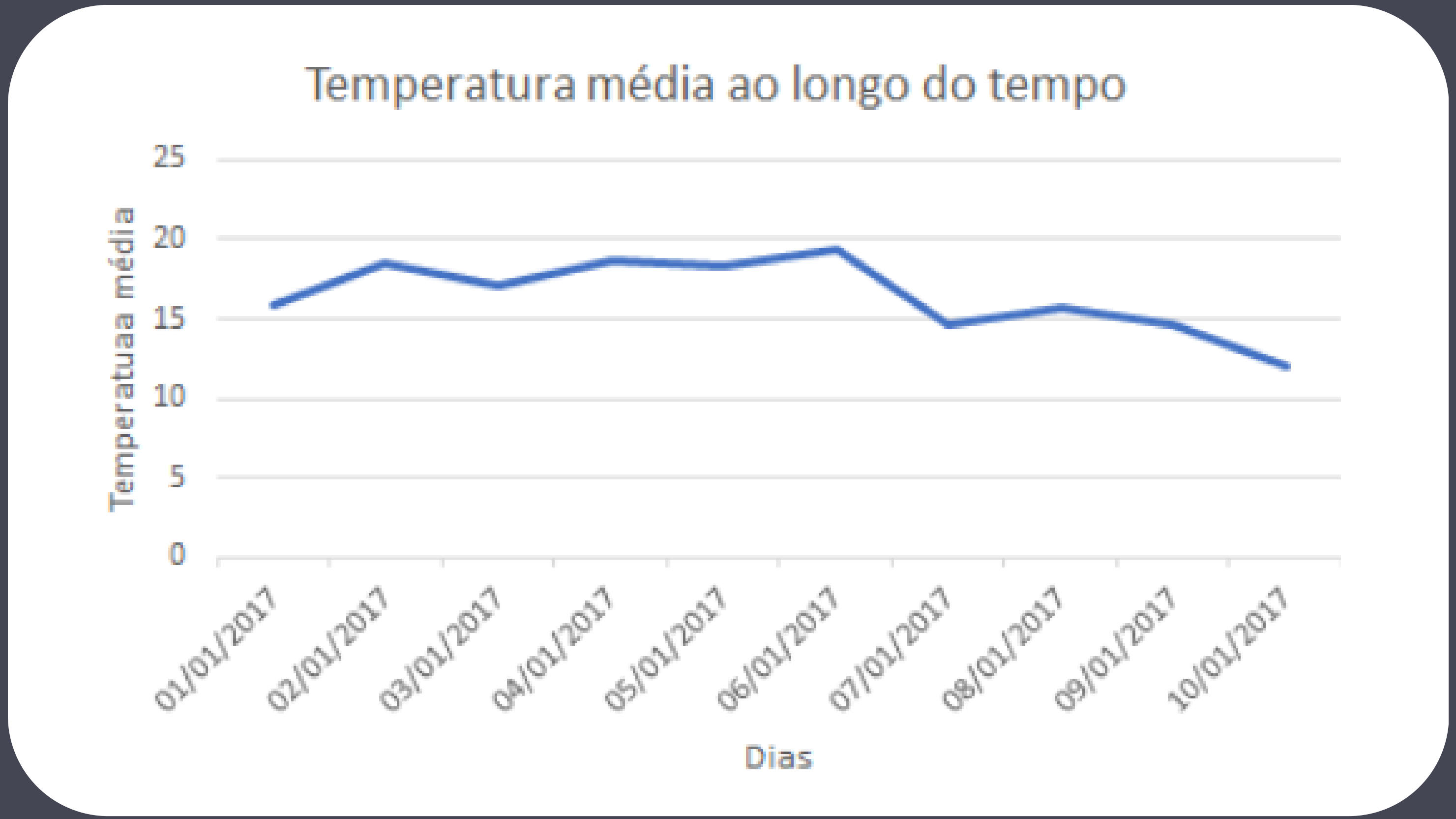
Séries Temporais

Análise e previsão pelos métodos ARMA e ARIMA

Julia Fischer

Definindo uma série temporal

- Conjunto de observações de dados coletados por um determinado período. São organizadas de forma sequencial em função do tempo.



Classificando uma série temporal

- Variáveis analisadas: univariada, multivariada.

(ex.: temperatura, pressão, altura...)

- Dimensões analisadas: unidimensional, multidimensional.

(ex.: tempo, latitude, longituge...)

- Tempo de registro: série discreta ou contínua.

(ex.: registro diário, mensal, contínuo...)

Objetivos e aplicações da análise

Objetivos:

- Investigar mecanismo gerador;
- Fazer previsões de curto e longo prazo;
- Descrever comportamento e variações;
- Procurar periodicidade relevante;

Aplicações:

- Demandas empresariais;
- Análises financeiras e modelos econômicos;
- Previsões metereológicas;
- Estudos de fenômenos geofísicos e astronômicos;
- Modelos epidemiológicos;

Entre outras...

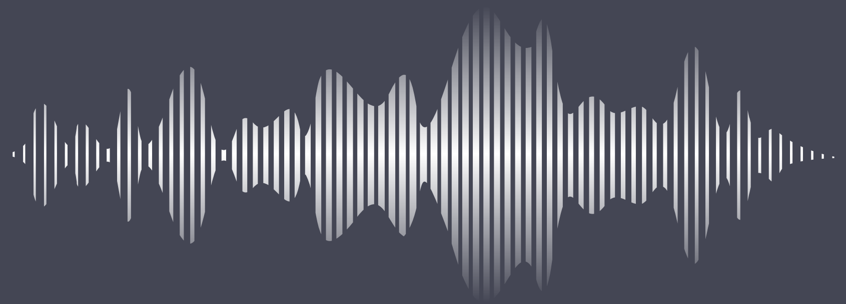
Componentes das Séries Temporais

Tendência

- Comportamento geral da série ao longo do tempo: crescente ou decrescente.

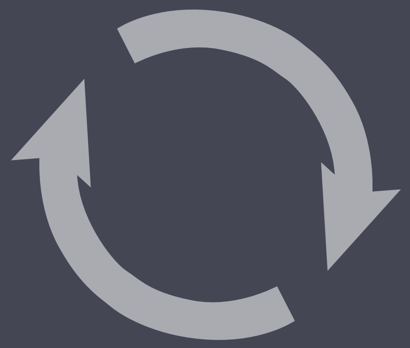
Sazonalidade

- Padrões que se repetem em intervalos de tempo fixos, influenciados por fatores sazonais.



Ruído

- Flutuações aleatórias, mas contínuas na série temporal: eventos imprevistos, erros de medição...



Ciclos

- Variações de longo prazo, que não se repentes em intervalos fixos de tempo e não estão ligadas a fatores sazonais.

Análise de séries temporais

Método ARMA:

- ARMA - AutoRegressive Moving Average (Média móvel autoregressiva).
- Combina componente auto-regressiva, indicada pela ordem ‘p’. Com a componente da média móvel de ordem ‘q’.

Método ARMA:

- Componente AutoRegressiva (AR):

Modela como os valores passados da série influenciam os valores futuros. Usa as ‘**p**’ observações anteriores para construir suas previsões.

- Modelo AR(p):

$$X_t = c + \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + \varepsilon_t$$

Onde:

Xt é o valor da série temporal no tempo t.

c é uma constante.

ϕ1,ϕ2,...,ϕp são os coeficientes dos atrasos anteriores.

εt é o erro aleatório no tempo t.

Método ARMA:

- Componente de Média Móvel (MA):

Modela como os valores passados dos erros (resíduos) influenciam os valores atuais. Constrói sua previsão com as **'q'** observações anteriores dos erros.

- Modelo MA(q):

$$X_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q}$$

Onde:

Xt é o valor da série temporal no tempo t.

c é uma constante.

εt é o erro aleatório no tempo t.

θ1,θ2,...,θq são os coeficientes dos erros anteriores.

Método ARMA:

- Combinando as componentes, a série é representada pelo modelo ARMA (p,q):

$$X_t = c + \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q}$$

Onde:

Xt é o valor da série temporal no tempo t.

c é uma constante.

ϕ1,ϕ2,...,ϕp são os coeficientes dos atrasos anteriores.

εt é o erro aleatório no tempo t.

θ1,θ2,...,θq são os coeficientes dos erros anteriores.

Método ARIMA:

- ARIMA - AutoRegressive Integrated Moving Average (Média Móvel Integrada AutoRegressiva)
- Extensão do método ARMA que inclui etapa de diferenciação para trabalhar com séries não estacionárias.
- Quando a série é não estacionária, permite que se diferencie, calculando a diferença entre observações anteriores da série. A ordem necessária para tornar a série estacionária é denominada '**d**'.
- Modelo ARIMA(p,d,q).

Aplicando o método ARIMA

Determinando parâmetros (p,d,q)

Diferenciação (d):

- Verificar a estacionariedade da série temporal aplicando testes, por ex.: Augmented Dickey-Fuller (ADF) ou o teste Kwiatkowski-Phillips-Schmidt-Shin (KPSS).

- Disponíveis na biblioteca "statsmodels"-PYTHON.

- Resulta em um p-valor, se $p <= 0.05$ a série é estacionária, se $p > 0.05$ a série não parece estacionária.

- Se ela não for estacionária, ou seja, não se desenvolver ao redor de uma média constante, aplicar diferenciação e testar novamente a estacionariedade.

- Cálculo disponível na biblioteca "numpy" -PYTHON.

EX.:

```
numpy.diff(arr[, n[, axis]])
Saída -> out[i] = arr[i+1] – arr[i]
```

Determinando parâmetros (p,d,q)

- O trabalho de estimar os parâmetros ‘p’ e ‘q’ é difícil e segue diferentes regras dependendo do manual ou documentação.
- Como a figura ao lado mostra, podem ser estimados visualmente, apesar de não muito preciso, utilizando os gráficos ACF e PACF que serão explicados abaixo.
- Outra possibilidade, é o cálculo de Critério de Informação, método mais preciso, mas que demanda tempo e poder computacional, que vai ser explicado na implementação da função iterativa ‘auto_arima’.

Rules to Interpret Non-Seasonal ACF and PACF Plots

	AR(p)	MA(q)	ARMA(p,q)
ACF	Tails off	Cuts off lag q	Tails off
PACF	Cuts off lag p	Tails off	Tails off

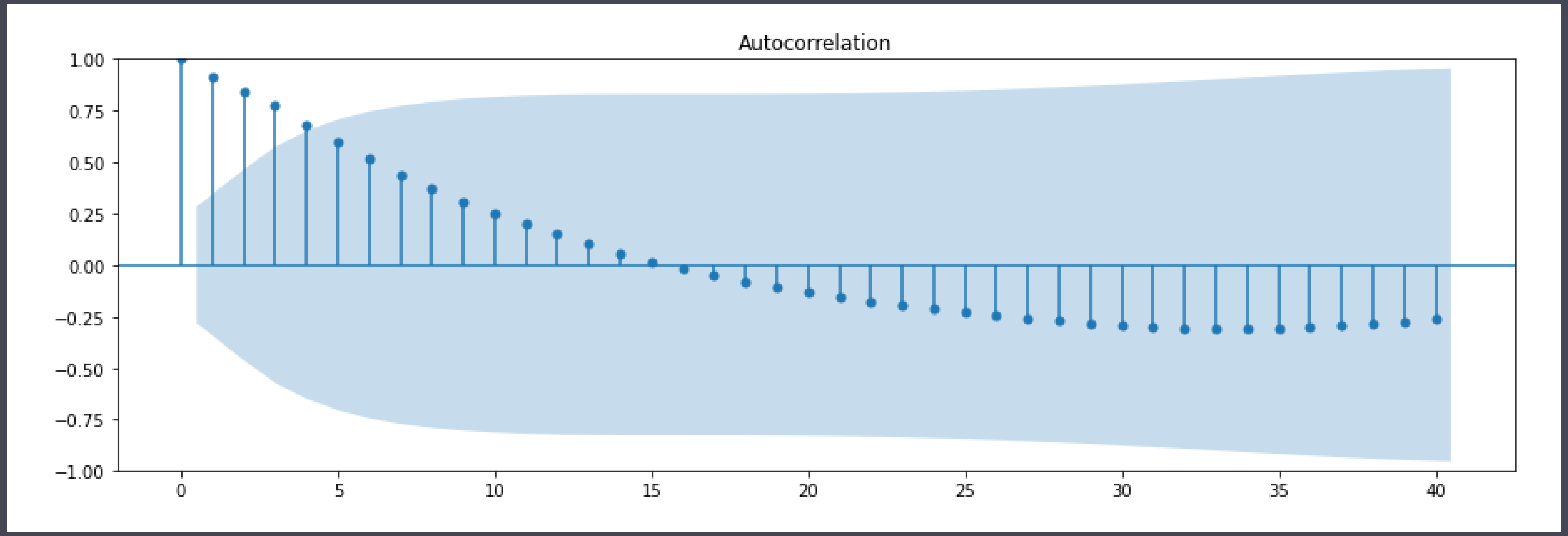
Determinando parâmetros (p,d,q)

Média Móvel (q):

- Verificar o gráfico de autocorrelação (ACF) para determinar o valor de ‘q’. O valor deve ser o lag onde a autocorrelação começa a cair para zero. Corresponde ao nº da média móvel no modelo.

- Disponíveis nas bibliotecas “statsmodels” ou “pandas” -PYTHON.

Exemplo de gráfico de autocorrelação:



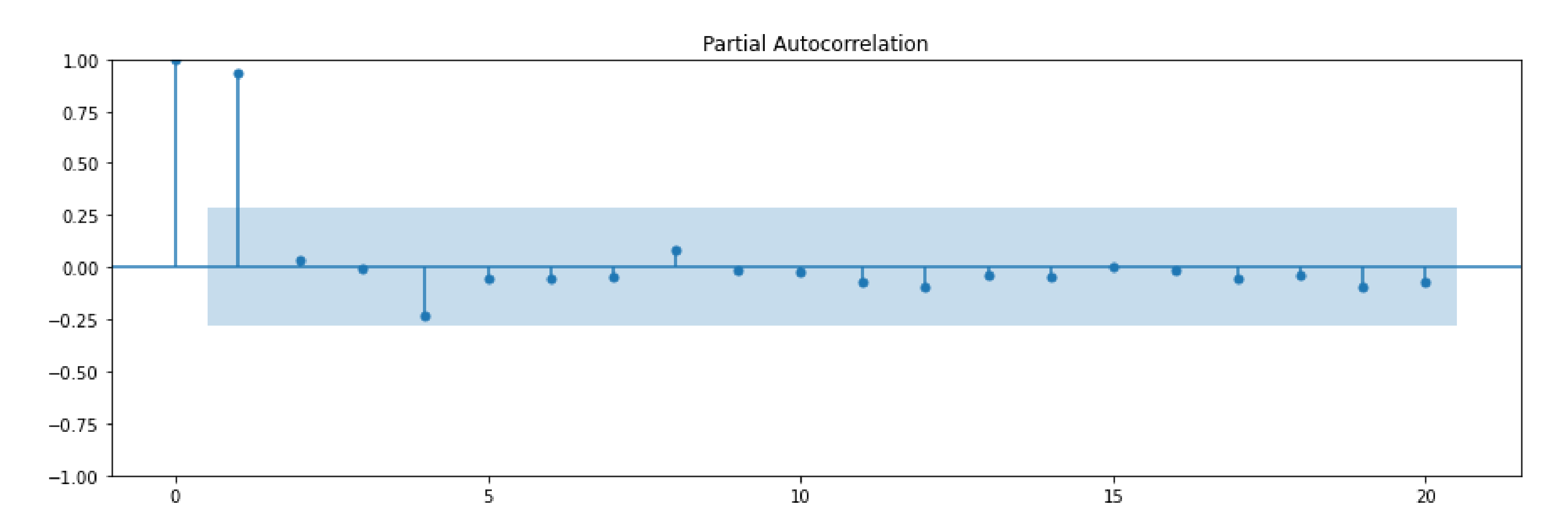
Determinando parâmetros (p,d,q)

Auto-regressão (p):

- Verificar o gráfico de autocorrelação parcial (PACF) para determinar o valor de 'p'. O valor deve ser o lag onde a autocorrelação parcial começa a cair para zero. Corresponde ao nº de termos autoregressivos no modelo.

- Disponíveis nas bibliotecas "statsmodels" ou "pandas" -PYTHON.

Exemplo de gráfico de autocorrelação parcial:



Determinando parâmetros (p,d,q)

ARIMA (p,d,q):

- Para determinar os parâmetros pode-se utilizar o Critério de Informação de Akaike (AIC) ou o Critério de Informação Bayesiana (BIC).

AIC = -2*log(L) +2*k

L - representa a verossimilhança do modelo (quão bem ele se ajusta aos dados).
k - representa o número de parâmetros no modelo. Quanto mais parâmetros, maior será o termo de penalização

- Modelos com menor valor de AIC são selecionados.

- Podem ser calculados automaticamente, utilizando a implementação do método ARIMA, pelo pacote “pmdarima” e a função “auto_arima” - PYTHON.

Avaliando o método

- Calcular o erro, ou resíduo do modelo, basicamente por:

RESÍDUO = VALOR REAL- VALOR PREVISTO

- Analisar o resíduo calculado - Comportamento como ruído branco: independente e de média zero.

- Avaliar pelo treino e teste a assertividade do modelo. Separa-se a amostra total em:

~80% para treino do modelo, ~20%para teste e valida-se o modelo verificando o erro nos dados de teste (dados que não foram "vistos" pelo modelo).

Avaliando o método

Algumas métrica utilizadas:

- "MFE" -MEAN FORECAST ERROR - (ERRO MÉDIO DA PREVISÃO/VIÉS)

Calcula a diferença entre os valores previstos e os valores observados para cada período e, em seguida, tira a média dessas diferenças. O resultado pode ser positivo ou negativo, dependendo se as previsões estão superestimando (valor positivo) ou subestimando (valor negativo) os valores reais.

  - MFE = Σ (Valor Observado - Valor Previsto) / N

- "MAE" - MEAN ABSOLUTE ERROR - (ERRO MÉDIO ABSOLUTO)

Calcula a diferença absoluta entre os valores previstos e os valores observados e tira a média dessas diferenças. Isso garante que todas as diferenças contribuam igualmente para a métrica, independentemente de serem superestimações ou subestimações.

  - MAE = Σ |Valor Observado - Valor Previsto| / N

- "MSE" - MEAN SQUARED ERROR - (ERRO QUADRÁTICO MÉDIO)

Calcula a diferença quadrática entre os valores previstos e os valores observados e tira a média dessas diferenças. Como resultado, o MSE atribui maior peso a erros maiores, tornando-o mais sensível a discrepâncias significativas entre as previsões e os valores reais.

  - MSE = Σ (Valor Observado - Valor Previsto)^2 / N

Avaliando o método

Algumas métrica utilizadas:

- "RMSE" - ROOT MEAN SQUARED ERROR - (ERRO QUADRÁTICO MÉDIO DA RAIZ)

EO RMSE é calculado tomando a raiz quadrada do Mean Squared Error (MSE), que é a média das diferenças quadráticas entre os valores previstos e os valores observados. Ao tomar a raiz quadrada, o RMSE retorna uma métrica que está na mesma unidade que os dados originais, tornando-a mais facilmente interpretável do que o MSE.

  - RMSE = $\sqrt{}$(Σ (Valor Observado - Valor Previsto)^2 / N)

- "MAPE" - MEAN ABSOLUTE PERCENTAGE ERROR - (ERRO PERCENTUAL MÉDIO ABSOLUTO)

O MAPE calcula a diferença percentual média entre os valores previstos e os valores observados, levando em consideração a magnitude dos valores reais. Em essência, o MAPE mede a precisão relativa das previsões em termos de porcentagem do erro em relação aos valores reais.

  - MAPE = (1 / N) * Σ (|Valor Observado - Valor Previsto| / |Valor Observado|) * 100%

Avaliando o método

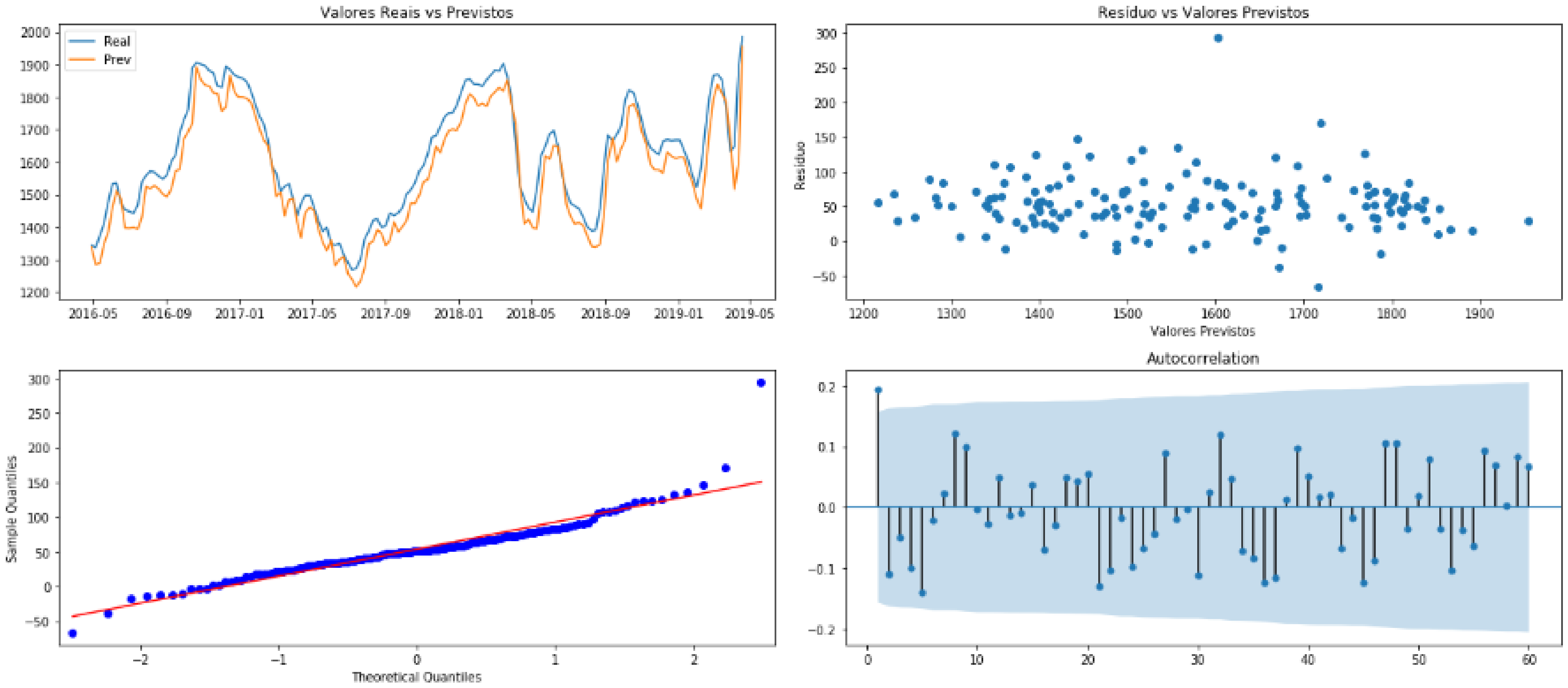
Algumas métrica utilizadas: Exemplo de código pela biblioteca sklearn

- from sklearn.metrics import mean_squared_error, mean_absolute_error, mean_absolute_error

```
def check_erro(orig, prev, nome_col='', nome_indice=''):
    vies = np.mean(orig - prev)
    mse = mean_squared_error(orig, prev)
    rmse = sqrt(mean_squared_error(orig, prev))
    mae = mean_absolute_error(orig, prev)
    mape = np.mean(np.abs((orig - prev) / orig)) * 100

grupo_erro = [vies, mse, rmse, mae, mape]
serie = pd.DataFrame(grupo_erro, index=['VIÉS','MSE','RMSE','MAE', 'MAPE'], columns=[nome_col])
serie.index.name = nome_indice
return serie
```

MODELO AUTO REGRESSIVO NA BASE DE TESTE



Base Treino	Simples	Média Simples	Média Móvel 5D	Média Móvel Exp.	Auto Regr.
VIÉS	-0.366072	20.948470	-0.581179	-0.540009	53.897706
MSE	3101.487631	72120.810997	6422.525818	6364.944623	4509.416686
RMSE	55.691001	268.553181	80.140663	79.780603	67.152190
MAE	35.180332	207.147333	54.574449	53.450252	56.164377
MAPE	2.578871	15.498293	4.004098	3.917916	3.466456

Disponível no Github/leandrovrabelo
Link do repositório:
https://github.com/leandrovrabelo/tsmodels/blob/master/notebooks/portugues/Princípios%20Básicos%20para%20Prever%20Séries%20Temporais.ipynb

Referências

Algumas das referências usadas:

- MORETTIN, Pedro A.; TOLOI, Clélia MC. Análise de séries temporais: modelos lineares univariados. Editora Blucher, 2018.

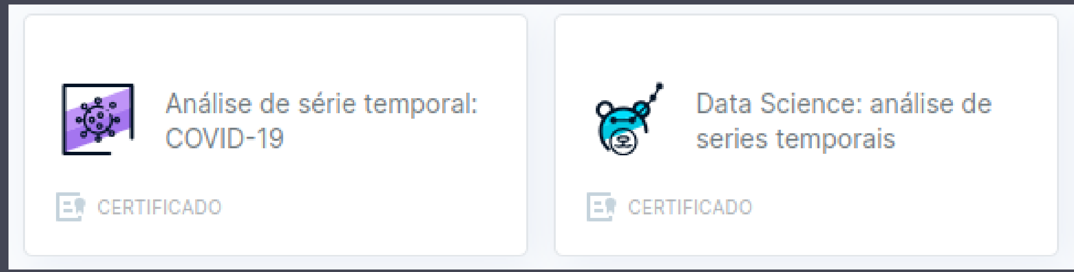
Github:

- https://github.com/leandrovrabelo/tsmodels/blob/master/notebooks/portugues/Princípios%20Básicos%20para%20Prever%20Séries%20Temporais.ipynb

- https://github.com/AirtonLira/artigo_series_arima.git

Alura:

- https://github.com/alura-cursos/COVID-Alura.git



Muito obrigado!

juh.fischer@gmail.com