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

**Jorge Luis Zanguettin**

# ALGORITMO DE MACHINE LEARNING PARA A PREDIÇÃO DE OCORRÊNCIA DA SPODOPTERA FRUGIPERDA NA LAVOURA DO ALGODÃO

Belo Horizonte
2021

**Jorge Luis Zanguettin**

# ALGORITMO DE MACHINE LEARNING PARA A PREDIÇÃO DE OCORRENCIA DA SPODOPTERA FRUGIPERDA NA LAVOURA DO ALGODÃO

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

Quando se fala de agronegócio, é extremamente comum associar o termo somente a produção in natura, como grãos, leite, frutas e hortaliças por exemplo, no entanto, o termo vai muito além da produção. O termo agronegócio é utilizado para fazer referência ao contexto da produção agropecuária, incluindo todos as técnicas, serviços e equipamentos diretamente ou indiretamente associados. O agronegócio é extremamente importante para o PIB do Brasil, segundo dados do CEPEA, somente o setor movimentou 26,6% do PIB brasileiro, sendo equivalente a quase 2 trilhões de reais somente no ano de 2020. Segundo a Embrapa, o Brasil ocupa o 5° lugar entre os maiores países produtores de algodão e em exportação, o Brasil ocupa o segundo maior exportador.

Dentre as diversas empresas, a Holambra Agrícola é uma empresa voltada para esse mercado e tem como principal objetivo a recepção, armazenagem e comercialização de cereais e algodão, utilizando diversas técnicas e visando sempre uma boa prestação de serviços atrelados a custos satisfatórios aos produtores associados. A empresa implementou uma rotina de monitoramento em algumas glebas de Algodão, localizadas nas fazendas de seus clientes, com o intuito de monitorar a ocorrência de diversas pragas e fazer o controle populacional das pragas. Tendo início em 2018, as glebas são monitoradas diariamente até os dias atuais, contendo dados de ocorrência das pragas, dos pontos monitorados e estágio da praga.

*Machine Learning* (ML) ou Aprendizado de Máquina é uma área da *Artificial Inteligence* (AI) ou Inteligência Artificial que tem como objetivo o foco na construção de algoritmos que consigam aprender de forma autônoma tendo como consequência, a capacidade de reconhecer e extrair padrões de grandes volumes de dados.

Os tipos de aprendizado de máquina atualmente são quatro: aprendizado supervisionado, onde o desenvolvedor do algoritmo fornece dados de treinamento que contenham as soluções desejadas (também conhecidas como rótulos) ao algoritmo; aprendizado não supervisionado, diferente do supervisionado, neste aprendizado os dados de treinamento são entregues ao algoritmo sem os rótulos; aprendizado semi-supervisionado, onde os algoritmos trabalham com dados de treinamento parcialmente rotulados, tendo na maioria dos casos, uma grande parte dos dados não rotulados e uma

pequena parte dos dados rotulados; aprendizado por reforço, onde o algoritmo (também pode ser chamado de agente, neste contexto) pode analisar o ambiente em que ele foi inserido e realizar ações mediante a recompensas ou penalidades, o principal objetivo deste tipo de aprendizado é a obtenção da melhor estratégia por parte do agente de forma autônoma.

No presente trabalho, foram implementados e comparados algoritmos de Aprendizagem Supervisionada focadas em classificação através da linguagem de programação *Python*, com suas bibliotecas *Scikit-Learn*, *Pandas* e *Numpy*, para o desenvolvimento foi utilizado a *IDE* iterativa *Jupyter Notebook* e para o armazenamento de todo o código fonte foi utilizado um repositório do GitHub do autor.

### 1.2. O problema proposto

No algodoeiro, existem diversas pragas que podem causar danos severos a lavoura, a *Spodoptera frugiperda* ou lagarta-do-cartucho pode ser considerada a principal praga-alvo não só no algodão, mas também em diversas outras culturas, isso acontece por conta da sua ampla distribuição temporal e geográfica, constituindo uma das espécies mais nocivas tropicais das Américas.

Utilizando os dados fornecidos pela Holambra Agrícola de monitoramento das pragas juntamente com os dados climáticos da região obtidos da plataforma *OpenWeatherMap* o presente trabalho tem como objetivo prever a ocorrência da praga lagarta-do-cartucho através da construção de um algoritmo classificatório de *Machine Learning* com aprendizagem supervisionada utilizando *Python* e suas bibliotecas de apoio.

Serão utilizados os últimos 3 anos de safra das diferentes propriedades e glebas apresentadas na base de dados de monitoramento concedida pela Holambra Agrícola.

### 1.3. Objetivos

O presente trabalho tem como objetivo analisar e correlacionar as bases de dados fornecidas pela Holambra Agrícola, sendo elas o banco de dados de monitoramento de pragas em glebas de Algodão e dados meteorológicos obtidos da plataforma *OpenWeatherMap*, comparando algoritmos de *Machine Learning*, utilizando o aprendizado supervisionado, para prever a ocorrência da praga *Spodoptera Frugiperda* na cultura do

algodão levando em consideração o histórico de ocorrências e o histórico meteorológico das lavouras.

## 2. Coleta de Dados

O conjunto analisado contém dados de ocorrências da praga *Spodoptera Frugiperda* em 71 propriedades por 356 dias não sequenciais. Cada propriedade foi monitorada manualmente em pontos aleatórios distribuídos dentro da Gleba. Este conjunto foi enviado via *E-mail*, sendo composto originalmente pelos campos apresentados na Tabela 1.

**Tabela 1 – Características do conjunto de dados de ocorrências.**

| Nome da coluna/campo | Descrição | Tipo |
|---|---|---|
| DATA | Data da coleta da ocorrência | Date |
| NOME DO PRODUTOR | Nome do produtor da propriedade | String |
| PROPRIEDADE | Nome da propriedade | String |
| GLEBA | Nome da gleba presente na propriedade | String |
| LAG S FRUGIPERDA | Ocorrência da praga *Spodoptera frugiperda* | String |
| DATA PLANTIO | Data do plantio da lavoura | Date |

O conjunto de dados secundário utilizado neste trabalho foi obtido pela plataforma *OpenWeatherMap*, que disponibilizou uma base de dados meteorológica da região de Holambra II, cidade onde se encontram as propriedades presentes no conjunto anterior. Este conjunto é composto pelos campos apresentados na Tabela 2.

**Tabela 2 – Características do conjunto de dados climáticos.**

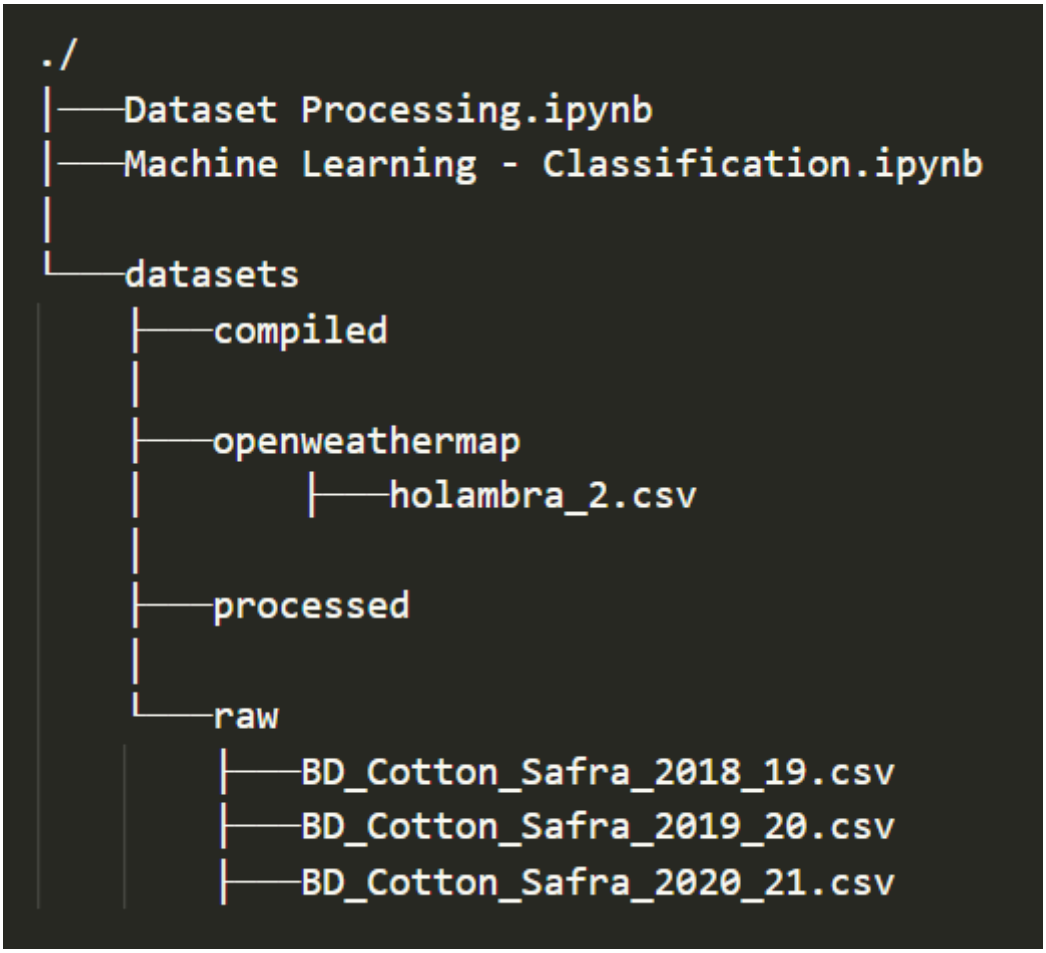
| Nome da coluna/campo | Descrição | Tipo |
|---|---|---|
| DATA | Data da coleta da temperatura | Date |
| TEMP | Temperatura | Double |
| TEMP_MIN | Temperatura Mínima | Double |
| TEMP_MAX | Temperatura Máxima | Double |
| PREASSURE | Pressão Atmosférica (ao nível do mar) em hPa | Double |
| HUMIDITY | Porcentagem de Umidade | Double |

| | | |
|---|---|---|
| WIND_SPEED | Velocidade do vento em M/S | Double |
| WIND_DEG | Direção do vento em Graus (meteorológicos) | Double |
| CLOUDS_ALL | Porcentagem de céu nublado | Double |

## 3. Processamento/Tratamento de Dados

Inicialmente, foi criado uma hierarquia de pastas no projeto para uma melhor organização dos arquivos presentes nesse trabalho.

**Figura 1 – Hierarquia de pastas do projeto.**




Como mostrado na Figura 1, o projeto possui diversos diretorios, onde:

- *Dataset Processing.ipynb* – *Jupyter Notebook* responsavel pela limpeza, balanceamento e compilação dos conjuntos de dados;
- *Machine Learning – Jupyter Notebook* responsavel pela comparação e obtenção das métricas dos modelos de *Machine Learning* de Classificação;
- *Datasets/raw* – Diretório onde se encontram os conjuntos de dados brutos;
- *Datasets/processed* – Diretório onde se encontram os conjuntos de dados tratados e balanceados;
- *Datasets/openweathermap* – Diretório onde se encotra o conjunto de dados climaticos;

- *Datasets/compiled* – Diretório onde se encontra o conjunto de dados de ocorrencias compilado com o conjunto de dados climaticos.

Utilizando pandas, no arquivo *"Dataset Processing.ipynb"*, foi decidido unir todos os arquivos contendo as ocorrencias para facilitar os procedimentos futuros. Utilizando o trecho de código a seguir, os arquivos foram compilados em apenas um arquivo chamado "BD_Cotton_Safra_2018_21.csv" encontrado no diretorio "/datasets/compiled", como mostrado na Figura 2.

**Figura 2 – Merge dos conjuntos de dados de ocorrência.**

```
In [4]: files = [
            'BD_Cotton_Safra_2018_19',
            'BD_Cotton_Safra_2019_20',
            'BD_Cotton_Safra_2020_21',
        ]

In [6]: dfCompiled = pd.DataFrame()

        for filename in files:
            print ('Compilação ->',filename)

            df = pd.read_csv(f'./datasets/raw/{filename}.csv', delimiter = ';', low_memory=False)
            dfCompiled = pd.concat([dfCompiled, df])

        dfCompiled.to_csv('./datasets/compiled/BD_Cotton_Safra_2018_21.csv', sep=';', index=False)

        Compilação -> BD_Cotton_Safra_2018_19
        Compilação -> BD_Cotton_Safra_2019_20
        Compilação -> BD_Cotton_Safra_2020_21
```

Após a realização do merge, o conjunto de dados de ocorrências passou a ser apenas um arquivo contendo todas as ocorrências dos 3 arquivos inicialmente obtidos. O conjunto de dados de ocorrências apresentou as características apresentadas na Figura 3.

**Figura 3 – Características do conjunto de dados de ocorrências após merge.**

Out[7]:

| | DATA | Nome do Produtor | Propriedade | GLEBA | Lag S frugiperda | DATA PLANTIO |
|---|---|---|---|---|---|---|
| 0 | 2018-10-15 | Jacobus Derks | Santa Fé | P3 | 0 | 2018-10-03 |
| 1 | 2018-10-15 | Jacobus Derks | Santa Fé | P3 | 0 | 2018-10-03 |
| 2 | 2018-10-15 | Jacobus Derks | Santa Fé | P3 | 0 | 2018-10-03 |
| 3 | 2018-10-15 | Jacobus Derks | Santa Fé | P3 | 0 | 2018-10-03 |
| 4 | 2018-10-15 | Jacobus Derks | Santa Fé | P3 | 0 | 2018-10-03 |
| ... | ... | ... | ... | ... | ... | ... |
| 528487 | 2021-12-01 | José Theodoro Swart | NS do Carmo | C4 cima | 0 | 2021-10-21 |
| 528488 | 2021-12-01 | José Theodoro Swart | NS do Carmo | C4 cima | 0 | 2021-10-21 |
| 528489 | 2021-12-01 | José Theodoro Swart | NS do Carmo | C4 cima | 0 | 2021-10-21 |
| 528490 | 2021-12-01 | José Theodoro Swart | NS do Carmo | C4 cima | 0 | 2021-10-21 |
| 528491 | 2021-12-01 | José Theodoro Swart | NS do Carmo | C4 cima | 0 | 2021-10-21 |

528492 rows × 6 columns

```
In [8]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 528492 entries, 0 to 528491
Data columns (total 6 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   DATA              528492 non-null  datetime64[ns]
 1   Nome do Produtor  528492 non-null  object
 2   Propriedade       528492 non-null  object
 3   GLEBA             528492 non-null  object
 4   Lag S frugiperda  528492 non-null  object
 5   DATA PLANTIO      528492 non-null  datetime64[ns]
dtypes: datetime64[ns](2), object(4)
memory usage: 24.2+ MB
```

Observando as informações obtidas pela função *"df.info()"* do Pandas, nota-se que o conjunto de dados de ocorrências não havia linhas nulas, não sendo necessário nenhum procedimento. Entretanto, o mesmo se encontrava extremamente desbalanceado, pois no conjunto total havia 528.492 linhas e dentre elas, somente 6.943 ocorrências da praga aqui analisada (Figura 4), portanto, optou-se pela mudança no *layout* dos dados, visando um balanceamento das classes antes do início do desenvolvimento dos modelos de classificação.

**Figura 4 – Características das classes presentes no conjunto de dados de classificação.**

```
In [6]: df['Lag S frugiperda'].value_counts()

Out[6]: 0      521549
        P        2478
        M        2311
        G        1525
        ovo       628
        p           1
        Name: Lag S frugiperda, dtype: int64
```

A abordagem escolhida para que o conjunto de dados se tornasse mais balanceado, foi agrupar os monitoramentos por propriedade e por data. Ou seja, ao invés de ter uma linha para cada ponto monitorado dentro da propriedade (Figura 5), uma linha irá conter a somatória de ocorrências na propriedade no dia em questão.

**Figura 5 – Características dos registros presentes no conjunto de dados de classificação**

| | DATA | Nome do Produtor | Propriedade | GLEBA | Lag S frugiperda | DATA PLANTIO |
|---|---|---|---|---|---|---|
| 0 | 2018-10-15 | Jacobus Derks | Santa Fé | P3 | 0 | 2018-10-03 |
| 1 | 2018-10-15 | Jacobus Derks | Santa Fé | P3 | 0 | 2018-10-03 |
| 2 | 2018-10-15 | Jacobus Derks | Santa Fé | P3 | 0 | 2018-10-03 |
| 3 | 2018-10-15 | Jacobus Derks | Santa Fé | P3 | 0 | 2018-10-03 |
| 4 | 2018-10-15 | Jacobus Derks | Santa Fé | P3 | 0 | 2018-10-03 |

Conferindo se o balanceamento foi feito corretamente, na Figura 6 conseguimos perceber que os dados ficaram mais homogêneos e a diferença quantitativa das classes não é mais tão significativa quanto anteriormente.

**Figura 5 – Características das classes presentes no conjunto de dados de classificação.**

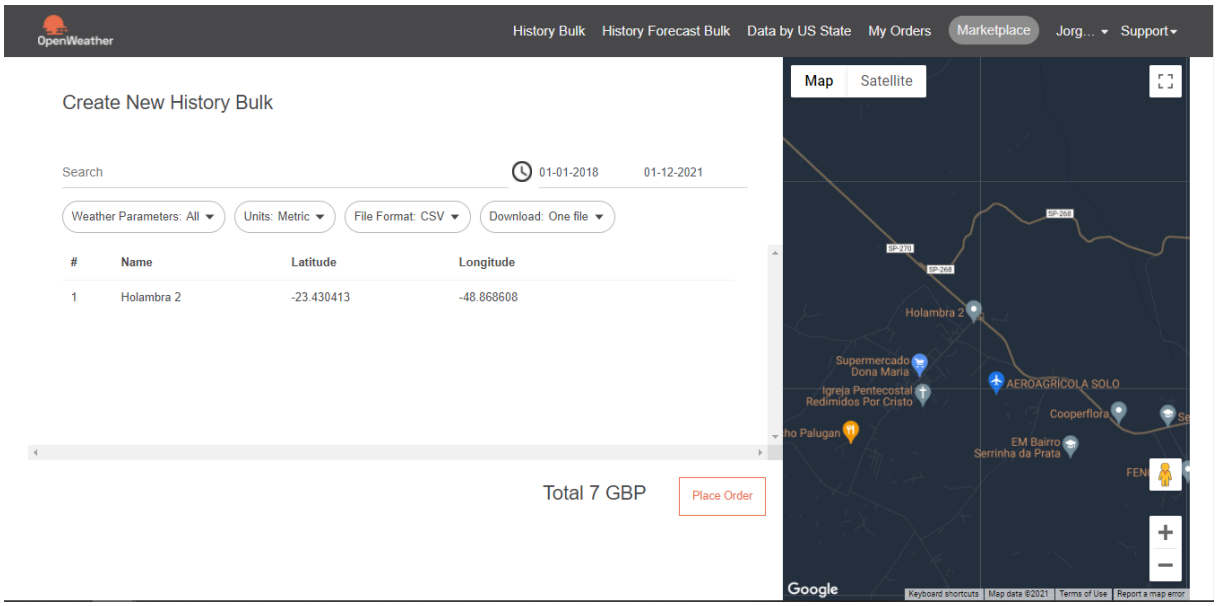
```
print (df_occurrence)
df_occurrence['ocorrencia'].value_counts()
```

```
            estagio  idade  ocorrencia
data
2018-10-15        0     12           0
2018-10-15        0      8           0
2018-10-15        0      6           0
2018-10-16        0      8           0
2018-10-16        0      5           0
...             ...    ...         ...
2021-12-01        0     35           0
2021-12-01        1     39           1
2021-12-01        1     41           0
2021-12-01        0     15           0
2021-12-01        1     41           0

[5724 rows x 3 columns]

0    3413
1    2311
Name: ocorrencia, dtype: int64
```

Após o balanceamento das classes, iniciou-se um estudo para entender qual outro conjunto de dados poderia compor a *Machine Learning*. Como o conjunto de dados aborda a ocorrência da *Spodoptera Frugiperda* no algodão, após pesquisas sobre a praga, foi identificado que o clima influencia diretamente no desenvolvimento da praga.

Segundo o site "*AgroLink*", "As lagartas recém eclodidas raspam as folhas e se alojam no cartucho, onde se observa seus excrementos. Pela destruição do cartucho, principalmente na fase próxima ao florescimento, podem causar danos expressivos que se acentuam em períodos de seca.". Com isso, foi adquirido pela plataforma *OpenWeatherMap*, um segundo conjunto de dados contendo dados climáticos de Holambra II, cidade onde as propriedades estão localizadas (Figura 6).

**Figura 5 – Características das classes presentes no conjunto de dados de classificação.**




Abrindo o conjunto de dados climáticos no *Pandas*, sendo o arquivo com o nome de "Holambra_2.csv" no diretório *"datasets/openweathermap/"*, nota-se que ele não possui valores nulos, ou seja, não foi necessário realizar nenhum procedimento implicando nessa situação, conforme mostrado na Figura 6.

**Figura 6 – Características das classes presentes no conjunto de dados climáticos.**

| | dt_iso | temp | temp_min | temp_max | pressure | humidity | wind_speed | wind_deg | clouds_all |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 2017-01-01 00:00:00 | 23.70 | 21.96 | 24.40 | 1013 | 86 | 1.32 | 334 | 97 |
| 1 | 2017-01-01 01:00:00 | 23.92 | 21.36 | 24.63 | 1013 | 85 | 1.38 | 326 | 100 |
| 2 | 2017-01-01 02:00:00 | 23.98 | 21.26 | 24.64 | 1014 | 85 | 1.52 | 314 | 100 |
| 3 | 2017-01-01 03:00:00 | 22.96 | 21.34 | 23.47 | 1013 | 94 | 1.37 | 299 | 99 |
| 4 | 2017-01-01 04:00:00 | 22.10 | 20.76 | 22.53 | 1013 | 95 | 1.39 | 302 | 95 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 43291 | 2021-12-09 19:00:00 | 27.11 | 27.11 | 27.11 | 1011 | 22 | 2.56 | 185 | 0 |
| 43292 | 2021-12-09 20:00:00 | 22.72 | 22.72 | 22.72 | 1012 | 39 | 3.62 | 136 | 0 |
| 43293 | 2021-12-09 21:00:00 | 22.72 | 22.72 | 22.72 | 1012 | 39 | 3.62 | 136 | 0 |
| 43294 | 2021-12-09 22:00:00 | 22.72 | 22.72 | 22.72 | 1012 | 39 | 3.62 | 136 | 0 |
| 43295 | 2021-12-09 23:00:00 | 15.25 | 15.25 | 15.25 | 1015 | 88 | 3.60 | 123 | 0 |

43296 rows × 9 columns

```
df_temp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 43296 entries, 0 to 43295
Data columns (total 9 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   dt_iso      43296 non-null  datetime64[ns]
 1   temp        43296 non-null  float64
 2   temp_min    43296 non-null  float64
 3   temp_max    43296 non-null  float64
 4   pressure    43296 non-null  int64
 5   humidity    43296 non-null  int64
 6   wind_speed  43296 non-null  float64
 7   wind_deg    43296 non-null  int64
 8   clouds_all  43296 non-null  int64
dtypes: datetime64[ns](1), float64(4), int64(4)
memory usage: 3.0 MB
```

No conjunto de dados climáticos foi necessário realizar a média diária de cada uma das colunas por ele apresentadas, pois inicialmente o mesmo apresenta coletas meteorológicas de hora em hora e o conjunto de dados de ocorrência contém dados diários.

**Figura 7 – Código responsável pela média dos valores do conjunto de dados climáticos.**

```
new_dates = []

for x in datas:
    rows = df_temp.loc[
        (df_temp['data'] == x)
    ]
    rows = rows.drop(['dt_iso', 'data'] , axis=1)
    new_row = [x]

    for collun in rows.columns:
        mean = round(sum(rows[collun].values) / len(rows), 2)
        new_row.append(mean)

    new_dates.append(
        new_row
    )

df_temp_new = pd.DataFrame(new_dates, columns=['data', 'temp','temp_min','temp_max','pressure','humidity','wind_speed','wind_deg'
df_temp_new.index = pd.to_datetime(df_temp_new['data'])
df_temp_new = df_temp_new.drop(['data'], axis=1)
df_temp_new
```

Após as médias serem realizadas, os conjuntos de dados foram unidos utilizando a coluna data como chave para união. O arquivo resultante dessa operação foi nomeado de *"BD_Cotton_Safra_Occurrence_And_Temp.csv"* presente no diretório *"/datasets/processed/"*, contendo os dados climáticos e dados de ocorrência lado a lado, como na Figura 8.

**Figura 8 – Informações do conjunto de dados final utilizado no trabalho.**

| data | temp | temp_min | temp_max | pressure | humidity | wind_speed | wind_deg | clouds_all | estagio | idade | ocorrencia |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2018-10-15 | 19.80 | 19.03 | 20.55 | 1012.92 | 86.25 | 2.43 | 141.62 | 81.67 | 0 | 12 | 0 |
| 2018-10-15 | 19.80 | 19.03 | 20.55 | 1012.92 | 86.25 | 2.43 | 141.62 | 81.67 | 0 | 8 | 0 |
| 2018-10-15 | 19.80 | 19.03 | 20.55 | 1012.92 | 86.25 | 2.43 | 141.62 | 81.67 | 0 | 6 | 0 |
| 2018-10-16 | 20.73 | 19.59 | 21.47 | 1012.75 | 82.50 | 2.91 | 112.50 | 87.83 | 0 | 8 | 0 |
| 2018-10-16 | 20.73 | 19.59 | 21.47 | 1012.75 | 82.50 | 2.91 | 112.50 | 87.83 | 0 | 5 | 0 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 2021-12-01 | 22.33 | 22.33 | 22.33 | 1008.54 | 65.79 | 2.30 | 185.42 | 3.88 | 0 | 35 | 0 |
| 2021-12-01 | 22.33 | 22.33 | 22.33 | 1008.54 | 65.79 | 2.30 | 185.42 | 3.88 | 1 | 39 | 1 |
| 2021-12-01 | 22.33 | 22.33 | 22.33 | 1008.54 | 65.79 | 2.30 | 185.42 | 3.88 | 1 | 41 | 0 |
| 2021-12-01 | 22.33 | 22.33 | 22.33 | 1008.54 | 65.79 | 2.30 | 185.42 | 3.88 | 0 | 15 | 0 |
| 2021-12-01 | 22.33 | 22.33 | 22.33 | 1008.54 | 65.79 | 2.30 | 185.42 | 3.88 | 1 | 41 | 0 |

5724 rows × 11 columns

```
df_occurrence_temp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 5724 entries, 2018-10-15 to 2021-12-01
Data columns (total 11 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   temp        5724 non-null   float64
 1   temp_min    5724 non-null   float64
 2   temp_max    5724 non-null   float64
 3   pressure    5724 non-null   float64
 4   humidity    5724 non-null   float64
 5   wind_speed  5724 non-null   float64
 6   wind_deg    5724 non-null   float64
 7   clouds_all  5724 non-null   float64
 8   estagio     5724 non-null   int64
 9   idade       5724 non-null   int64
 10  ocorrencia  5724 non-null   int64
dtypes: float64(8), int64(3)
memory usage: 536.6 KB
```

=

+

Finalizado o tratamento de ambos os conjuntos de dados e unificando os mesmos, é valido destacar que grande parte dos esforços deste trabalho foram concentrados nessa etapa, pois sem ela, todo o restante do trabalho era completamente inviável de ser realizado, dado o tamanho da desproporção das classes do conjunto de dados de ocorrências, um *Overfitting* do modelo era evidente.

## 4. Análise e Exploração dos Dados

Após o tratamento dos dados mostrado anteriormente, também foi gerado três novas colunas para o conjunto de dados: sendo a idade em dias da safra, sendo a data da ocorrência subtraída pela data de plantio da safra; O estágio da safra utilizando a coluna idade comparada aos dados encontrados na Figura 9, obtida em uma apresentação sobre a Cultura do Algodão ministrada na Faculdade de Tecnologia *Shunji Nishimura* de Pompeia, São Paulo; A criação de uma coluna chamada "ocorrência", que contém um binário mantendo "0" para nenhuma ocorrência da praga e "1" se houver ocorrência da praga naquele registro. As funções que foram utilizadas para a criação dessas 3 colunas estão presentes na Figura 10.

**Figura 9 – Informações dos estágios de desenvolvimento/crescimento do algodão.**

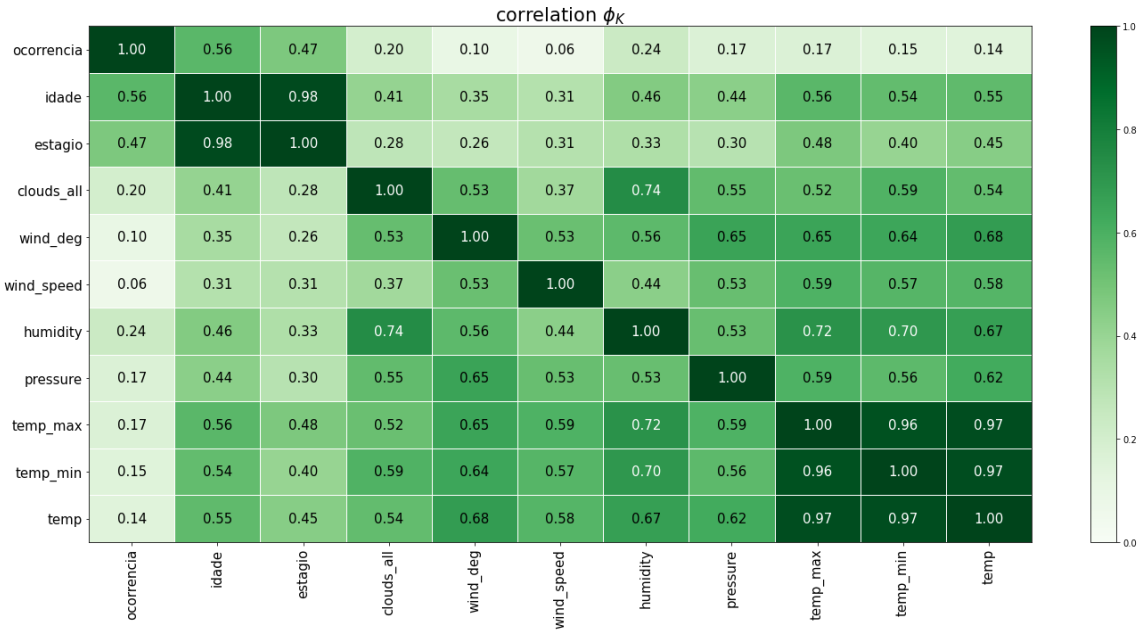
**Figura 10 – Funções para criação de novas colunas.**

```
def get_days_difference(initial_day, final_day):
    diferenca = final_day - initial_day
    idade_atual = diferenca.days
    if idade_atual < 0:
        idade_atual = idade_atual * -1

    return idade_atual

def get_estagio(idade_atual):
    if int(idade_atual) <= 35:
        return 0
    elif int(idade_atual) > 35 and int(idade_atual) <= 70:
        return 1
    elif int(idade_atual) > 70 and int(idade_atual) <= 110:
        return 2
    elif int(idade_atual) > 110:
        return 3

def is_occurrence(row):
    if row['ovo'] != 0 or row['P'] != 0 or row['M'] != 0 or row['G'] != 0:
        val = 1
    else:
        val = 0
    return val
```

Além das modificações anteriores, também foram removidas as colunas com identificação das propriedades e glebas, uma vez que esses dados não são *features* uteis ao nosso modelo de classificação. O conjunto de dados de ocorrências balanceado e com as novas características foi salvo como *"BD_Cotton_Safra_Occurrence.csv"* no diretório *"/datasets/compiled/"*.

Na Figura 11, podemos analisar a matriz de correlação das variáveis do conjunto de dados final. Para obter essa matriz de correlação, foi utilizado o cálculo de coeficiente de correlação de *Phik,* por ser um dos mais indicados quando o conjunto de dados possui variáveis contínuas e/ou categóricas.

**Figura 11 – Funções para criação de novas colunas.**




Analisando a matriz de correlação, podemos observar uma correlação muito forte entre as Temperaturas médias, máximas e mínimas. Também percebemos uma correlação entre a variável que dita a ocorrência e a umidade do ar.

Com essa matriz, podemos reforçar os estudos anteriores que apontam a relação entre o clima com a ocorrência da praga na cultura do algodão.

## 5. Criação de Modelos de Machine Learning

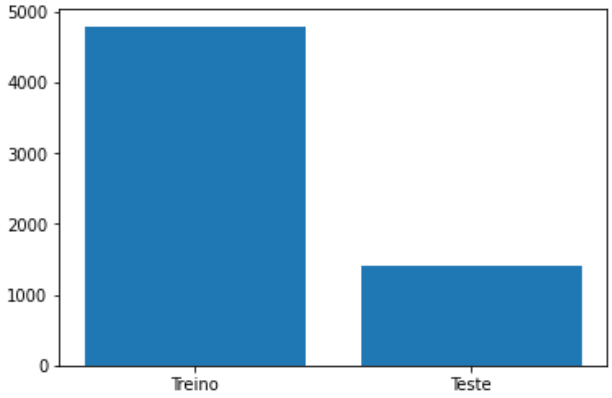
Dado o problema aqui proposto, foram escolhidos 8 modelos de *Machine Learning* de classificação com aprendizagem supervisionada, sendo eles *DecisionTreeClassifier, AdaBoostClassifier, XGBClassifier, RandomForestClassifier, MLPClassifier, LogisticRegression, KNeighborsClassifier* e *LGBMClassifier.* Na Tabela 3 estão presentes cada um dos modelos e quais foram seus hiper parâmetros utilizados para esse trabalho.

**Tabela 3 – Hiper parâmetros dos modelos de classificação.**

| **Nome do modelo** | **Hiper parâmetros** |
|---|---|
| *DecisionTreeClassifier* | * |
| *AdaBoostClassifier* | * |
| *XGBClassifier* | *objective* = *'binary*:*logistic'*,<br>*booster* = *'gbtree'*,<br>*eval_metric* = 'auc',<br>*tree_method* = 'hist',<br>*grow_policy* = 'lossguide',<br>*use_label_encoder* = False |
| *RandomForestClassifier* | * |
| *MLPClassifier* | * |
| *LogisticRegression* | *solver* = 'lbfgs',<br>*max_iter* = 100 |
| *KNeighborsClassifier* | * |
| *LGBMClassifier* | * |

Os modelos de *Machine Learning* foram implementados utilizando *Python* no *Jupyter Notebook*, o conjunto de dados final contendo 5.724 registros foi dividido entre dois conjuntos, um conjunto para o treino do modelo, contendo 4006 registros (70%) e um conjunto para o teste do modelo, contendo 1718 registros (30%).

**Figura 12 – Distribuição dos dados entre os conjuntos de treino e teste.**




Após treinar um modelo de *Machine Learning* é importante testá-lo para definir se o modelo é capaz de generalizar bem para novos dados e cumprir com o seu propósito. Se o modelo é capaz de prever muito bem os dados de treino, mas é ruim ao prever dados de teste, temos um problema de *Overfitting*.

Para avaliar os modelos apresentados e escolher o mais interessante para o problema, foram utilizados a Matriz de Confusão e suas métricas e a Curva ROC.

## 6. Interpretação dos Resultados

Antes de apresentar os resultados de cada um dos modelos, é necessário entender o que cada uma das métricas utilizadas para avaliar os modelos de classificação significam no contexto de Machine Learning. No presente trabalho, foram utilizadas as métricas de Precisão, Sensibilidade, F1-Score e Acurácia obtidas de cada um dos modelos pela Matriz de Confusão e utilizadas na obtenção da Curva ROC de cada modelo, com o intuito de definir qual modelo é o mais assertivo dado nosso problema.

### 6.1. Matriz de Confusão

A matriz de confusão é uma matriz que contêm todas as classes que nosso modelo contém para a classificação, sendo elas nesse trabalho, a **Ocorrência** e a **Não Ocorrência**. Nesta matriz deve conter a quantidade de valores reais de cada uma das classes e a quantidade de valores preditos pelo nosso modelo em cada uma das classes, assim, podemos visualizar o número de casos que o modelo preditivo acertou e errou.

Na Figura 13, podemos observar um exemplo de Matriz de Confusão e as métricas resultantes dessa matriz.

**Figura 13 – Exemplo de Matriz de Confusão.**

| | | Previsto | |
|---|---|---|---|
| | | **Não Ocorrência** | **Ocorrência** |
| **Real** | **Não Ocorrência** | **Verdadeiro Negativo (VN)** | **Falso Positivo (FP)** |
| | **Ocorrência** | **Falso Negativo (FN)** | **Verdadeiro Positivo (VP)** |

Na diagonal do quadrante superior esquerdo, nós temos os acertos denominados de **Verdadeiro Negativo** e **Verdadeiro Positivo**. Por sua vez, na diagonal do quadrante superior direito temos o número de erros que o modelo obteve em sua previsão, sendo o **Falso Positivo** e o **Falso Negativo**. Em outras palavras, quando observamos uma matriz de confusão nós temos:

- **Verdadeiros Positivos**: Classificação realizada corretamente pelo modelo, onde a classe é **positivo**;
- **Falsos Negativos**: Classificação realizada incorretamente, onde o modelo previu a classe Negativo quando a real classe era **positivo**;
- **Falsos Positivos**: Classificação realizada incorretamente, onde o modelo previu a classe Positivo quando a real classe era **negativo**;
- **Verdadeiros Negativos**: Classificação realizada corretamente pelo modelo, onde a classe é **negativo**.

Obtendo esses valores, conseguimos também obter as métricas de acurácia, precisão, sensibilidade e medida de *F1-score*. A Acurácia indica uma performance geral do modelo, utilizando em seu cálculo todas as classificações feitas corretamente, como mostrado na fórmula abaixo:

$$acurácia = \frac{VP + VN}{Total}$$

A Precisão é definida como a proporção de predições corretas de uma categoria em relação a todas as previsões feitas dessa categoria, como mostrado na fórmula abaixo:

$$precisão = \frac{VP}{VP + FP}$$

A sensibilidade ou *Recall* é definida como a proporção de verdadeiros positivos em relação a soma dos verdadeiros positivos com os falsos negativos, como mostrado na fórmula abaixo:

$$recall = \frac{VP}{VP + FN}$$

*F1-Score* é dado pela média harmônica entre a precisão e sensibilidade, sendo uma métrica que representa em um número único a qualidade geral do modelo, como mostrado na fórmula abaixo:

$$F1 = \frac{2 * precisão * recall}{precisão + recall}$$

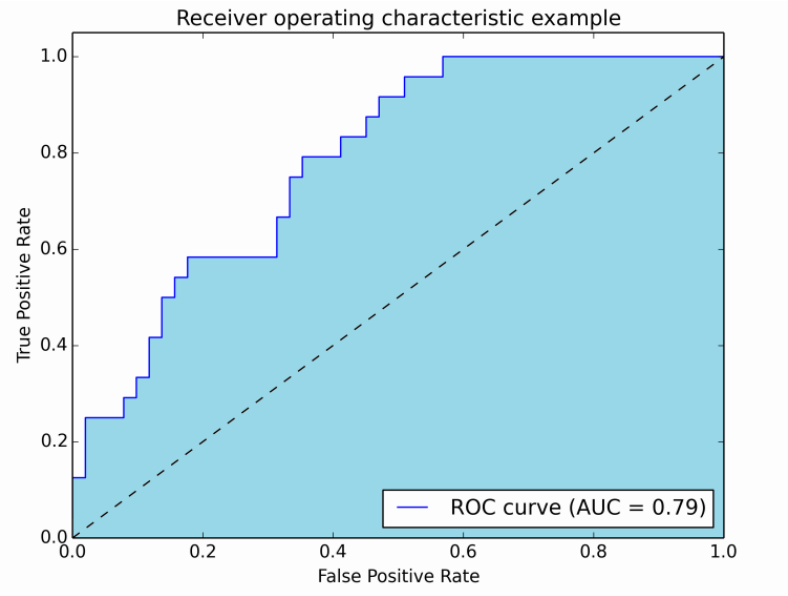
**6.2. Curva ROC**

Outra forma de avaliar o modelo é por meio da curva ROC, que é uma métrica que ilustra o desempenho de um modelo de classificação binário à medida que o seu limiar de discriminação varia. A ROC possui dois parâmetros: a medida de sensibilidade (taxa de verdadeiro positivo) e a medida de especificidade (taxa de falso positivo). Em resumo, curva ROC resulta da representação gráfica dos índices de sensibilidade e especificidade.

Uma forma de simplificar a análise da curva ROC é por meio da AUC (*Area Under the ROC Curve*), que nada mais é que uma maneira de resumir a curva ROC em um único valor, calculando a área sob a curva. O valor do AUC varia entre 0 e 1 e o limiar entre a classe é 0,5. Ou seja, acima desse limite, o algoritmo classifica em uma classe e abaixo na outra classe. Quanto mais próximo de 1 o valor do AUC é, melhor foi o desempenho do modelo na classificação entre as duas classes.

Na Figura 14, podemos observar um exemplo de plotagem da Curva ROC e sua medida AUC.

**Figura 14 – Exemplo da Curva ROC e medida AUC.**




## 6.3. Comparativo entre modelos

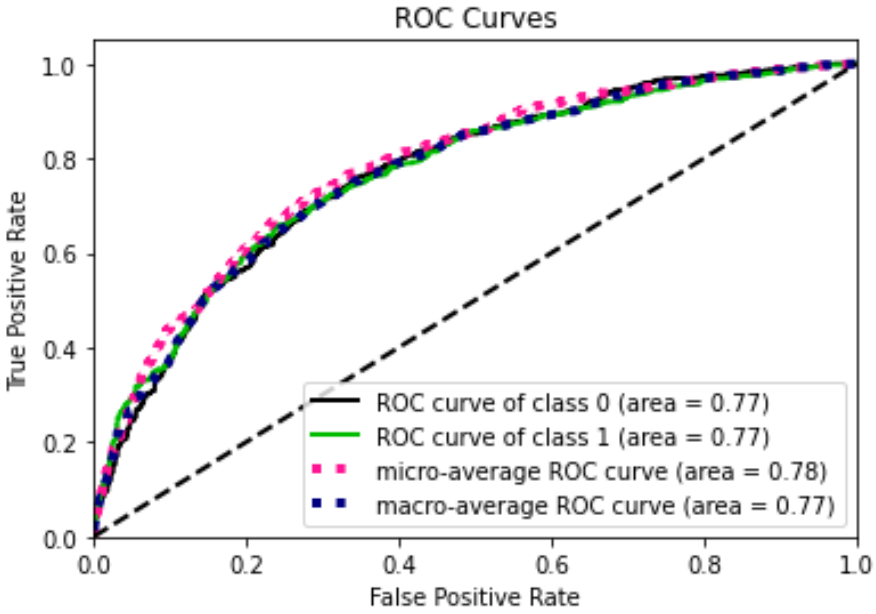
Na Tabela 4 contém todos os modelos de classificação e suas métricas de desempenho em ambas as classes. Ao final de cada conjunto de métricas, há também a acurácia geral de cada um dos modelos.

**Tabela 4 – Métricas de desempenho dos modelos de classificação.**

| Modelo | Classe | Precisão | Sensibilidade | F1-Score | Acurácia |
|---|---|---|---|---|---|
| DecisionTree Classifier | Não Ocorrência | 0.70 | 0.76 | 0.73 | 0.66 |
| | Ocorrência | 0.60 | 0.53 | 0.56 | |
| AdaBoost Classifier | Não Ocorrência | 0.73 | 0.81 | 0.77 | 0.71 |
| | Ocorrência | 0.67 | 0.57 | 0.61 | |
| XGB Classifier | Não Ocorrência | 0.74 | 0.77 | 0.76 | 0.71 |
| | Ocorrência | 0.65 | 0.61 | 0.63 | |
| RandomForest Classifier | Não Ocorrência | 0.72 | 0.73 | 0.73 | 0.67 |
| | Ocorrência | 0.60 | 0.59 | 0.59 | |
| MLP Classifier | Não Ocorrência | 0.60 | 0.99 | 0.75 | 0.61 |
| | Ocorrência | 0.85 | 0.05 | 0.09 | |
| Logistic Regression | Não Ocorrência | 0.64 | 0.82 | 0.72 | 0.62 |
| | Ocorrência | 0.55 | 0.32 | 0.41 | |
| KNeighbors Classifier | Não Ocorrência | 0.73 | 0.79 | 0.76 | 0.70 |
| | Ocorrência | 0.65 | 0.58 | 0.62 | |
| LGBM Classifier | Não Ocorrência | 0.74 | 0.80 | 0.77 | 0.72 |
| | Ocorrência | 0.67 | 0.60 | 0.63 | |

Observando a tabela anterior, conclui-se que o modelo que obteve mais desempenho nos nossos testes foi o *LGBMClassifier*. Para efetivamente considerarmos que o *LGBMClassifier* é o modelo com maior desempenho em nossos testes, vamos comparar a Curva ROC dele com os demais, lembrando que, quanto mais próximo de "1" o AUC for, mais capacidade de generalizar as previsões o modelo analisado possui.

**Figura 15 – Curva ROC do modelo de classificação *LGBMClassifier.***




Contudo, também podemos analisar a Matriz de confusão do mesmo modelo, permitindo uma melhor análise de características a serem melhoradas futuramente.

**Figura 16 – Matriz de confusão do modelo de classificação *LGBMClassifier.***

| | | Previsto | |
|---|---|---|---|
| | | **Não Ocorrência** | **Ocorrência** |
| **Real** | **Não Ocorrência** | **812** | **207** |
| | **Ocorrência** | **281** | **418** |

Analisando a Matriz de Confusão do nosso modelo, podemos observar que o modelo tem mais dificuldade em classificar um Não Ocorrência corretamente. Futuramente, pode-se adicionar outras variáveis para facilitar essa distinção entre as classes e assim, consigam trazer mais acurácia para o modelo de predição.

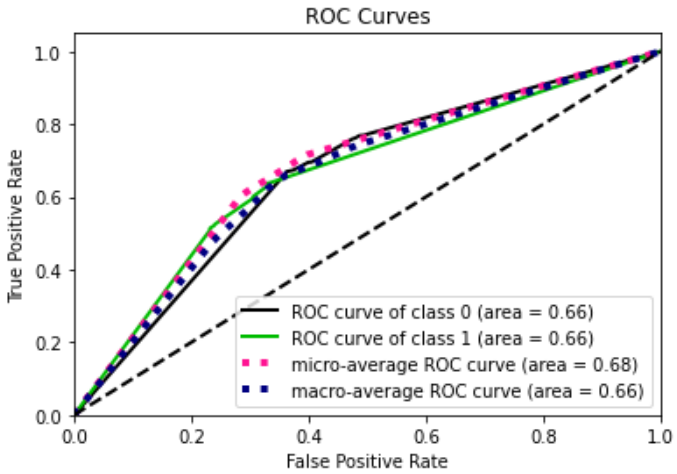
Abaixo, pode-se conferir as métricas dos outros modelos apresentados neste trabalho. São apresentadas a Matriz de Confusão e a Curva ROC de cada modelo separadamente, ambas as métricas obtidas utilizando a classificação do conjunto de dados de teste.

### 6.3.1. Decision Tree Classifier

**Figura 17 – Matriz de confusão do modelo de classificação *DecisionTreeClassifier.***

| | | Previsto | |
|---|---|---|---|
| | | **Não Ocorrência** | **Ocorrência** |
| **Real** | **Não Ocorrência** | **785** | **249** |
| | **Ocorrência** | **313** | **371** |

**Figura 18 – Curva ROC do modelo de classificação *DecisionTreeClassifier.***

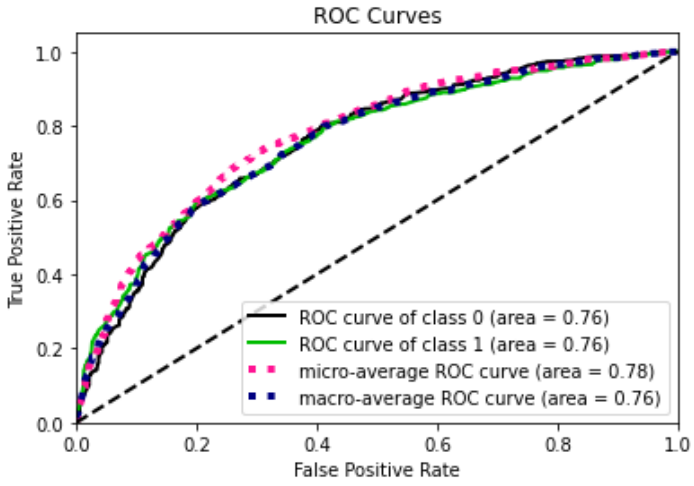


### 6.3.2. AdaBoostClassifier

**Figura 19 – Matriz de confusão do modelo de classificação *AdaBoostClassifier.***

| | | Previsto | |
|---|---|---|---|
| | | Não Ocorrência | Ocorrência |
| Real | Não Ocorrência | 825 | 194 |
| | Ocorrência | 303 | 396 |

**Figura 20 – Curva ROC do modelo de classificação *AdaBoostClassifier.***

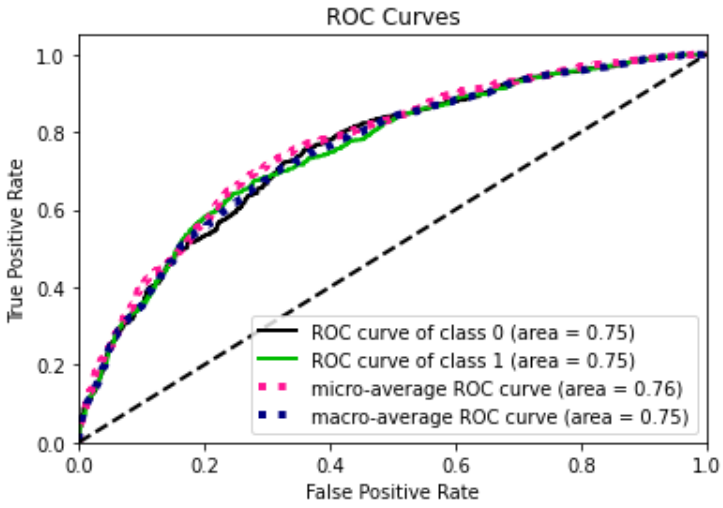


### 6.3.3. XGBClassifier

**Figura 21 – Matriz de confusão do modelo de classificação *XGBClassifier.***

| | | Previsto | |
|---|---|---|---|
| | | Não Ocorrência | Ocorrência |
| Real | Não Ocorrência | 789 | 230 |
| | Ocorrência | 274 | 425 |

**Figura 22 – Curva ROC do modelo de classificação *XGBClassifier.***

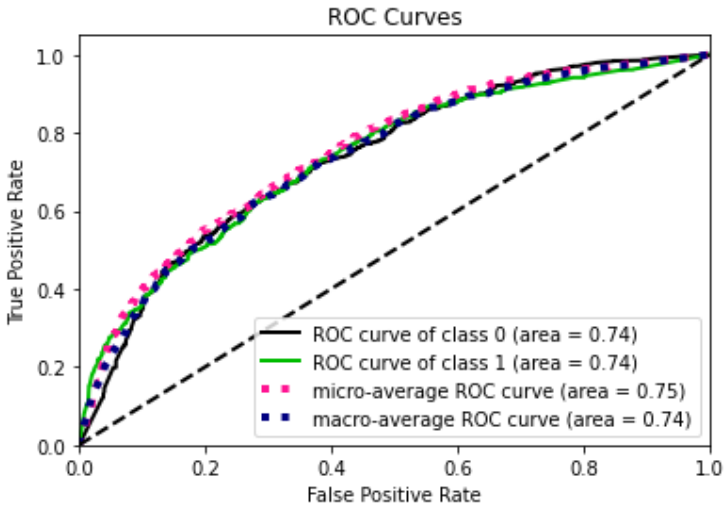


### 6.3.4. RandomForestClassifier

**Figura 23 – Matriz de confusão do modelo de classificação *RandomForestClassifier*.**

| | | Previsto | |
|---|---|---|---|
| | | Não Ocorrência | Ocorrência |
| Real | Não Ocorrência | 749 | 270 |
| | Ocorrência | 285 | 414 |

**Figura 24 – Curva ROC do modelo de classificação *RandomForestClassifier*.**

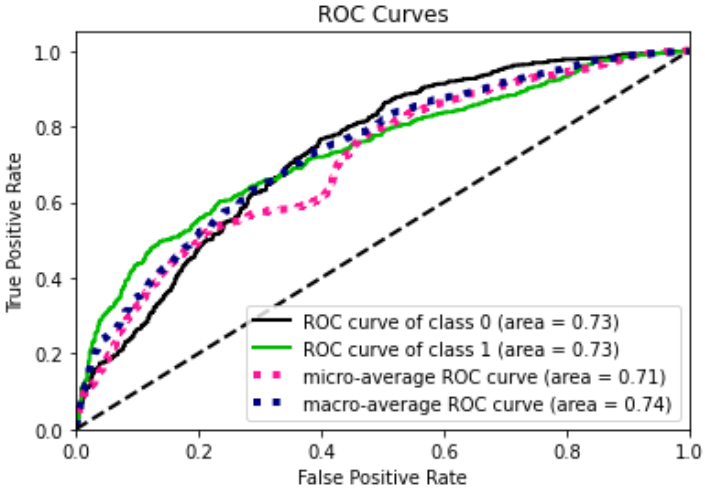


### 6.3.5. MLPClassifier

**Figura 25 – Matriz de confusão do modelo de classificação *MLPClassifier.***

| | | Previsto | |
|---|---|---|---|
| | | Não Ocorrência | Ocorrência |
| Real | Não Ocorrência | 1014 | 5 |
| | Ocorrência | 671 | 28 |

**Figura 26 – Curva ROC do modelo de classificação *MLPClassifier.***

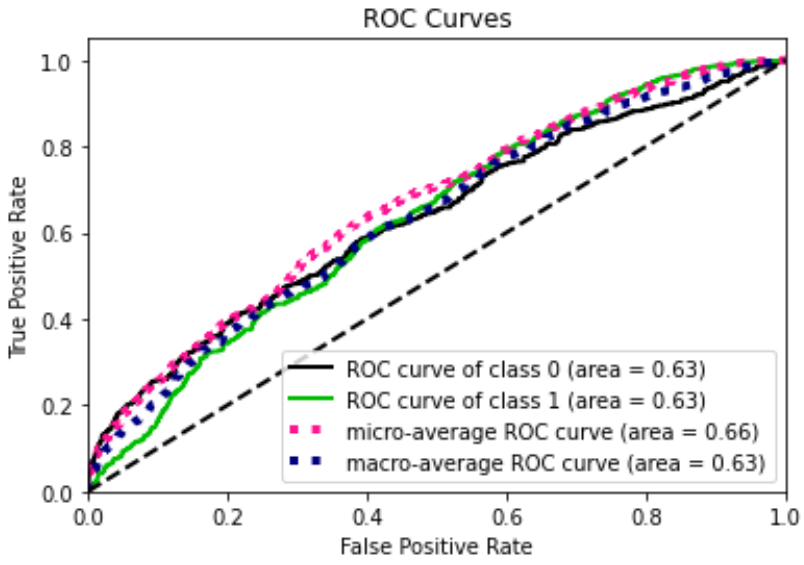


### 6.3.6. LogisticRegression

**Figura 27 – Matriz de confusão do modelo de classificação *LogisticRegression*.**

| | | Previsto | |
|---|---|---|---|
| | | Não Ocorrência | Ocorrência |
| Real | Não Ocorrência | 836 | 183 |
| | Ocorrência | 472 | 227 |

**Figura 28 – Curva ROC do modelo de classificação *LogisticRegression*.**

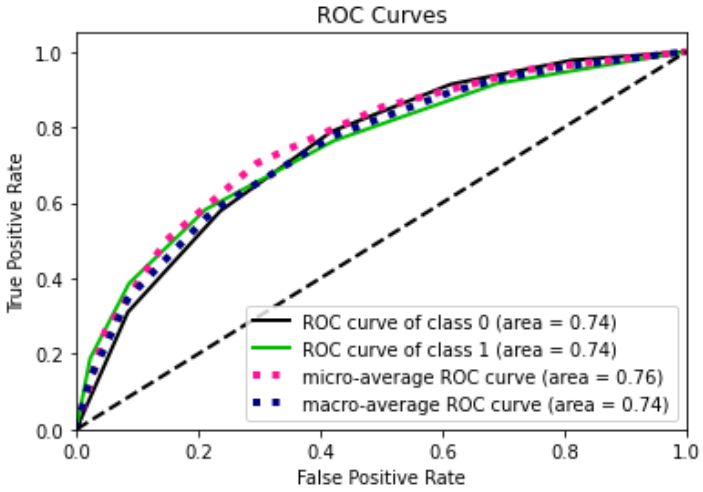


### 6.3.7. KNeighborsClassifier

**Figura 29 – Matriz de confusão do modelo de classificação *KNeighborsClassifier.***

| | | Previsto | |
|---|---|---|---|
| | | Não Ocorrência | Ocorrência |
| Real | Não Ocorrência | 803 | 216 |
| | Ocorrência | 292 | 407 |

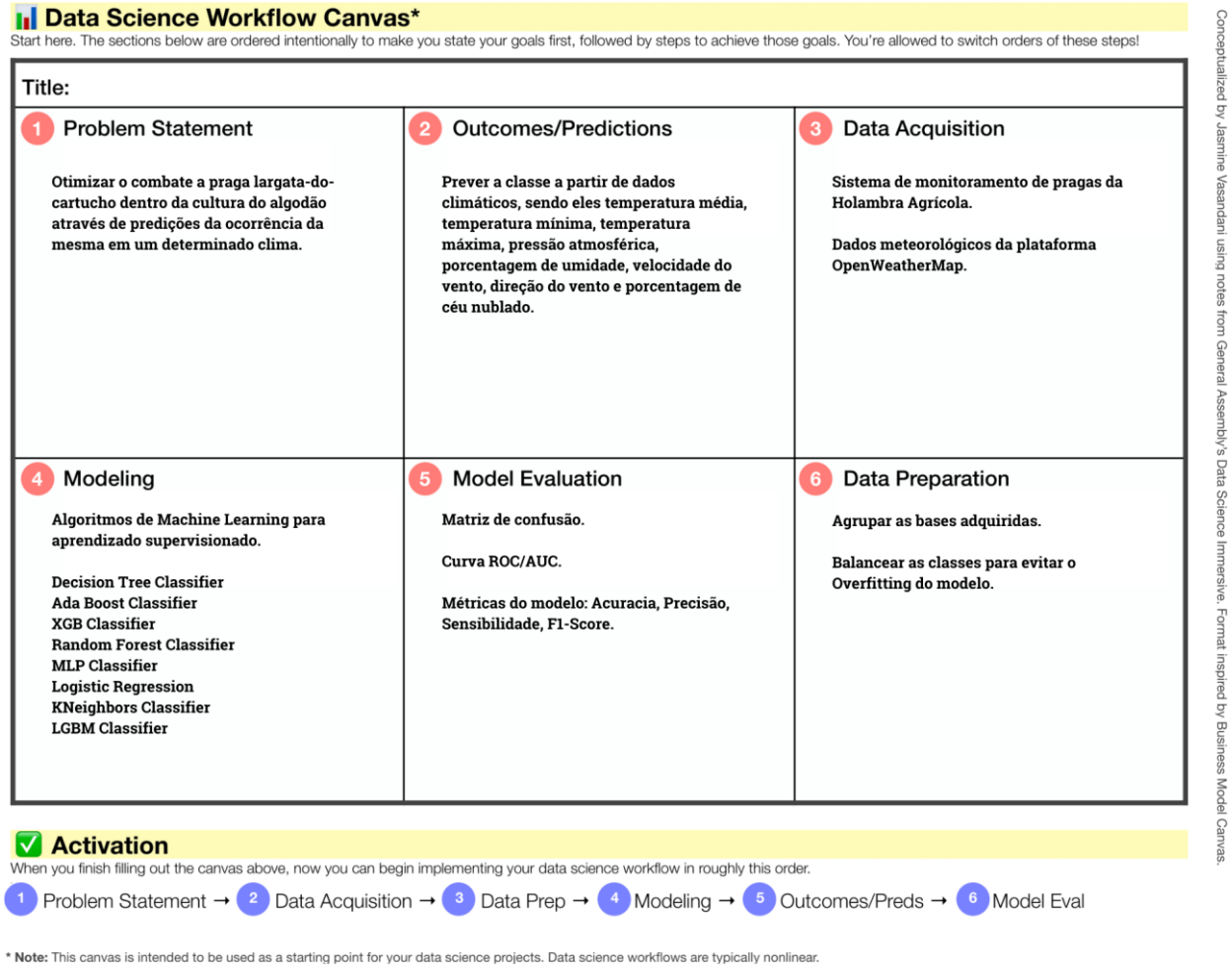
**Figura 30 – Curva ROC do modelo de classificação *KNeighborsClassifier.***



## 7. Apresentação dos Resultados

Os resultados obtidos no presente trabalho foram previamente estabelecidos e mapeados no modelo de *Canvas* de *Vasandani*, apresentado na Figura 30.

**Figura 31 – Canvas de Vasandani para o presente trabalho.**




Durante o trabalho, foram implementados oito modelos de *Machine Learning* com o objetivo de realizar a predição da ocorrência da praga lagarta-do-cartucho na cultura do algodão a partir do agrupamento de um conjunto de dados de ocorrências com um conjunto de dados climáticos do mesmo período.

Previamente, foi notado analisando o conjunto de dados de ocorrências que a classe a ser predita era muito desbalanceada, contendo aproximadamente 98% dos dados da classe com o valor correspondente a “Não Ocorrência”. Utilizando esse conhecimento prévio, o conjunto de dados foi balanceado e o mesmo valor compõe 59% do conjunto de dados, tornando possível e viável a construção de um modelo de classificação.

Por fim, concluiu-se que os modelos de classificação analisados tiveram desempenhos semelhantes, mas o modelo que se destacou nas métricas neste trabalho analisadas foi o modelo "*LGBMClassifier*", que obteve uma melhor acurácia (0.72) e um melhor desempenho tanto nas demais métricas analisadas quanto na análise da Curva ROC com a métrica AUC (0.77).

**8. Links**

Link para o vídeo: https://youtu.be/T94aqFZl-Q0

Link para o repositório: https://drive.google.com/drive/folders/1-Gzdp5VS4KUS4KUmWx9d2DwVRyqArIKy?usp=sharing

# APÊNDICE

## Programação/Scripts

Cole aqui seus scripts em Python e/ou R.