



DengAI: prevendo a propagação da doença



Daniel Vargas Shimamoto
Diego Machado de Assis



DengAI: prevendo a propagação da doença

Daniel Vargas Shimamoto
Diego Machado de Assis

NASA Earth Observatory map by Lauren Dauphin based on data from Janey Messina, University of Oxford

Motivação

Grave problema epidêmico

Doença tropical e subtropical

400 milhões de casos por ano

1 milhão de casos no Brasil em 2020

Dinâmica da transmissão

Mosquito *aedes aegypti* como vetor

Variáveis climáticas como temperatura e precipitação

Relação entre clima e potencial da doença



Conjunto de dados

DrivenData

Plataforma de **competições** de Ciência de Dados

- 10.000+ competidores
- *Just for fun*

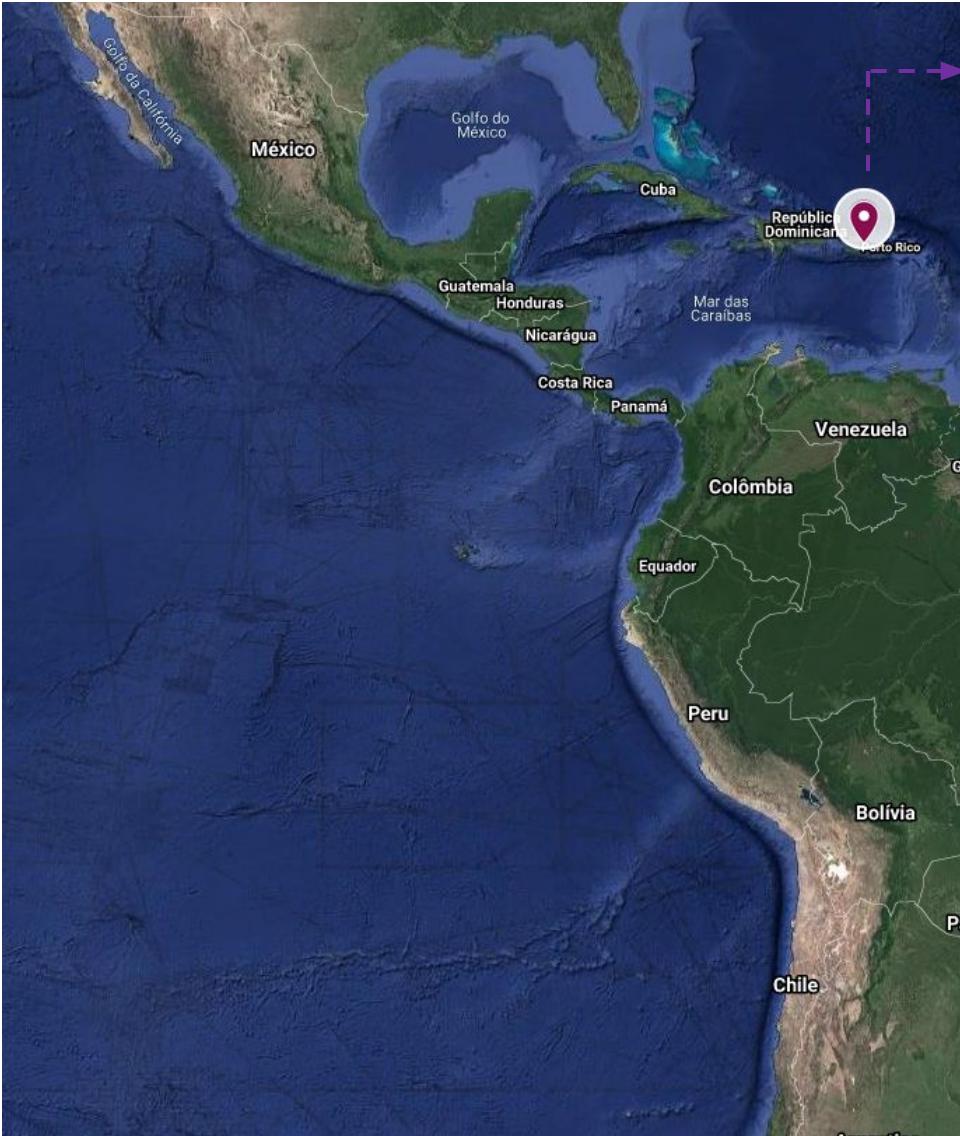
The screenshot shows the homepage of the DrivenData website. At the top, there is a navigation bar with links for COMPETITIONS, ABOUT, CAREERS, DRIVENDATA+, BLOG, MY PROFILE, and LOG OUT. Below the navigation, a large, colorful radial bar chart is displayed, radiating from a central white circle through various colors (yellow, orange, red, purple, blue, green) to represent data science competitions.

In the center of the page, the text "Data science competitions to build a better world" is displayed, along with two buttons: "I want to join a competition" and "I want to run a competition".

Below this, a section titled "Data Science + Social Impact" explains DrivenData's mission: "DrivenData works on projects at the intersection of data science and social impact, in areas like international development, health, education, research and conservation, and public services. We want to give more organizations access to the capabilities of data science, and engage more data scientists with social challenges where their skills can make a difference." It also mentions that they have worked with over 35 organizations across 100+ projects.

At the bottom of the main content area, there are three small images with captions: "PUBLIC HEALTH" (a map), "CONSERVATION" (a chimpanzee), and "DEVELOPMENT" (people in a community setting).

Localizando o problema



San Juan (Porto Rico)

Capital, maior e mais populoso município de Porto Rico

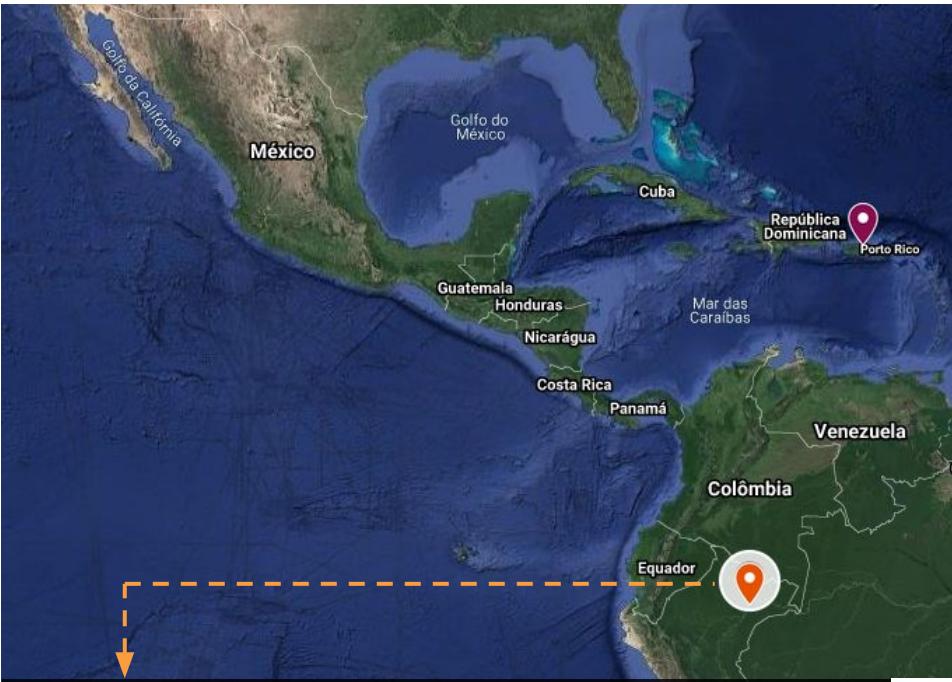
Área total: 199,25 km²

Área urbana: 120,15 km²

População total (2010): 442.447 hab.



Localizando o problema



Iquitos (Peru)

Capital da Amazônia Peruana

Área total: 369 km²
População total (2015): 465.817 hab.



San Juan (Porto Rico)

Capital, maior e mais populoso município de Porto Rico

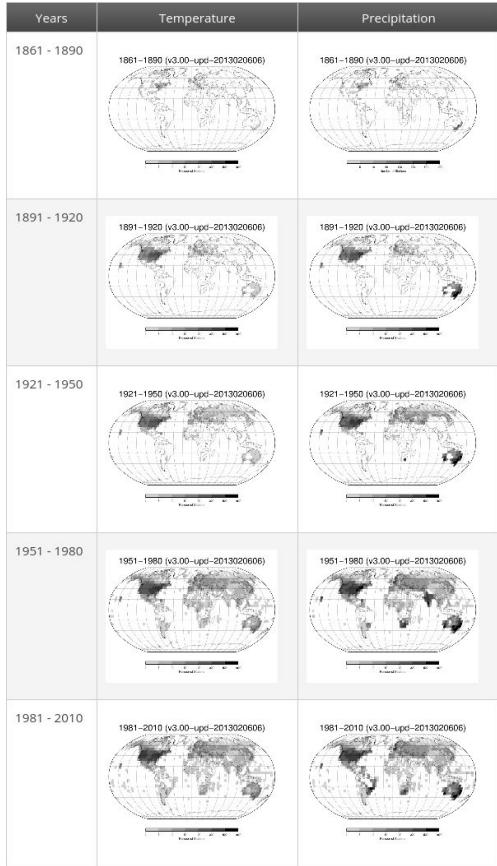
Área total: 199,25 km²
Área urbana: 120,15 km²
População total (2010): 442.447 hab.



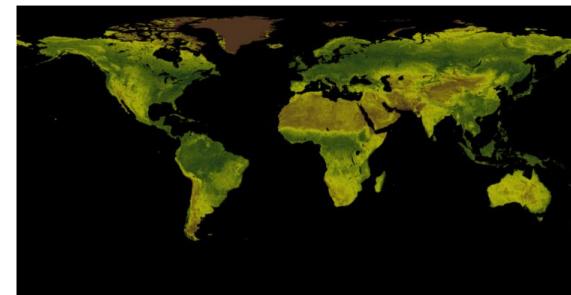
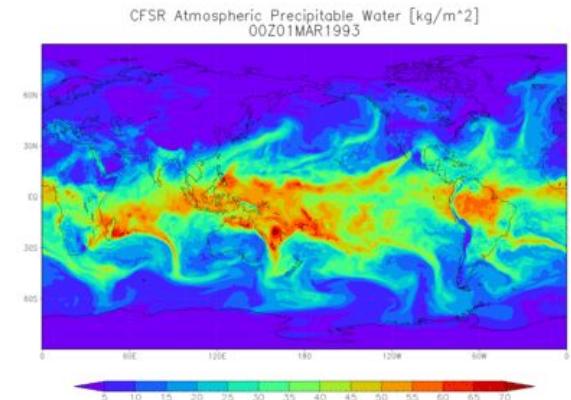
Origem dos dados



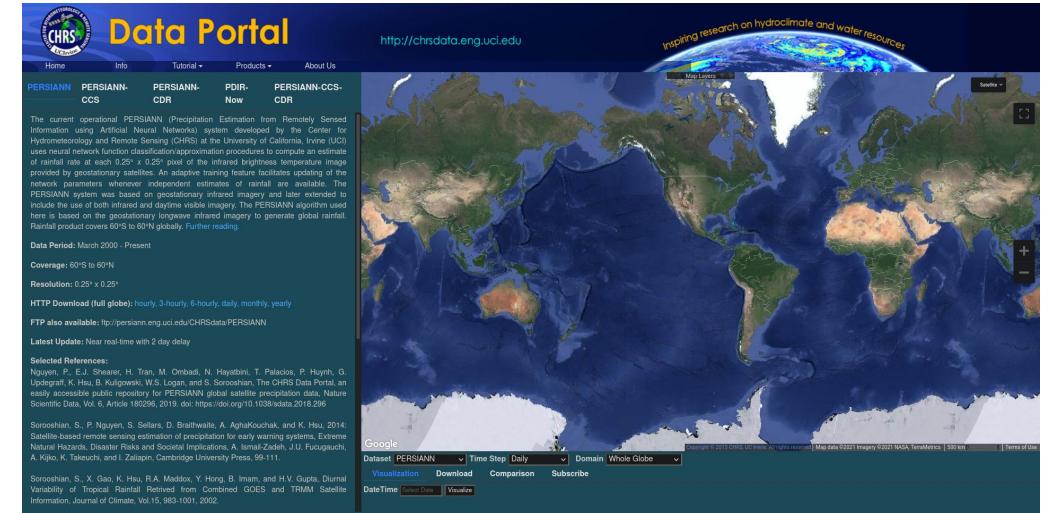
NOAA NATIONAL CENTERS FOR ENVIRONMENTAL INFORMATION
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION



Global Historical Climate Network
(GHCN-Daily)



Climate Data Record (CDR) of Normalized Differential Vegetation Index (NDVI)



PERSIANN
Precipitation Estimation from Remotely Sensed Information using Artificial Neural Networks

DengAI



Atributos

Conjunto de variáveis independentes

	city	year	weekofyear	week_start_date	ndvi_ne	ndvi_nw	...	reanalysis_tdtr_k	station_avg_temp_c	station_diur_temp_rng_c	station_max_temp_c	station_min_temp_c	station_precip_mm
0	sj	1990	18	1990-04-30	0.122600	0.103725	...	2.628571	25.442857	6.900000	29.4	20.0	16.0
1	sj	1990	19	1990-05-07	0.169900	0.142175	...	2.371429	26.714286	6.371429	31.7	22.2	8.6
2	sj	1990	20	1990-05-14	0.032250	0.172967	...	2.300000	26.714286	6.485714	32.2	22.8	41.4
3	sj	1990	21	1990-05-21	0.128633	0.245067	...	2.428571	27.471429	6.771429	33.3	23.3	4.0
4	sj	1990	22	1990-05-28	0.196200	0.262200	...	3.014286	28.942857	9.371429	35.0	23.9	5.8
...
1451	iq	2010	21	2010-05-28	0.342750	0.318900	...	9.800000	28.633333	11.933333	35.4	22.4	27.0
1452	iq	2010	22	2010-06-04	0.160157	0.160371	...	7.471429	27.433333	10.500000	34.7	21.7	36.6
1453	iq	2010	23	2010-06-11	0.247057	0.146057	...	7.500000	24.400000	6.900000	32.2	19.2	7.4
1454	iq	2010	24	2010-06-18	0.333914	0.245771	...	7.871429	25.433333	8.733333	31.2	21.0	16.0
1455	iq	2010	25	2010-06-25	0.298186	0.232971	...	11.014286	27.475000	9.900000	33.7	22.2	20.4

1456 rows × 24 columns

Atributos

Conjunto de variáveis independentes

CHAVES

	city	year	weekofyear	week_start_date	ndvi_ne	ndvi_nw	...	reanalysis_tdtr_k	station_avg_temp_c	station_diur_temp_rng_c	station_max_temp_c	station_min_temp_c	station_precip_mm
0	sj	1990	18	1990-04-30	0.122600	0.103725	...	2.628571	25.442857	6.900000	29.4	20.0	16.0
1	sj	1990	19	1990-05-07	0.169900	0.142175	...	2.371429	26.714286	6.371429	31.7	22.2	8.6
2	sj	1990	20	1990-05-14	0.032250	0.172967	...	2.300000	26.714286	6.485714	32.2	22.8	41.4
3	sj	1990	21	1990-05-21	0.128633	0.245067	...	2.428571	27.471429	6.771429	33.3	23.3	4.0
4	sj	1990	22	1990-05-28	0.196200	0.262200	...	3.014286	28.942857	9.371429	35.0	23.9	5.8
...
1451	iq	2010	21	2010-05-28	0.342750	0.318900	...	9.800000	28.633333	11.933333	35.4	22.4	27.0
1452	iq	2010	22	2010-06-04	0.160157	0.160371	...	7.471429	27.433333	10.500000	34.7	21.7	36.6
1453	iq	2010	23	2010-06-11	0.247057	0.146057	...	7.500000	24.400000	6.900000	32.2	19.2	7.4
1454	iq	2010	24	2010-06-18	0.333914	0.245771	...	7.871429	25.433333	8.733333	31.2	21.0	16.0
1455	iq	2010	25	2010-06-25	0.298186	0.232971	...	11.014286	27.475000	9.900000	33.7	22.2	20.4

1456 rows × 24 columns



Atributos

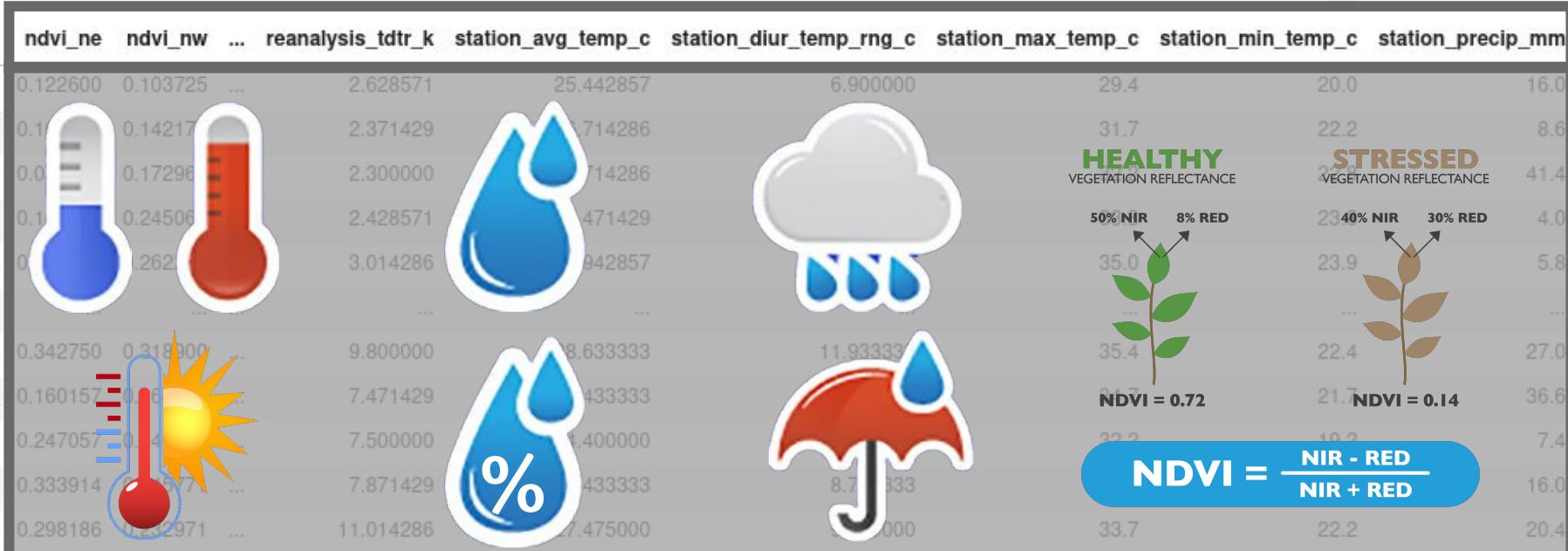
Conjunto de variáveis independentes

CHAVES

	city	year	weekofyear	week_start_date	ndvi_ne	ndvi_nw	...	reanalysis_tdtr_k	station_avg_temp_c	station_diur_temp_rng_c	station_max_temp_c	station_min_temp_c	station_precip_mm
0	sj	1990	18	1990-04-30	0.122600	0.103725	...	2.628571	25.442857	6.900000	29.4	20.0	16.0
1	sj	1990	19	1990-05-07	0.103725	0.142171	...	2.371429	27.714286	6.900000	31.7	22.2	8.6
2	sj	1990	20	1990-05-14	0.0	0.172961	...	2.300000	27.714286	6.900000	31.7	22.2	8.6
3	sj	1990	21	1990-05-21	0.1	0.245061	...	2.428571	27.714286	6.900000	31.7	22.2	8.6
4	sj	1990	22	1990-05-28	0.1	0.262121	...	3.014286	27.942857	6.900000	31.7	22.2	8.6
...
1451	iq	2010	21	2010-05-28	0.342750	0.318000	...	9.800000	28.633333	11.933333	35.4	22.4	27.0
1452	iq	2010	22	2010-06-04	0.160157	0.618000	...	7.471429	28.633333	11.933333	35.4	22.4	27.0
1453	iq	2010	23	2010-06-11	0.247057	0.418000	...	7.500000	28.400000	11.933333	35.4	22.4	27.0
1454	iq	2010	24	2010-06-18	0.333914	0.218000	...	7.871429	28.433333	8.733333	35.4	22.4	27.0
1455	iq	2010	25	2010-06-25	0.298186	0.2971	...	11.014286	27.475000	8.733333	33.7	22.2	20.4

1456 rows × 24 columns

CARACTERÍSTICAS



Rótulos

Variável dependente (alvo)

	city	year	weekofyear	total_cases
0	sj	1990	18	4
1	sj	1990	19	5
2	sj	1990	20	4
3	sj	1990	21	3
4	sj	1990	22	6
...
1451	iq	2010	21	5
1452	iq	2010	22	8
1453	iq	2010	23	1
1454	iq	2010	24	1
1455	iq	2010	25	4

1456 rows × 4 columns

Rótulos

Variável dependente (alvo)

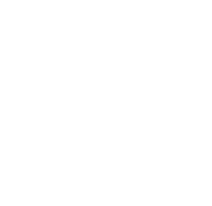
CHAVES

	city	year	weekofyear	total_cases
0	sj	1990	18	4
1	sj	1990	19	5
2	sj	1990	20	4
3	sj	1990	21	3
4	sj	1990	22	6
...
1451	iq	2010	21	5
1452	iq	2010	22	8
1453	iq	2010	23	1
1454	iq	2010	24	1
1455	iq	2010	25	4

1456 rows × 4 columns

Rótulos

Variável dependente (alvo)

	CHAVES	RÓTULOS
	city year weekofyear	total_cases
0	sj 1990 18	
1	sj 1990 19	
2	sj 1990 20	
3	sj 1990 21	
4	sj 1990 22	
...
1451	iq 2010 21	
1452	iq 2010 22	
1453	iq 2010 23	
1454	iq 2010 24	
1455	iq 2010 25	

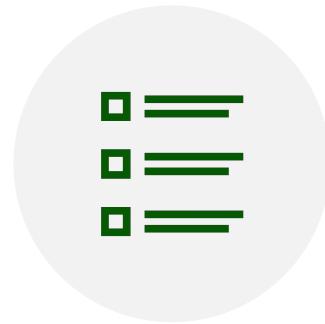
1456 rows × 4 columns

Pré-processamento



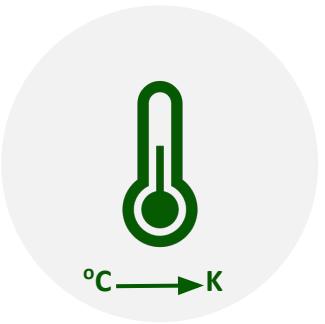
Tipos de Dados

Conversão do atributo week_start_date em data (`pd.to_datetime`)



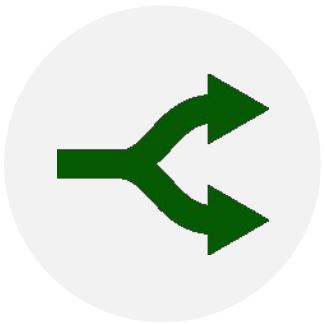
Definição dos Índices

Reindexação utilizando a chave única do registro (`city, year, weekofyear`)



Unidades de Medida

Padronização das temperaturas para Kelvin ($K = {}^\circ C - 273$)



Divisão registros de cada cidade

SAN JUAN
Atributos: (936, 21)
Rótulos: (936, 1)

IQUITOS
Atributos: (520, 21)
Rótulos: (520, 1)



Valores Nulos

Tratamento dos dados faltantes

Pré-processamento: Valores nulos

Valores faltantes para San Juan

```
sj_features.isna().sum()
```

week_start_date	0
ndvi_ne	191
ndvi_nw	49
ndvi_se	19
ndvi_sw	19
precipitation_amt_mm	9
reanalysis_air_temp_k	6
reanalysis_avg_temp_k	6
reanalysis_dew_point_temp_k	6
reanalysis_max_air_temp_k	6
reanalysis_min_air_temp_k	6
reanalysis_precip_amt_kg_per_m2	6
reanalysis_relative_humidity_percent	6
reanalysis_sat_precip_amt_mm	9
reanalysis_specific_humidity_g_per_kg	6
reanalysis_tdtr_k	6
station_diur_temp_rng_c	6
station_precip_mm	6
station_max_temp_k	6
station_min_temp_k	6
station_avg_temp_k	6

Valores faltantes para Iquitos

```
iq_features.isna().sum()
```

week_start_date	0
ndvi_ne	3
ndvi_nw	3
ndvi_se	3
ndvi_sw	3
precipitation_amt_mm	4
reanalysis_air_temp_k	4
reanalysis_avg_temp_k	4
reanalysis_dew_point_temp_k	4
reanalysis_max_air_temp_k	4
reanalysis_min_air_temp_k	4
reanalysis_precip_amt_kg_per_m2	4
reanalysis_relative_humidity_percent	4
reanalysis_sat_precip_amt_mm	4
reanalysis_specific_humidity_g_per_kg	4
reanalysis_tdtr_k	4
station_diur_temp_rng_c	37
station_precip_mm	16
station_max_temp_k	14
station_min_temp_k	8
station_avg_temp_k	37

Pré-processamento: Valores nulos

Valores faltantes para San Juan

```
sj_features.isna().sum()
```

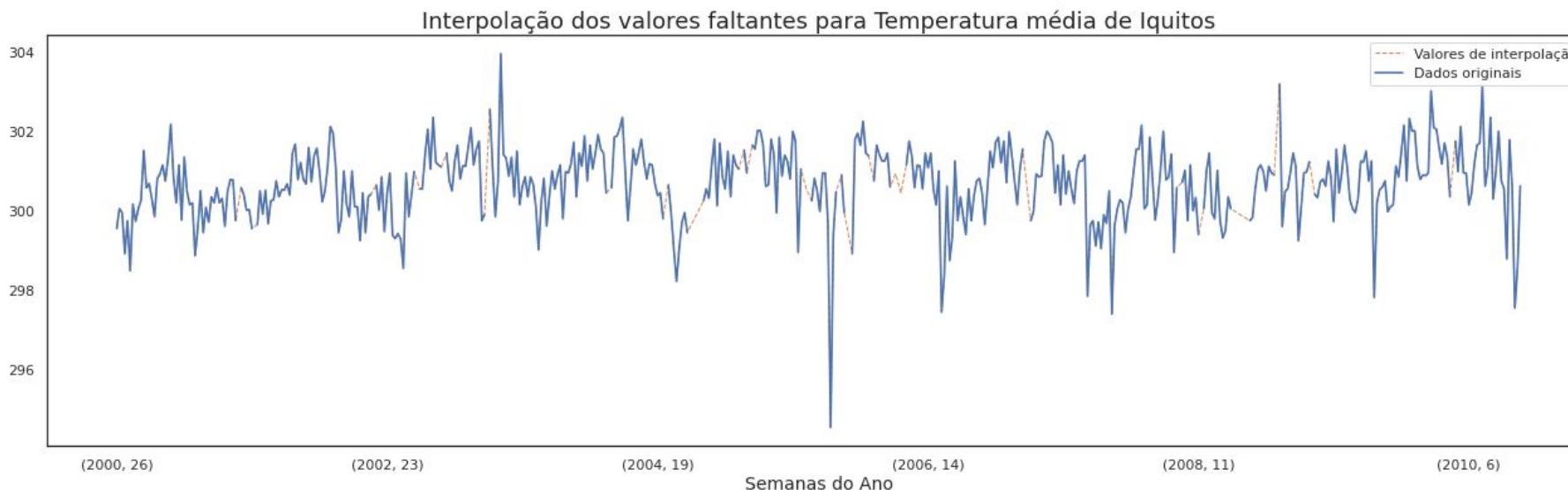
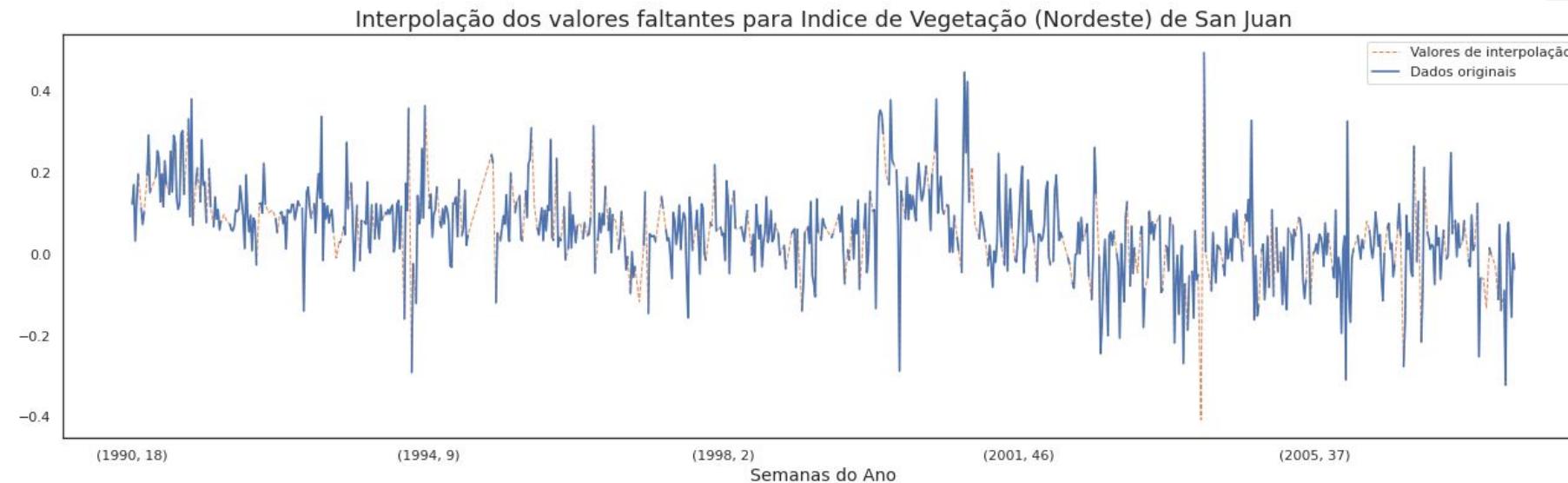
week_start_date	0
ndvi_ne	191
ndvi_nw	49
ndvi_se	19
ndvi_sw	19
precipitation_amt_mm	9
reanalysis_air_temp_k	6
reanalysis_avg_temp_k	6
reanalysis_dew_point_temp_k	6
reanalysis_max_air_temp_k	6
reanalysis_min_air_temp_k	6
reanalysis_precip_amt_kg_per_m2	6
reanalysis_relative_humidity_percent	6
reanalysis_sat_precip_amt_mm	9
reanalysis_specific_humidity_g_per_kg	6
reanalysis_tdtr_k	6
station_diur_temp_rng_c	6
station_precip_mm	6
station_max_temp_k	6
station_min_temp_k	6
station_avg_temp_k	6

Valores faltantes para Iquitos

```
iq_features.isna().sum()
```

week_start_date	0
ndvi_ne	3
ndvi_nw	3
ndvi_se	3
ndvi_sw	3
precipitation_amt_mm	4
reanalysis_air_temp_k	4
reanalysis_avg_temp_k	4
reanalysis_dew_point_temp_k	4
reanalysis_max_air_temp_k	4
reanalysis_min_air_temp_k	4
reanalysis_precip_amt_kg_per_m2	4
reanalysis_relative_humidity_percent	4
reanalysis_sat_precip_amt_mm	4
reanalysis_specific_humidity_g_per_kg	4
reanalysis_tdtr_k	4
station_diur_temp_rng_c	37
station_precip_mm	16
station_max_temp_k	14
station_min_temp_k	8
station_avg_temp_k	37

Pré-processamento: Valores nulos

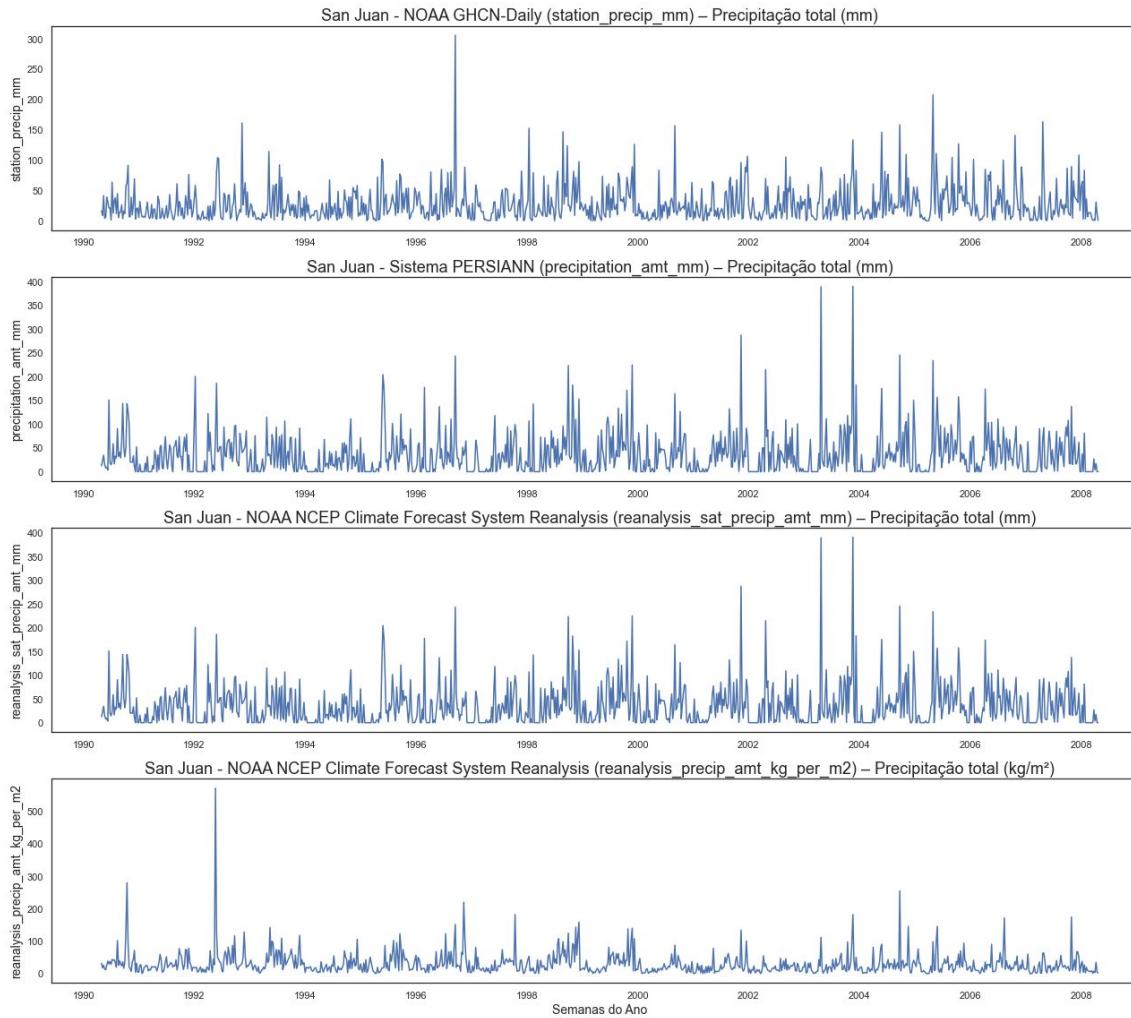


Análise Exploratória



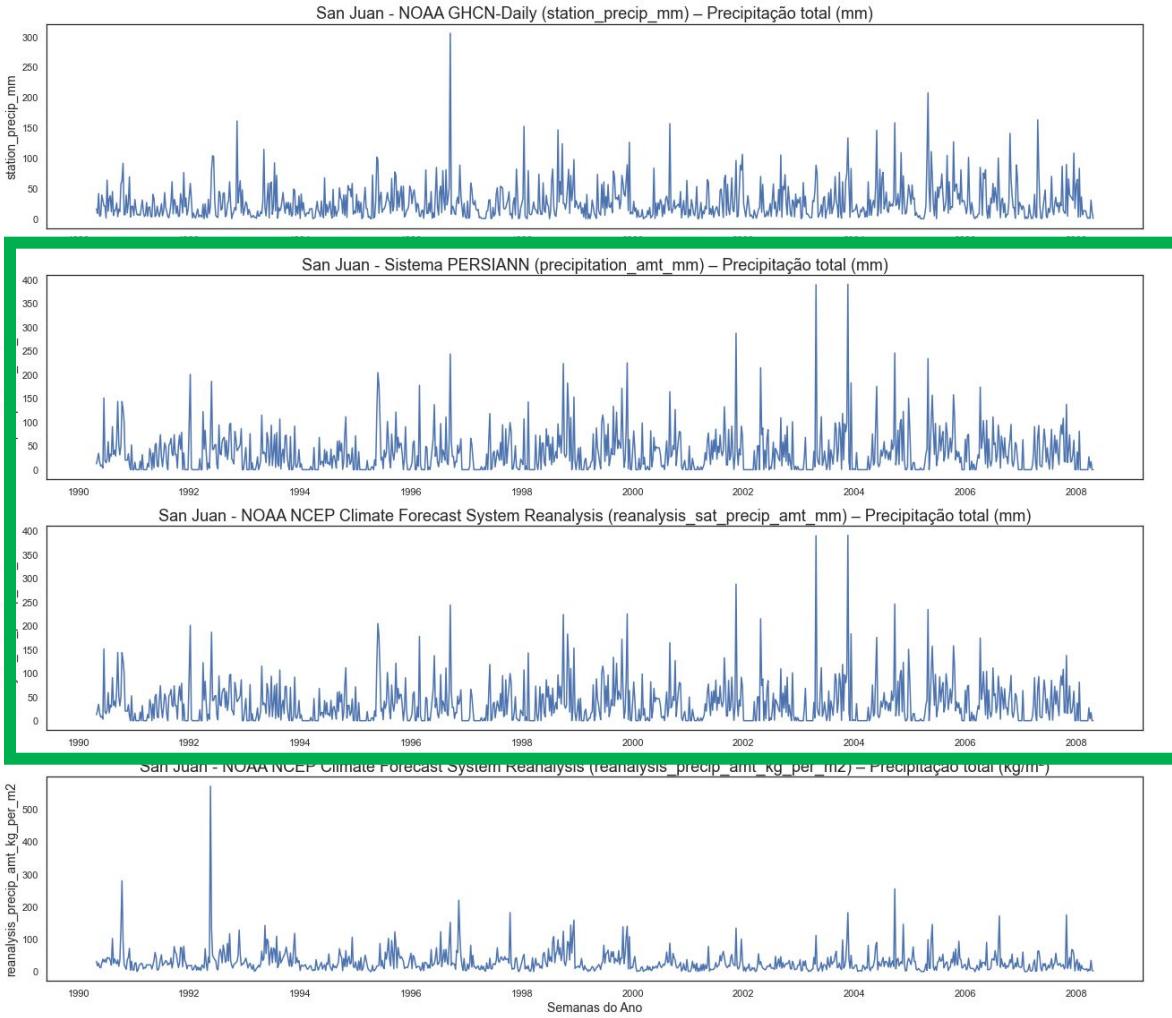
Taxa de Precipitação

4 atributos de precipitação



Taxa de Precipitação

4 atributos de precipitação

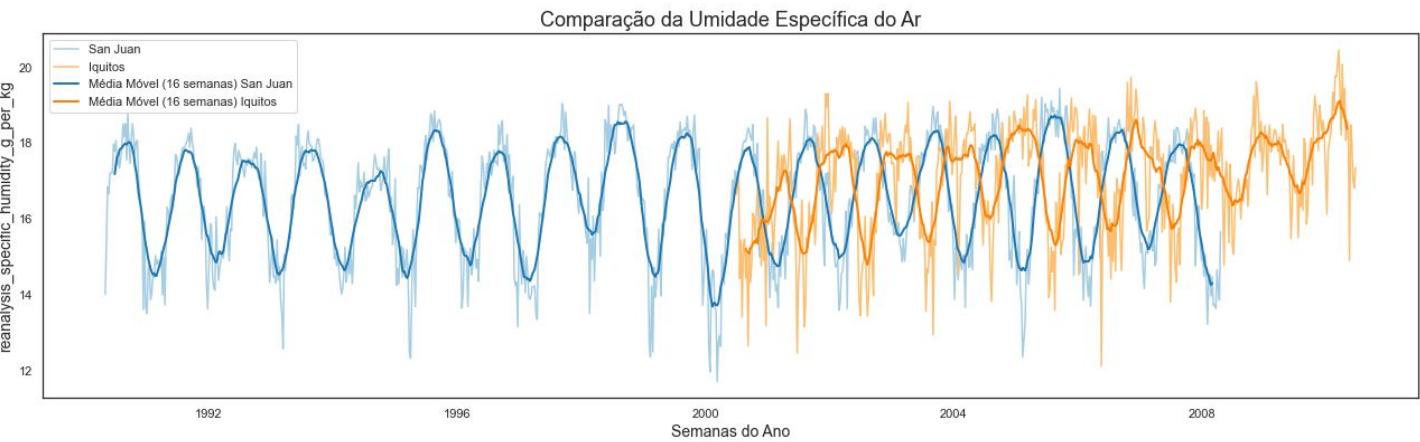
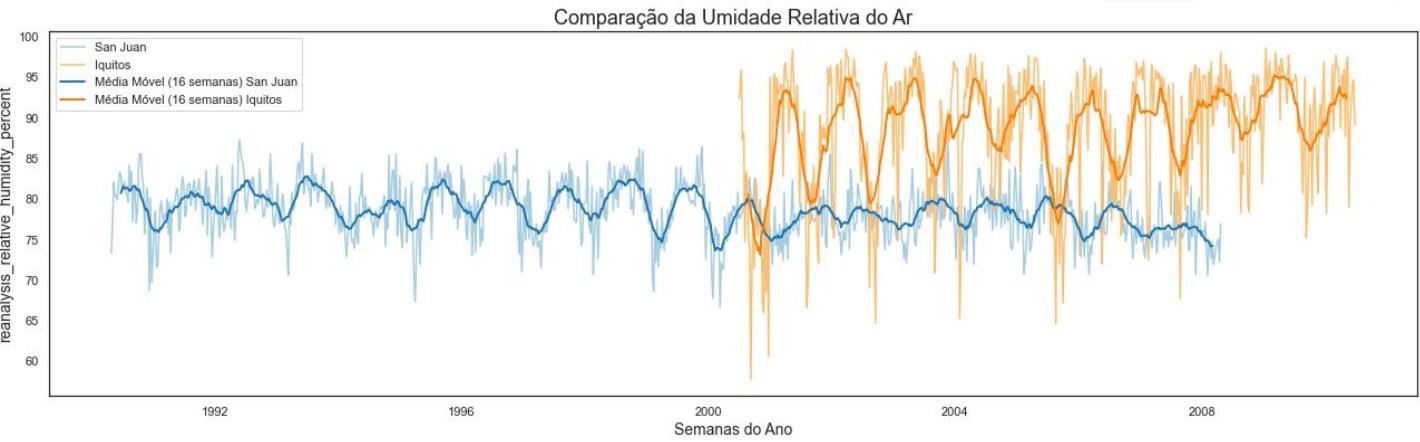


PERSIANN = NOAA NCEP CFSR

Umidade

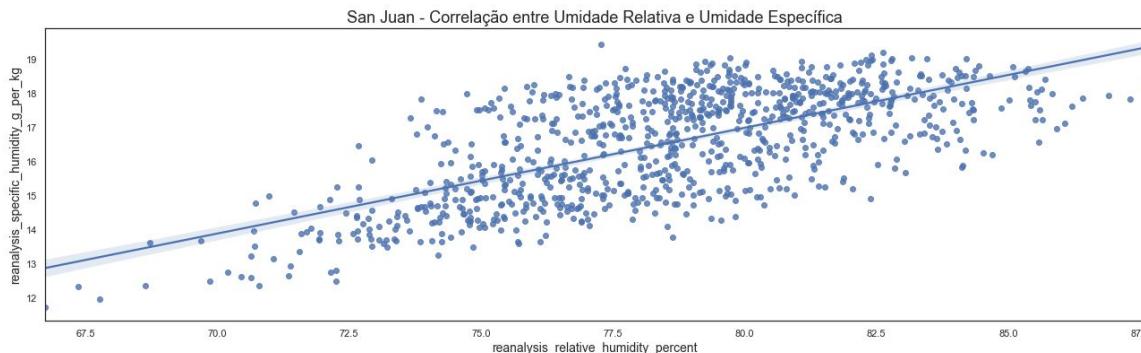
Umidade Relativa e Específica
para San Juan e Iquitos

Média móvel 16 semanas

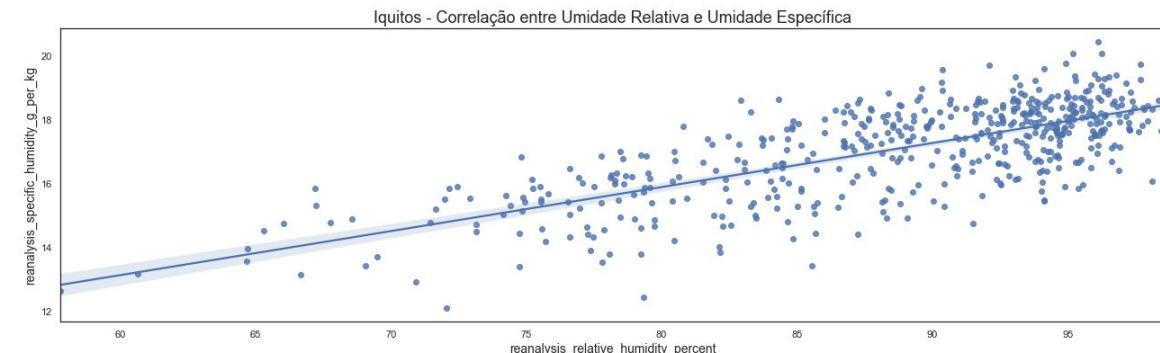


Umidade: correlação

Coeficiente de Correlação de Pearson: Umidade Relativa e Umidade Específica



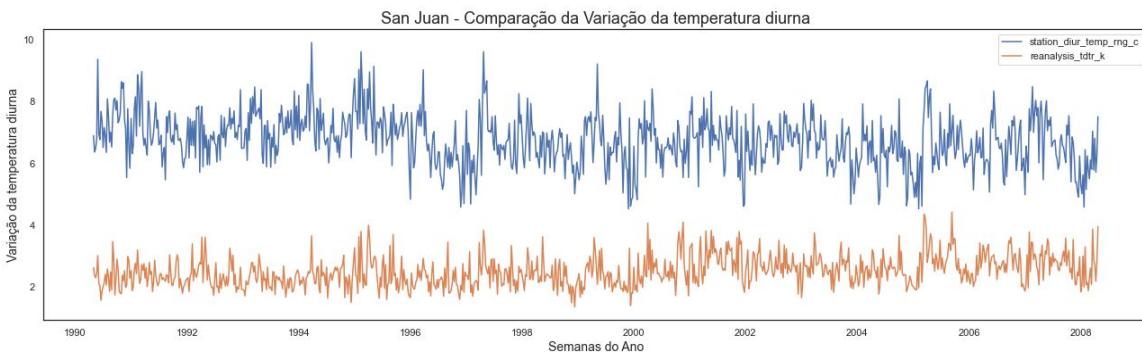
San Juan: $r = 0,67$



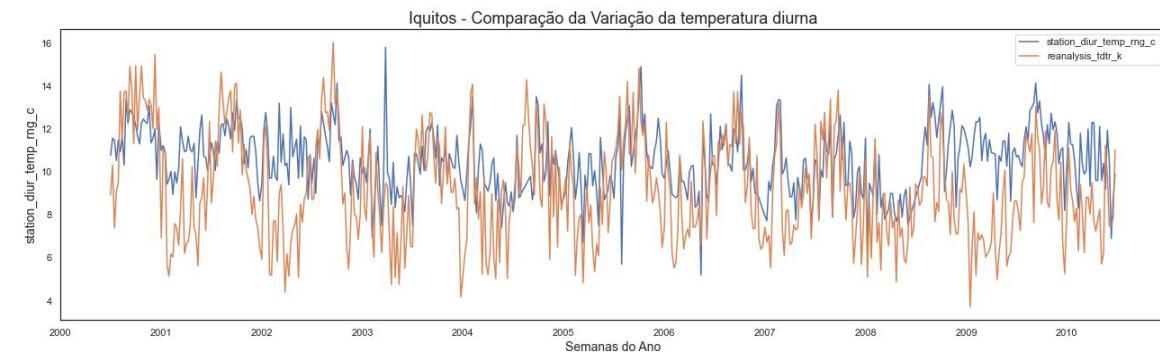
Iquitos: $r = 0,72$

Temperatura: variação diurna

Comparação **GHCN-Daily** e **NCEP CFSR**



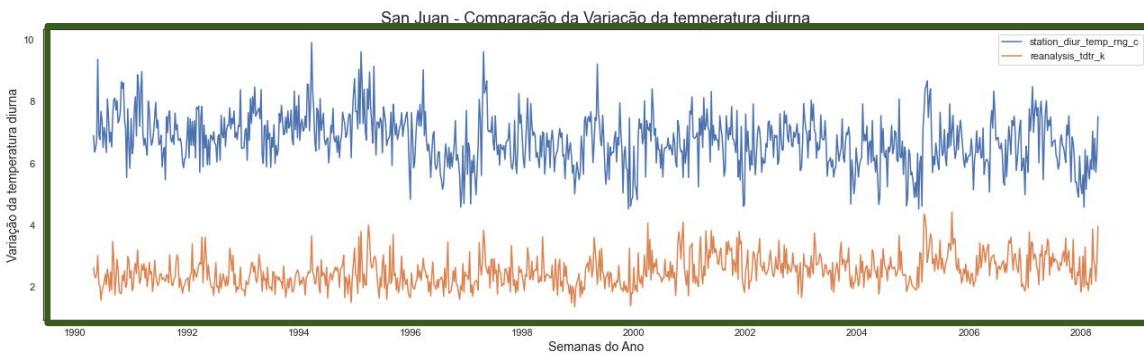
San Juan



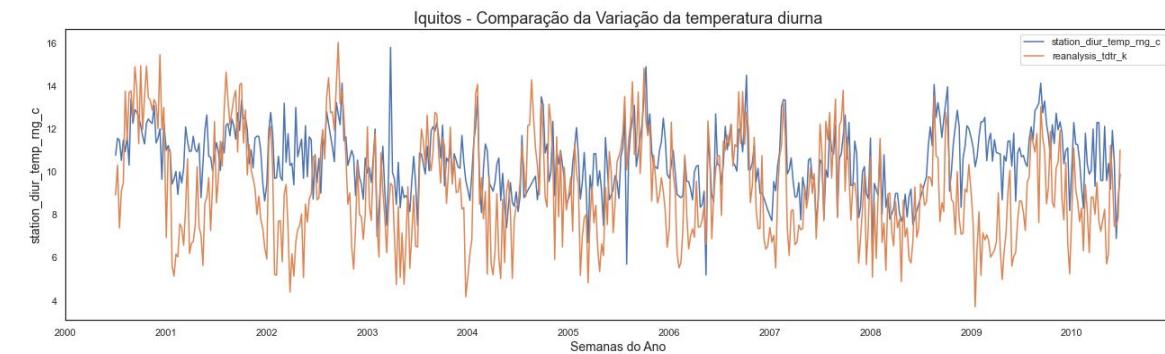
Iquitos

Temperatura: variação diurna

Comparação **GHCN-Daily** e **NCEP CFSR**



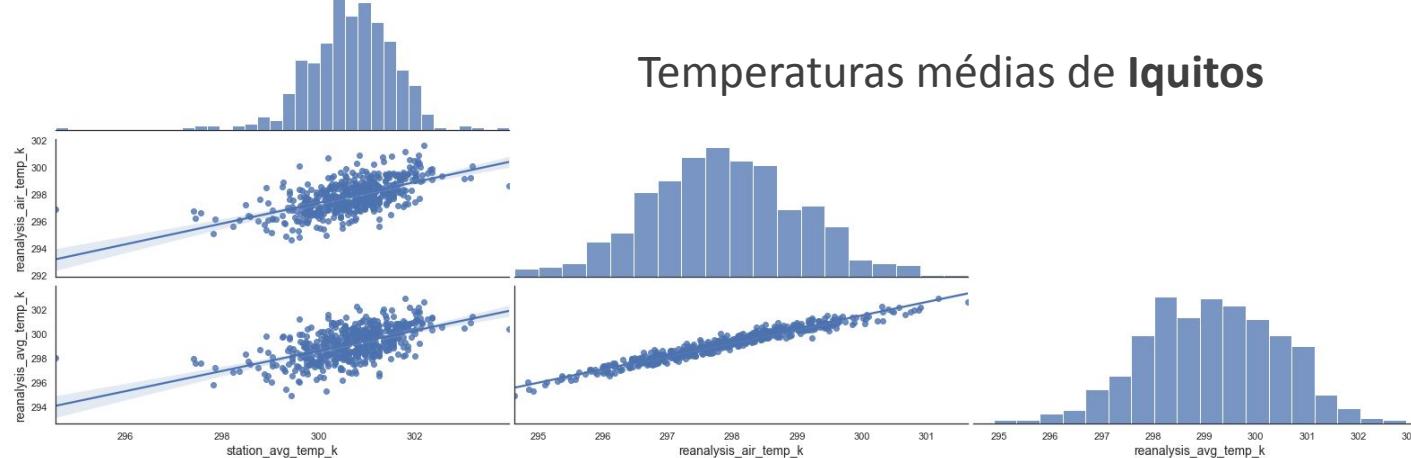
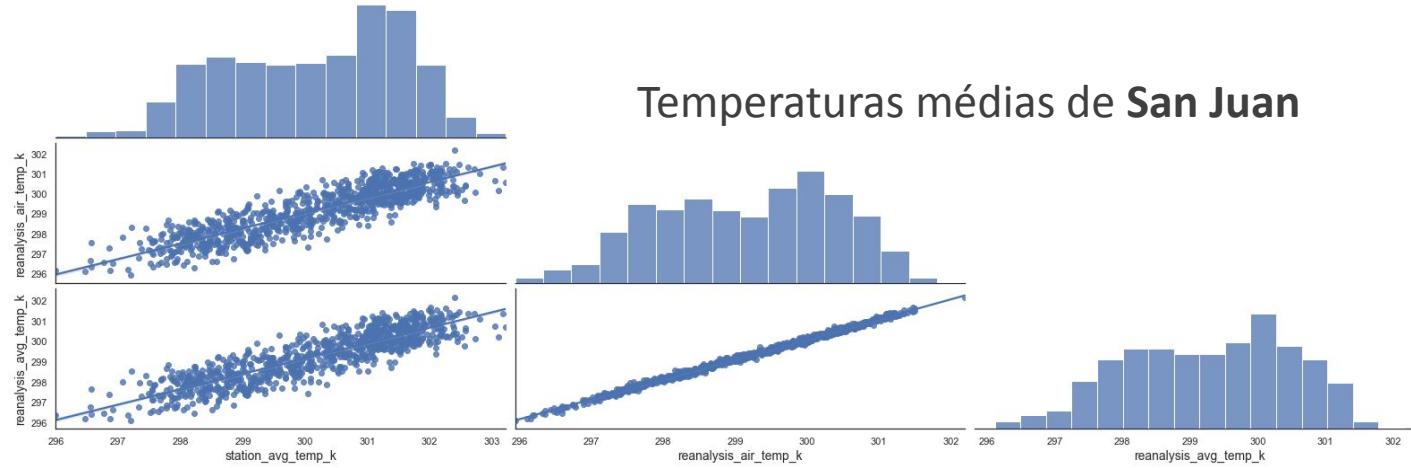
San Juan



Iquitos

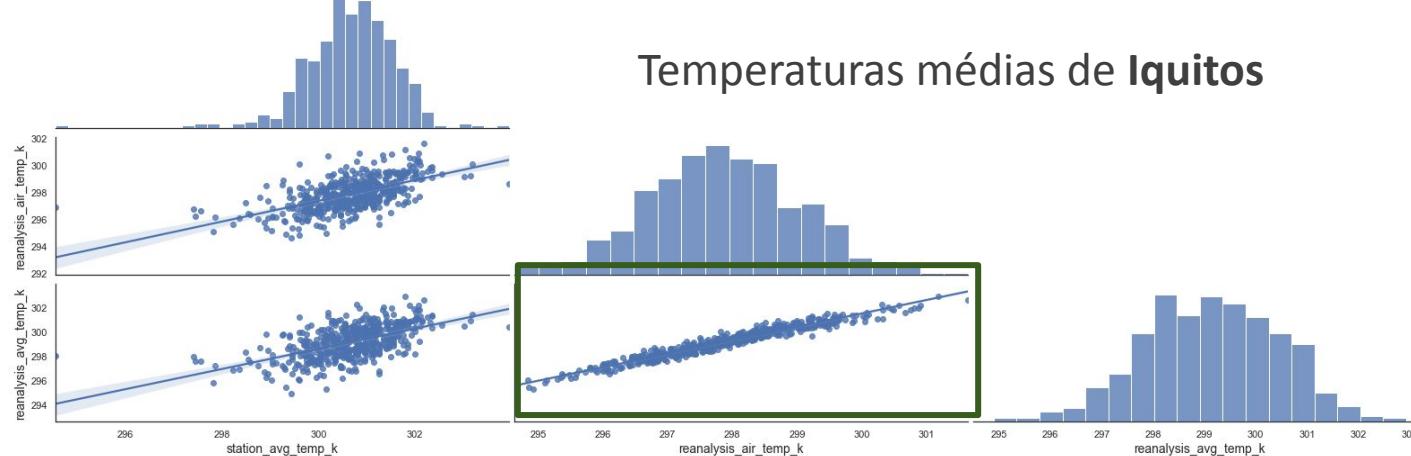
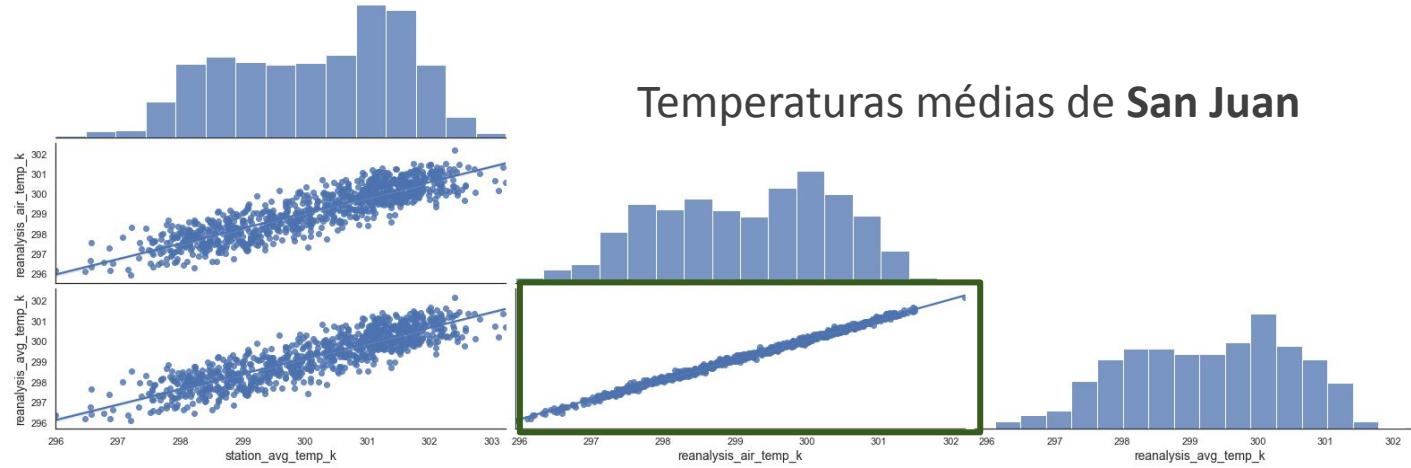
Temperatura: médias

Dados do **GHCN-Daily** e **NCEP CFSR**



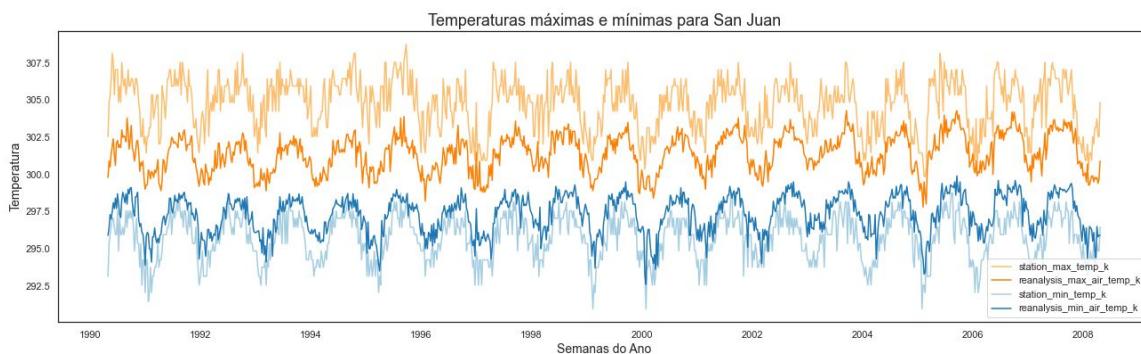
Temperatura: médias

Dados do **GHCN-Daily** e **NCEP CFSR**

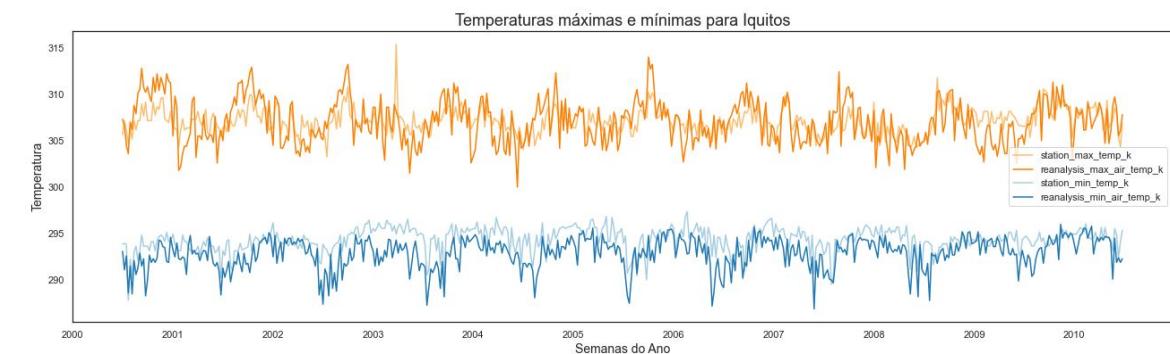


Temperatura: máximas e mínimas

Temperaturas **Máximas** e **Mínimas** em San Juan e Iquitos ao longo do tempo



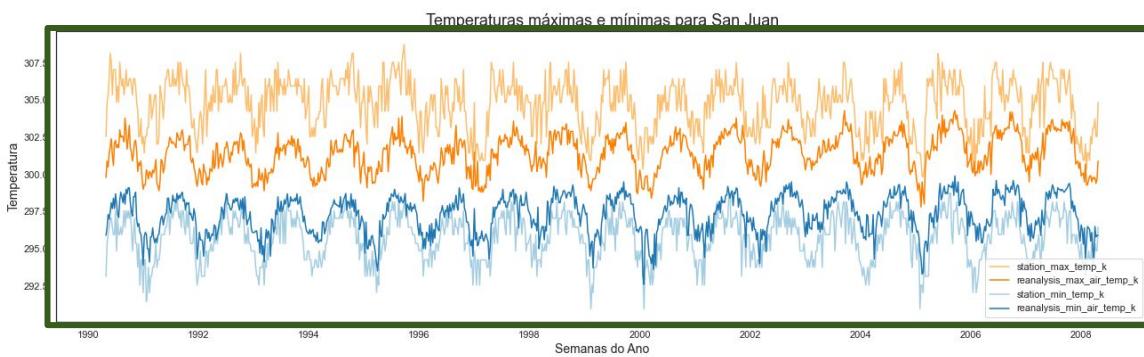
San Juan



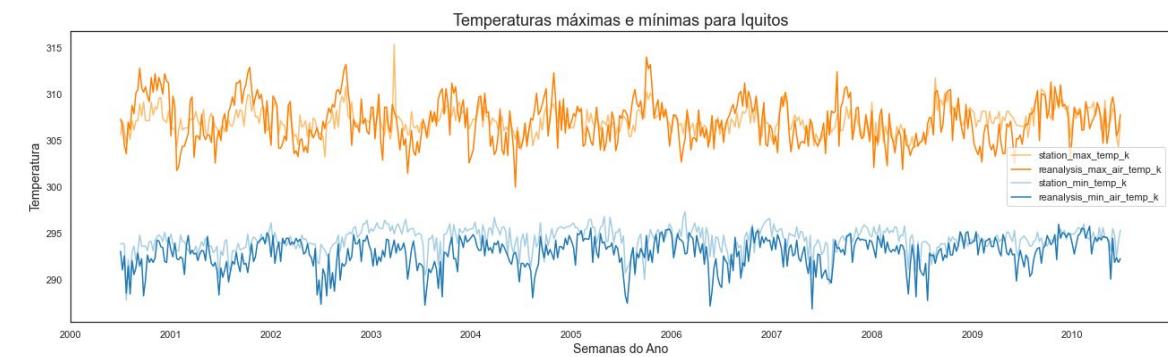
Iquitos

Temperatura: máximas e mínimas

Temperaturas **Máximas** e **Mínimas** em San Juan e Iquitos ao longo do tempo



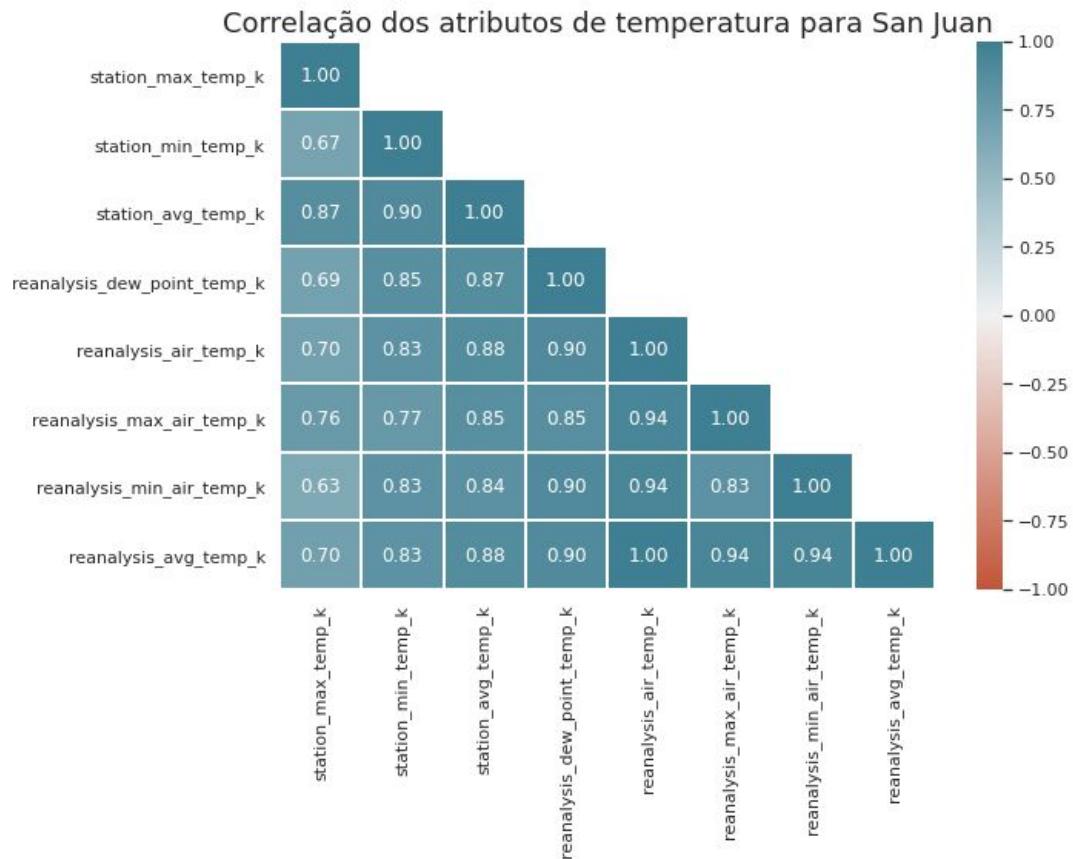
San Juan



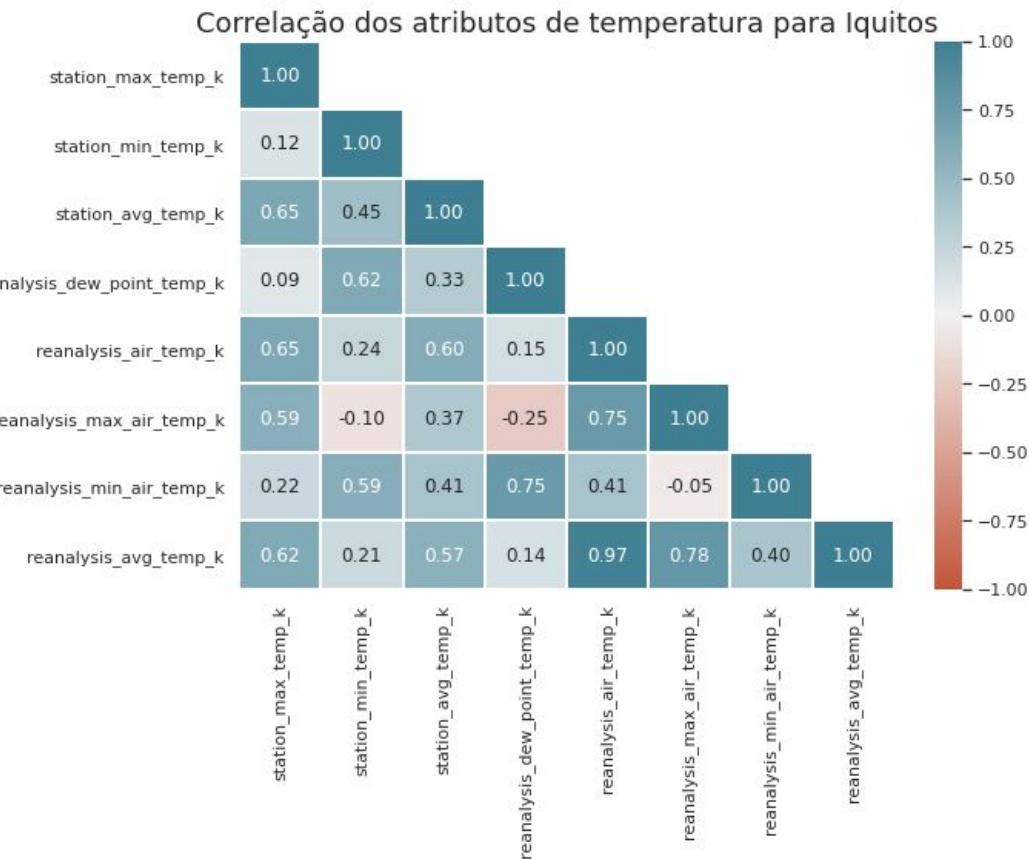
Iquitos

Temperatura: correlação

San Juan: Dados de temperatura com alta correlacionados



Iquitos: Menor correlação entre os dados
correlação negativa



Índice de vegetação: correlação

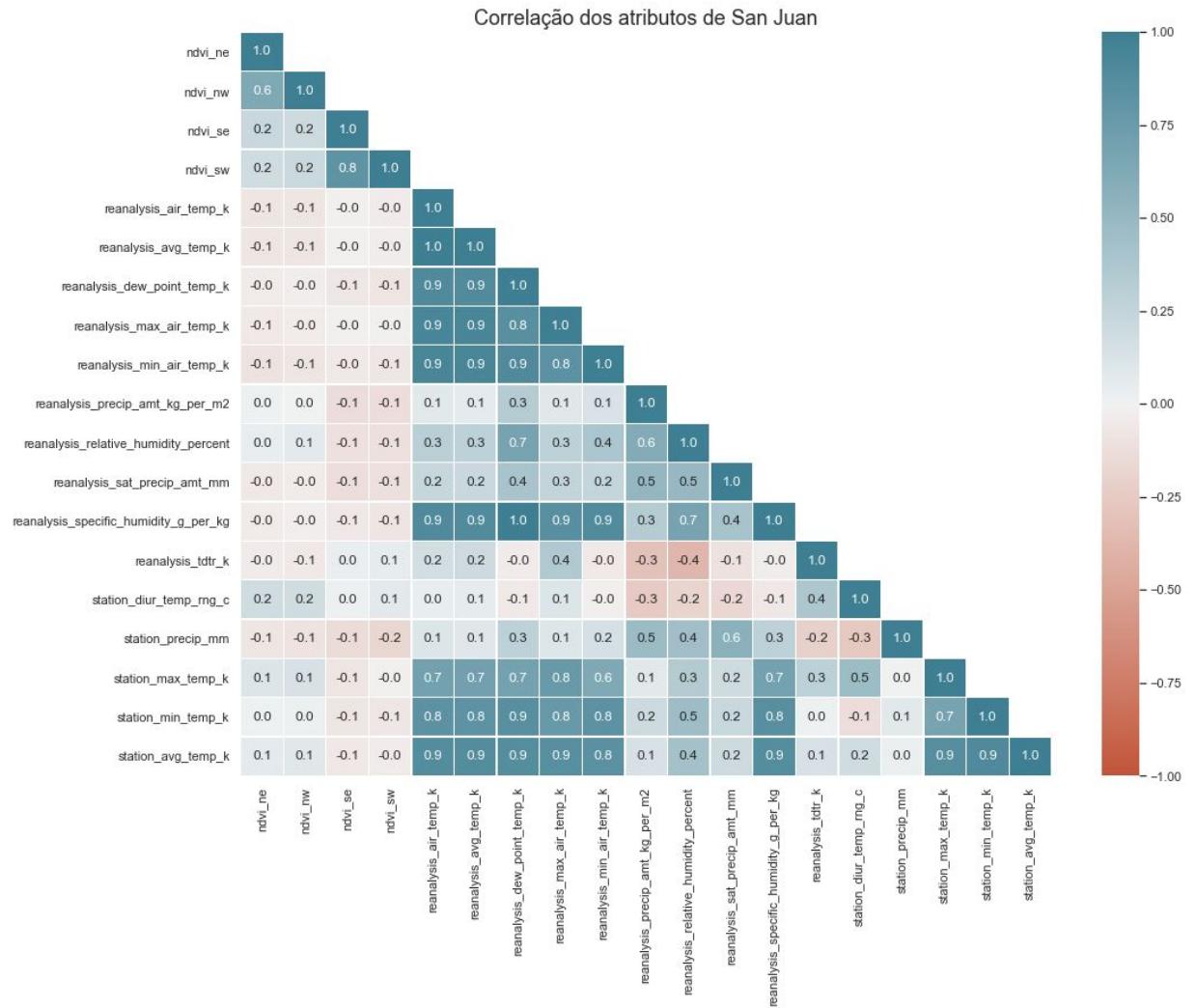
San Juan: Correlações Norte e Sul

	ndvi_se	ndvi_sw	ndvi_ne	ndvi_nw
ndvi_se	1.00	0.81	0.22	0.20
ndvi_sw	0.81	1.00	0.18	0.22
ndvi_ne	0.22	0.18	1.00	0.63
ndvi_nw	0.20	0.22	0.63	1.00

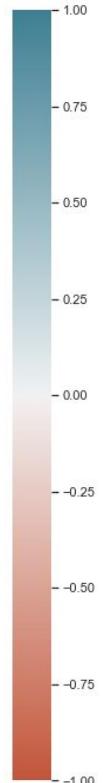
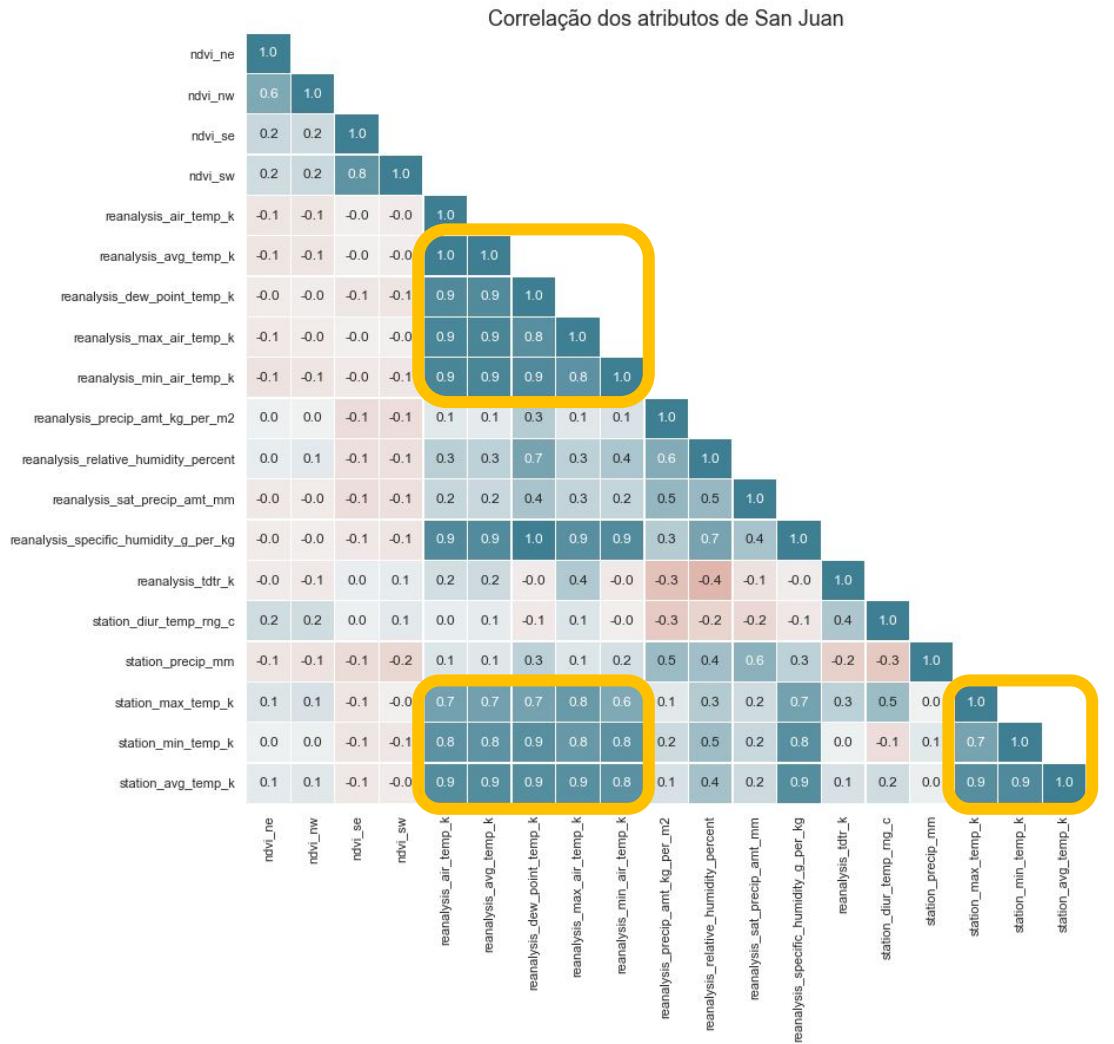
Iquitos: Forte correlação entre todas as regiões

	ndvi_se	ndvi_sw	ndvi_ne	ndvi_nw
ndvi_se	1.00	0.72	0.77	0.64
ndvi_sw	0.72	1.00	0.84	0.77
ndvi_ne	0.77	0.84	1.00	0.76
ndvi_nw	0.64	0.77	0.76	1.00

Correlação entre atributos: San Juan

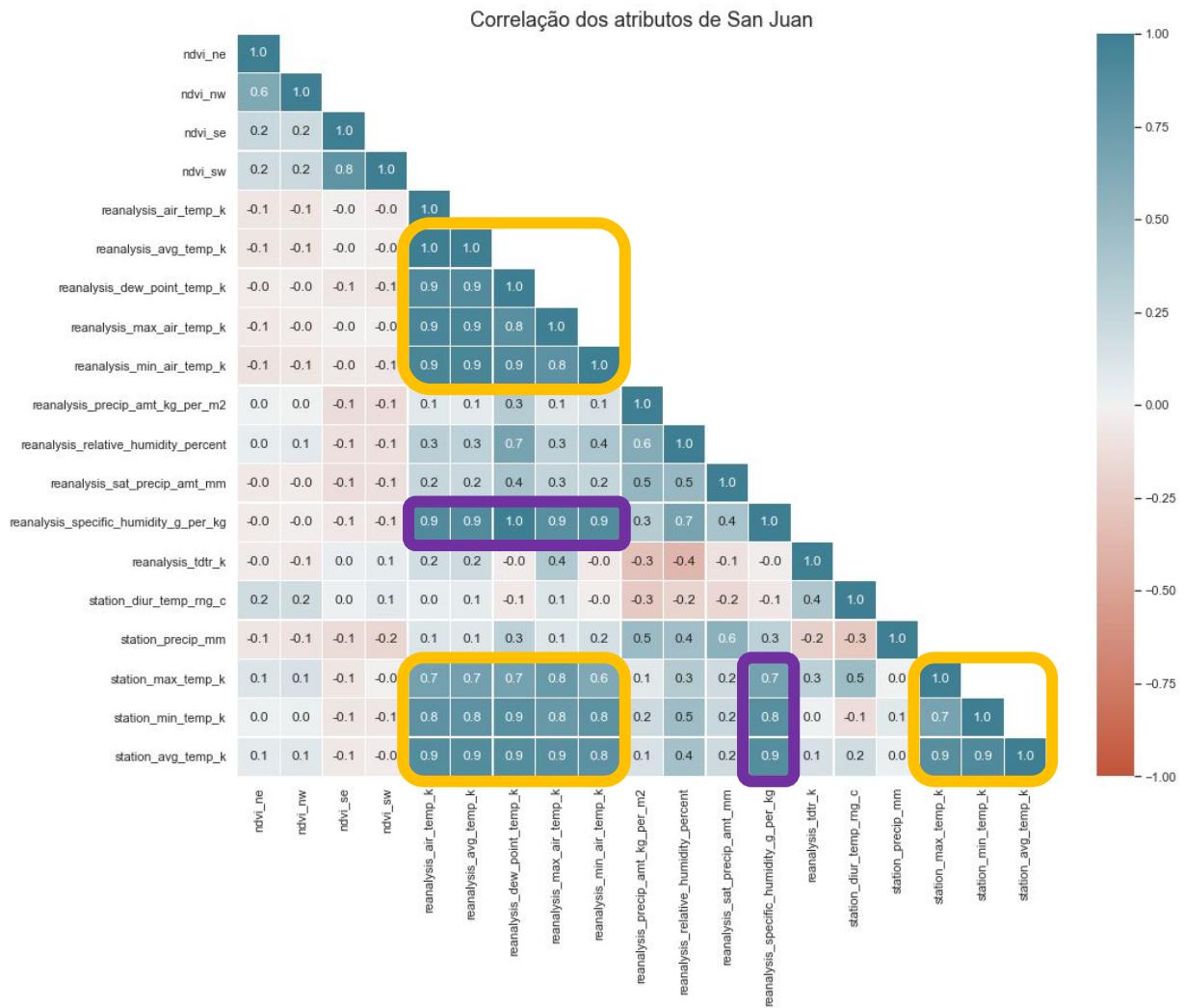


Correlação entre atributos: San Juan



Temperaturas: Alta correlação

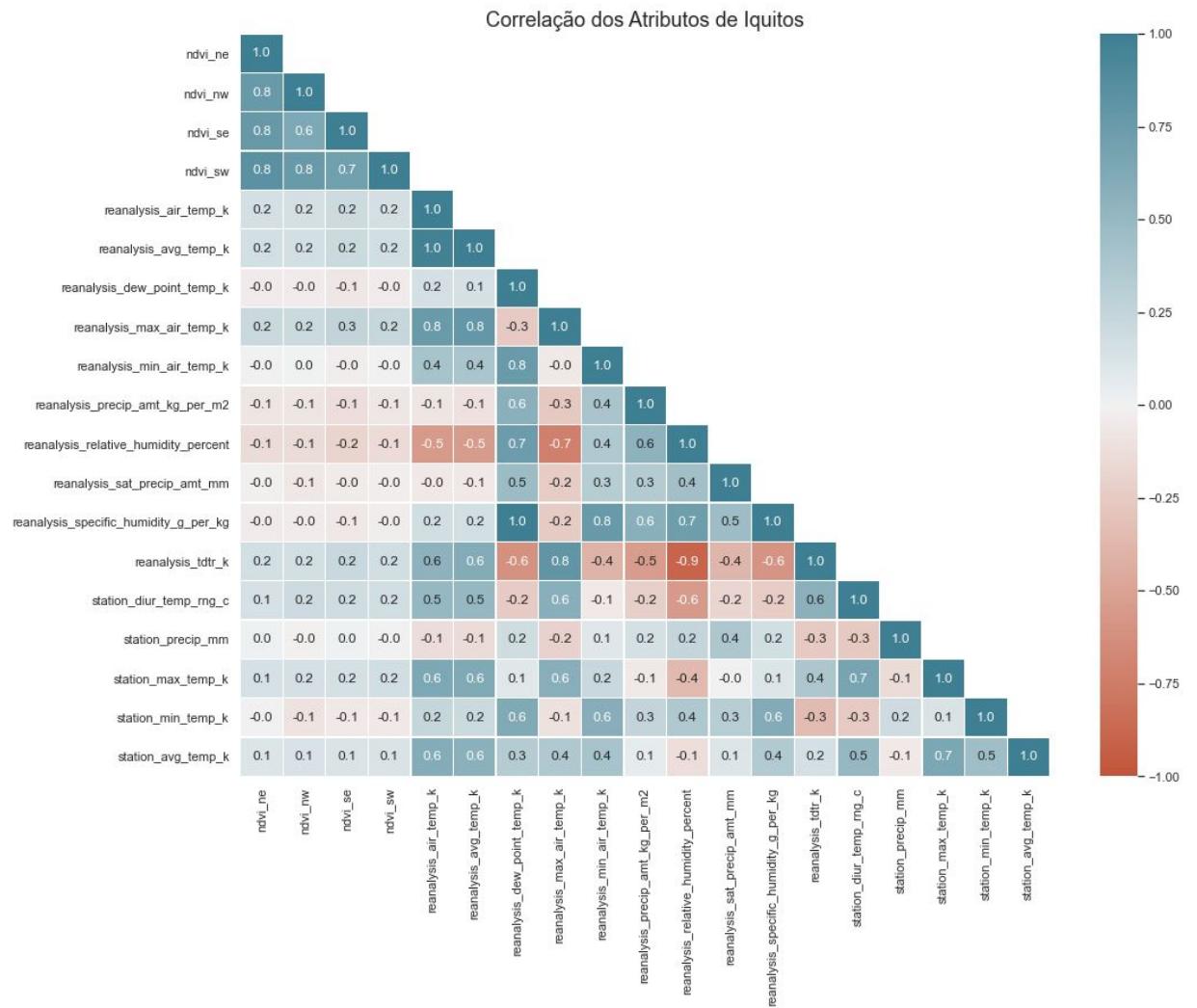
Correlação entre atributos: San Juan



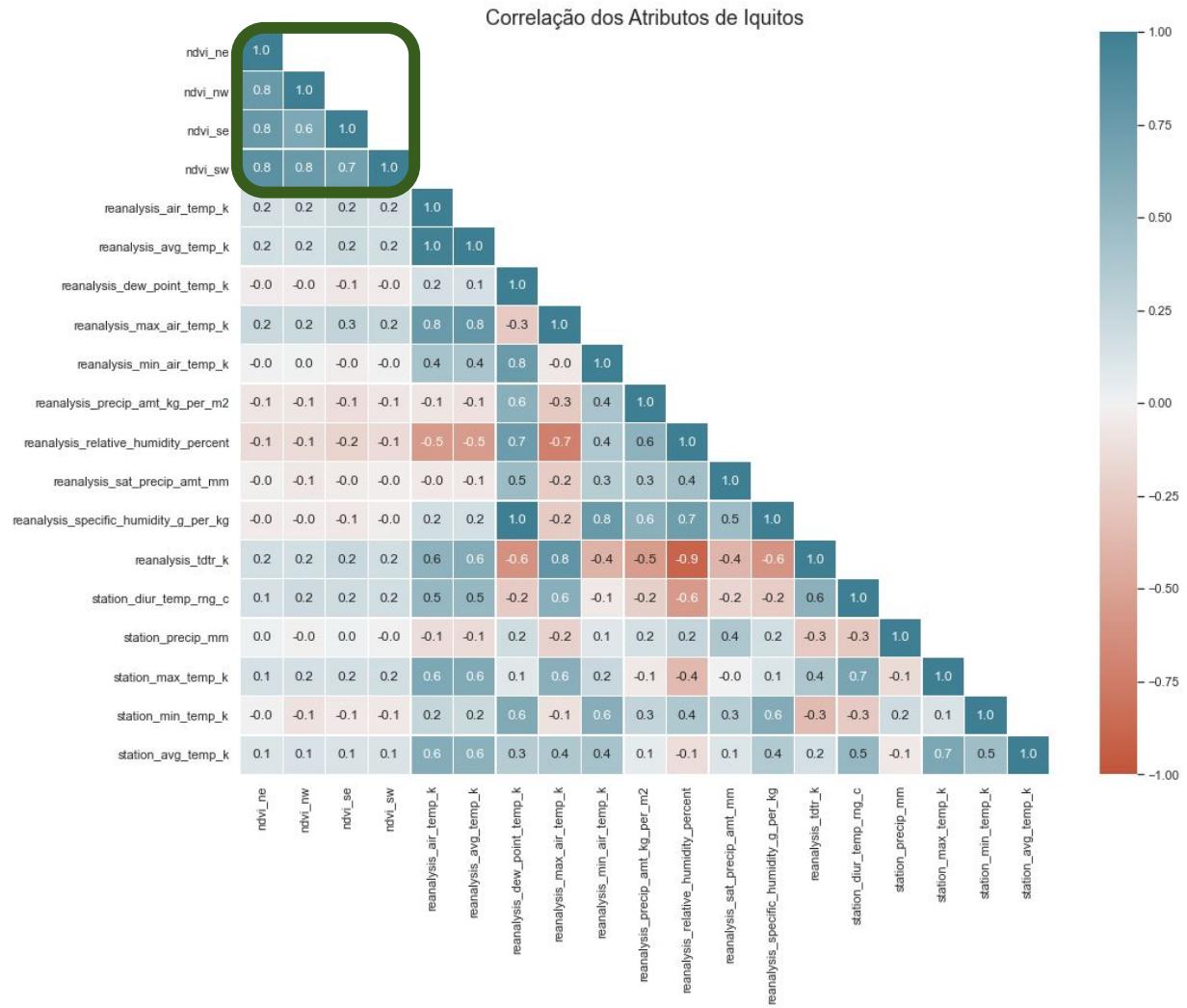
Temperaturas: Alta correlação

Umidade Específica e Temperaturas: Alta correlação

Correlação entre atributos: Iquitos

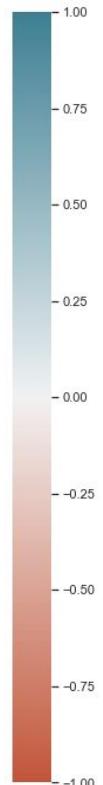
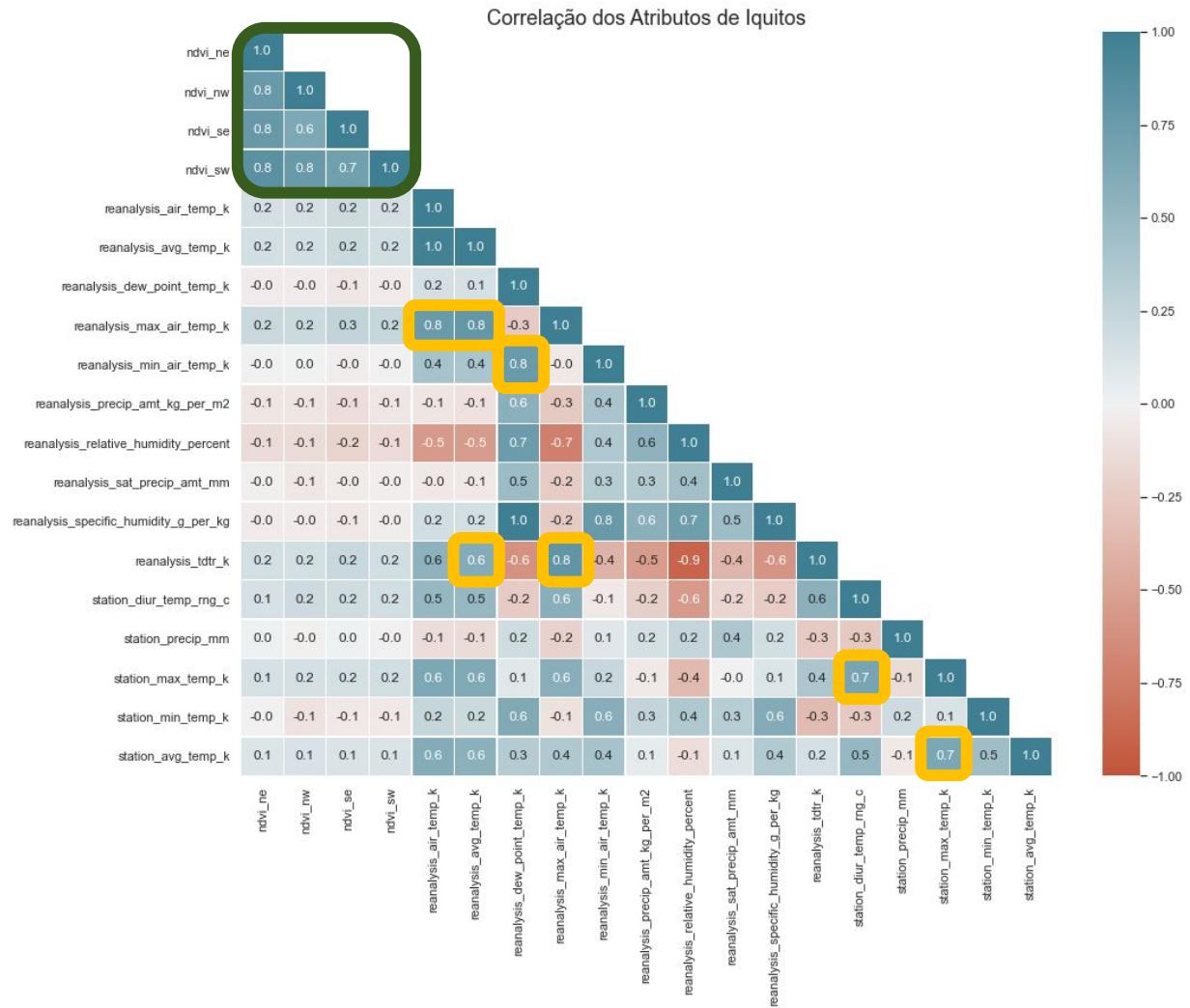


Correlação entre atributos: Iquitos



Índices de vegetação: Alta correlação

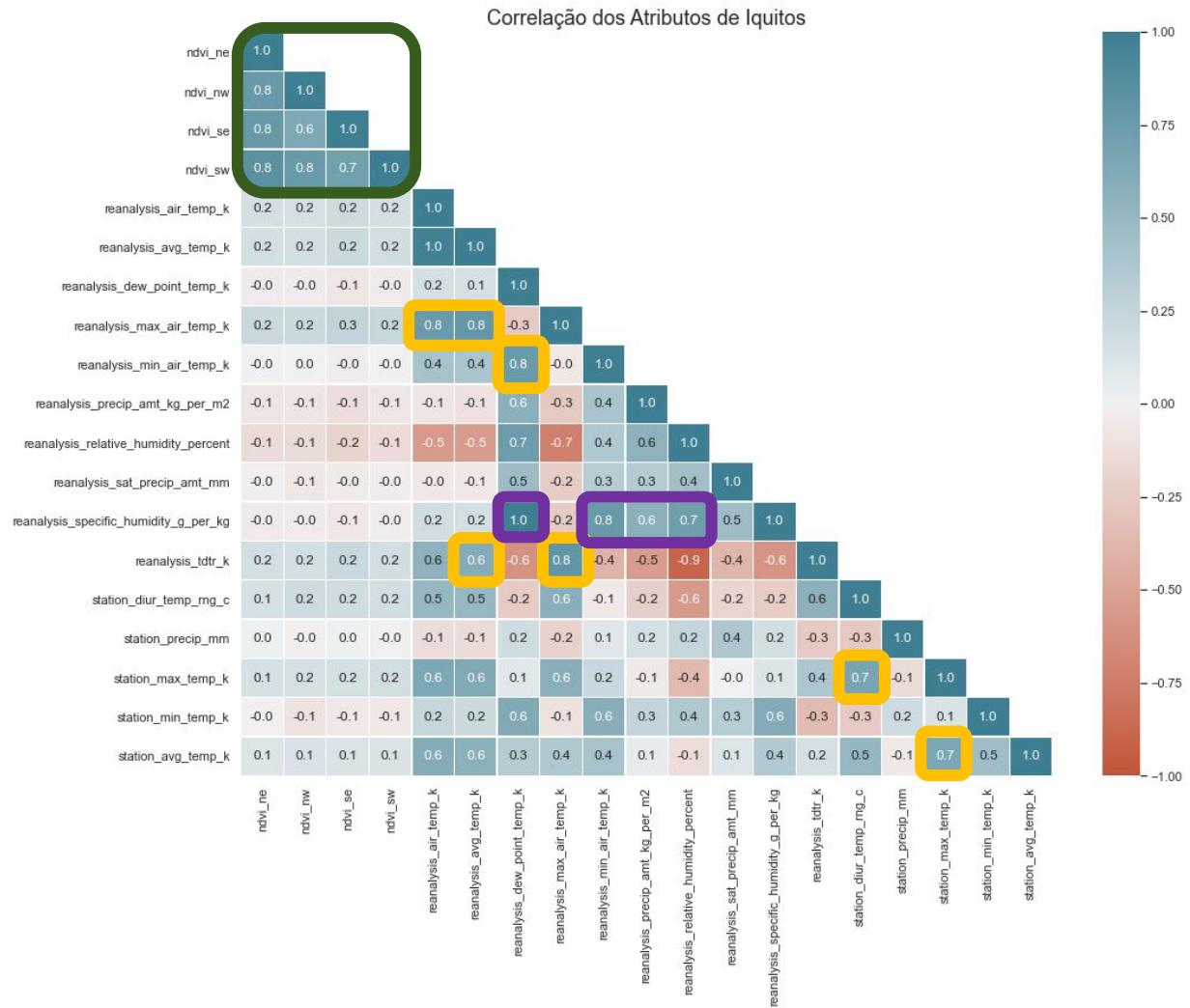
Correlação entre atributos: Iquitos



Índices de vegetação: Alta correlação

Temperaturas : Alguns valores

Correlação entre atributos: Iquitos

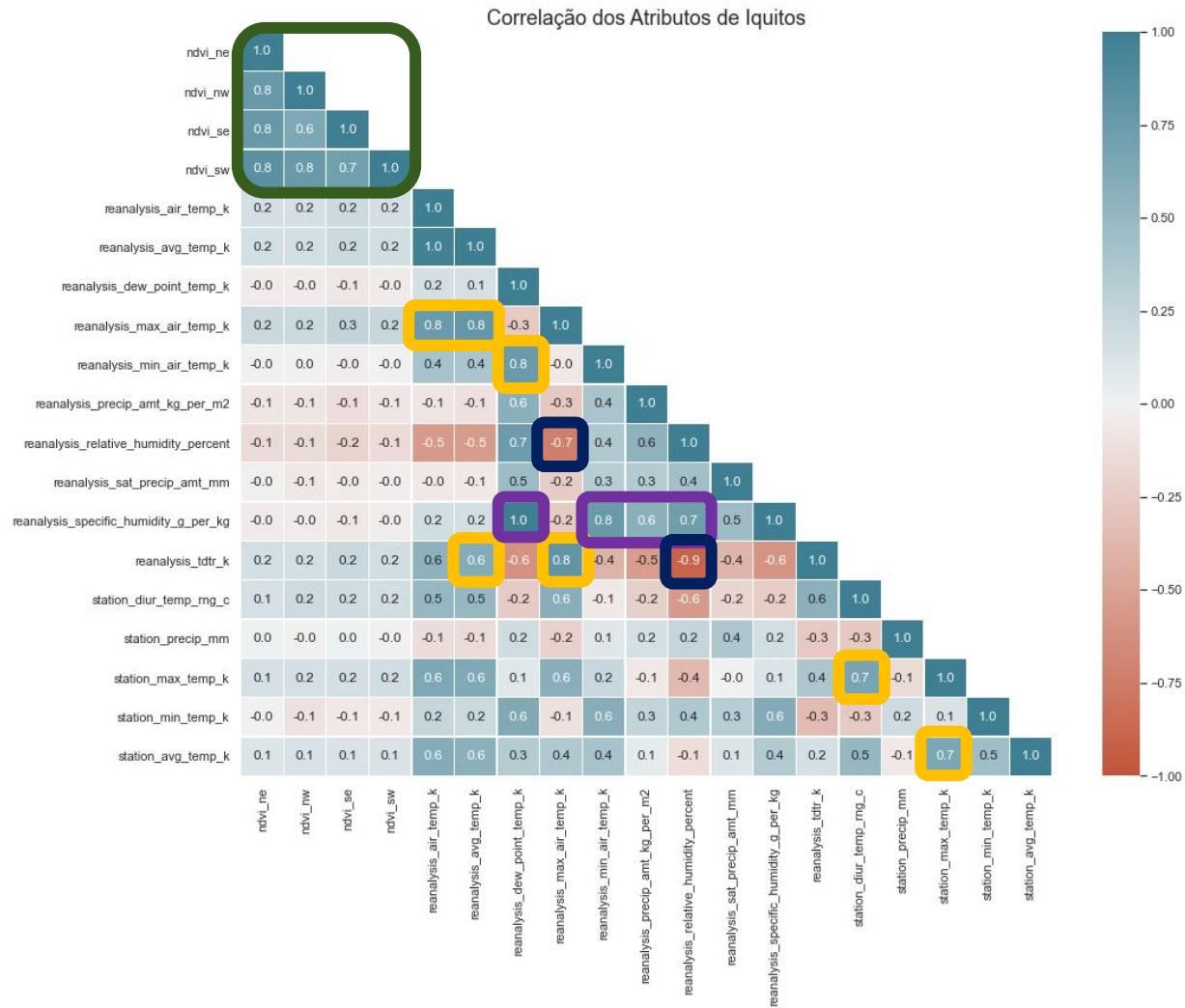


Índices de vegetação: Alta correlação

Temperaturas : Alguns valores

Umidade Específica e Temperatura do ponto de orvalho: Correlação quase perfeita

Correlação entre atributos: Iquitos



Índices de vegetação: Alta correlação

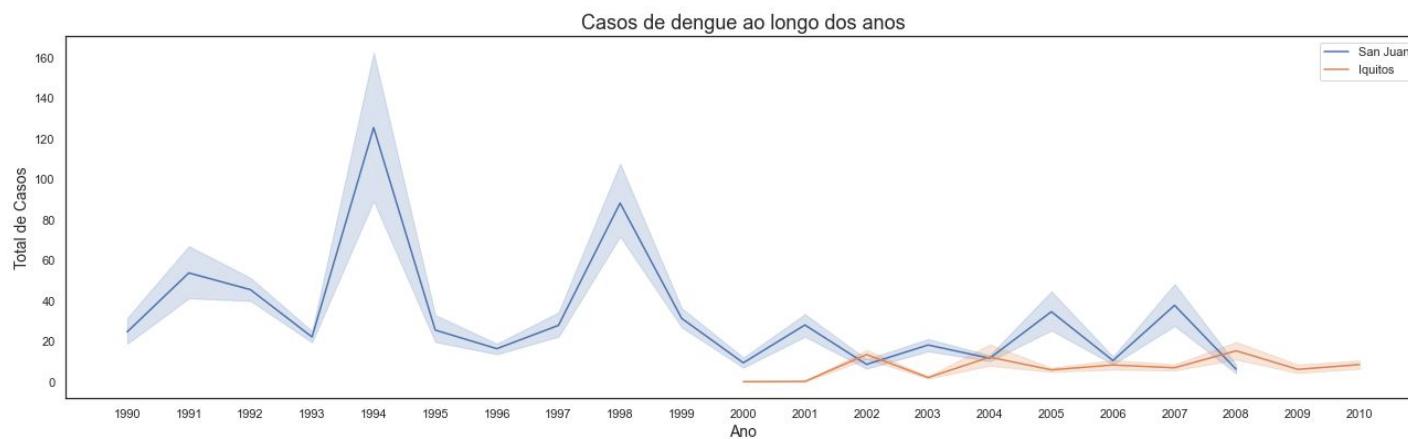
Temperaturas : Alguns valores

Umidade Específica e Temperatura do ponto de orvalho: Correlação quase perfeita

Umidade relativa e temperatura máxima e variação diurna: Correlação negativa forte

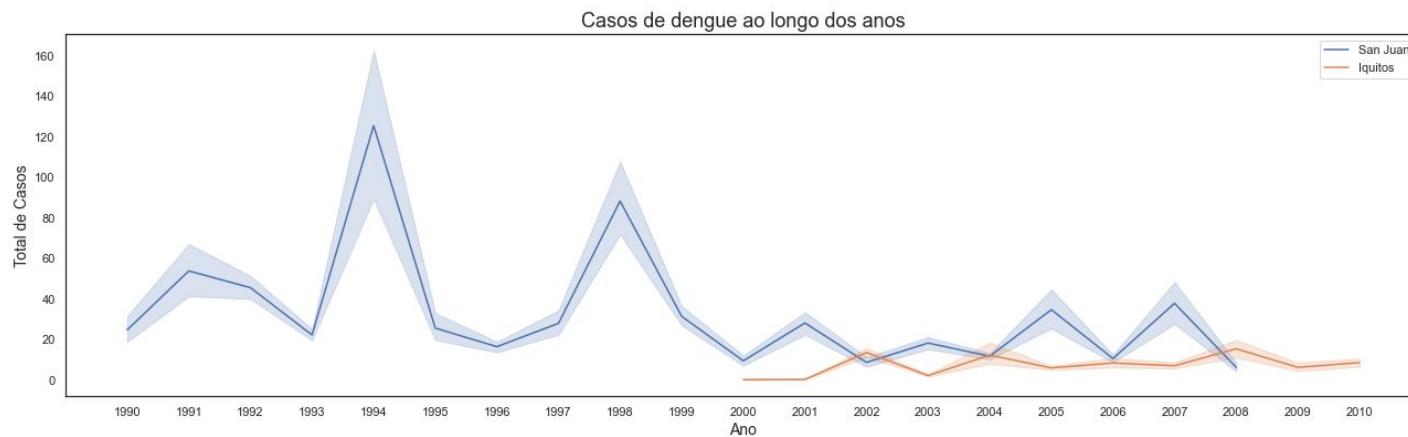
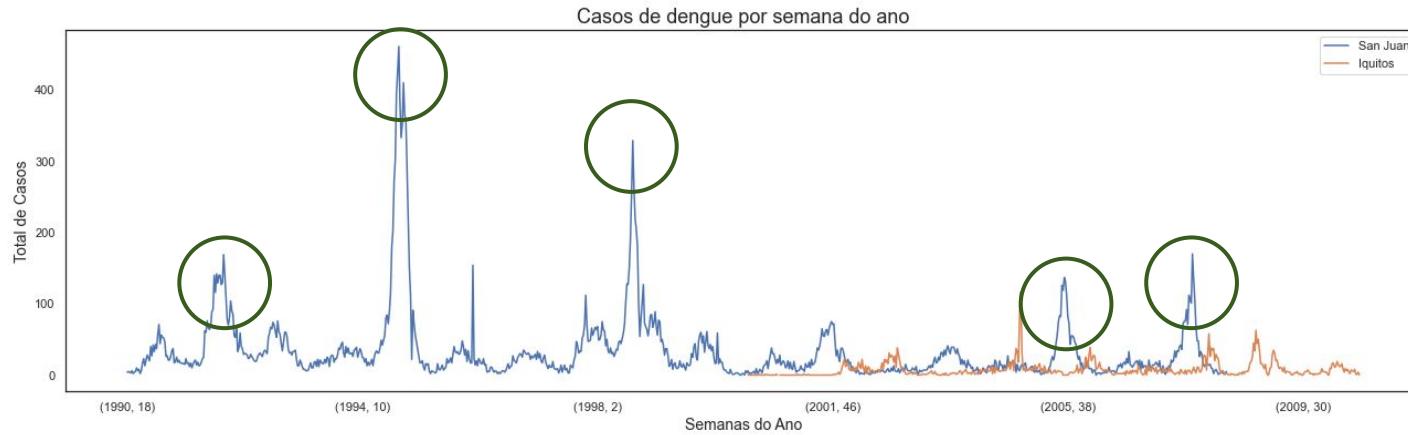
Rótulos: número de casos

Casos de dengue em **San Juan** e **Iquitos** a cada semana e agregados por ano

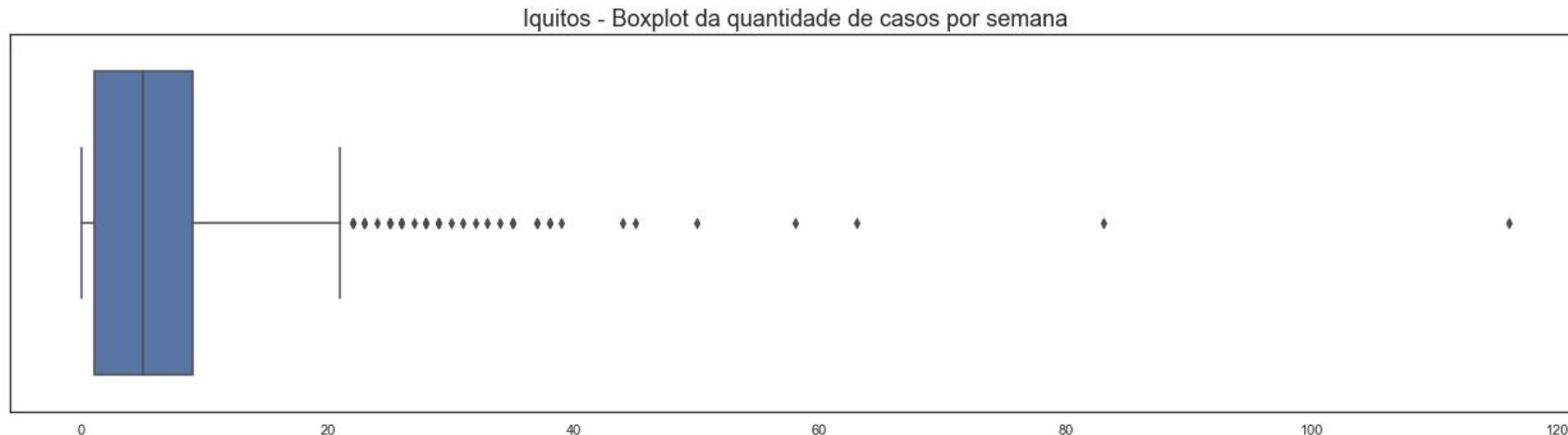
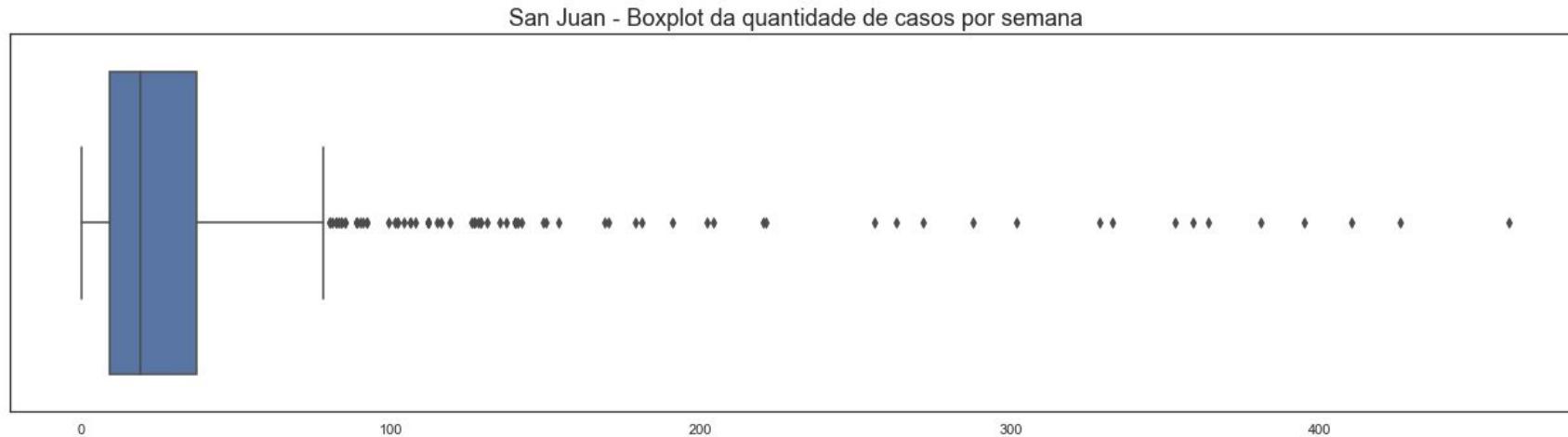


Rótulos: número de casos

Casos de dengue em **San Juan** e **Iquitos** a cada semana e agregados por ano

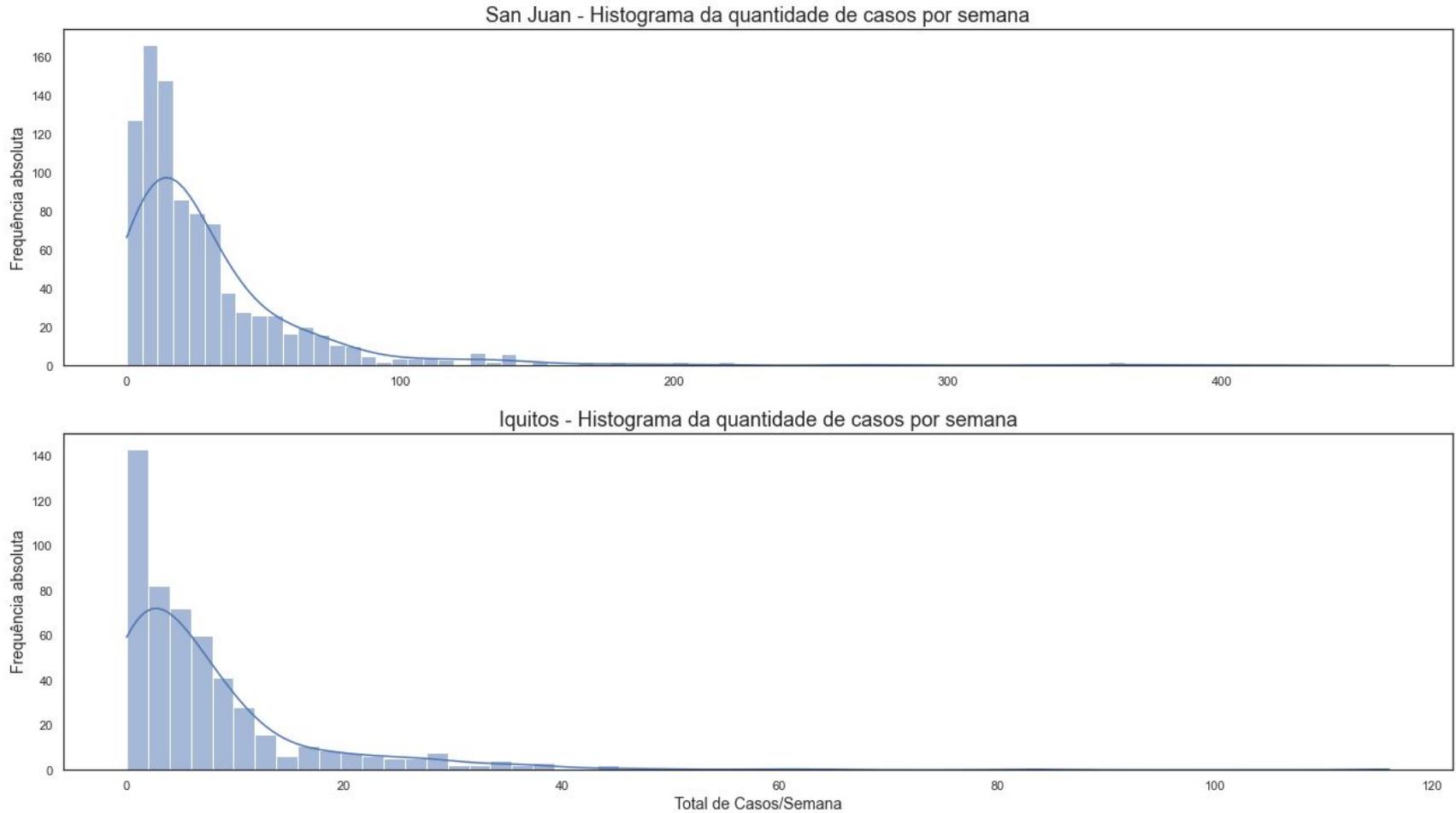


Rótulos: estatísticas dos casos - Boxplot



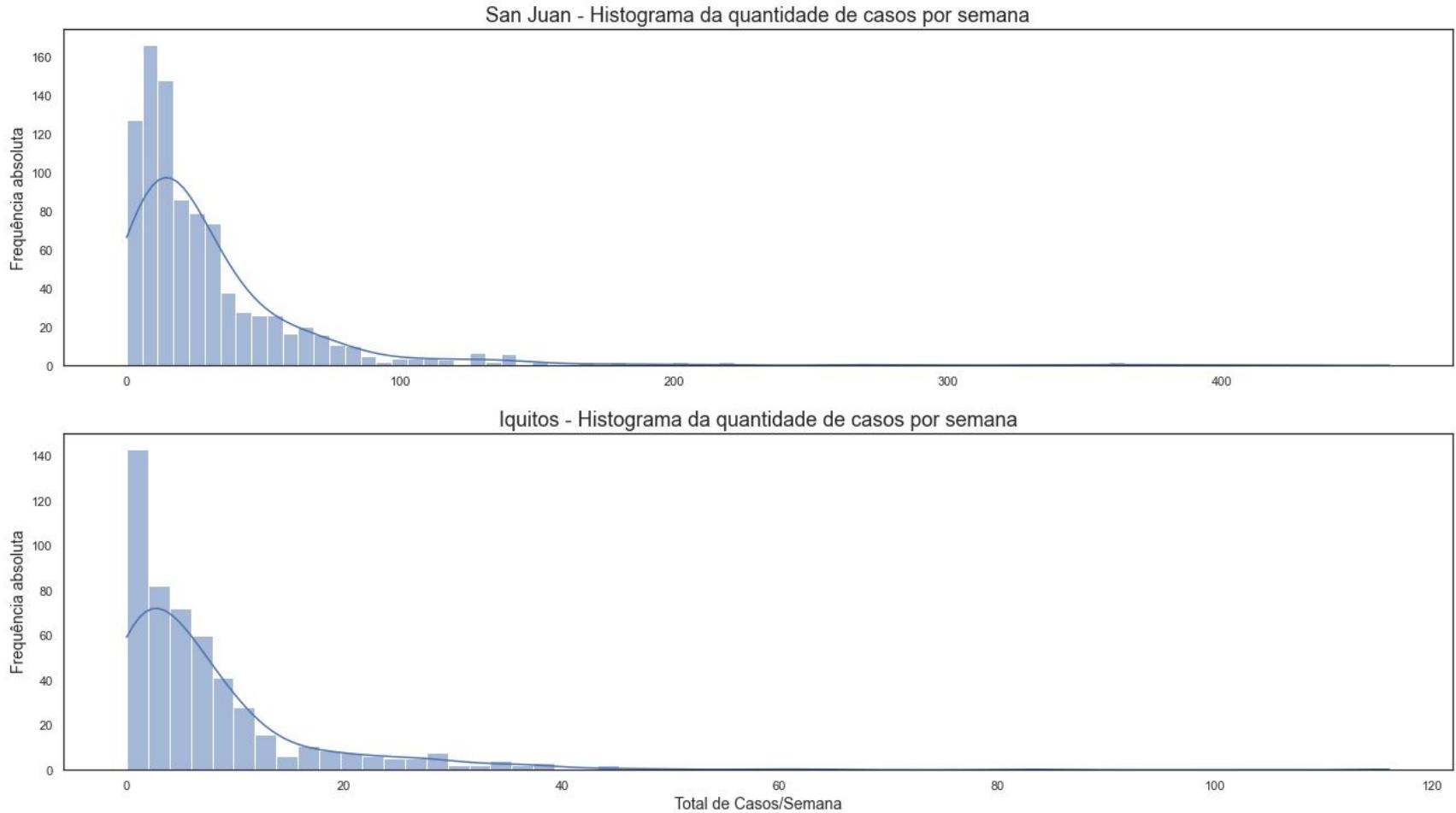
	San Juan	Iquitos
count	936.00	520.00
mean	34.18	7.57
var	2640.05	115.90
std	51.38	10.77
min	0.00	0.00
25%	9.00	1.00
50%	19.00	5.00
75%	37.00	9.00
max	461.00	116.00

Rótulos: estatísticas dos casos - Histograma



	San Juan	Iquitos
count	936.00	520.00
mean	34.18	7.57
var	2640.05	115.90
std	51.38	10.77
min	0.00	0.00
25%	9.00	1.00
50%	19.00	5.00
75%	37.00	9.00
max	461.00	116.00

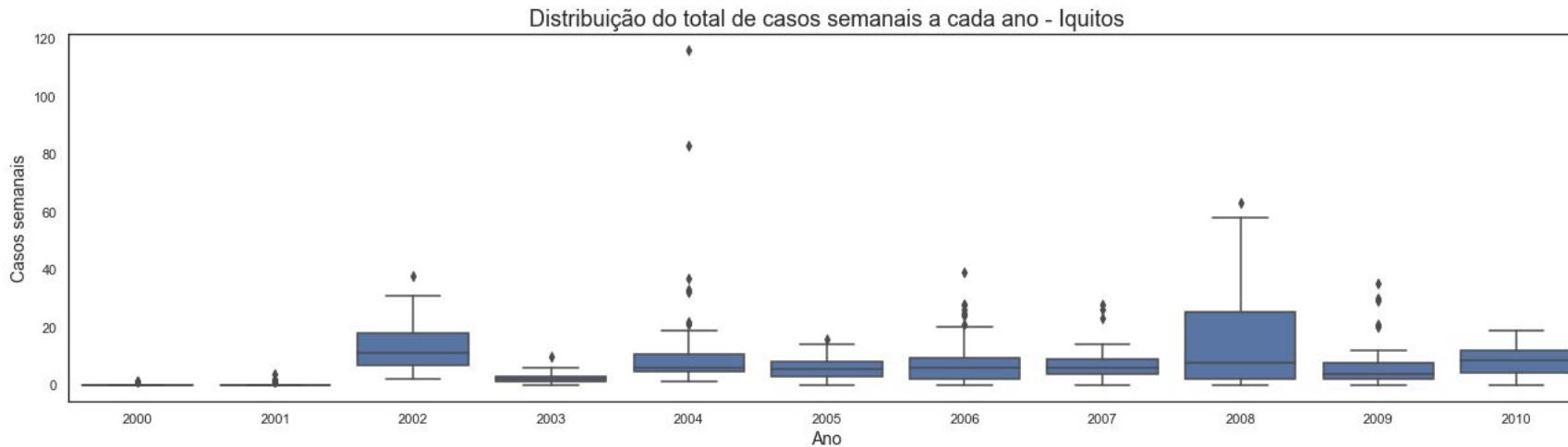
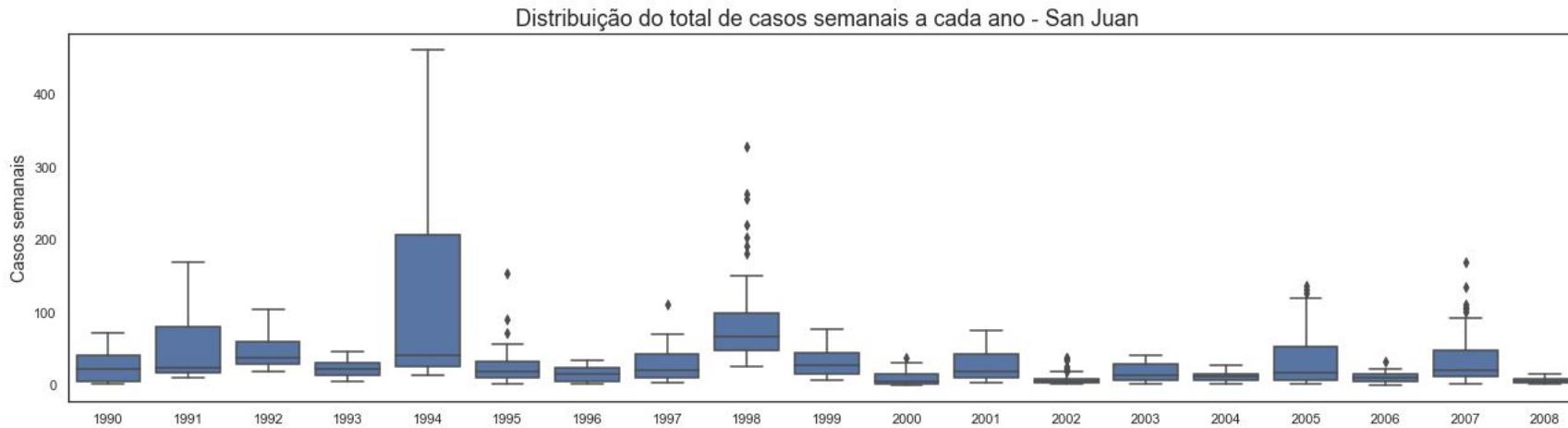
Rótulos: estatísticas dos casos - Histograma



	San Juan	Iquitos
count	936.00	520.00
mean	34.18	7.57
var	2640.05	115.90
std	51.38	10.77
min	0.00	0.00
25%	9.00	1.00
50%	19.00	5.00
75%	37.00	9.00
max	461.00	116.00

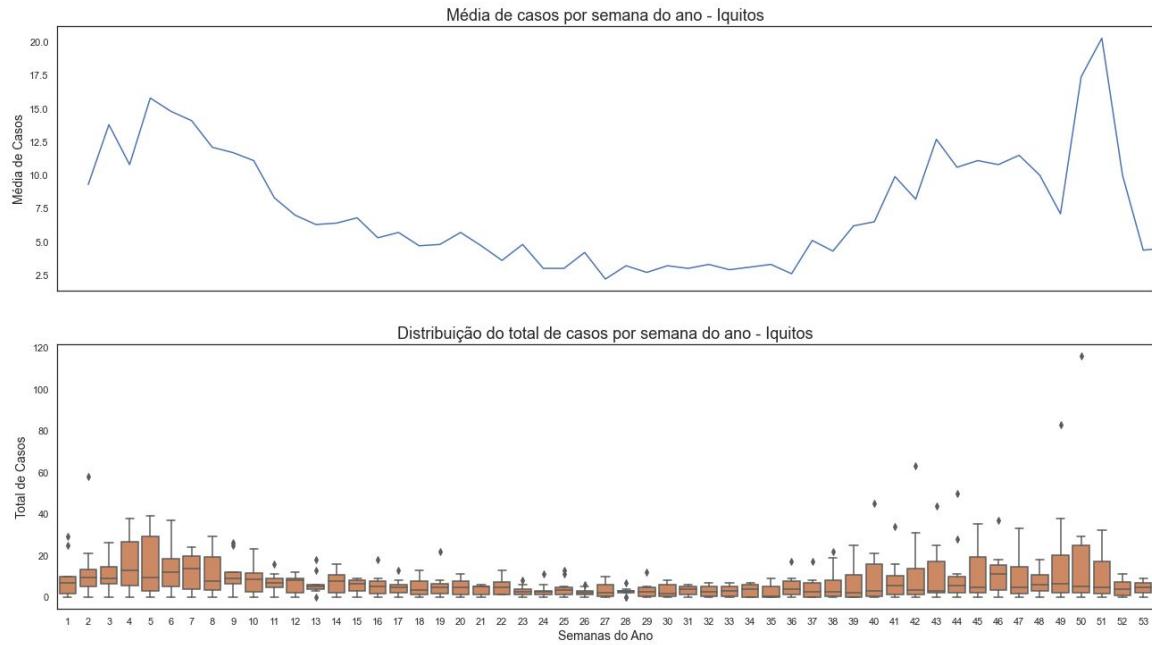
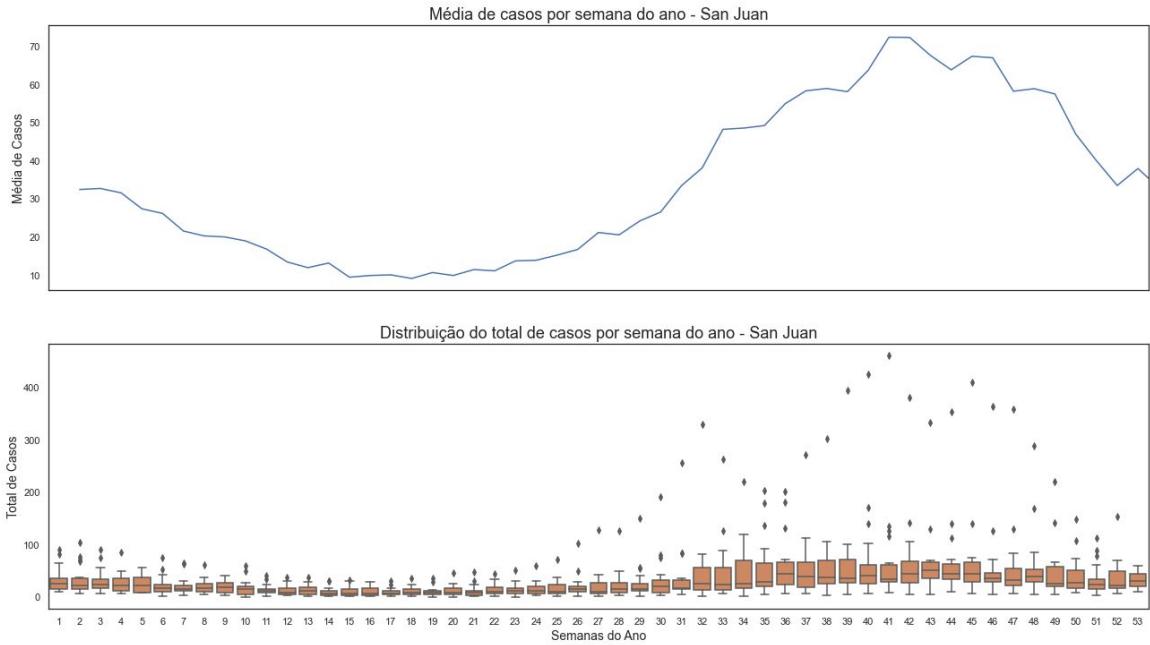
Rótulos: tendência

Nenhum indicativo de tendência ao longo dos anos



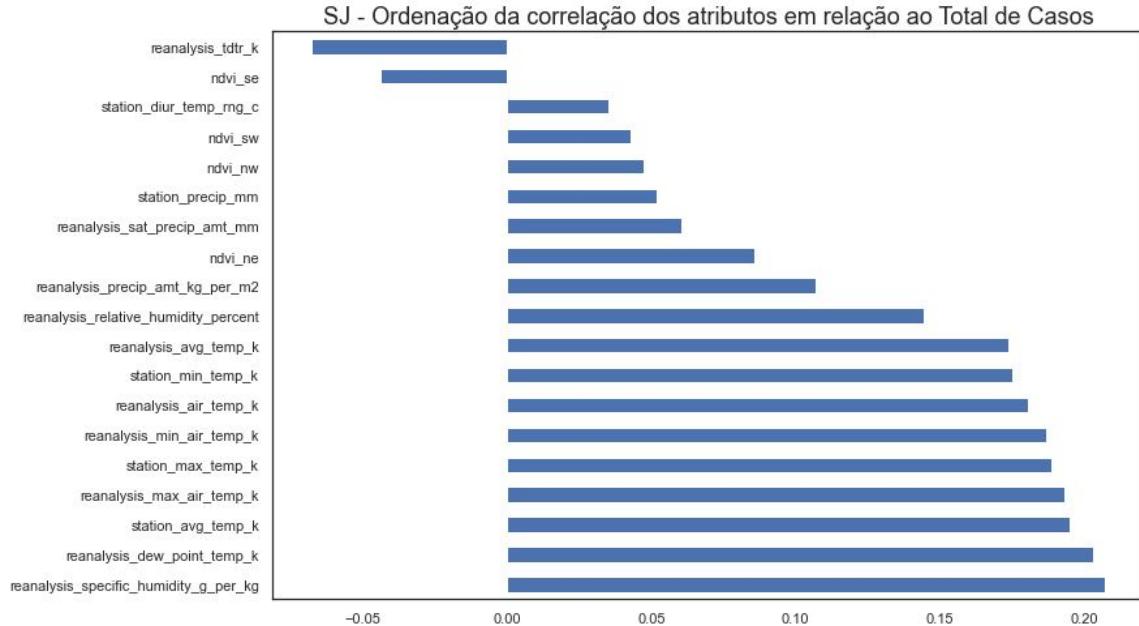
Rótulos: sazonalidade

Períodos mais ou menos definidos dentro do ano

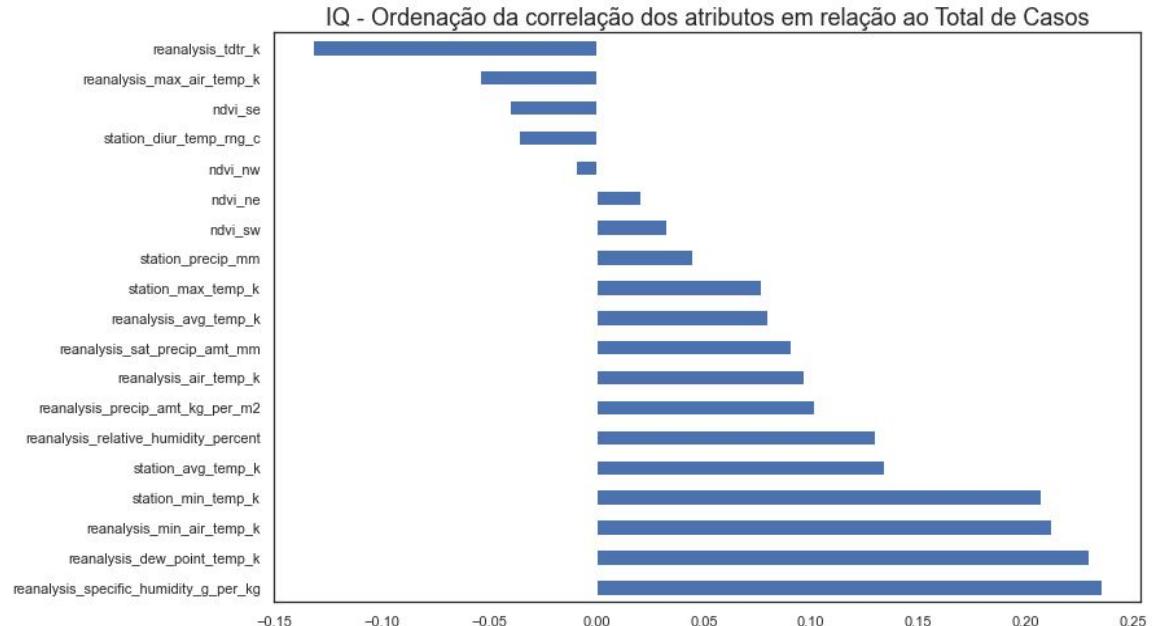


Rótulos: correlação com atributos

San Juan



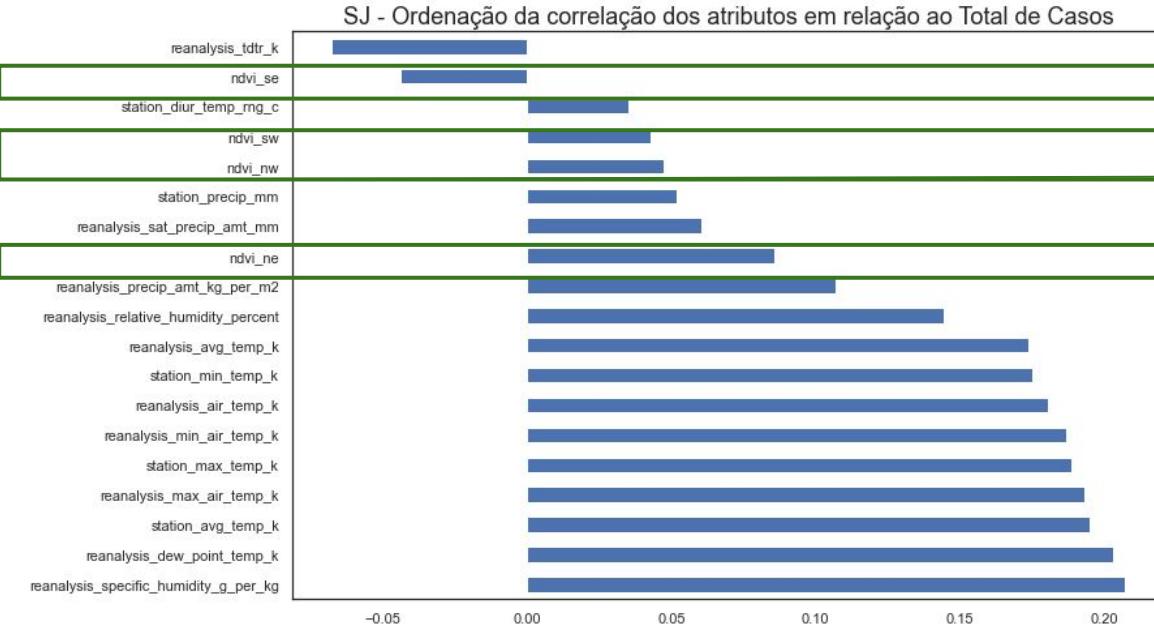
Iquitos



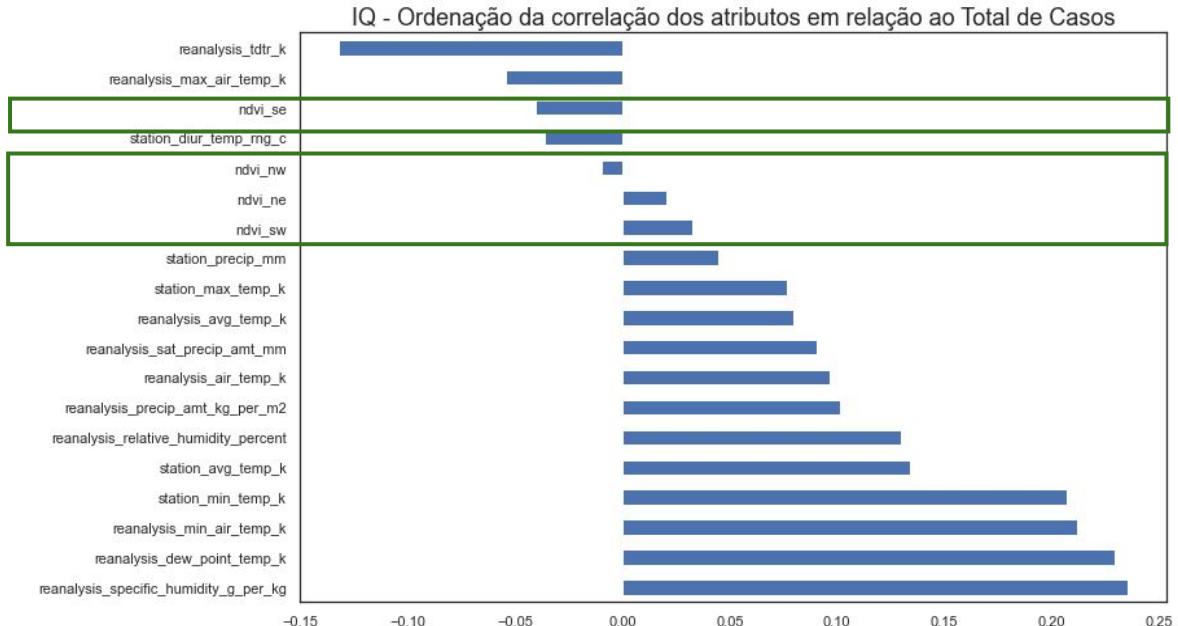
- Nenhuma correlação forte individualmente

Rótulos: correlação com atributos

San Juan



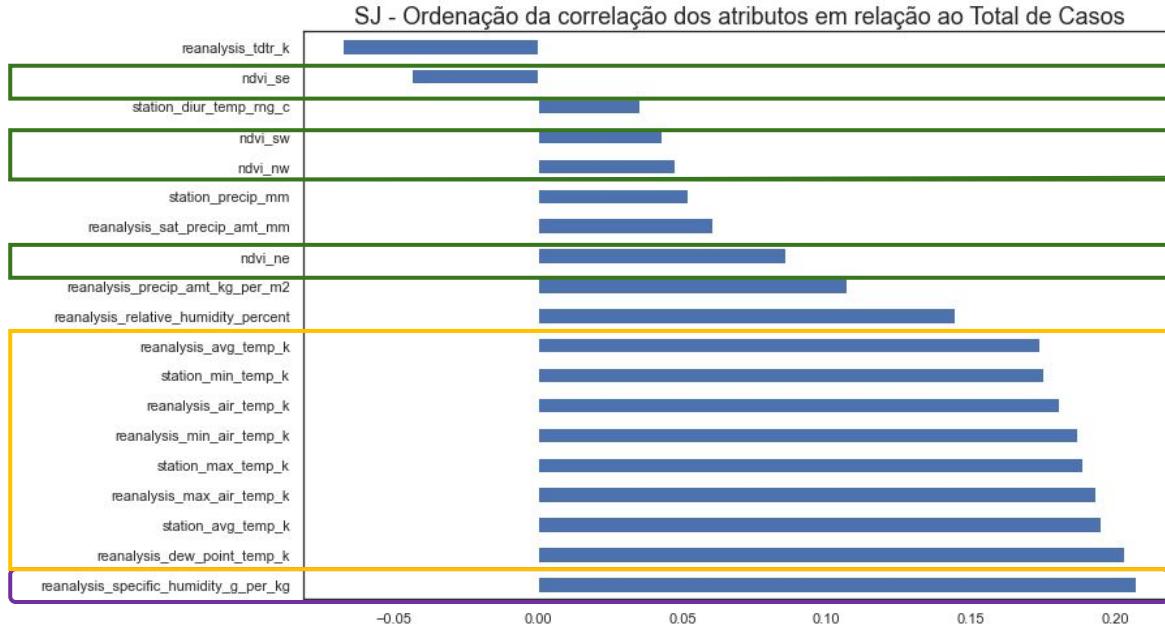
Iquitos



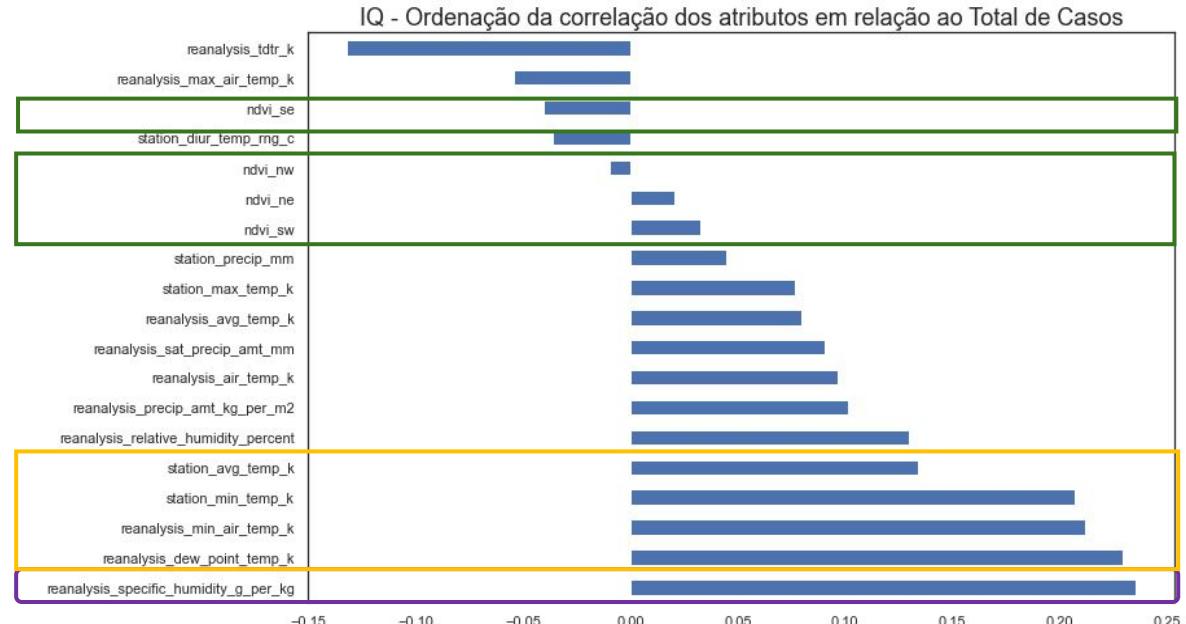
- Nenhuma correlação forte individualmente
- Pouca correlação com Vegetação

Rótulos: correlação com atributos

San Juan

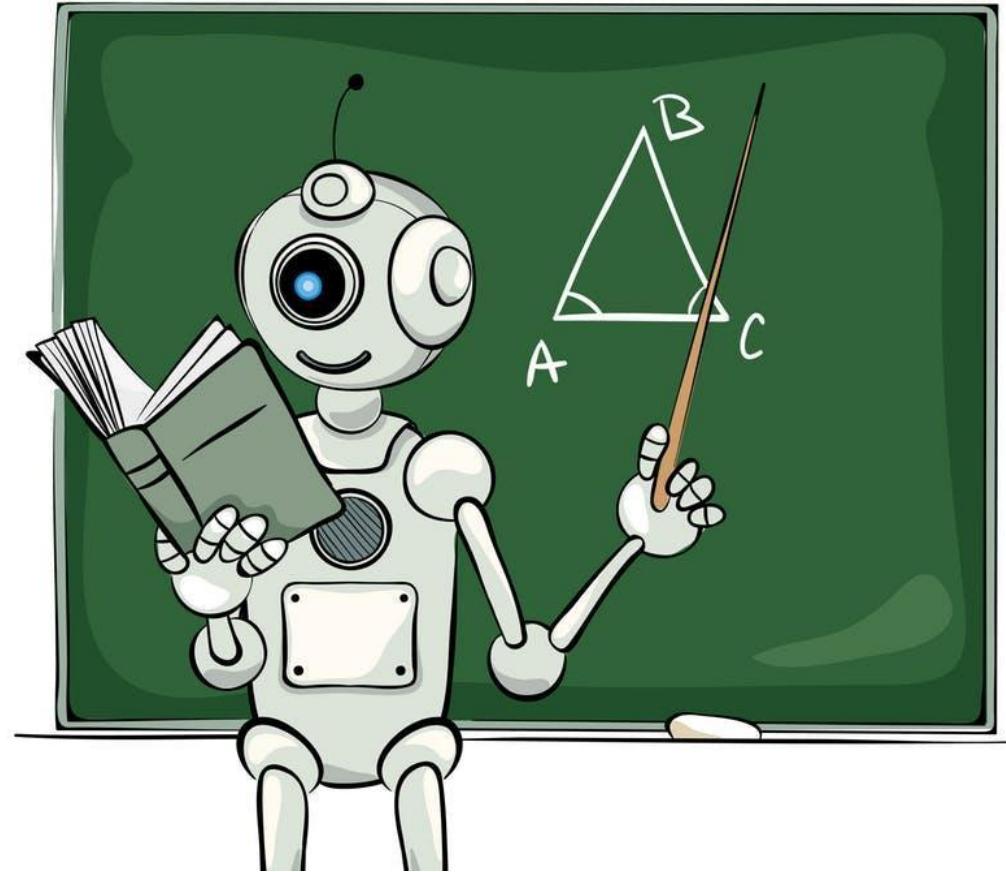


Iquitos



- Nenhuma correlação forte individualmente
- Pouca correlação com Vegetação
- Maiores correlações com Umidade e Temperaturas

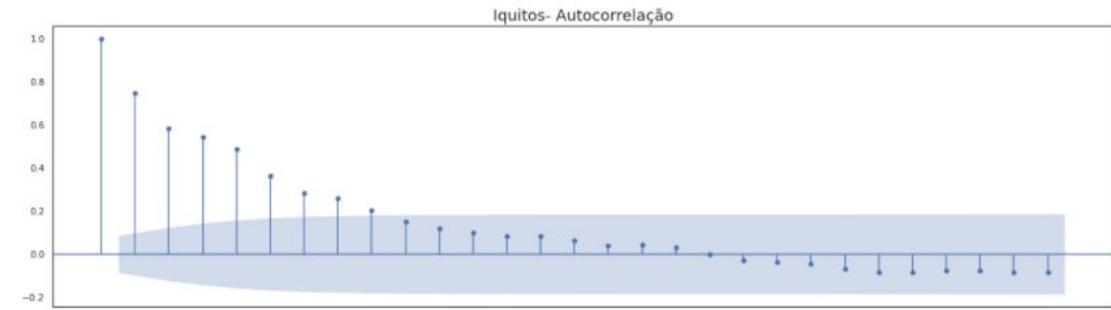
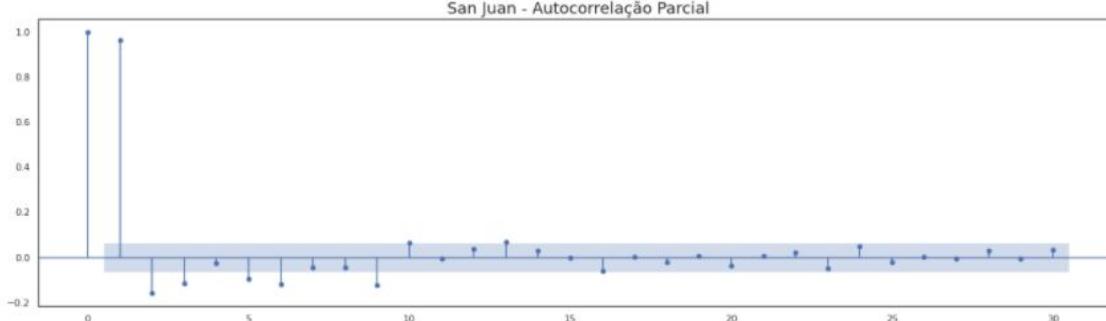
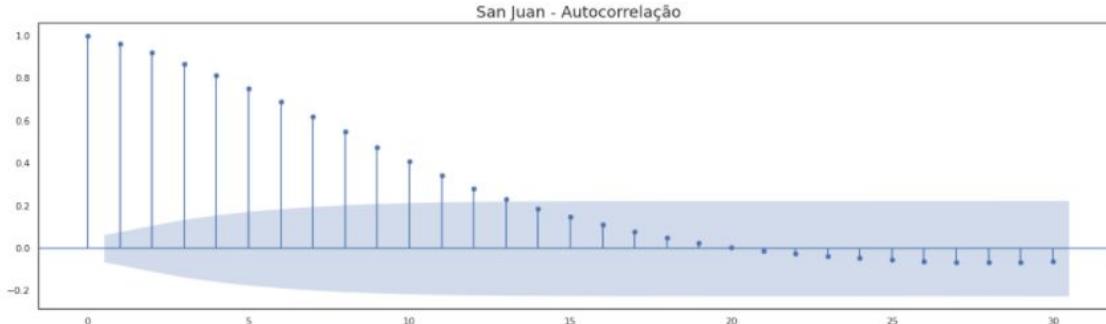
Modelo Preditivo



Modelo Baseline - Série Temporal

Autocorrelação: verifica se os dados anteriores interferem nos valores dos dados futuros.

Autocorrelação parcial: retira o efeito do LAG 0 desse cálculo (efeito da função sobre ela mesma)



Modelo Autoregressivo: Autocorrelação com decaimento suave e Autocorrelação Parcial com redução abrupta

Modelo Baseline - Série Temporal

Modelo Baseline: AutoReg da biblioteca Statsmodels.tsa.ar_model

```
# Número de lags a serem incluídas no modelo
lags = 13 # Valor com nível de significância maior que 5% para San Juan (maior que Iquitos)

# Definições do modelo
trend = 'n' # --> Nenhuma tendência
seasonal = True # --> Modelo Sazonal (Resultados anuais)
period = 52 # --> Período de Sazonalidade (um ano)

baseline_model = AutoReg(train_df[target].values, lags=lags, trend=trend, seasonal=seasonal, period=period, old_names=True)
baseline_results = baseline_model.fit()
```

Modelo Baseline - Série Temporal

Modelo Baseline: AutoReg da biblioteca Statsmodels.tsa.ar_model

```
# Número de lags a serem incluídas no modelo
lags = 13 # Valor com nível de significância maior que 5% para San Juan (maior que Iquitos)

# Definições do modelo
trend = 'n' # --> Nenhuma tendência
seasonal = True # --> Modelo Sazonal (Resultados anuais)
period = 52 # --> Período de Sazonalidade (um ano)

baseline_model = AutoReg(train_df[target].values, lags=lags, trend=trend, seasonal=seasonal, period=period, old_names=True)
baseline_results = baseline_model.fit()
```

Modelo Baseline - Série Temporal

Avaliação do Modelo: Mean Absolute Error*

$$MAE = \frac{1}{n} \sum \left| y - \hat{y} \right|$$

Divide by the total number of data points

Predicted output value

Actual output value

Sum of

The absolute value of the residual

53

*Parâmetro indicado pela competição

Modelo Baseline - Série Temporal

Avaliação do Modelo: Mean Absolute Error*

$$MAE = \frac{1}{n} \sum \left| y - \hat{y} \right|$$

Divide by the total number of data points

Predicted output value

Actual output value

Sum of

The absolute value of the residual

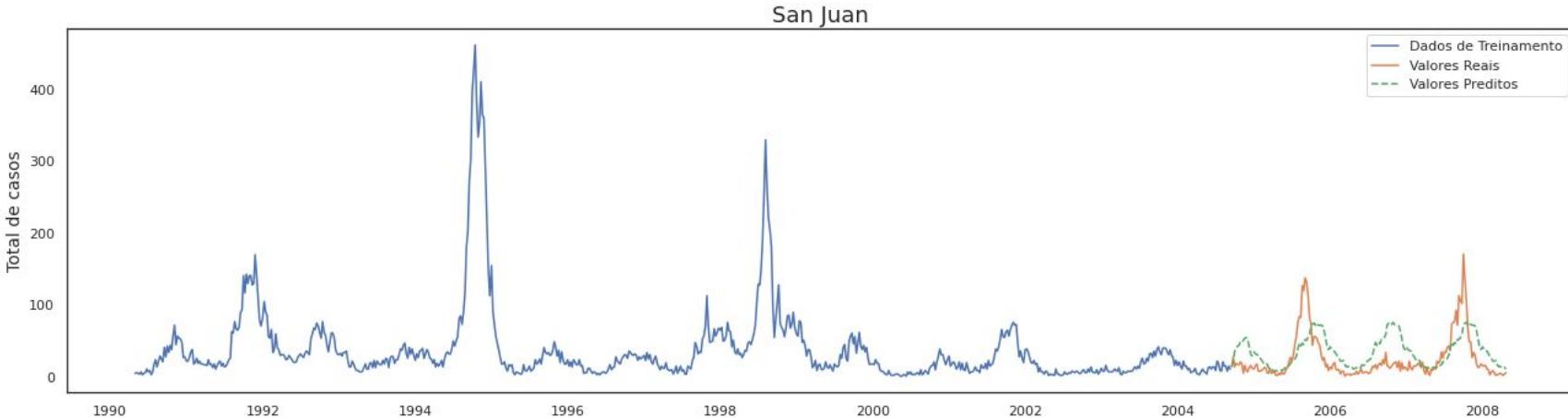
```
from sklearn.metrics import mean_absolute_error

def error_model(labels_test, predict):
    mae = mean_absolute_error(labels_test['total_cases'], predict)
    return mae
```

*Parâmetro indicado pela competição

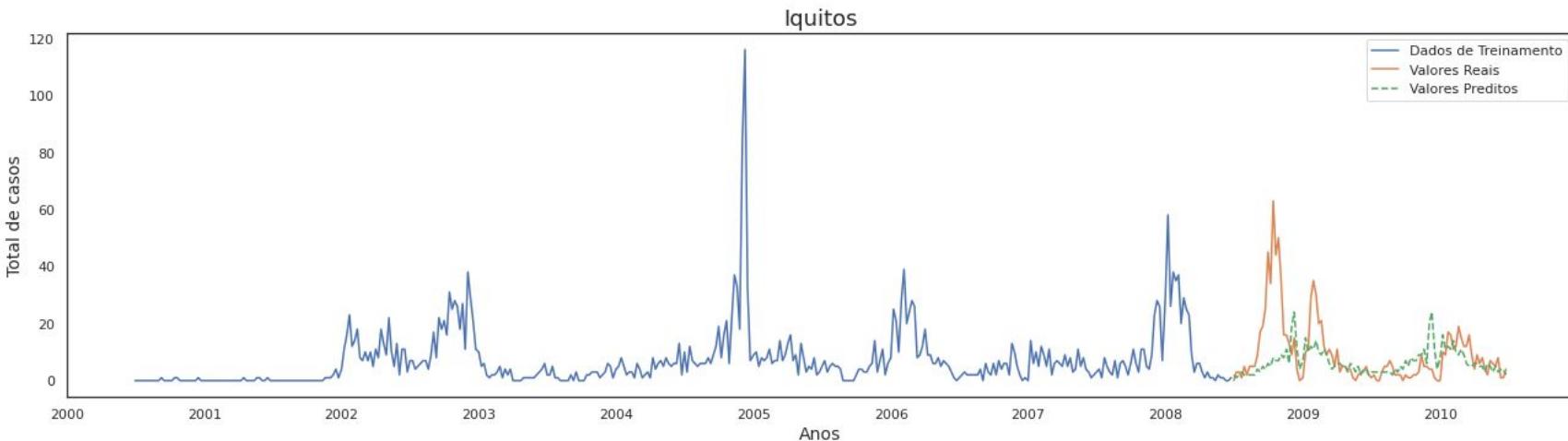
Modelo Baseline - Série Temporal

Modelo Baseline: AutoReg da biblioteca Statsmodels.tsa.ar_model



MAE

	San Juan	Iquitos	Geral
Baseline	22.82	6.71	17.08



Modelo Binomial Negativo

- Generalização da *Regressão de Poisson* (média \neq variância)
- Variável dependente contável, com **Distribuição Binomial Negativa**

$$Y \sim NB(\mu, \alpha)$$

- $E[Y | x] = \mu$
- α : Parâmetro auxiliar
 - Pode ser buscado
 - ou calculado

Modelo Binomial Negativo

Modelo 1: Varredura pelo melhor valor de α

```
def nb_search_training(train_df, test_df, features=None, target='total_cases'):

    # Utilizar todas as variáveis independentes como features
    if features == None:
        features = train_df.columns[train_df.columns != target]

    # Especificar a formula do modelo
    formula = f'{target} ~ 1' # `1` no segundo termo é o intercepto
    if len(features) > 0:
        formula += ' + ' + ' + '.join(features)

    # Definir o range de busca do alfa
    grid = 10 ** np.arange(-10, 2, dtype=np.float64)

    # Iniciando valor min_error com valor alto
    min_error = 1000
    best_alpha = None
    best_results = None

    # Buscar pelo melhor parâmetro alfa
    for alpha in grid:
        # Construir o modelo e treinar
        model = smf.glm(formula=formula, data=train_df, family=sm.families.NegativeBinomial(alpha=alpha))
        results = model.fit()

        # Predizer valores do DF teste e calcular o erro
        predictions = results.predict(test_df).astype(int)
        error = error_model(test_df, predictions)

        # Verificar se o modelo possui o menor erro
        if error < min_error:
            best_alpha = alpha
            best_results = results
            min_error = error

    return best_results, best_alpha
```

Modelo Binomial Negativo

Modelo 1: Varredura pelo melhor valor de α

```
def nb_search_training(train_df, test_df, features=None, target='total_cases'):

    # Utilizar todas as variáveis independentes como features
    if features == None:
        features = train_df.columns[train_df.columns != target]

    # Especificar a formula do modelo
    formula = f'{target} ~ 1' # `1` no segundo termo é o intercepto
    if len(features) > 0:
        formula += ' + ' + ' + '.join(features)

    # Definir o range de busca do alfa
    grid = 10 ** np.arange(-10, 2, dtype=np.float64)

    # Iniciando valor min_error com valor alto
    min_error = 1000
    best_alpha = None
    best_results = None

    # Buscar pelo melhor parâmetro alfa
    for alpha in grid:
        # Construir o modelo e treinar
        model = smf.glm(formula=formula, data=train_df, family=sm.families.NegativeBinomial(alpha=alpha))
        results = model.fit()

        # Predizer valores do DF teste e calcular o erro
        predictions = results.predict(test_df).astype(int)
        error = error_model(test_df, predictions)

        # Verificar se o modelo possui o menor erro
        if error < min_error:
            best_alpha = alpha
            best_results = results
            min_error = error

    return best_results, best_alpha
```

Modelo Binomial Negativo

Modelo 1: Varredura pelo melhor valor de α

```
def nb_search_training(train_df, test_df, features=None, target='total_cases'):

    # Utilizar todas as variáveis independentes como features
    if features == None:
        features = train_df.columns[train_df.columns != target]

    # Especificar a fórmula do modelo
    formula = f'{target} ~ 1' # `1` no segundo termo é o intercepto
    if len(features) > 0:
        formula += ' + ' + ' + '.join(features)

    # Definir o range de busca do alfa
    grid = 10 ** np.arange(-10, 2, dtype=np.float64)

    # Iniciando valor min_error com valor alto
    min_error = 1000
    best_alpha = None
    best_results = None

    # Buscar pelo melhor parâmetro alfa
    for alpha in grid:
        # Construir o modelo e treinar
        model = smf.glm(formula=formula, data=train_df, family=sm.families.NegativeBinomial(alpha=alpha))
        results = model.fit()

        # Predizer valores do DF teste e calcular o erro
        predictions = results.predict(test_df).astype(int)
        error = error_model(test_df, predictions)

        # Verificar se o modelo possui o menor erro
        if error < min_error:
            best_alpha = alpha
            best_results = results
            min_error = error

    return best_results, best_alpha
```

Modelo Binomial Negativo

Modelo 1: Varredura pelo melhor valor de α

```
def nb_search_training(train_df, test_df, features=None, target='total_cases'):

    # Utilizar todas as variáveis independentes como features
    if features == None:
        features = train_df.columns[train_df.columns != target]

    # Especificar a fórmula do modelo
    formula = f'{target} ~ 1' # `1` no segundo termo é o intercepto
    if len(features) > 0:
        formula += ' + ' + ' + '.join(features)

    # Definir o range de busca do alfa
    grid = 10 ** np.arange(-10, 2, dtype=np.float64)

    # Iniciano valor min_error com valor alto
    min_error = 1000
    best_alpha = None
    best_results = None

    # Buscar pelo melhor parâmetro alfa
    for alpha in grid:
        # Construir o modelo e treinar
        model = smf.glm(formula=formula, data=train_df, family=sm.families.NegativeBinomial(alpha=alpha))
        results = model.fit()

        # Predizer valores do DF teste e calcular o erro
        predictions = results.predict(test_df).astype(int)
        error = error_model(test_df, predictions)

        # Verificar se o modelo possui o menor erro
        if error < min_error:
            best_alpha = alpha
            best_results = results
            min_error = error

    return best_results, best_alpha
```

Modelo Binomial Negativo

Modelo 1: Varredura pelo melhor valor de α

```
def nb_search_training(train_df, test_df, features=None, target='total_cases'):

    # Utilizar todas as variáveis independentes como features
    if features == None:
        features = train_df.columns[train_df.columns != target]

    # Especificar a fórmula do modelo
    formula = f'{target} ~ 1' # `1` no segundo termo é o intercepto
    if len(features) > 0:
        formula += ' + ' + ' + '.join(features)

    # Definir o range de busca do alfa
    grid = 10 ** np.arange(-10, 2, dtype=np.float64)

    # Iniciano valor min_error com valor alto
    min_error = 1000
    best_alpha = None
    best_results = None

    # Buscar pelo melhor parâmetro alfa
    for alpha in grid:
        # Construir o modelo e treinar
        model = smf.glm(formula=formula, data=train_df, family=sm.families.NegativeBinomial(alpha=alpha))
        results = model.fit()

        # Predizer valores do DF teste e calcular o erro
        predictions = results.predict(test_df).astype(int)
        error = error_model(test_df, predictions)

        # Verificar se o modelo possui o menor erro
        if error < min_error:
            best_alpha = alpha
            best_results = results
            min_error = error

    return best_results, best_alpha
```

Modelo Binomial Negativo

Modelo 2: Cálculo prévio de α

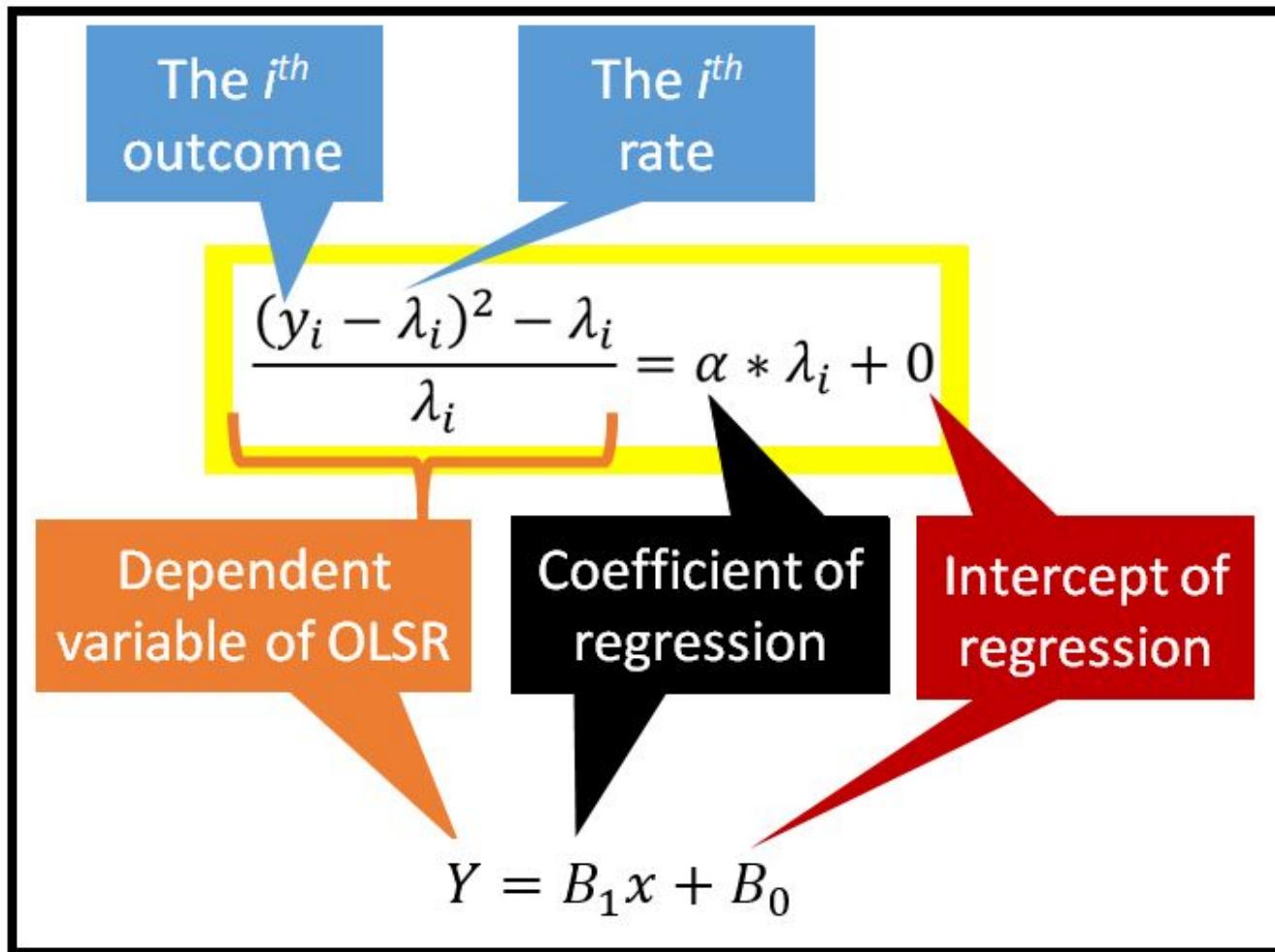
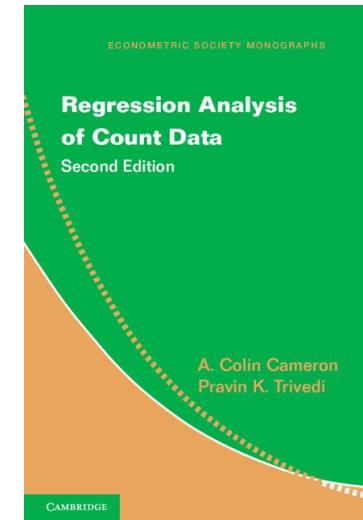


Imagen Sachin Date



Modelo Binomial Negativo

Modelo 2: Cálculo prévio de α

$$\frac{(y_i - \lambda_i)^2 - \lambda_i}{\lambda_i} = \alpha * \lambda_i + 0$$

```
def nb_calculate_training(train_df, features=None, target='total_cases'):

    # Utilizar todas as variáveis independentes como features
    if features == None:
        features = train_df.columns[train_df.columns != target]

    # Especificar a formula do modelo
    formula = f'{target} ~ 1' # `1` no segundo termo é o intercepto
    if len(features) > 0:
        formula += ' + ' + ' + '.join(features)

    # Obter o vetor lambda
    rate = lambda_vector(train_df, formula)
    # Obter o termo dependente da equação de regressão OLS
    ols_y = ols_dependent(train_df[target], rate)
    ols_df = pd.DataFrame({
        'rate': rate,
        'ols_y': ols_dependent(train_df[target], rate)
    })

    # Obter alpha a partir da regressão OLS
    alpha = ols_regression(ols_df, 'ols_y', 'rate')

    # Treinar o modelo de regressão utilizando a distribuição binomial negativa
    nb_train_model = smf.glm(formula=formula, data=train_df, family=sm.families.NegativeBinomial(alpha=alpha))
    nb_train_results = nb_train_model.fit()

    return nb_train_results, alpha
```

Modelo Binomial Negativo

Modelo 2: Cálculo prévio de α

$$\frac{(y_i - \lambda_i)^2 - \lambda_i}{\lambda_i} = \alpha * \lambda_i + 0$$

```
def nb_calculate_training(train_df, features=None, target='total_cases'):  
  
    # Utilizar todas as variáveis independentes como features  
    if features == None:  
        features = train_df.columns[train_df.columns != target]  
  
    # Especificar a formula do modelo  
    formula = f'{target} ~ 1' # `1` no segundo termo é o intercepto  
    if len(features) > 0:  
        formula += ' + ' + ' + '.join(features)  
  
    # Obter o vetor lambda  
    rate = lambda_vector(train_df, formula)  
    # Obter o termo dependente da equação de regressão OLS  
    ols_y = ols_dependent(train_df[target], rate)  
    ols_df = pd.DataFrame({  
        'rate': rate,  
        'ols_y': ols_dependent(train_df[target], rate)  
    })  
  
    # Obter alpha a partir da regressão OLS  
    alpha = ols_regression(ols_df, 'ols_y', 'rate')  
  
    # Treinar o modelo de regressão utilizando a distribuição binomial negativa  
    nb_train_model = smf.glm(formula=formula, data=train_df, family=sm.families.NegativeBinomial(alpha=alpha))  
    nb_train_results = nb_train_model.fit()  
  
    return nb_train_results, alpha
```

```
def lambda_vector(train_df, formula):  
    # Treinar o modelo de regressão utilizando Poisson  
    poisson_train_model = smf.glm(formula=formula, data=train_df, family=sm.families.Poisson())  
    poisson_train_results = poisson_train_model.fit()  
  
    return poisson_train_results.mu
```

Modelo Binomial Negativo

Modelo 2: Cálculo prévio de α

$$\frac{(y_i - \lambda_i)^2 - \lambda_i}{\lambda_i} = \alpha * \lambda_i + 0$$

```
def nb_calculate_training(train_df, features=None, target='total_cases'):  
  
    # Utilizar todas as variáveis independentes como features  
    if features == None:  
        features = train_df.columns[train_df.columns != target]  
  
    # Especificar a formula do modelo  
    formula = f'{target} ~ 1' # `1` no segundo termo é o intercepto  
    if len(features) > 0:  
        formula += ' + ' + ' + '.join(features)  
  
    # Obter o vetor lambda  
    rate = lambda_vector(train_df, formula)  
    # Obter o termo dependente da equação de regressão OLS  
    ols_y = ols_dependent(train_df[target], rate)  
    ols_df = pd.DataFrame({  
        'rate': rate,  
        'ols_y': ols_dependent(train_df[target], rate)  
    })  
  
    # Obter alpha a partir da regressão OLS  
    alpha = ols_regression(ols_df, 'ols_y', 'rate')  
  
    # Treinar o modelo de regressão utilizando a distribuição binomial negativa  
    nb_train_model = smf.glm(formula=formula, data=train_df, family=sm.families.NegativeBinomial(alpha=alpha))  
    nb_train_results = nb_train_model.fit()  
  
    return nb_train_results, alpha
```

```
def lambda_vector(train_df, formula):  
    # Treinar o modelo de regressão utilizando Poisson  
    poisson_train_model = smf.glm(formula=formula, data=train_df, family=sm.families.Poisson())  
    poisson_train_results = poisson_train_model.fit()  
  
    return poisson_train_results.mu
```

```
def ols_dependent(target_array, lambda_array):  
    return ((target_array - lambda_array)**2 - lambda_array) / lambda_array
```

Modelo Binomial Negativo

Modelo 2: Cálculo prévio de α

$$\frac{(y_i - \lambda_i)^2 - \lambda_i}{\lambda_i} = \alpha * \lambda_i + 0$$

```
def nb_calculate_training(train_df, features=None, target='total_cases'):  
  
    # Utilizar todas as variáveis independentes como features  
    if features == None:  
        features = train_df.columns[train_df.columns != target]  
  
    # Especificar a formula do modelo  
    formula = f'{target} ~ 1' # `1` no segundo termo é o intercepto  
    if len(features) > 0:  
        formula += ' + ' + ' + '.join(features)  
  
    # Obter o vetor lambda  
    rate = lambda_vector(train_df, formula)  
    # Obter o termo dependente da equação de regressão OLS  
    ols_y = ols_dependent(train_df[target], rate)  
    ols_df = pd.DataFrame({  
        'rate': rate,  
        'ols_y': ols_dependent(train_df[target], rate)  
    })  
  
    # Obter alpha a partir da regressão OLS  
    alpha = ols_regression(ols_df, 'ols_y', 'rate')  
  
    # Treinar o modelo de regressão utilizando a distribuição binomial negativa  
    nb_train_model = smf.glm(formula=formula, data=train_df, family=sm.families.NegativeBinomial(alpha=alpha))  
    nb_train_results = nb_train_model.fit()  
  
    return nb_train_results, alpha
```

```
def lambda_vector(train_df, formula):  
    # Treinar o modelo de regressão utilizando Poisson  
    poisson_train_model = smf.glm(formula=formula, data=train_df, family=sm.families.Poisson())  
    poisson_train_results = poisson_train_model.fit()  
  
    return poisson_train_results.mu
```

```
def ols_dependent(target_array, lambda_array):  
    return ((target_array - lambda_array)**2 - lambda_array) / lambda_array
```

```
def ols_regression(ols_df, ols_y, rate):  
    # Especificar a fórmula do modelo  
    # `-1` no final é para indicar intercepto = 0  
    formula = f'{ols_y} ~ {rate} - 1'  
  
    # Treinar modelo de regressão OLS  
    ols_train_model = smf.ols(formula=formula, data=ols_df)  
    ols_train_results = ols_train_model.fit()  
  
    # Valor de alpha é retornado (coeficiente da regressão)  
    return ols_train_results.params[0]
```

Modelo Binomial Negativo

Atributos do modelo

- **Versão 0 - Nenhum atributo:** avaliar como o modelo se comporta sem nenhuma variável independente, apenas com os valores do rótulo

Modelo Binomial Negativo

Atributos do modelo

- **Versão 0 - Nenhum atributo:** avaliar como o modelo se comporta sem nenhuma variável independente, apenas com os valores do rótulo
- **Versão 1 - Todos atributos :** avaliação considerando toda informação de entrada disponível (com exceção de `week_start_date`)

Modelo Binomial Negativo

Atributos do modelo

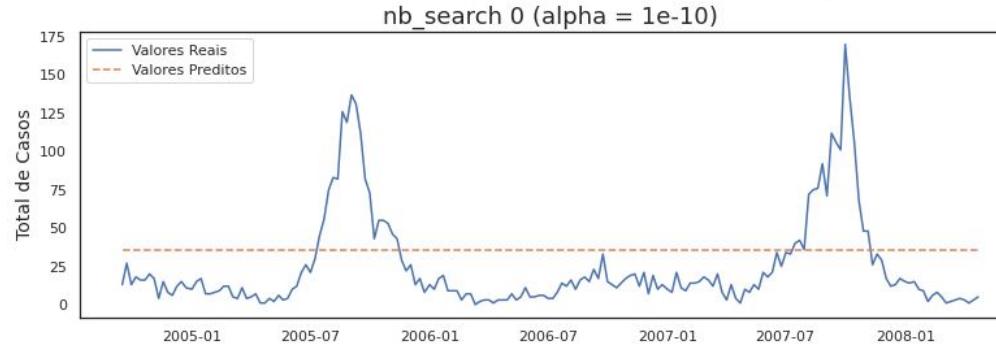
- **Versão 0 - Nenhum atributo:** avaliar como o modelo se comporta sem nenhuma variável independente, apenas com os valores do rótulo
- **Versão 1 - Todos atributos :** avaliação considerando toda informação de entrada disponível (com exceção de `week_start_date`)
- **Versão 2 - Subconjunto de atributos:** a partir da análise exploratória, utilizando os 4 atributos mais fortemente correlacionados com o total de casos:
 - `reanalysis_specific_humidity_g_per_kg`: Umidade específica média (g/kg)
 - `reanalysis_dew_point_temp_k`: Temperatura média do ponto de orvalho (K)
 - `station_avg_temp_k`: Temperatura média (K)
 - `station_min_temp_k`: Temperatura mínima (K)

Modelo Binomial Negativo - San Juan

V0

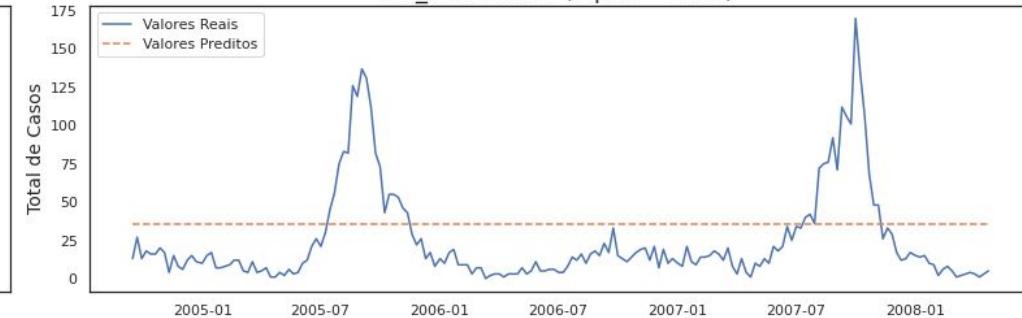
Modelo 1

SJ - Comparação dos modelos com os valores de treino

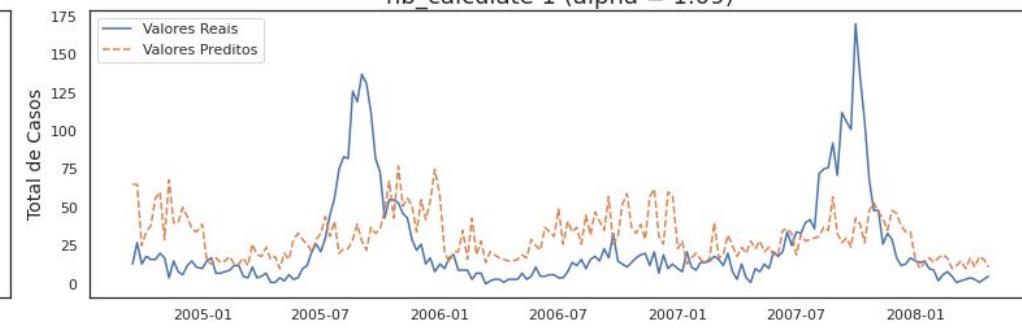
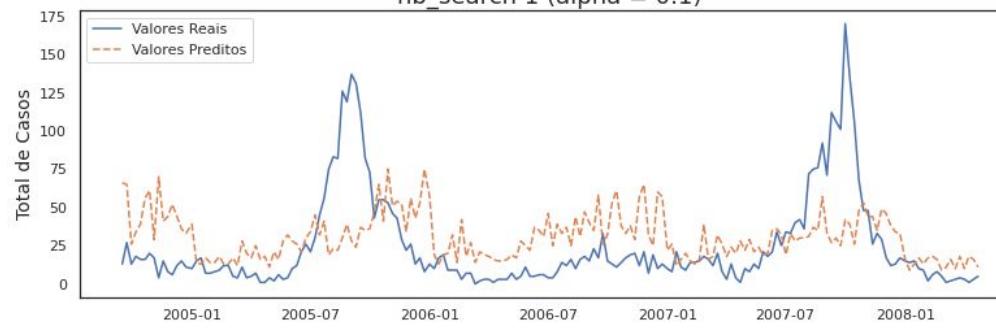


Modelo 2

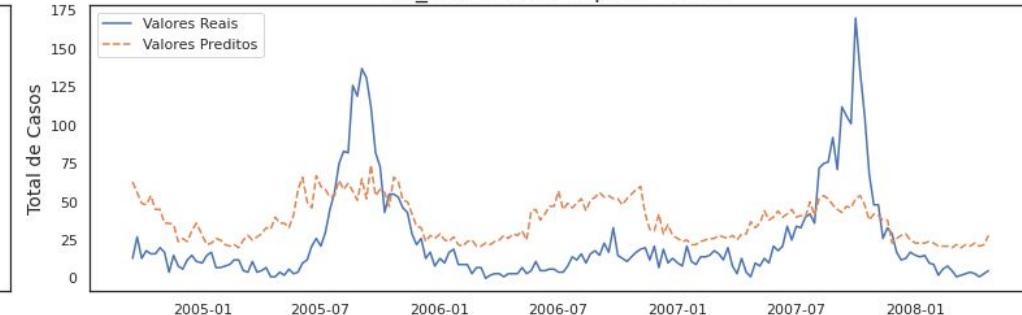
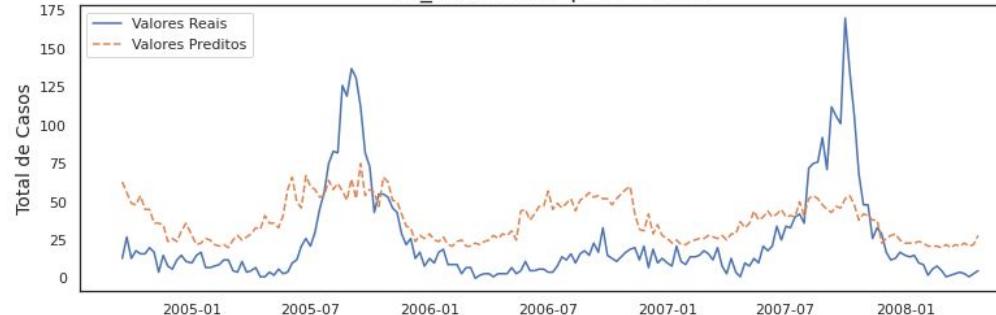
nb_calculate 0 ($\alpha = 2.24$)



V1



V2



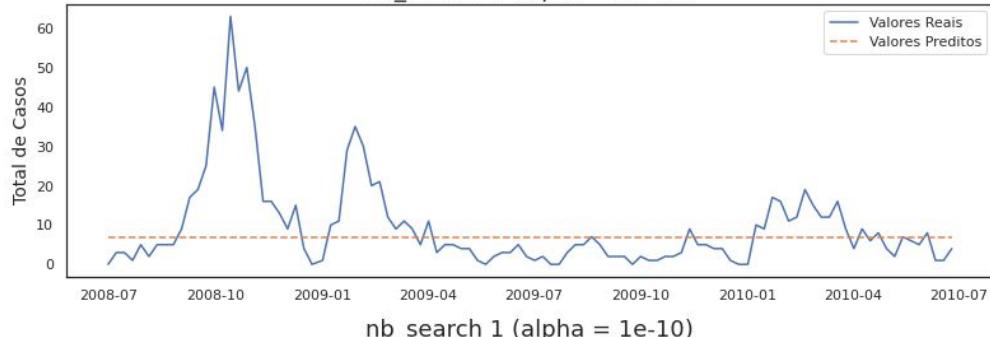
Modelo Binomial Negativo - Iquitos

V0

Modelo 1

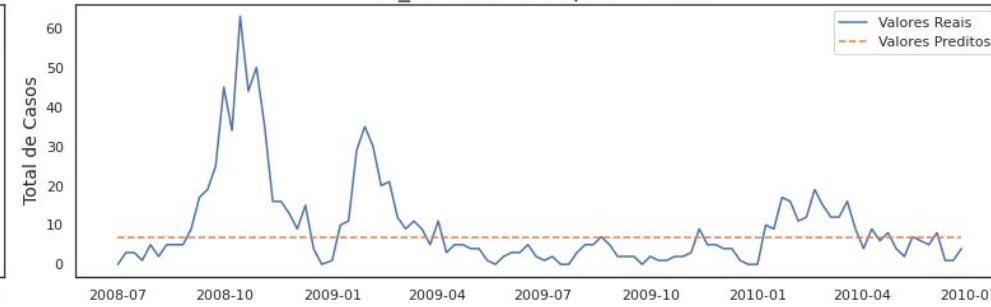
IQ - Comparação dos modelos com os valores de treino

nb_search 0 ($\alpha = 1e-10$)



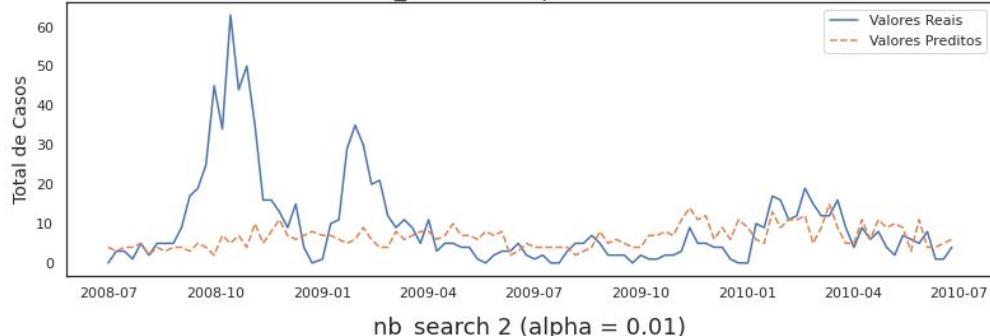
Modelo 2

nb_calculate 0 ($\alpha = 2.07$)

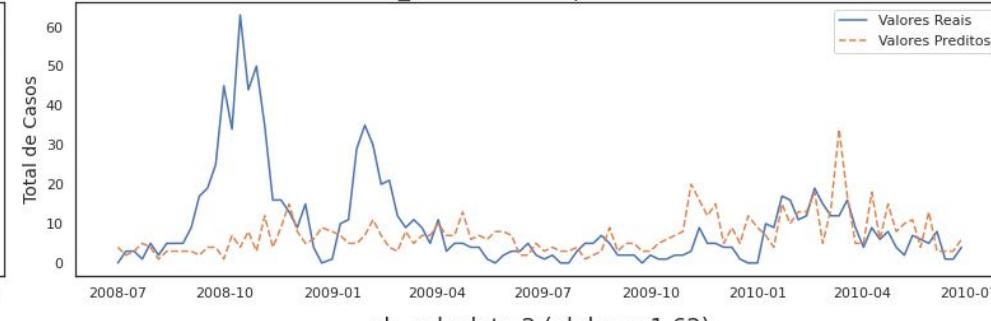


V1

nb_search 1 ($\alpha = 1e-10$)

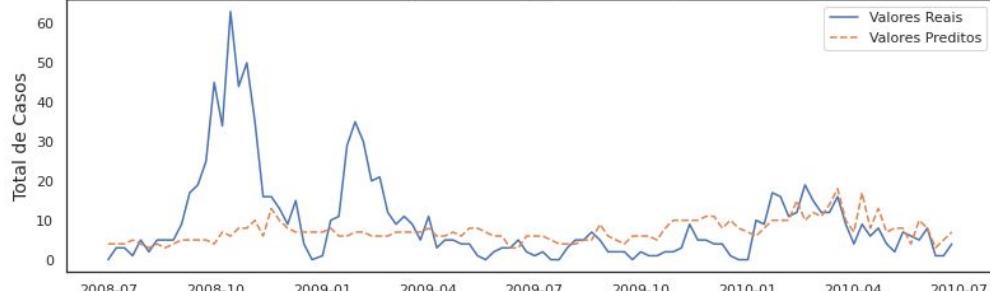


nb_calculate 1 ($\alpha = 1.47$)

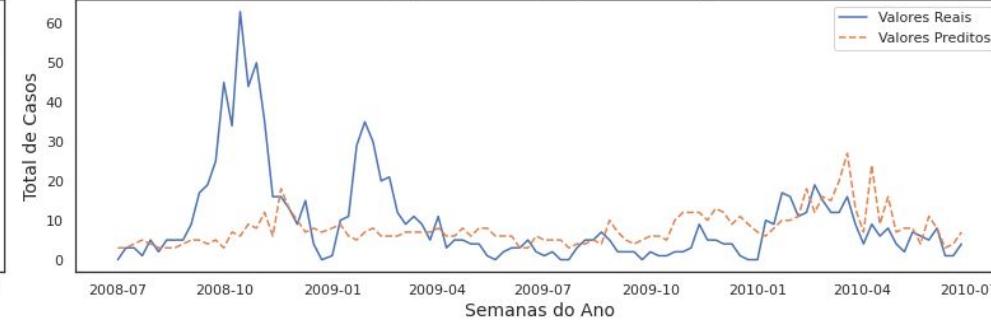


V2

nb_search 2 ($\alpha = 0.01$)



nb_calculate 2 ($\alpha = 1.62$)



Avaliação dos modelos e Submissão

	San Juan	Iquitos	Geral
baseline	22.82	6.71	17.08
nb_search 0	27.85	7.27	20.52
nb_calculate 0	27.85	7.27	20.52
nb_search 1	23.08	7.21	17.43
nb_calculate 1	23.13	7.52	17.57
nb_search 2	24.27	6.82	18.05
nb_calculate 2	24.28	7.17	18.18

Avaliação dos modelos e Submissão

	San Juan	Iquitos	Geral
baseline	22.82	6.71	17.08
nb_search 0	27.85	7.27	20.52
nb_calculate 0	27.85	7.27	20.52
nb_search 1	23.08	7.21	17.43
nb_calculate 1	23.13	7.52	17.57
nb_search 2	24.27	6.82	18.05
nb_calculate 2	24.28	7.17	18.18

Avaliação dos modelos e Submissão

Baseline

Woohoo! We processed your submission!

Your score for this submission is:

25.8870

nb_search 1

Woohoo! We processed your submission!

Your score for this submission is:

26.6442

nb_calculate 1

Woohoo! We processed your submission!

Your score for this submission is:

26.7620

nb_search 2

Woohoo! We processed your submission!

Your score for this submission is:

25.7115

nb_calculate 2

Woohoo! We processed your submission!

Your score for this submission is:

25.9255

Avaliação dos modelos e Submissão

Baseline

Woohoo! We processed your submission!

Your score for this submission is:

25.8870

nb_search 1

Woohoo! We processed your submission!

Your score for this submission is:

26.6442

nb_calculate 1

Woohoo! We processed your submission!

Your score for this submission is:

26.7620

nb_search 2

Woohoo! We processed your submission!

Your score for this submission is:

25.7115

nb_calculate 2

Woohoo! We processed your submission!

Your score for this submission is:

25.9255



Conclusão

- **Modelo Binomial Negativo:** Não existe uma diferença significativa no MAE entre os valores de α calculado e estimado. Uma diferença relevante é em relação ao tempo de processamento, a tabela ao lado mostra o tempo médio de processamento do método de busca (*nb_search*) e do método calculado (*nb_calculate*).

Tempo Médio	
nb_search 0	150
nb_calculate 0	29
nb_search 1	1779
nb_calculate 1	290
nb_search 2	360
nb_calculate 2	52

Conclusão

- **Modelo Binomial Negativo:** Não existe uma diferença significativa no MAE entre os valores de α calculado e estimado. Uma diferença relevante é em relação ao tempo de processamento, a tabela ao lado mostra o tempo médio de processamento do método de busca (*nb_search*) e do método calculado (*nb_calculate*).
- **Atributos:** A redução da quantidade de atributo melhorou o valor encontrado de MAE, indicando a possibilidade que a maior quantidade de atributo acaba prejudicando a performance do modelo.

Tempo Médio	
<i>nb_search 0</i>	150
<i>nb_calculate 0</i>	29
<i>nb_search 1</i>	1779
<i>nb_calculate 1</i>	290
<i>nb_search 2</i>	360
<i>nb_calculate 2</i>	52

Conclusão

- **Modelo Binomial Negativo:** Não existe uma diferença significativa no MAE entre os valores de α calculado e estimado. Uma diferença relevante é em relação ao tempo de processamento, a tabela ao lado mostra o tempo médio de processamento do método de busca (*nb_search*) e do método calculado (*nb_calculate*).
- **Atributos:** A redução da quantidade de atributo melhorou o valor encontrado de MAE, indicando a possibilidade que a maior quantidade de atributo acaba prejudicando a performance do modelo.
- **Modelo Baseline:** O modelo AR com parâmetro sazonal teve a melhor performance nos dados supervisionados, mas quando exposto à dados não supervisionados teve um resultado levemente inferior ao *nb_search* 2. Isso pode ser um indício que o modelo se ajustou à base de treino/teste mas não reflete os resultados em uma base desconhecida.

Tempo Médio	
<i>nb_search</i> 0	150
<i>nb_calculate</i> 0	29
<i>nb_search</i> 1	1779
<i>nb_calculate</i> 1	290
<i>nb_search</i> 2	360
<i>nb_calculate</i> 2	52

Conclusão

- **Modelo Binomial Negativo:** Não existe uma diferença significativa no MAE entre os valores de α calculado e estimado. Uma diferença relevante é em relação ao tempo de processamento, a tabela ao lado mostra o tempo médio de processamento do método de busca (*nb_search*) e do método calculado (*nb_calculate*).
- **Atributos:** A redução da quantidade de atributo melhorou o valor encontrado de MAE, indicando a possibilidade que a maior quantidade de atributo acaba prejudicando a performance do modelo.
- **Modelo Baseline:** O modelo AR com parâmetro sazonal teve a melhor performance nos dados supervisionados, mas quando exposto à dados não supervisionados teve um resultado levemente inferior ao *nb_search* 2. Isso pode ser um indício que o modelo se ajustou à base de treino/teste mas não reflete os resultados em uma base desconhecida.
- **Oportunidade:** Treinar com diferentes combinações de atributos, procurar outros modelos

Tempo Médio	
<i>nb_search</i> 0	150
<i>nb_calculate</i> 0	29
<i>nb_search</i> 1	1779
<i>nb_calculate</i> 1	290
<i>nb_search</i> 2	360
<i>nb_calculate</i> 2	52

CURRENT RANK	# COMPETITORS
1424	10744



Links úteis

Dados

- [DrivenData](#)
- [Índice de Vegetação por Diferença Normalizada \(NDVI\)](#)
- [NOAA CDR](#)
- [NOAA GHCN-Daily](#)
- [NOAA NCEP Climate Forecast System Reanalysis](#)
- [PERSIANN](#)
- [Umidade Relativa do Ar](#)
- [Umidade Especifica](#)

Python

- [AutoReg](#)
- [Fitting Models using R-style Formulas](#)
- [GLM](#)
- [Interpolate \(Pandas\)](#)
- [Negative Binomial](#)

Links úteis

Teoria

- [Coeficiente de correlação de Pearson](#)
- [Functional forms for the negative binomial model for count data](#)
- [Mean Absolute Error](#)
- [Negative binomial distribution](#)
- [Negative Binomial Regression](#)
- [Negative Binomial Regression: A Step by Step Guide](#)
- [Ordinary least squares](#)
- [Poisson Regression](#)
- [Time Series Analysis: Identifying AR and MA using ACF and PACF Plots](#)
- [Zero-inflated model](#)