project.org/web/packages/tsoutliers/tsoutliers.pdf. Acesso em: 03 abr. 2021.

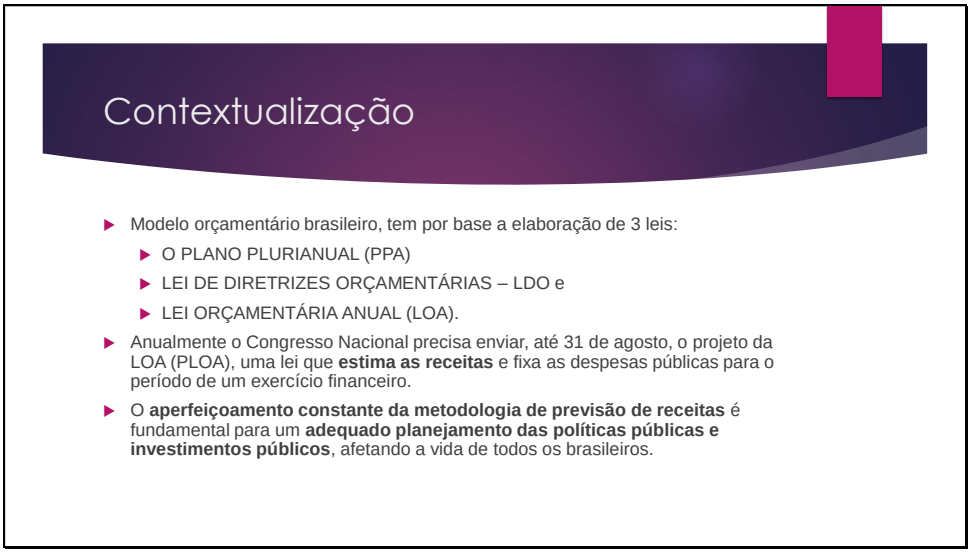
pmdarima: ARIMA estimators for Python. Disponível em: http://alkaline-ml.com/pmdarima/index.html. Acesso em 03 abr. 2021.

APÊNDICE

Apresentação

Slide 1

Slide 2



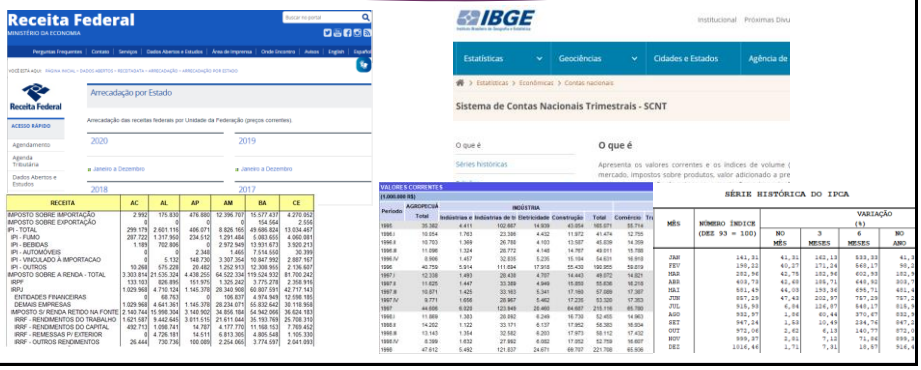
Slide 3

Problema Proposto

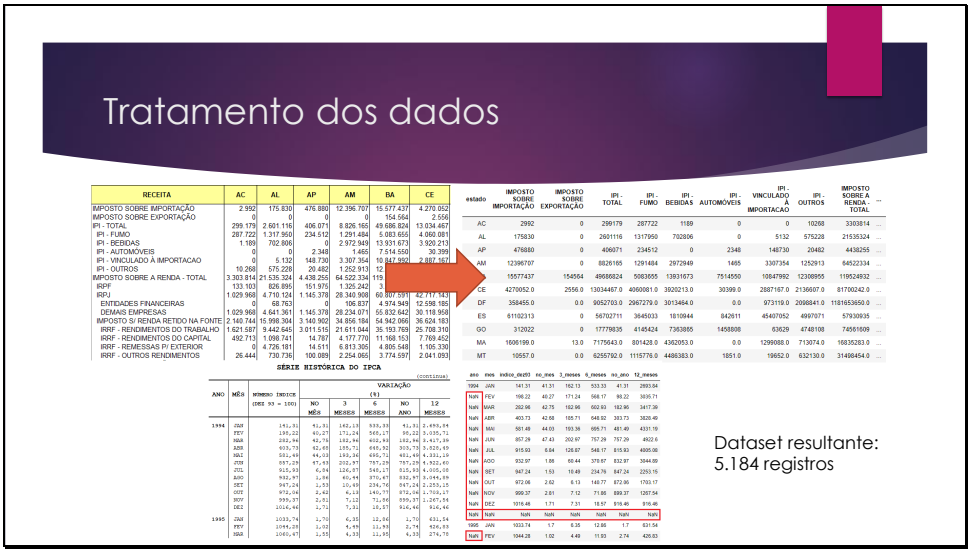
O que?	**Por que?**
Aplicação de técnicas de ciência de dados para analisar a série histórica da arrecadação fazendária federal total e realizar previsões para períodos futuros.	Melhoria da qualidade da previsão de receitas, com impactos positivos na definição orçamentária do gasto público, investimentos e políticas públicas.
Abrangência?	**Período?**
Dados da arrecadação fazendária federal nacional, dados do PIB e do IPCA.	O período que está sendo analisado compreende os anos de 2004 a 2019
Como?	**De Quem?**
Uso predominante da linguagem Python no ambiente Jupyter, com os pacotes Auto-ARIMA e Facebook Prophet.	Dados públicos, disponibilizados pela Secretaria da Receita Federal do Brasil - RFB e pelo Instituto Brasileiro de Geografia e Estatística - IBGE.

Slide 4

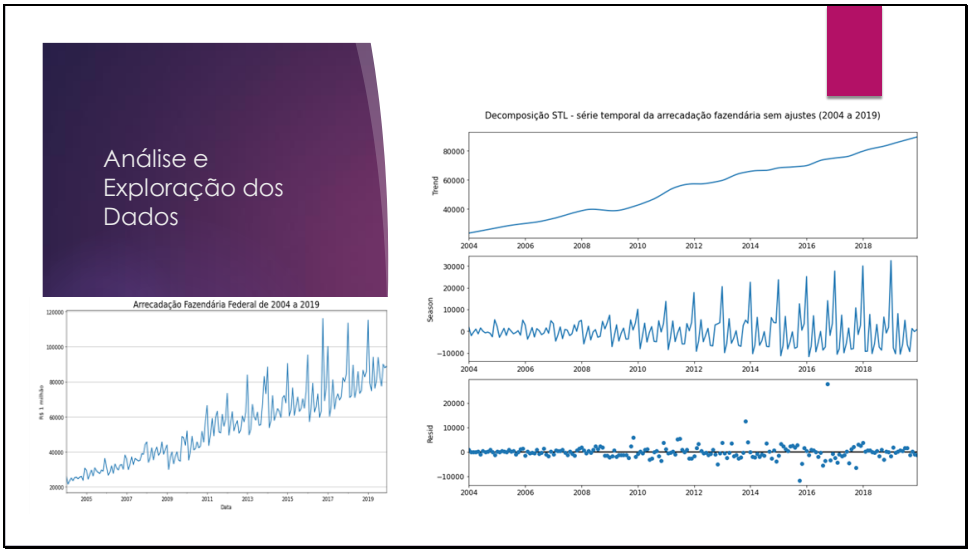
Coleta dos dados



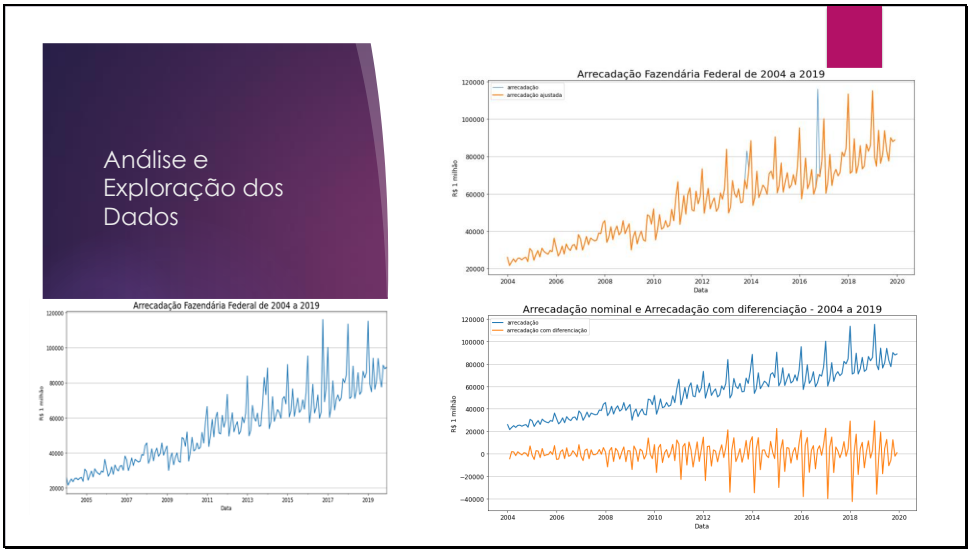
Slide 5



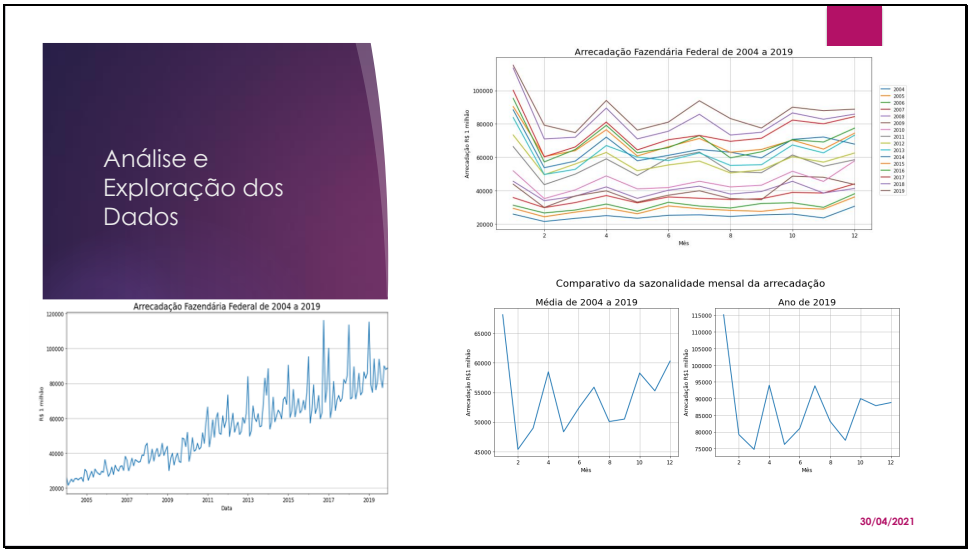
Slide 6



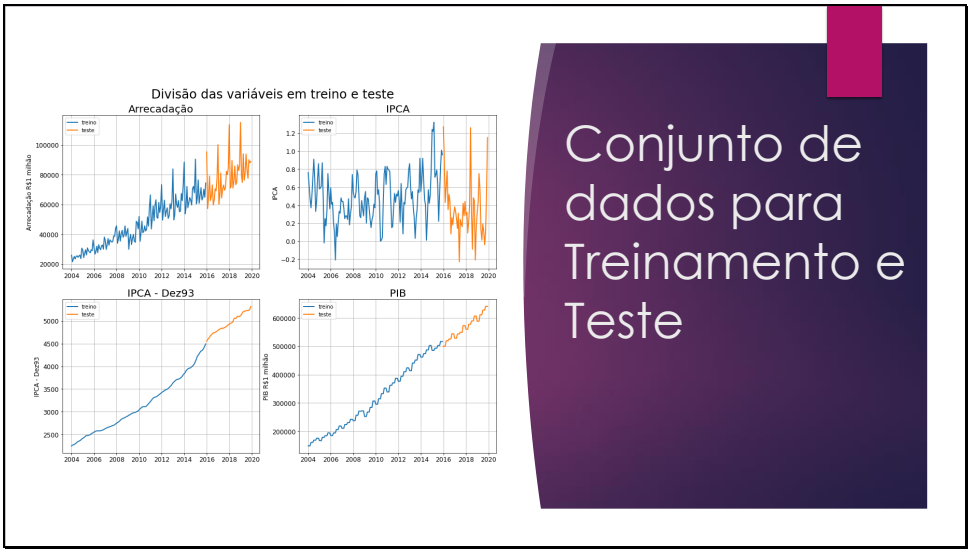
Slide 7



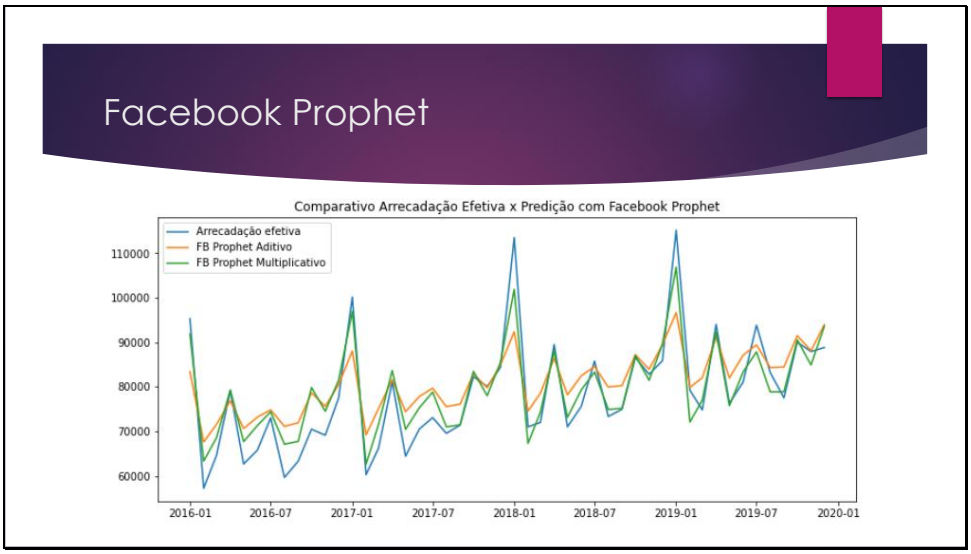
Slide 8



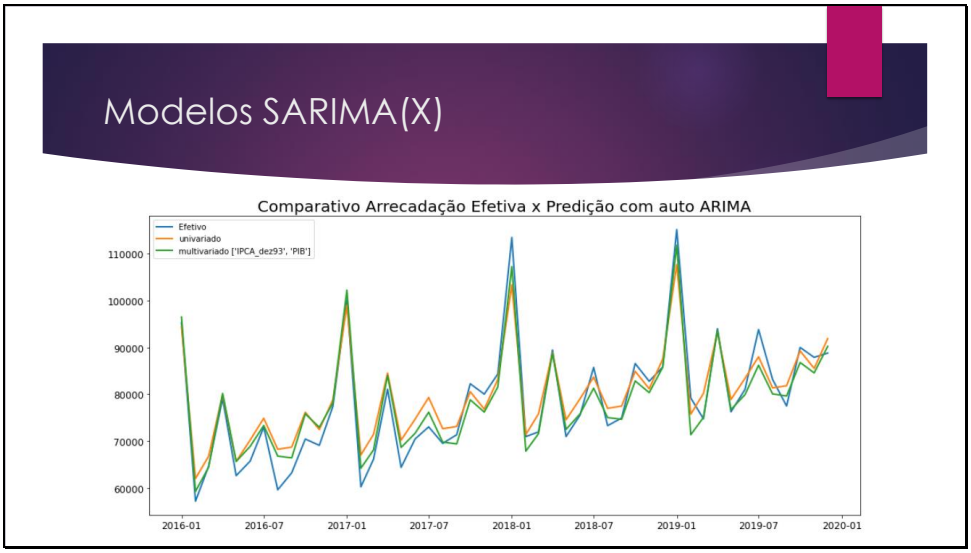
Slide 9



Slide 10



Slide 11



Slide 12