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

Simone Tardin Fagundes

ANÁLISE E PREVISÃO DO PREÇO DE COMBUSTÍVEIS

Belo Horizonte
2021

Simone Tardin Fagundes

ANÁLISE E PREVISÃO DO PREÇO DE COMBUSTÍVEIS

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

1.1. Contextualização

Os preços de determinados produtos, como os relacionados a petróleo, tiveram grande alta de 2020 em diante, devido ao cenário de grande instabilidade gerado pela pandemia de Covid-19. A escassez tanto de matérias primas quanto de produtos destinados ao consumidor final tem gerado uma crise no Brasil, com grande queda no poder de compra dos brasileiros.

Considerando o contexto acima, estudos que ajudem a explicar e prever variações de preços de produtos essenciais para a economia são fundamentais para o planejamento e tomada de decisões, inclusive pelos responsáveis por políticas públicas.

A análise de séries temporais pode ser aplicada à medicina, engenharia, economia, entre tantas outras áreas, e com objetivos e técnicas diferentes, tais como: *Query by content*, em que se busca localizar padrões conhecidos em um banco de dados de séries temporais; detecção de anomalias, em que se busca detectar padrões incomuns em séries; descoberta de *motifs*, para descobrir subsequências dentro de uma série temporal que se repetem; classificação, usada para se classificar séries temporais em um certo número de classes; segmentação, para criar uma representação reduzida da série temporal; previsão, em que deseja estimar valores futuros baseado nos valores conhecidos (passados); clusterização, para agrupar diferentes séries temporais em clusteres similares entre si (Esling & Agon, 2012).

Assim, as séries temporais e suas técnicas de análise são ferramenta crucial nesta época de abundância de dados e podem ajudar a resolver diversos problemas.

1.2. O problema proposto

Este trabalho propõe-se a analisar os dados de preços de combustíveis registrados semanalmente pela Agência Nacional do Petróleo, Gás Natural e

Biocombustíveis – ANP. Tal análise pode auxiliar na tomada de decisões relativas à implementação de políticas públicas.

O objetivo é analisar os dados em séries temporais relativas aos preços do etanol e da gasolina, de modo a encontrar os componentes das séries, como tendência, ciclos e sazonalidade, e realizar a previsão para algumas semanas, verificando posteriormente a qualidade dos modelos. Ainda, verificar o impacto da alta do dólar e se a adição dos dados relacionados à taxa de câmbio poderiam melhorar os modelos.

Para os tratamentos de dados e análises realizadas, foram utilizadas as ferramentas Jupyter Notebook 6.3.0 / Anaconda Navigator 2.0.3 e Python 3.

2. Coleta de Dados

Os dados analisados relacionados aos preços dos combustíveis provém do site Base de dados (https://basedosdados.org/dataset), e estão disponíveis na página https://basedosdados.org/dataset/br-anp-precos-combustiveis. O arquivo é produto das coletas realizadas pela ANP de 10/05/2004 a 15/10/2021 e reunidas pelos responsáveis do site Base de Dados. Além dos preços de combustíveis, foi também obtido arquivo contendo a série histórica da taxa de câmbio do dólar, de 1996 a outubro de 2021, na página https://www.cepea.esalq.usp.br/br/serie-de-preco/dolar.aspx, do Centro de Estudos Avançados em Economia Aplicada – Cepea, do Departamento de Economia, Administração e Sociologia da Escola Superior de Agricultura “Luiz de Queiroz” (Esalq), da Universidade de São Paulo (USP).

O arquivo de preços de combustível foi obtido no formato .csv (arquivos de valores separados por vírgula) contendo os preços pesquisados a cada semana em vários postos de combustível do país e apresenta a estrutura descrita tabela abaixo.

Nome da coluna/campo	Descrição	Tipo
ano	Ano da coleta	Numérico, inteiro
sigla_uf	Sigla da Unidade Federativa da revenda	Textual, contendo uma das 27 siglas possíveis

	pesquisada	
id_municipio	ID Município IBGE - 7 Dígitos	Numérico, inteiro
bairro_revenda	Nome do bairro da revenda pesquisada	Textual
cep_revenda	Número do Código do Endereço Postal (CEP) do logradouro da revenda pesquisada	Textual
endereço_revenda	Endereço de revenda	Textual
cnpj_revenda	Número do Cadastro Nacional de Pessoa Jurídica da revenda	Textual
nome_estabelecimento	Nome do estabelecimento	Textual
bandeira_revenda	Nome da Bandeira da revenda	Textual”
data_coleta	Data da coleta do(s) preço(s)	Textual
produto	Nome do combustível	Textual
unidade_medida	Unidade de Medida	Textual
preco_compra	Preço de venda da distribuidora para o posto revendedor de combustível	Numérico, ponto flutuante
preco_venda	Preço de venda ao consumidor final praticado pelo revendedor na data da coleta	Numérico, ponto flutuante

Os dados relacionados à série histórica da taxa de câmbio estavam organizados em um arquivo .xlsx com a seguinte estrutura:

Nome da coluna/campo	Descrição	Tipo
Data da série	Data de registro da taxa de câmbio	Data
DóLAR COMERCIAL ATUALIZADO àS 16H30	Valor do dólar, convertido para real	Numérico, ponto flutuante

3. Processamento/Tratamento de Dados

Após *download* dos arquivos com os dados, eles foram transformados para dataframes, estruturas de dados construídas pela biblioteca pandas.

O arquivo de dados sobre o dólar, contendo 6.312 registros, teve as colunas renomeadas conforme tabela a seguir, para simplificar o processamento.

Nome original	Nome dado
Data da série	data
DóLAR COMERCIAL ATUALIZADO àS 16H30	cambio

Então, foi verificado se havia algum valor nulo na série.

```
# Verificando a existência de valores nulos
df_dolar.isnull().sum()
```

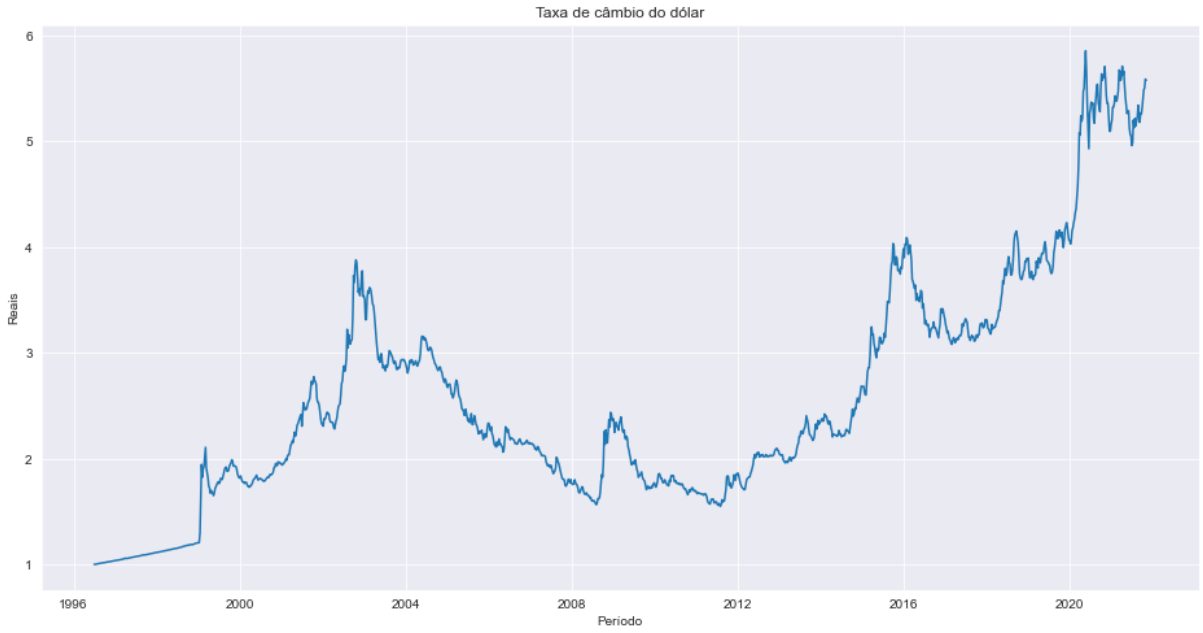
```
data      0
cambio    0
dtype: int64
```

O conjunto de dados pré-processados para análise resultou com a estrutura de acordo com a amostra a seguir.

```
df_dolar.head()
```

	data	cambio
0	2021-10-28	5.628
1	2021-10-27	5.570
2	2021-10-26	5.567
3	2021-10-25	5.546
4	2021-10-22	5.640

O gráfico da série da taxa de câmbio original pode ser visto a seguir.



Pode-se perceber que a série de dados do dólar inicia em 1996, enquanto a de preços dos combustíveis, apenas em 2004. Logo, para coincidir os períodos, os dados extras da série de câmbio foram removidos.

```
df_dolar_resumo.drop(df_dolar_resumo.iloc[-2:].index, axis=0, inplace=True)
df_dolar_resumo.drop(df_dolar_resumo.iloc[0:len(df_dolar_resumo)-len(df_etanol_resumo)].index, axis=0, inplace=True)
```

```
print(len(df_dolar_resumo)==len(df_etanol_resumo))
print(len(df_dolar_resumo)==len(df_gasolina_resumo))
```

```
True
True
```

O arquivo de preços de combustíveis possuía 26.514.824 registros, englobando os preços relativos a etanol, gasolina, diesel, GNV, GLP, diesel S50, diesel S10 e gasolina aditivada.

Após análise das colunas existentes, optou-se por remover as colunas 'ano', 'sigla_uf', 'id_municipio', 'bairro_revenda', 'cep_revenda', 'endereco_revenda', 'cnpj_revenda', 'nome_estabelecimento', 'bandeira_revenda', 'unidade_medida', 'preco_compra', por não conterem informação relevante para a análise. E então, procedeu-se à verificação de existência de valores nulos.

```
# Verificando a existência de valores nulos
df_combustiveis.isnull().sum()
```

```
data_coleta    1
produto        0
preco_venda    1
dtype: int64
```

Os valores nulos foram removidos. A seguir pode-se visualizar o formato do dataframe.

```
df_combustiveis.head()
```

	data_coleta	produto	preco_venda
0	2004-05-10	etanol	1.280
1	2004-05-10	gasolina	1.899
2	2004-05-10	diesel	1.399
3	2004-05-10	etanol	0.799
4	2004-05-10	gasolina	1.899

Com base nos dados anteriores, foram gerados os conjuntos de dados filtrados para etanol e gasolina. Optou-se por utilizar apenas os dados relativos à gasolina comum, não incluindo a aditivada.

```
df_gasolina = df_combustiveis[df_combustiveis.produto == 'gasolina']
df_etanol = df_combustiveis[df_combustiveis.produto == 'etanol']
```

Após a filtragem, foi calculada a média para cada produto referente a cada semana de coleta.

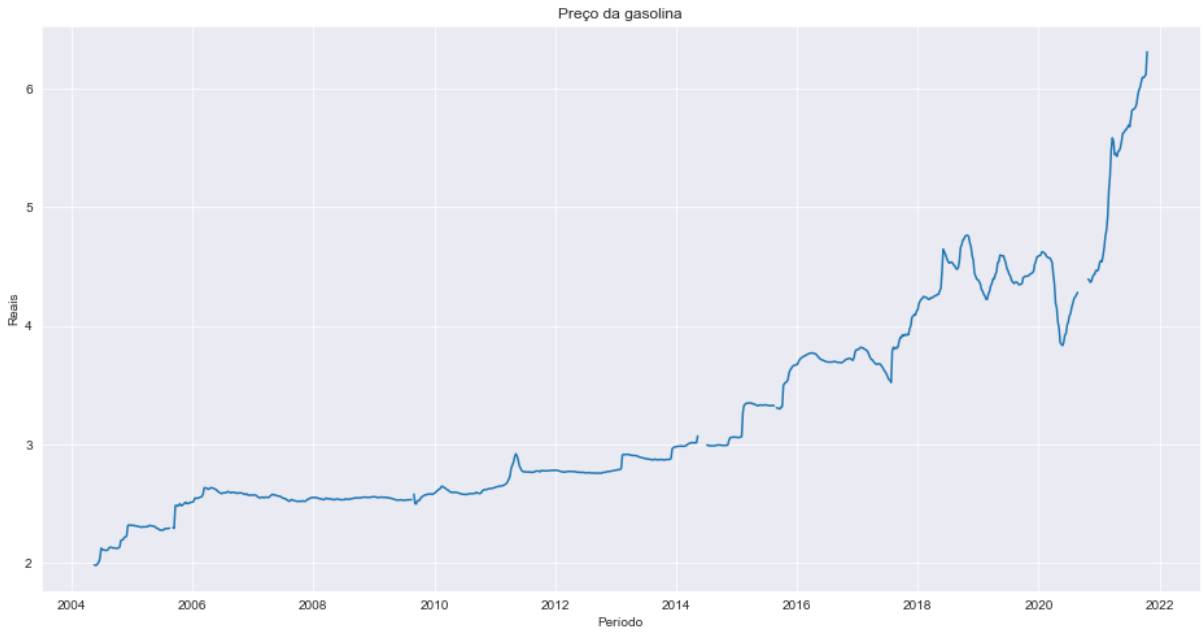
```
df_etanol_resumo = df_etanol.groupby(pd.Grouper(key='data_coleta', axis=0, freq='W')).mean()
df_gasolina_resumo = df_gasolina.groupby(pd.Grouper(key='data_coleta', axis=0, freq='W')).mean()
```

O conjunto de dados, após o cálculo da média, possuía diversos valores nulos, devido a semanas em que não houve pesquisa de preços. A figura a seguir mostra como ficou o conjunto de dados sobre os preços de etanol. Nela podem ser percebidos valores inexistentes em várias semanas, em meses como agosto de 2005, junho de 2014 e setembro de 2020.

```
df_etanol_resumo[df_etanol_resumo.preco_venda.isnull()]
```

data_coleta	preco_venda
2005-08-21	NaN
2005-08-28	NaN
2009-08-23	NaN
2014-05-18	NaN
2014-05-25	NaN
2014-06-01	NaN
2014-06-08	NaN
2014-06-15	NaN
2014-06-22	NaN
2014-06-29	NaN
2015-08-23	NaN
2020-08-30	NaN
2020-09-06	NaN
2020-09-13	NaN
2020-09-20	NaN
2020-09-27	NaN
2020-10-04	NaN
2020-10-11	NaN
2020-10-18	NaN

Os gráficos abaixo possibilitam visualizar os períodos com dados faltantes.



Como as séries aparentemente não são estacionárias, optou-se por preencher esses valores com a média móvel, a fim de dar prosseguimento à tendência e não prejudicar os algoritmos. Dessa forma, após o preenchimento dos valores nulos, as séries foram verificadas novamente.

```
df_gasolina_resumo.isnull().sum()
```

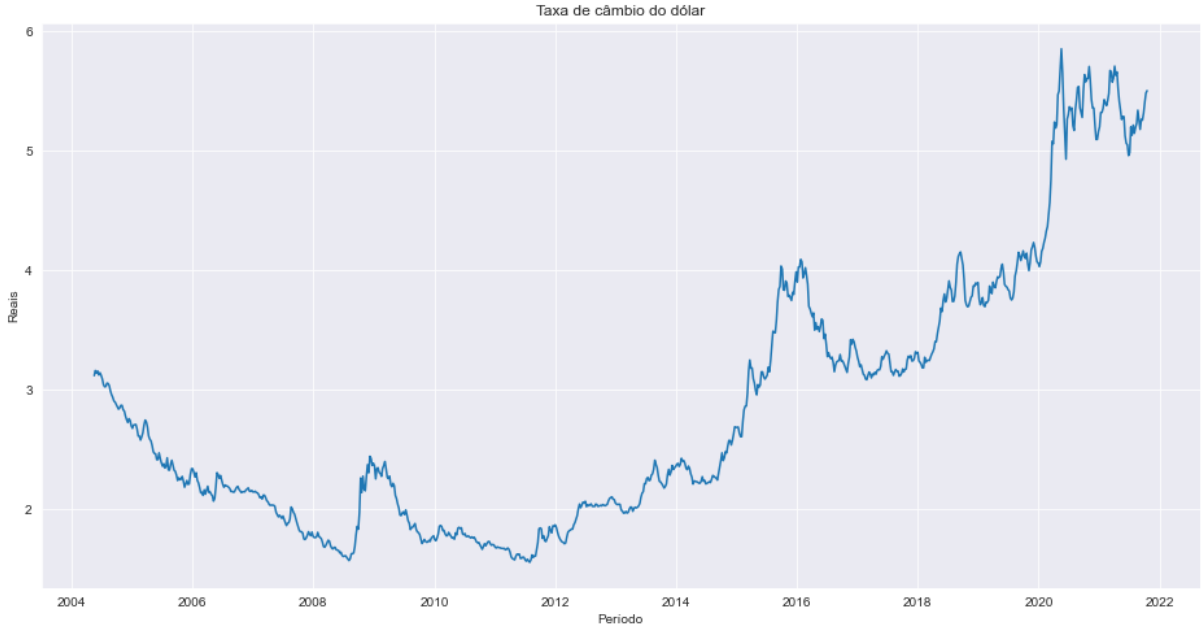
```
preco_venda    0
dtype: int64
```

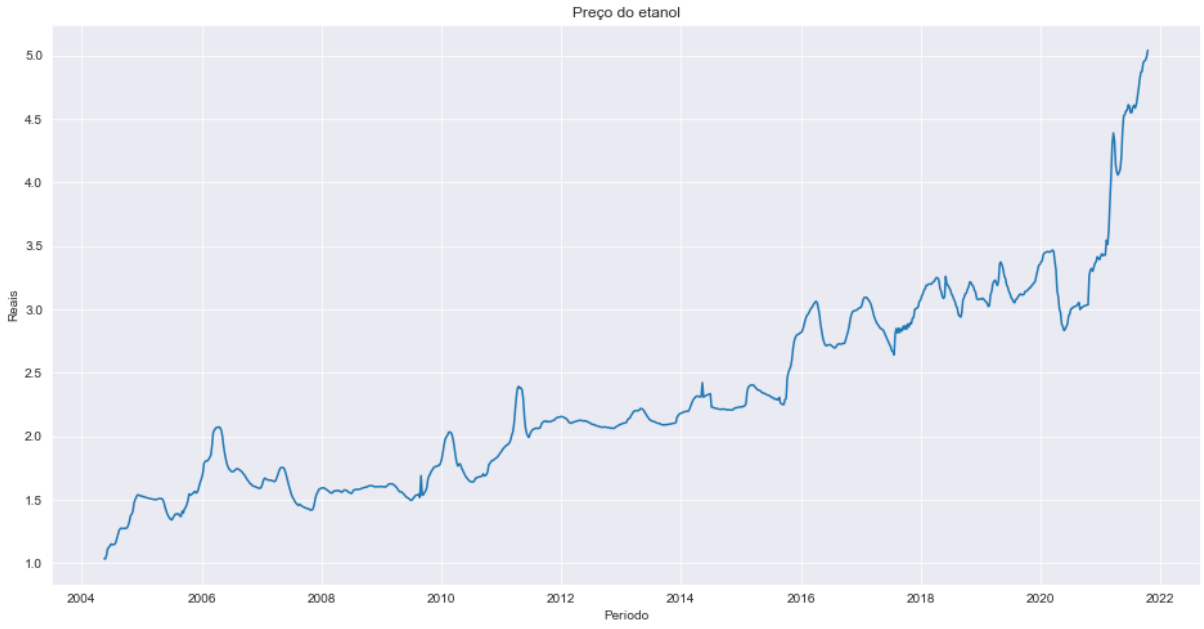
```
df_etanol_resumo.isnull().sum()
```

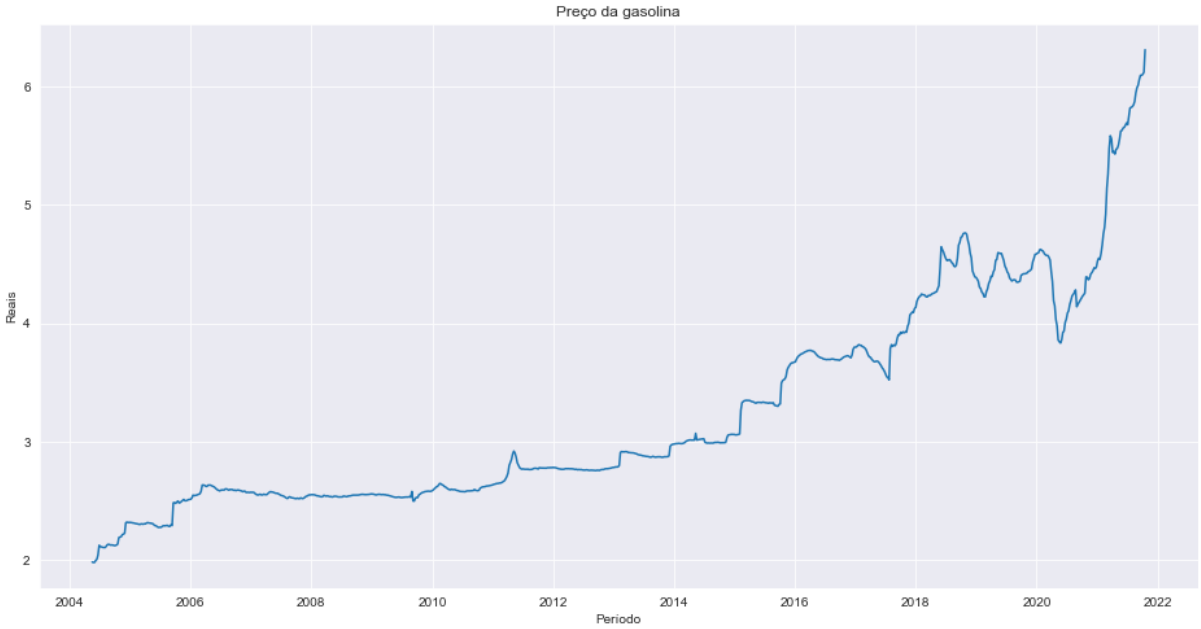
```
preco_venda    0
dtype: int64
```

4. Análise e Exploração dos Dados

Posteriormente à etapa de pré-processamento, os conjuntos de dados e as séries possuíam o seguinte formato.







Procedeu-se então à verificação das informações estatísticas, as quais podem ser vistas abaixo.

```
df_etanol.describe()
```

	preco_venda
count	6.554200e+06
mean	2.094484e+00
std	7.316561e-01
min	5.900000e-01
25%	1.599000e+00
50%	1.948000e+00
75%	2.448000e+00
max	7.199000e+00

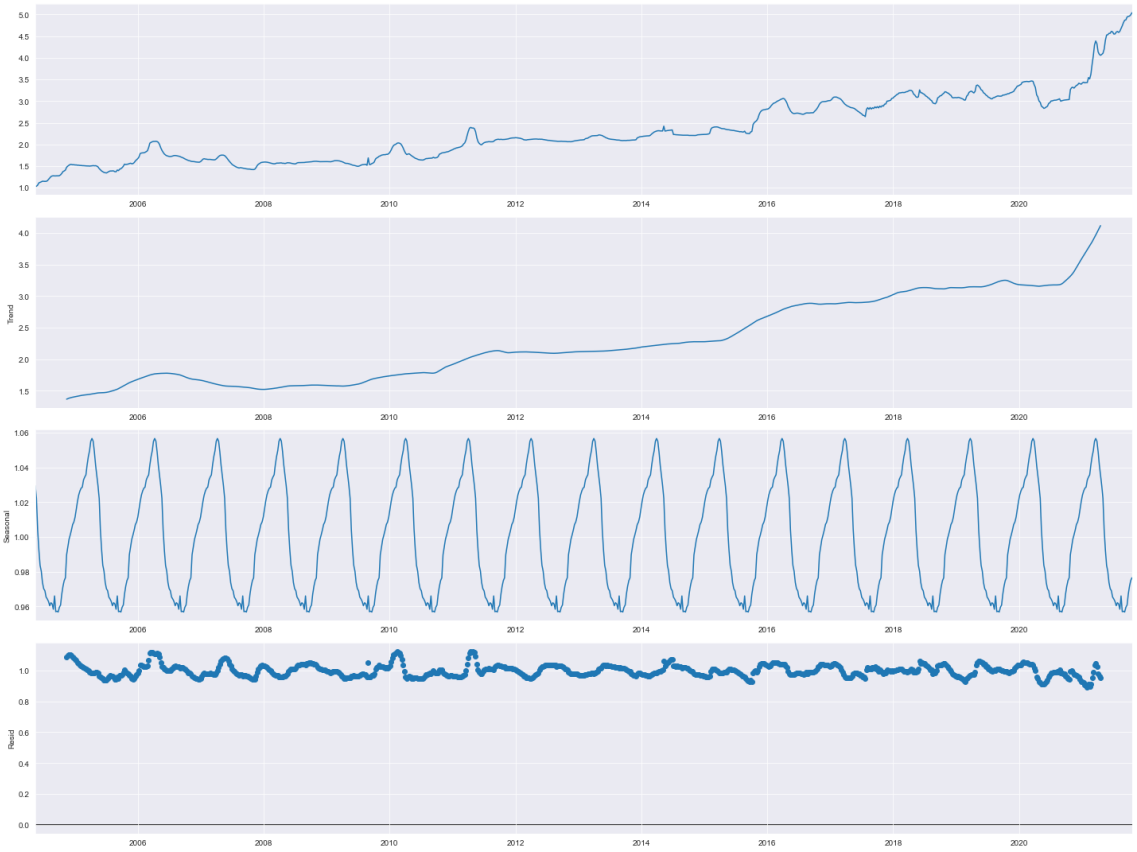
```
df_gasolina.describe()
```

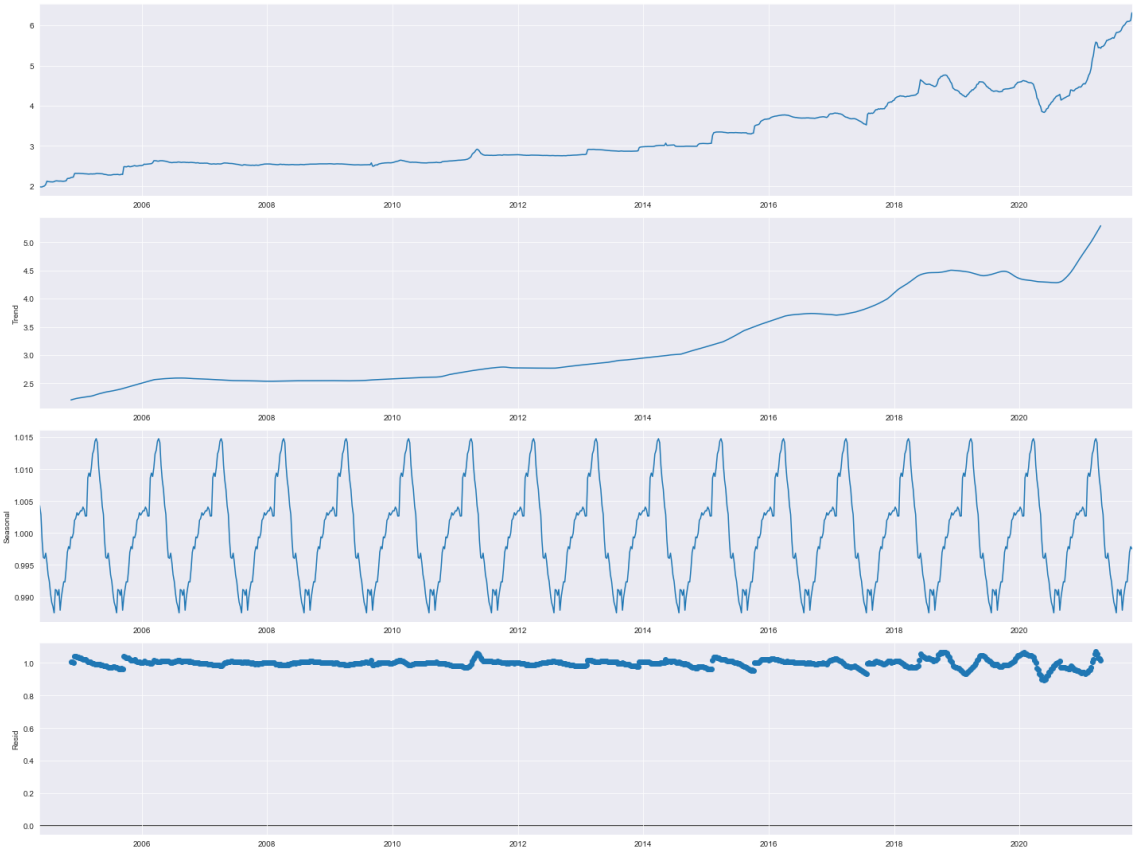
	preco_venda
count	7.066140e+06
mean	2.997807e+00
std	7.886034e-01
min	1.390000e+00
25%	2.499000e+00
50%	2.720000e+00
75%	3.240000e+00
max	7.499000e+00

```
df_dolar.describe()
```

	cambio
count	6312.000000
mean	2.573707
std	1.086591
min	1.004000
25%	1.806000
50%	2.266000
75%	3.178250
max	5.915000

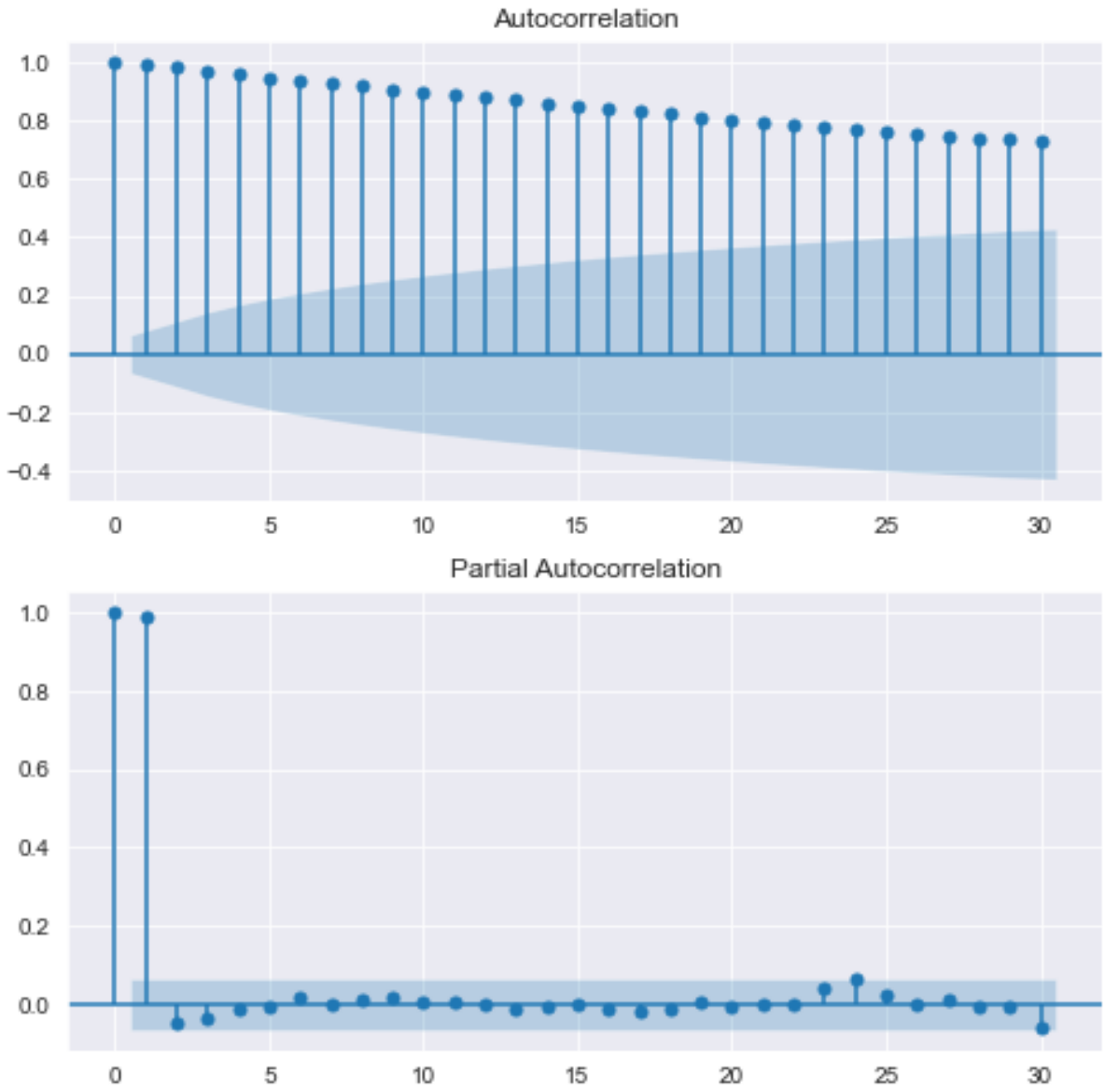
As decomposições das séries de preços de etanol e gasolina, mostradas respectivamente a seguir, sugeriram modelos com sazonalidade.


Trend
Seasonal
Resid
2006
2008
2010
2012
2014
2016
2018
2020

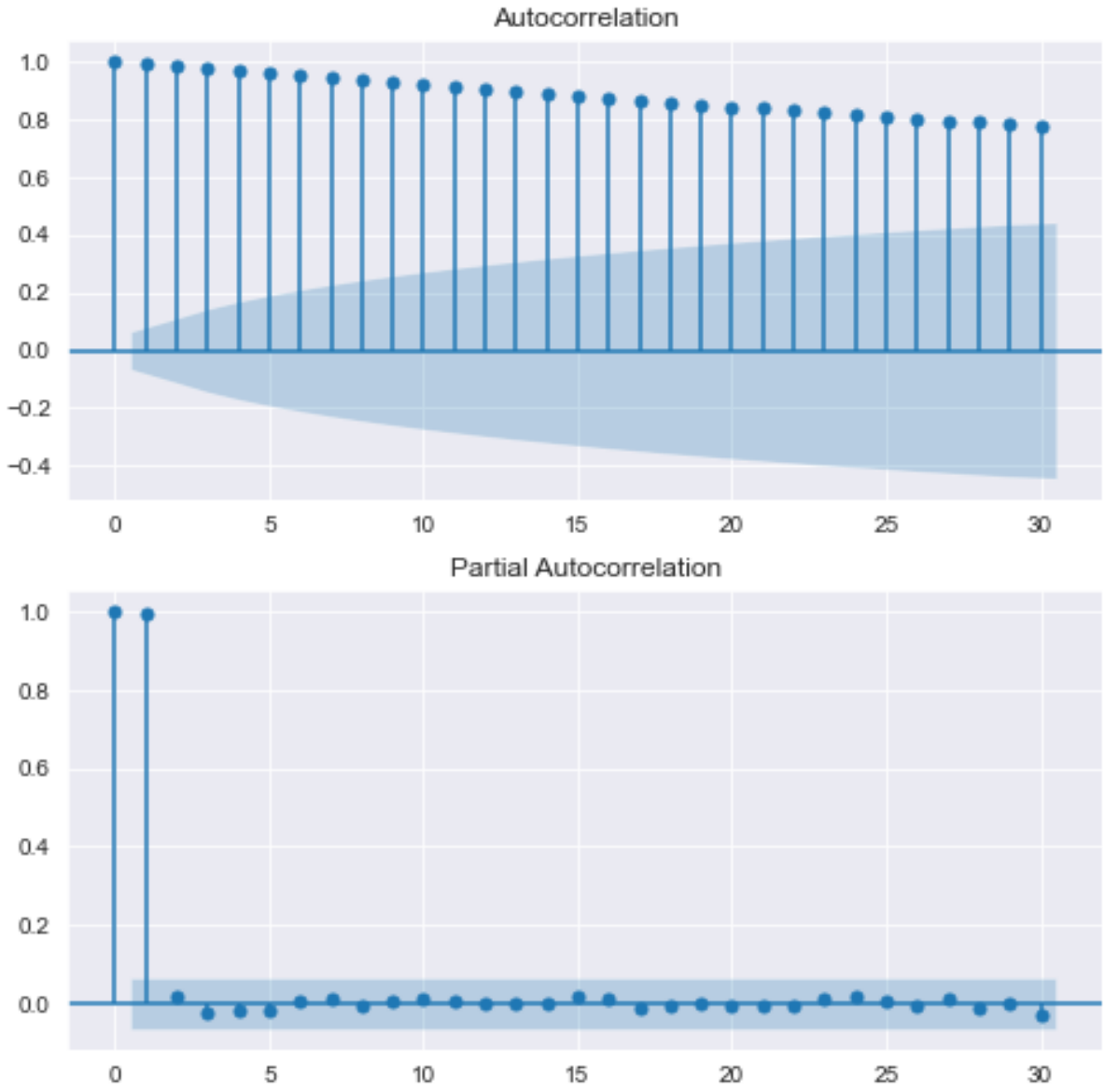

Trend
Seasonal
Resid
2006
2008
2010
2012
2014
2016
2018
2020

A análise dos correlogramas não deixou claro um modelo específico: a autocorrelação decai lentamente e a autocorrelação parcial não demonstra picos significativos após h=1.

```
# Correlogramas: ACF e PACF da série de preços do etanol
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(8,8))
plot_acf(df_etanol_resumo, ax = ax1)
plot_pacf(df_etanol_resumo, ax = ax2)
plt.show()
```



```
# Correlogramas: ACF e PACF da série de preços da gasolina
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(8,8))
plot_acf(df_gasolina_resumo, ax = ax1)
plot_pacf(df_gasolina_resumo, ax = ax2)
plt.show()
```



O teste de Dickey-Fuller apresentou valor de p alto para ambas as séries, refutando a hipótese de estacionariedade.

```
#Análise de estacionariedade para os preços do etanol
X = df_etanol_resumo.preco_venda
result = adfuller(X)
print('ADF Estatíticas: %f' % result[0])
print('Valor de P: %f' % result[1])
print('Valores Críticos:')
for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))
```

```
ADF Estatíticas: 1.154095
Valor de P: 0.995653
Valores Críticos:
        1%: -3.438
        5%: -2.865
        10%: -2.568
```

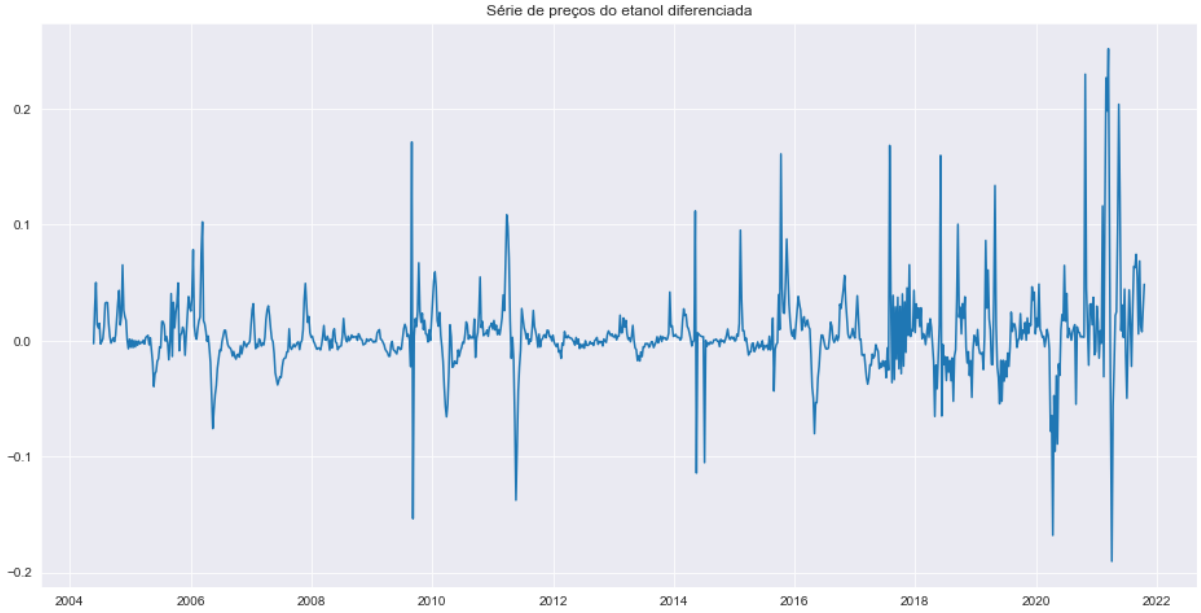
```
#Análise de estacionariedade para os preços da gasolina
X = df_gasolina_resumo.preco_venda
result = adfuller(X)
print('ADF Estatíticas: %f' % result[0])
print('Valor de P: %f' % result[1])
print('Valores Críticos:')
for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))
```

```
ADF Estatíticas: 2.224368
Valor de P: 0.998902
Valores Críticos:
        1%: -3.438
        5%: -2.865
        10%: -2.568
```

Porém, após diferenciação, as séries se tornam estacionárias, com valores de p abaixo de 0.05, conforme pode-se ver a seguir.

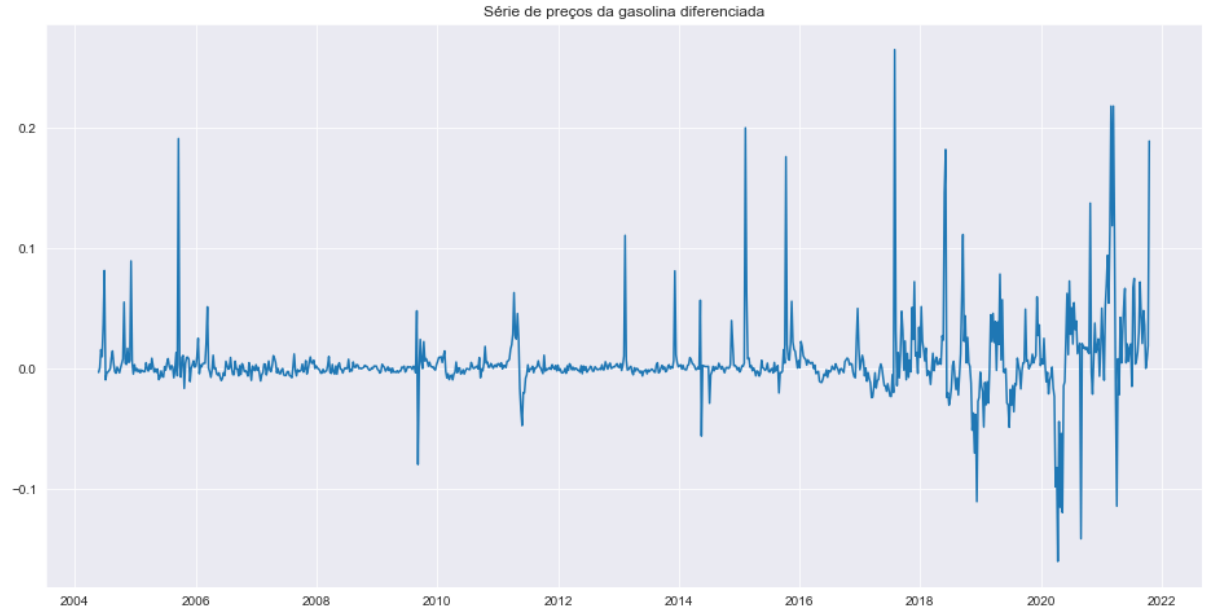
```
#Análise de estacionariedade para a série diferenciada de preços do etanol
X = etanol_diff.preco_venda
result = adfuller(X)
print('ADF Estatíticas: %f' % result[0])
print('Valor de P: %f' % result[1])
print('Valores Críticos:')
for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))
```

```
ADF Estatíticas: -13.195032
Valor de P: 0.000000
Valores Críticos:
        1%: -3.438
        5%: -2.865
        10%: -2.568
```




```
#Análise de estacionariedade para a série diferenciada de preços da gasolina
X = gasolina_diff.preco_venda
result = adfuller(X)
print('ADF Estatíticas: %f' % result[0])
print('Valor de P: %f' % result[1])
print('Valores Críticos:')
for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))
```

```
ADF Estatíticas: -13.324073
Valor de P: 0.000000
Valores Críticos:
        1%: -3.438
        5%: -2.865
        10%: -2.568
```



A série de taxa de câmbio não teve uma análise estatística profunda, dado que será apenas suporte para previsão dos dados de preços dos combustíveis.

5. Criação de Modelos de Machine Learning

Dois modelos foram utilizados neste trabalho: ARIMA – *AutoRegressive Integrated Moving Average* e Redes Neurais do tipo LSTM – *Long Short Term Memory*.

As redes neurais LSTM – *Long Short Term Memory*, buscam resolver os problemas de memória curta dos outros modelos de redes neurais, trazendo neurônios mais robustos, com blocos de memória.

Já o modelo ARIMA trata-se de uma generalização do modelo ARMA para séries não estacionárias. A parte AR (*autoregressive*) trata da regressão da variável em seus valores anteriores, enquanto a parte MA (*moving average*) envolve a modelagem da variável como combinação linear de erros passados. Os parâmetros p e q se referem à ordem da parte regressiva e da parte de médias móveis, respectivamente. A parte I (*integrated*) refere-se à ordem de diferenciação necessária para que a série se torne estacionária, com d sendo o parâmetro representativo. Dessa forma, podemos generalizar um modelo ARIMA como ARIMA(p, d, q).

Nos casos de séries sazonais, os modelos possuem também os parâmetros relativos à sazonalidade e passam a ser denotados como ARIMA(p, d, q)(P, D, Q)$_m$, em que P, D e Q se referem à ordem dos termos de auto regressão, diferenciação e médias móveis da parte sazonal, respectivamente, e m representa o número de períodos para um ciclo sazonal.

5.1. ARIMA

Primeiramente, criou-se o modelo para análise da série de preços do etanol. Foram estipulados, para o método auto_arima da biblioteca pmdarima, os parâmetros de p e q com valores de 1 a 6, m igual a 52, para verificação de sazonalidade anual em dados semanais, e d=1, visto que a série se torna estacionária após diferenciação de primeira ordem, conforme verificado anteriormente.

```
arima_etanol = auto_arima(df_etanol_resumo, start_p=1, start_q=1, max_p=6, max_q=6,
                          m=52, d=1, seasonal=True, trace=True, error_action='ignore', suppress_warnings=True, stepwise=True)
```

Após o processamento, o melhor modelo encontrado foi o SARIMAX(3, 1, 0), ou seja, um modelo AR, com 3 termos auto regressivos e diferenciação de primeira ordem. Não foi sugerido nenhum termo sazonal no resultado, conforme pode-se visualizar na figura abaixo.

SARIMAX Results

Dep. Variable:	y	No. Observations:	764
Model:	SARIMAX(3, 1, 0)	Log Likelihood	1778.960
Date:	Sat, 30 Oct 2021	AIC	-3547.921
Time:	15:45:58	BIC	-3524.734
Sample:	0	HQIC	-3538.994
	- 764		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	0.0012	0.001	1.204	0.229	-0.001	0.003
ar.L1	0.2589	0.016	15.907	0.000	0.227	0.291
ar.L2	0.2778	0.022	12.511	0.000	0.234	0.321
ar.L3	0.0115	0.030	0.385	0.700	-0.047	0.070
sigma2	0.0006	1.02e-05	54.383	0.000	0.001	0.001

Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	16566.61
Prob(Q):	0.97	Prob(JB):	0.00
Heteroskedasticity (H):	4.18	Skew:	0.75
Prob(H) (two-sided):	0.00	Kurtosis:	25.78

Para a análise da série de preços da gasolina, foram utilizados os mesmos parâmetros da série de etanol para a busca do melhor modelo (p e q com valores de 1 a 6, m igual a 52 e d=1).

```
arima_gasolina = auto_arima(df_gasolina_resumo, start_p=1, start_q=1, max_p=6, max_q=6,
                      m=52, d=1, seasonal=True, trace=True, error_action='ignore', suppress_warnings=True, stepwise=True)
```

Neste caso, o melhor modelo encontrado foi o SARIMAX(2, 1, 0), ou seja, também um modelo AR, mas com 2 termos auto regressivos. Aqui também não foi indicada a existência de termos sazonais no modelo.

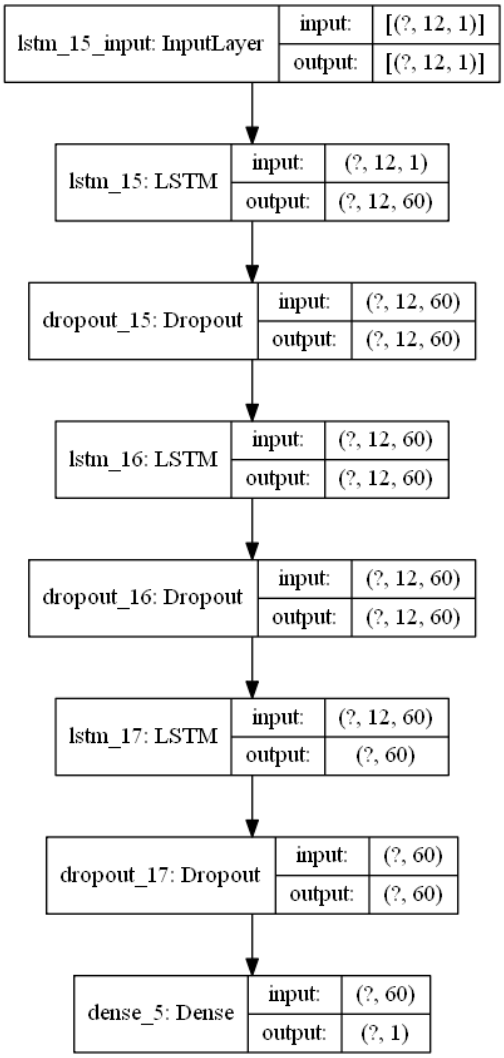
```
                               SARIMAX Results
==============================================================================
Dep. Variable:                      y   No. Observations:                  764
Model:               SARIMAX(2, 1, 0)   Log Likelihood                1828.481
Date:                Sat, 30 Oct 2021   AIC                            -3648.962
Time:                        15:50:11   BIC                            -3630.413
Sample:                             0   HQIC                           -3641.820
                                  - 764
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      0.0022      0.001      1.642      0.101      -0.000       0.005
ar.L1          0.2449      0.019     13.189      0.000       0.209       0.281
ar.L2          0.0630      0.026      2.464      0.014       0.013       0.113
sigma2         0.0005   7.43e-06     65.298      0.000       0.000       0.000
===================================================================================
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):             97737.61
Prob(Q):                              0.98   Prob(JB):                         0.00
Heteroskedasticity (H):               3.98   Skew:                             5.61
Prob(H) (two-sided):                  0.00   Kurtosis:                        57.30
===================================================================================
```

5.2. Rede Neural LSTM

Para a configuração da rede neural foi escolhida uma janela de 12. Este parâmetro representa o número de observações anteriores que a rede avalia para prever o próximo valor. Os dados foram normalizados e foi utilizada uma rede com 3 camadas LSTM com 60 neurônios cada, batch_size de 16 e 150 épocas de treinamento.

```
# Modelo LSTM - univariado

Model_uni = Sequential()

Model_uni.add(LSTM(units = 60, return_sequences = True, input_shape = (X_treino.shape[1],1)))
Model_uni.add(Dropout(0.2))

Model_uni.add(LSTM(units = 60, return_sequences = True))
Model_uni.add(Dropout(0.2))

Model_uni.add(LSTM(units = 60))
Model_uni.add(Dropout(0.2))

Model_uni.add(Dense(units = 1))

Model_uni.compile(optimizer = 'adam', loss = 'mean_squared_error')

Model_uni.fit(X_treino, y_treino, epochs = 150, batch_size = 16)

Model_uni.summary()
```

Após o treinamento com os dados da série de etanol, o modelo resultante pode ser visto abaixo.



O código para geração do modelo para a série de preços da gasolina utilizou os mesmos parâmetros e gerou um modelo similar.

Após a criação dos modelos univariados, optou-se pela criação de modelos multivariados, incluindo a série de valores do dólar.

O modelo criado seguiu a mesma estrutura, conforme código abaixo.

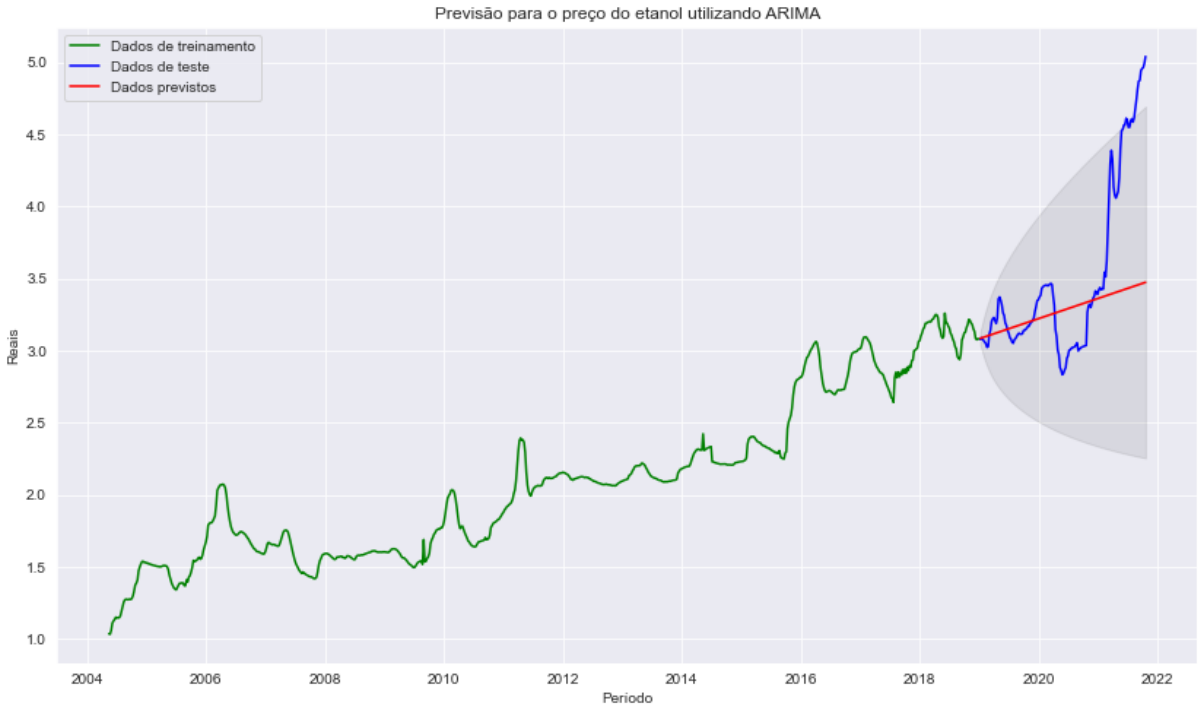
```
# Modelo LSTM - multivariado

Model_multi = Sequential()

Model_multi.add(LSTM(units = 60, return_sequences = True, input_shape = (X_treino_m.shape[1], 2)))
Model_multi.add(Dropout(0.2))

Model_multi.add(LSTM(units = 60, return_sequences = True))
Model_multi.add(Dropout(0.2))

Model_multi.add(LSTM(units = 60))
Model_multi.add(Dropout(0.2))

Model_multi.add(Dense(units = 1))

Model_multi.compile(optimizer = 'adam', loss = 'mean_squared_error')

Model_multi.fit(X_treino_m, y_treino_m, epochs = 150, batch_size = 16)

Model_multi.summary()
```

6. Apresentação dos Resultados

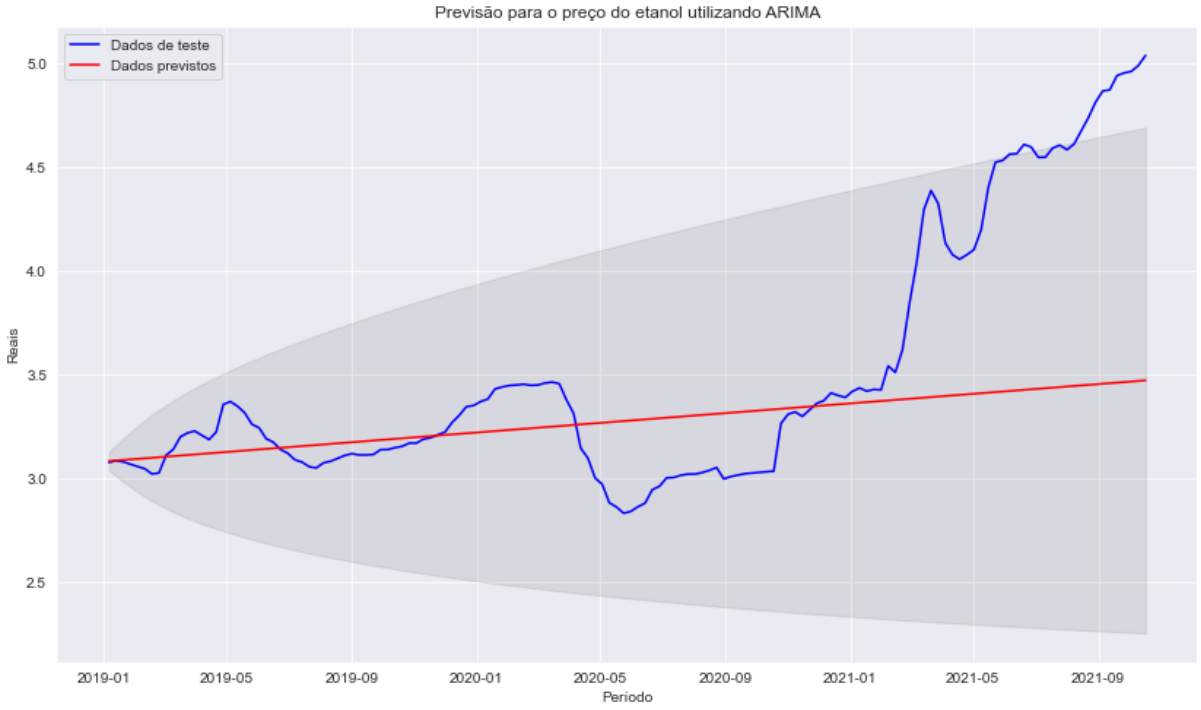
6.1. ARIMA

Os dados de preços do etanol e da gasolina foram divididos em conjuntos de dados para treino (até dezembro de 2018) e teste (janeiro de 2019 a agosto de 2021).

O modelo ARIMA encontrado pelo método auto_arima para a série de preços do etanol, quando aplicado aos dados de teste, forneceu como previsão os valores demonstrados no gráfico abaixo.



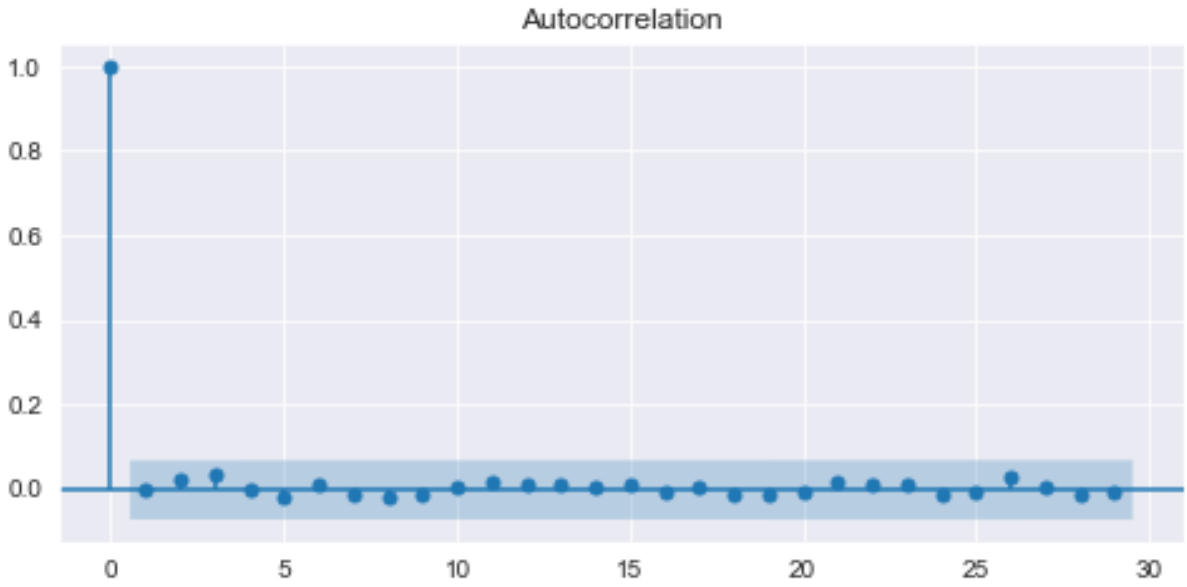
No detalhe do gráfico a seguir, percebe-se que a grande maioria dos valores observados se encontra dentro do intervalo de confiança de 95% dos valores previstos, à exceção dos valores posteriores a junho de 2021.

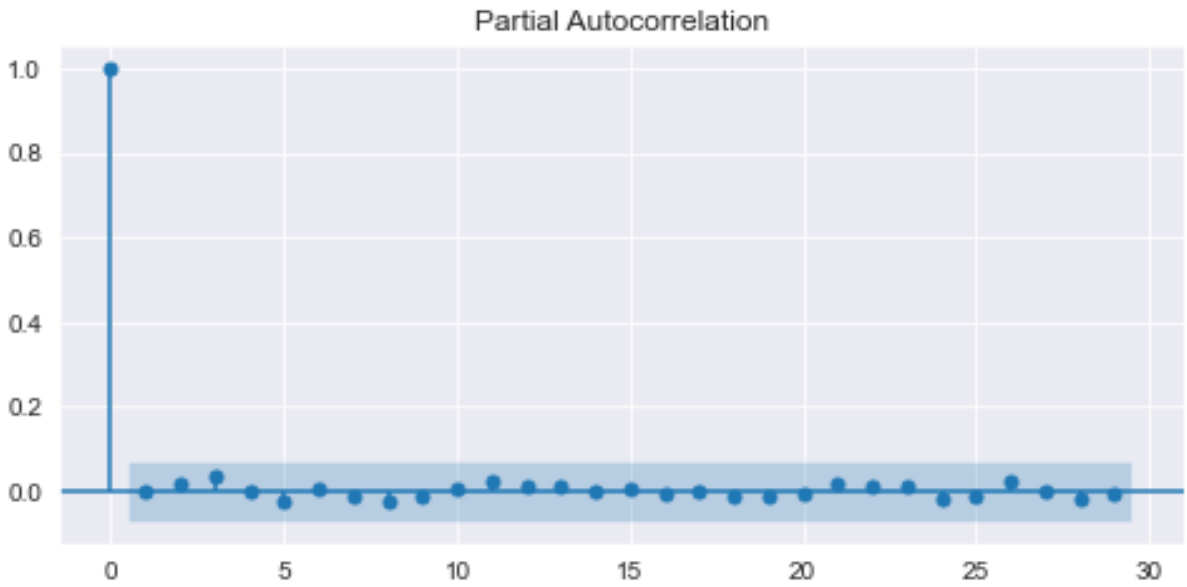


O erro obtido e os correlogramas para os erros seguem abaixo.

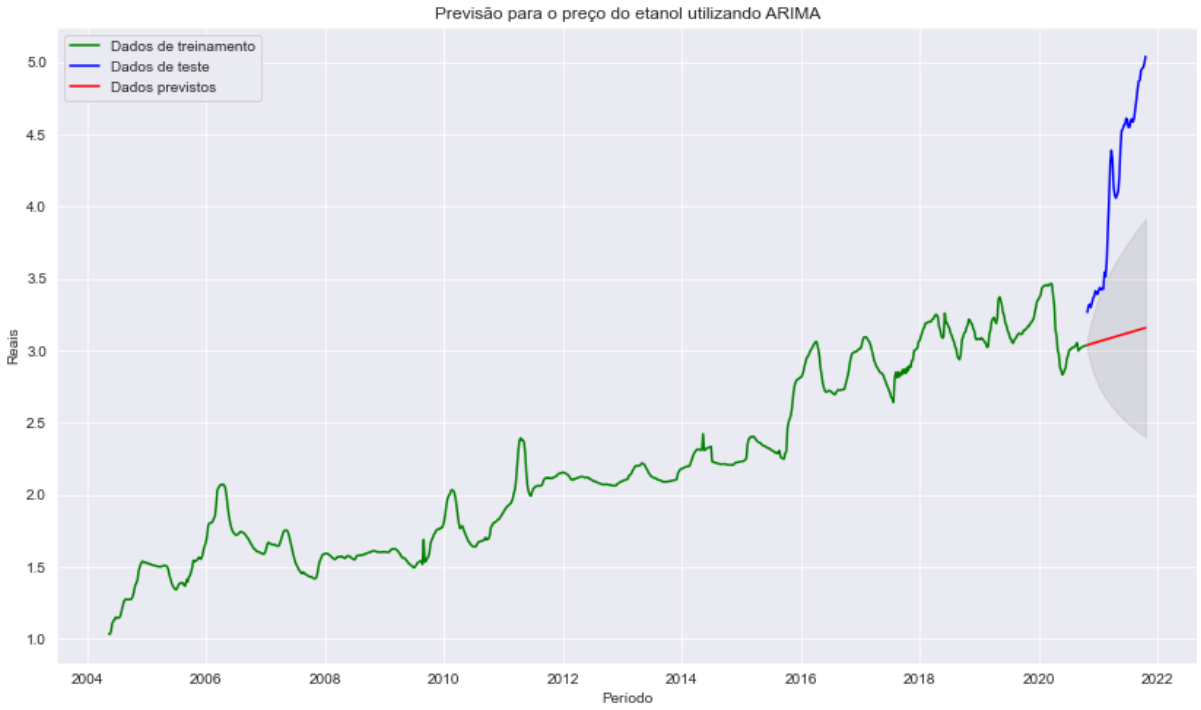
```
#Cálculo do erro
mse = mean_squared_error(teste, previsao)
print('MSE: '+str(mse))
mae = mean_absolute_error(teste, previsao)
print('MAE: '+str(mae))
rmse = math.sqrt(mean_squared_error(teste, previsao))
print('RMSE: '+str(rmse))
```

```
MSE: 0.3236850893827139
MAE: 0.3646505568114602
RMSE: 0.5689332908019304
```

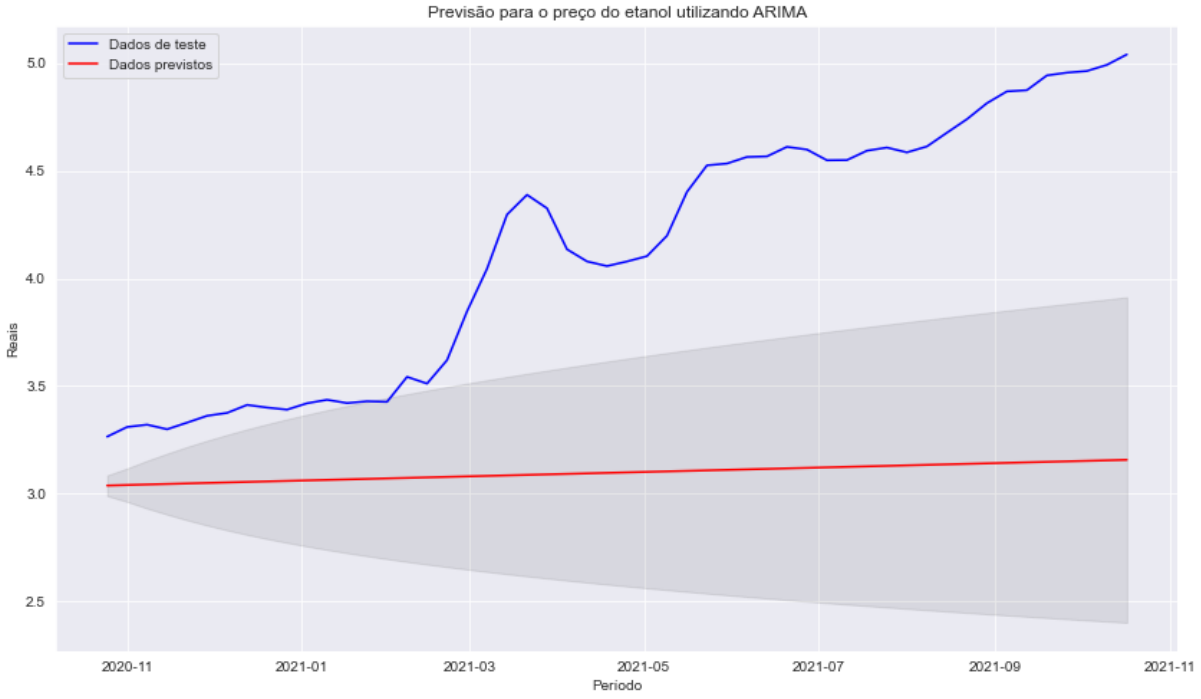




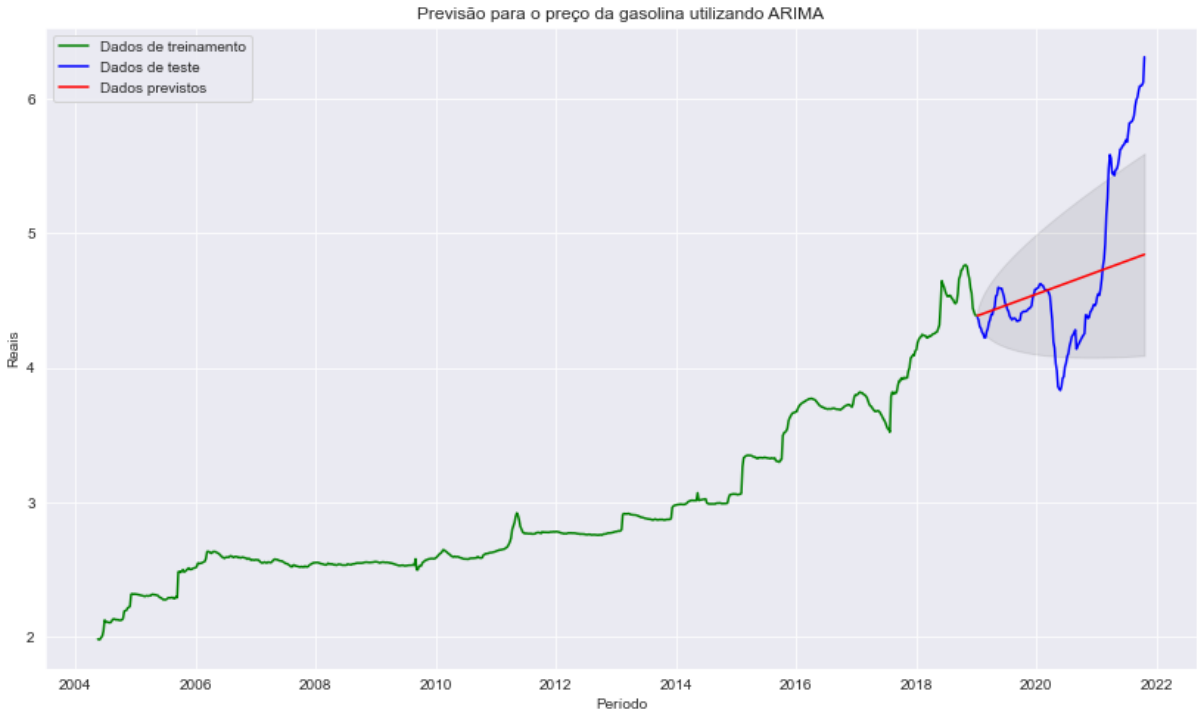
Porém, quando se realiza uma previsão para o último ano com dados disponíveis (52 semanas), obtém-se um resultado pior do que o anterior, como se pode verificar no gráfico abaixo.



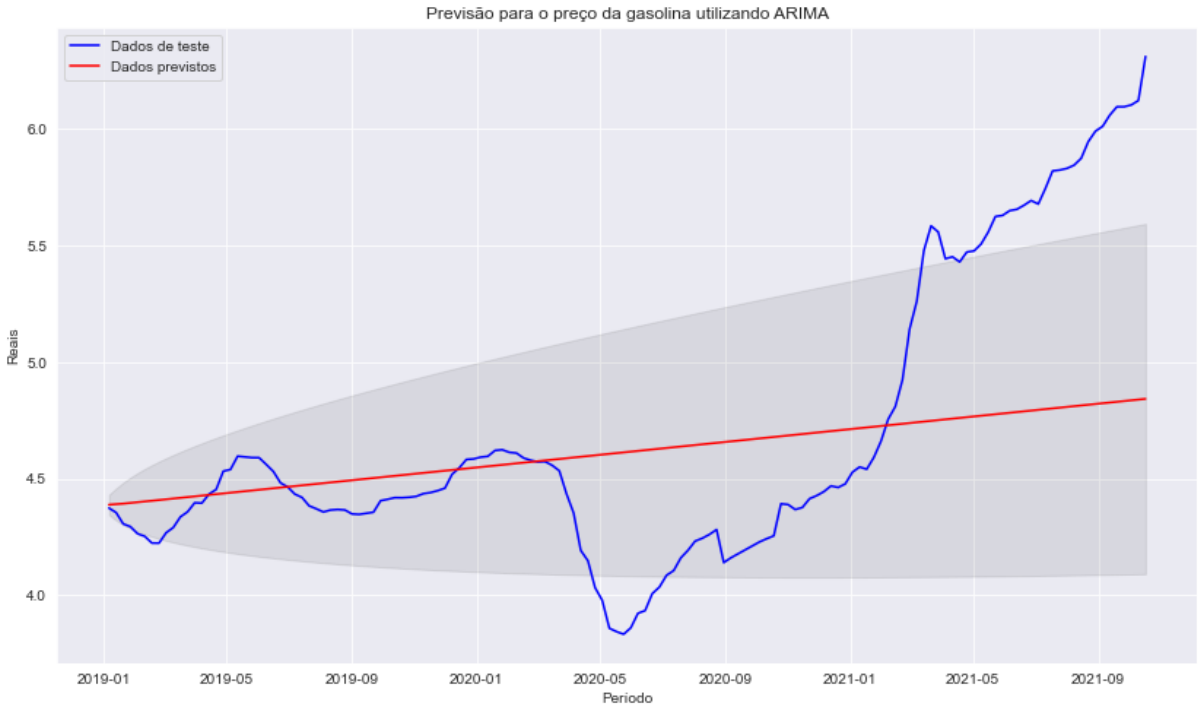
Abaixo, segue o gráfico ampliado para melhor visualização.



Na análise da série de preços da gasolina, para o modelo ARIMA correspondente, os resultados são parecidos. A previsão levando-se em conta as últimas 146 semanas gera resultados em sua maioria dentro do intervalo de confiança.



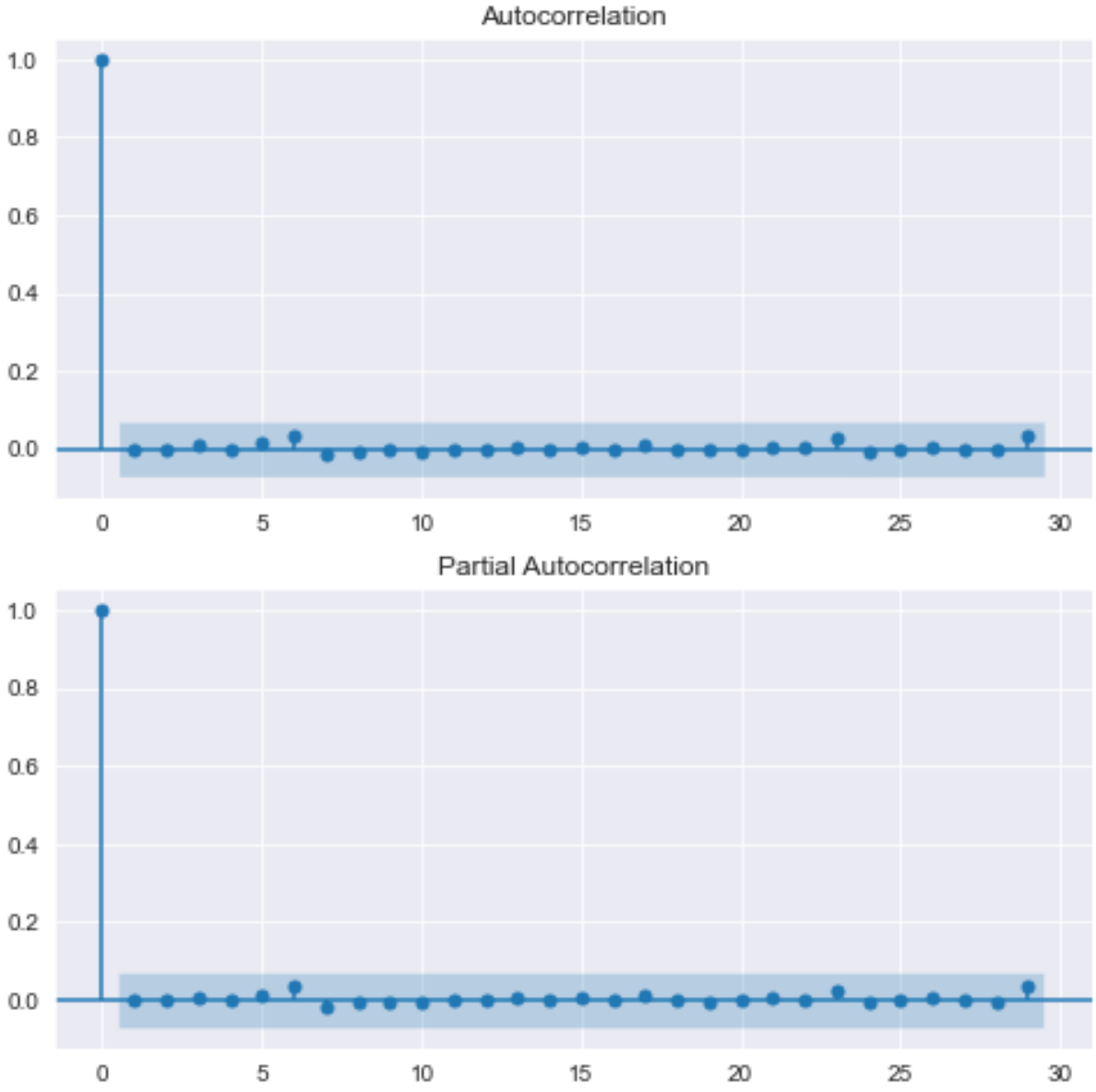
Abaixo, o gráfico mostra mais detalhadamente.



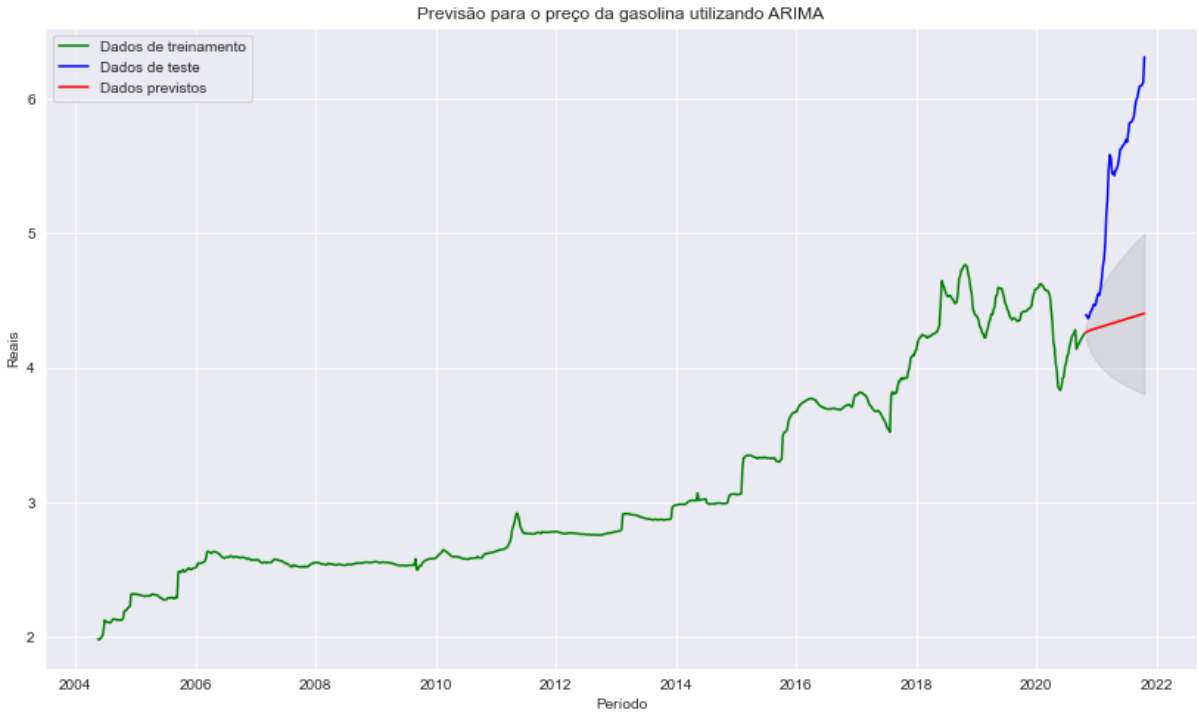
Segue o erro obtido e os correlogramas dos erros:

```
#Cálculo do erro
mse = mean_squared_error(teste, previsao)
print('MSE: '+str(mse))
mae = mean_absolute_error(teste, previsao)
print('MAE: '+str(mae))
rmse = math.sqrt(mean_squared_error(teste, previsao))
print('RMSE: '+str(rmse))
```

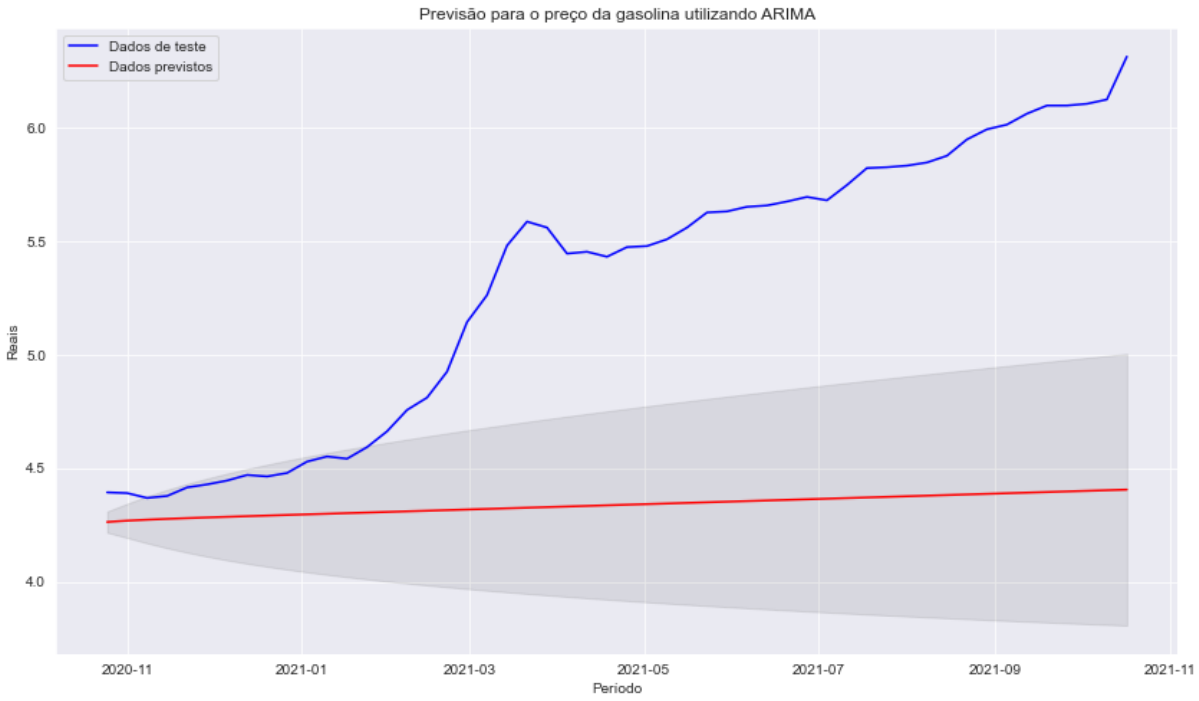
```
MSE: 0.2853004224666578
MAE: 0.3840371857849808
RMSE: 0.5341352099109904
```



A previsão para o último ano com dados disponíveis (52 semanas) para a série de preços da gasolina também gera um resultado pior.



Segue gráfico ampliado para melhor visualização.



6.2. Rede Neural LSTM

Os dados das séries de preços de combustíveis, antes de serem aplicados aos modelos de redes neurais descritos na seção anterior, foram normalizados, divididos em conjuntos de dados para treino e teste e, por fim, estruturados em janelas de análise de tamanho 12.

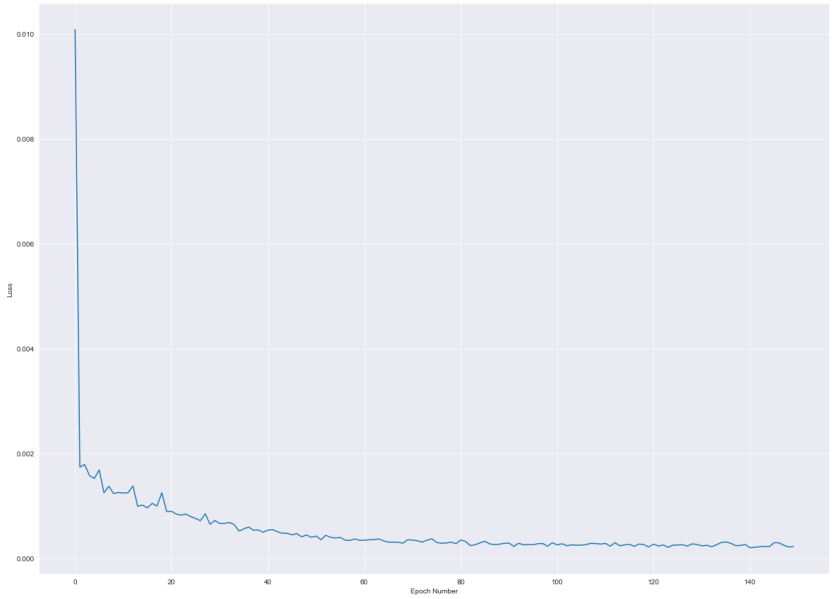
```
# Normalização
scaler = MinMaxScaler(feature_range = (0,1))
df_etanol_scaled = scaler.fit_transform(df_etanol_resumo)
```

```
# Divisão do conjunto de dados em treino e teste
etanol_treino = df_etanol_scaled[0:-n_periodos_previsao]
etanol_teste = df_etanol_scaled[-n_periodos_previsao:]
```

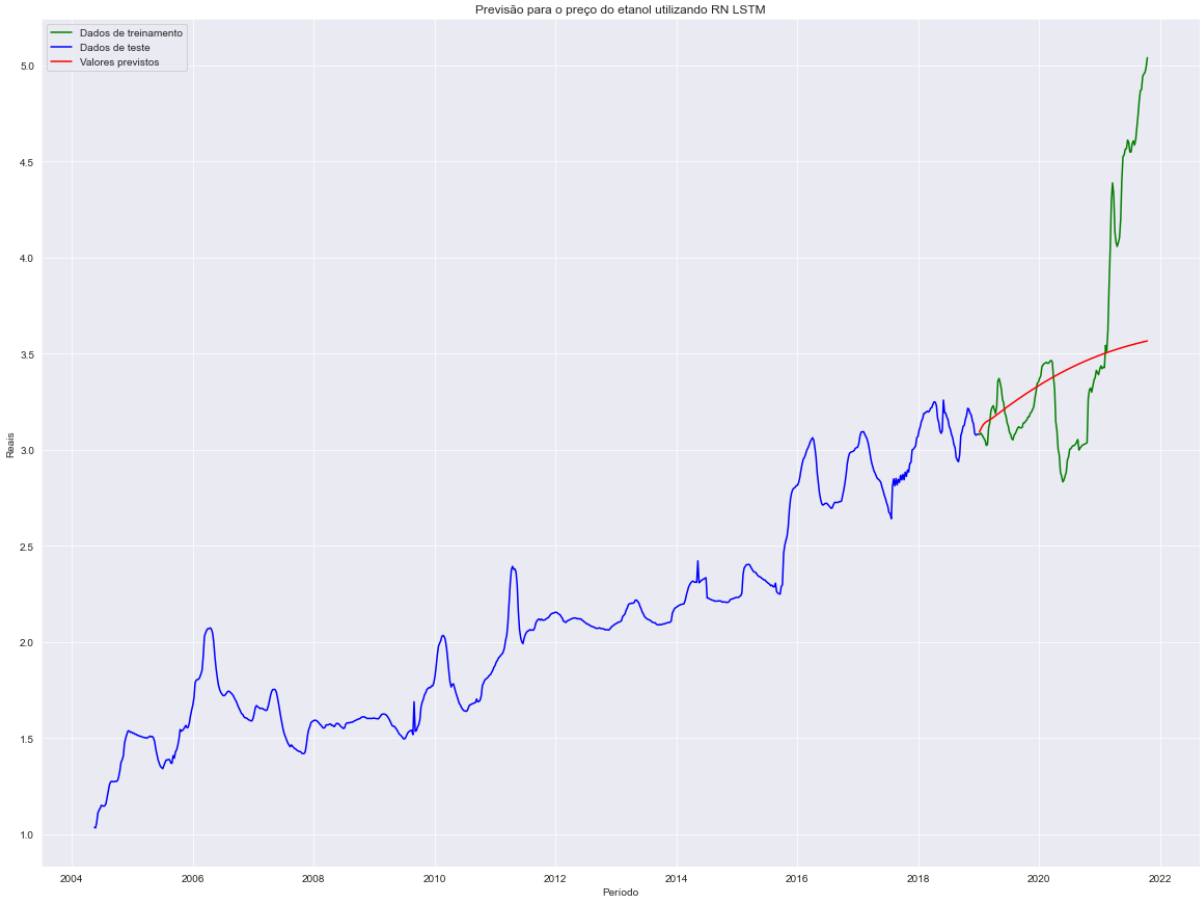
```
# Criação da estrutura de dados com janelas
X_treino = []
y_treino = []

for i in range(janela, len(etanol_treino)):
    X_treino.append(etanol_treino[i-janela:i, :])
    y_treino.append(etanol_treino[i, :])

X_treino, y_treino = np.array(X_treino), np.array(y_treino)
```

O gráfico da perda por época ao se realizar o treinamento da rede neural para os dados de preços do etanol pode ser visto a seguir. O gráfico para os dados de preços da gasolina é similar.

Vemos que não seria necessário configurar um número maior de épocas para treinamento, pois, aproximadamente a partir da época 80, não há uma melhora significativa.



A previsão realizada com o modelo treinado pode ser vista abaixo.

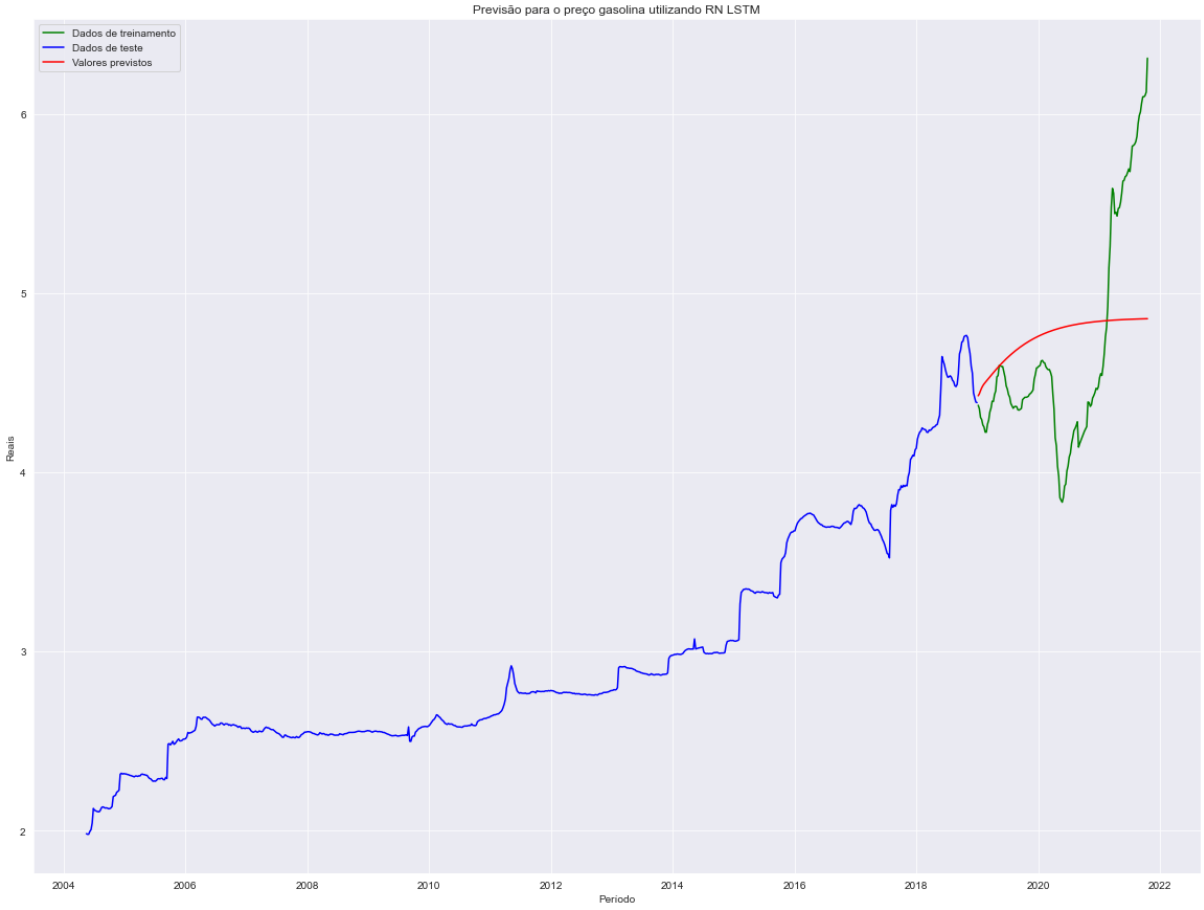


Abaixo verifica-se o erro.

```
# Cálculo do erro
print('MAE: ', mean_absolute_error(etanol_teste, previsao))
print('MSE: ', mean_squared_error(etanol_teste,previsao))
print('RMSE: ', np.sqrt(mean_squared_error(etanol_teste,previsao)))
```

```
MAE:  0.3675552555329129
MSE:  0.28958388158614157
RMSE:  0.5381299857712275
```

Vemos que a previsão não teve um resultado tão bom. O mesmo ocorreu com a previsão para a série de preços da gasolina, cujo gráfico e erros calculados podem ser vistos a seguir.



```
# Cálculo do erro e R²
print('MAE: ', mean_absolute_error(gasolina_teste, previsao))
print('MSE: ', mean_squared_error(gasolina_teste,previsao))
print('RMSE: ', np.sqrt(mean_squared_error(gasolina_teste,previsao)))
```

```
MAE:  0.47551817797886503
MSE:  0.3347905382945724
RMSE:  0.5786108694922455
```

Com base nestes resultados, optou-se por adicionar uma variável ao modelo, criando uma rede neural LSTM multivariada, levando-se em conta a série de taxa de câmbio do dólar.

Assim, criou-se uma estrutura com duas séries: a série com a variável alvo, o preço do combustível desejado, e a série com a variável de suporte, a taxa de câmbio do dólar, conforme pode-se ver abaixo.

```
etanol_multi = df_dolar_resumo.copy()
etanol_multi['preco_venda'] = df_etanol_resumo.preco_venda
```

```
etanol_multi.head()
```

data	cambio	preco_venda
2004-05-16	3.11680	1.035783
2004-05-23	3.15940	1.033012
2004-05-30	3.13800	1.061085
2004-06-06	3.15540	1.110978
2004-06-13	3.12225	1.124888

Os dados foram então normalizados, e divididos em conjunto de teste e treino, de acordo com a janela de doze observações.

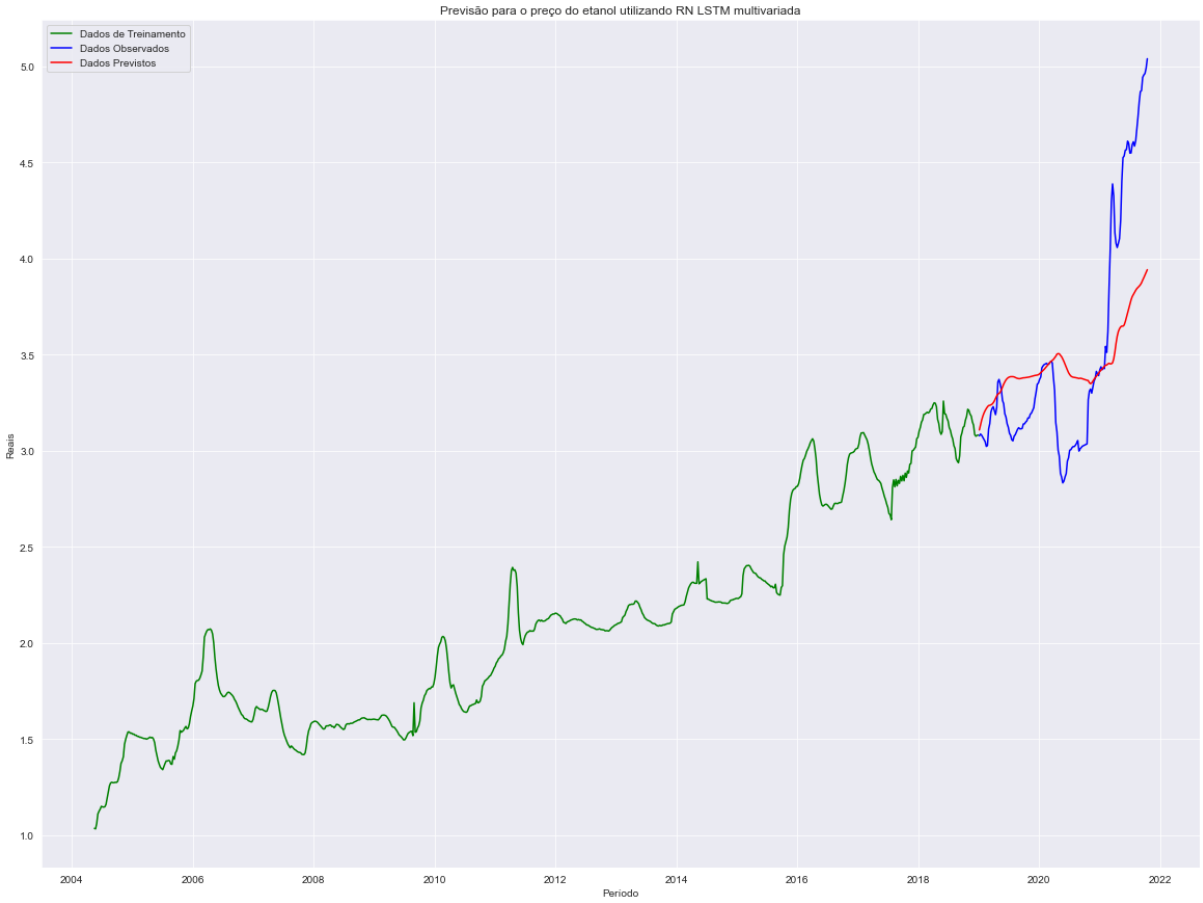
```
# Normalização
scaler = MinMaxScaler(feature_range = (0,1))
etanol_multi_scaled = scaler.fit_transform(etanol_multi)
```

```
# Divisão do conjunto de dados em treino e teste
treino_multi = etanol_multi_scaled[:-n_periodos_previsao,:]
teste_multi = etanol_multi_scaled[-n_periodos_previsao:,1]
```

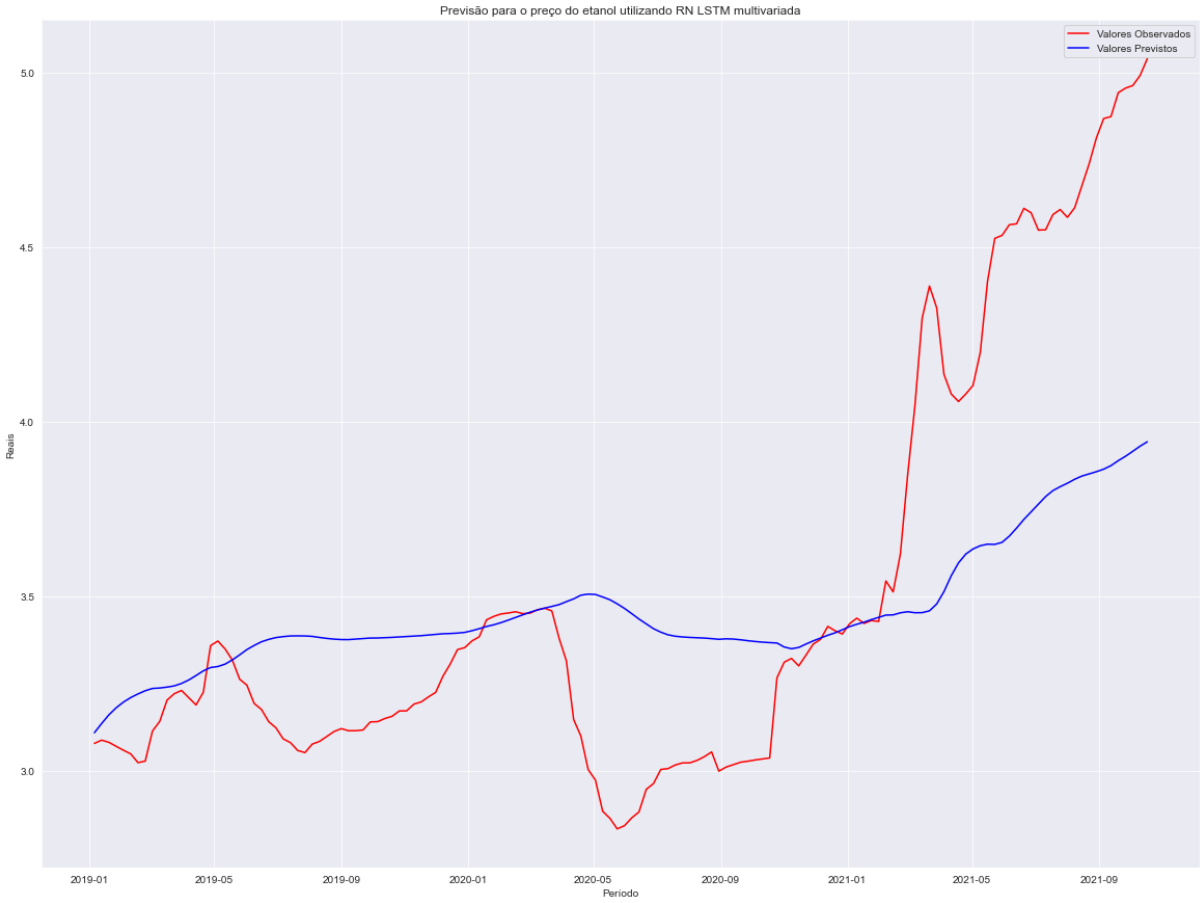
```
# Criação da estrutura de dados com janelas
X_treino_m = []
y_treino_m = []

for i in range(janela, len(treino_multi)):
    X_treino_m.append(treino_multi[i-janela:i, 0:2])
    y_treino_m.append(treino_multi[i, 1])

X_treino_m, y_treino_m = np.array(X_treino_m), np.array(y_treino_m)
```

A previsão foi então realizada e o gráfico com os resultados pode ser visto mais abaixo.

```
# Prevendo os valores
previsao = []

lote_um = treino_multi[-janela:]
novo_lote = lote_um.reshape((1,janela,2))

for i in range(n_periodos_previsao):
    previsao_unitaria = Model_multi.predict(novo_lote)[0]
    #print(previsao_unitaria)
    previsao.append(previsao_unitaria)
    aux = teste_multi[i]
    aux = aux.reshape(1,1)
    novo_item_teste = np.insert(aux, 1, previsao_unitaria, axis = 1)
    novo_item_teste= novo_item_teste.reshape(1,1,2)
    novo_lote = np.append(novo_lote[:, 1:,:], novo_item_teste, axis=1)

previsao = np.array(previsao)
```



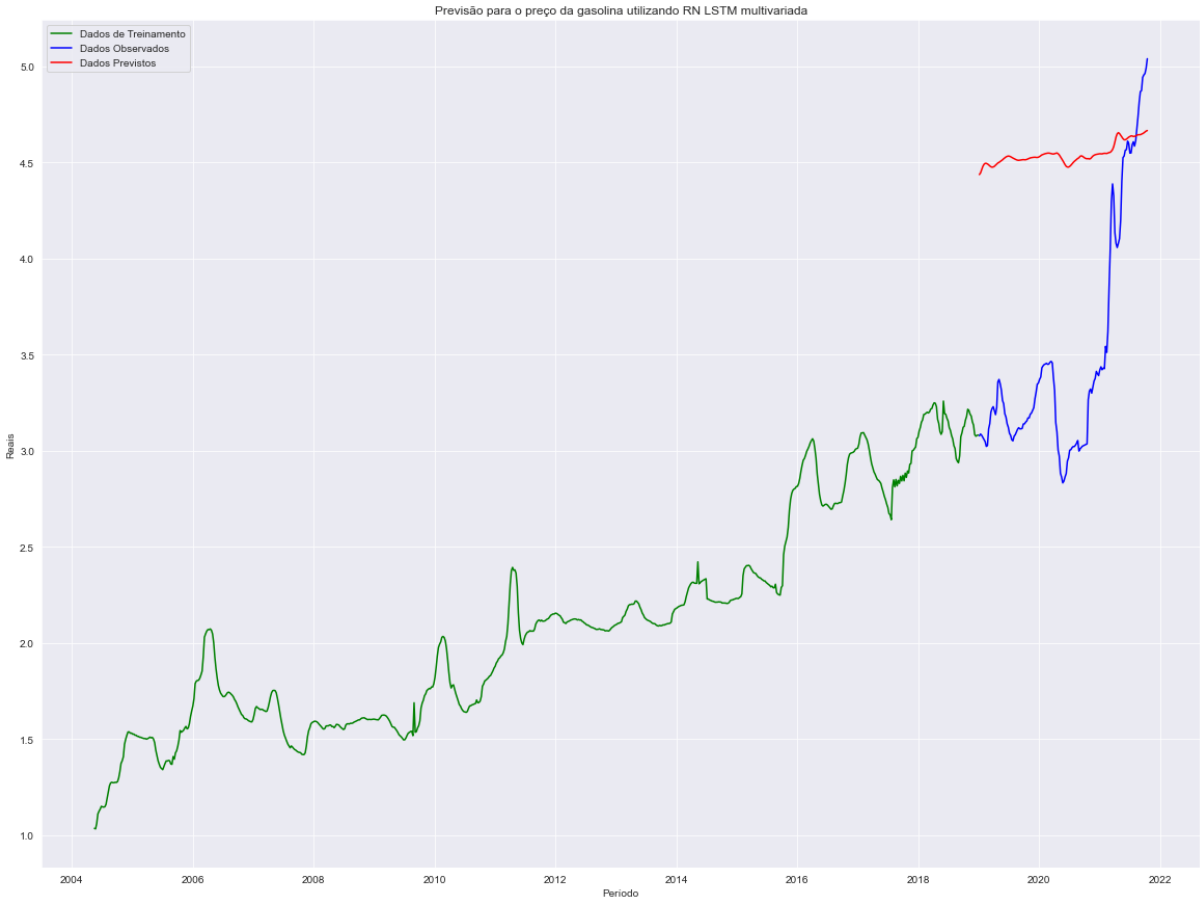
Abaixo é possível visualizar o gráfico ampliado.

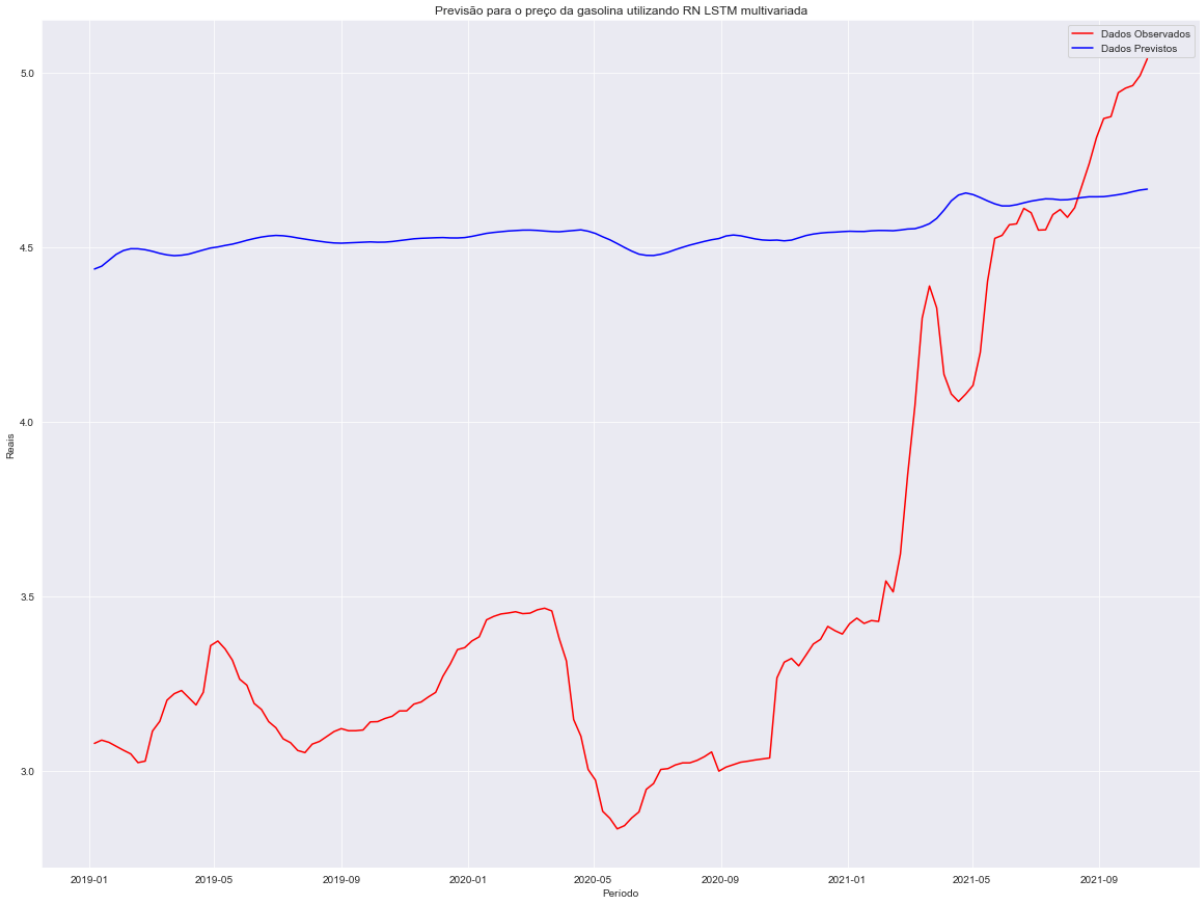



```
# Cálculo do erro
print('MAE: ', mean_absolute_error(teste_multi,previsao_plot))
print('MSE: ', mean_squared_error(teste_multi,previsao_plot))
print('RMSE: ', np.sqrt(mean_squared_error(teste_multi,previsao_plot)))
```

```
MAE:  2.8371285109059157
MSE:  8.05675578140416
RMSE:  2.8384424921784412
```

A seguir podem ser vistos os resultados para a previsão multivariada dos preços da gasolina, bem como o erro.


Previsão para o preço da gasolina utilizando RN LSTM multivariada
Dados de Treinamento
Dados Observados
Dados Previstos
Reais
Periodo
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
2004
2006
2008
2010
2012
2014
2016
2018
2020
2022




```
# Cálculo do erro
print('MAE: ', mean_absolute_error(teste_multi,previsao_plot))
print('MSE: ', mean_squared_error(teste_multi,previsao_plot))
print('RMSE: ', np.sqrt(mean_squared_error(teste_multi,previsao_plot)))
```

```
MAE:  3.917973750350518
MSE:  15.35975930616792
RMSE:  3.9191528811935776
```

Podemos ver que a rede neural multivariada consegue identificar variações. Na série de preços da gasolina, ela conseguiu prever uma alta nos valores, porém não foi capaz de identificar o período correto de ocorrência, gerando um alto valor de erro. O erro para a previsão dos preços de etanol foi menor, mas ainda alto.

As previsões realizadas pelo modelo ARIMA tiveram um erro menor, neste caso particular.

7. Links

Link para o repositório com os códigos:
https://github.com/SissiFagundes/TCCPUCMinas.git

Link para o vídeo:
https://youtu.be/6wFn0MUUnzg

REFERÊNCIAS

BOX, GEORGE et al. Time Series Analysis: Forecasting and Control. New Jersey: Wiley, 2016.

BROCKWELL, PETER J.; DAVIS, RICHARD A. Introduction to Time Series and Forecasting. New York: Springer, 2002.

ESLING, PHILIPPE; AGON, CARLOS. Time-Series Data Mining. ACM Computing Surveys, 2012.