UNEAMÉRICA

MACHINE LEARNING APLICADA PARA CRIAR UM MODELO PREDITIVO DE EGRESSOS DE APOSENTADORIAS PARA A UNIVERSIDADE FEDERAL DO TRIÂNGULO MINEIRO

Tiago José Chaves Touso

09/2020

Uberaba, MG

2020

RESUMO

O estudo tem como objetivo verificar a viabilidade de usar Inteligência Artificial (Machine Learning) para criar um modelo preditivo para fazer previsões de egresso por aposentadoria na Universidade Federal do Triângulo Mineiro, um órgão do serviço público federal, identificando assim os seus cargos e os seus setores que mais serão afetados com as aposentadorias neste ano.

Foram analisados, com estatística descritiva, os dados dos servidores ativos e servidores inativos (Aposentados), buscando os atributos que são mais relevantes para o desenvolvimento, e para escolha de uma técnica de Marchine Learning.

Após essa análise, será criado o modelo preditivo de egresso, e também será feita a verificação da acurácia e da precisão do modelo, verificando assim se o modelo poderá ou não ser utilizado para fazer as previsões.

Verificado a acurácia e a precisão será feita a previsão, gerando assim o relatório com as quantidades de servidores que poderão aposentar neste ano. E assim, podemos também aproveitar o modelo para gerar a previsão de aposentadoria para os próximos anos, sendo necessário somente fazer o treinamento do modelo com os dados atuais.

Palavras-chave:
Inteligência Artificial – Machine Learning – Modelo Preditivo – Serviço Público – Servidor Público - Aposentadoria

ABSTRACT

The study aims to verify the feasibility of using Artificial Intelligence (Machine Learning) to create a predictive model to make predictions of retirement egress at the Federal University of Triângulo Mineiro, a federal public service body, thus identifying their positions and their sectors that will be most affected by pensions this year.

Data from active and inactive servers (Retired) were analyzed with descriptive statistics, looking for the attributes that are most relevant for development, and for choosing a Marchine Learning technique.

After this analysis, the egress predictive model will be created, and the accuracy and precision of the model will also be checked, thus verifying whether or not the model can be used to make the predictions.

The accuracy is verified and the forecast will be made, thus generating the report with the number of servers that may retire this year. And so, we can also take advantage of the model to generate the retirement forecast for the coming years, requiring only training the model with the current data.

Key words:

Artificial Intelligence - Machine Learning - Predictive Model - Public Service - Public Servant - Retirement

Índice

1. INTRODUÇÃO

Tudo muda e se atualiza muito rápido no mundo. Não basta ter seus sistemas informatizados, seus processos bem estruturados. Hoje o grande truque é poder fazer previsões do futuro. Mas não são previsões mágicas ou truque. São previsões com base bem sólidas e científicas, com técnicas computacionais de inteligência artificial.

Pensando sobre essa premissa, na era do Big Data, um dos maiores ativos que uma instituição tem são seus dados (banco de dados), e ter a possibilidade de prever os acontecimentos será o maior diferencial de competitividade de todos os tempos para as instituições, seja pública ou seja privada.

E o egresso por aposentadoria é um dos grandes gargalos no serviço público, pois a reposição da vaga é um processo complexo, e legalista, e moroso. Sendo assim, ter uma ferramenta estratégica, que possa prever esse egressos, permitirá que a instituição minimize os impactos, tanto de logística, quanto de talentos humanos, a fim de garantir a continuação e a melhor prestação de serviço público a população.

2. ANÁLISE DO PROJETO

O projeto tem como objetivo, estudar uma base de dados dos servidores ativos e inativos da Universidade Federal do Triângulo Mineiro (UFTM), e criar um modelo de Machine Learning que possa fazer a previsão de aposentadoria para os servidores ativos.

No Brasil, para se aposentar, além da regra geral constitucional, que já traz algumas possibilidades, há também as legislações complementares, ou normas especiais para algumas profissões.

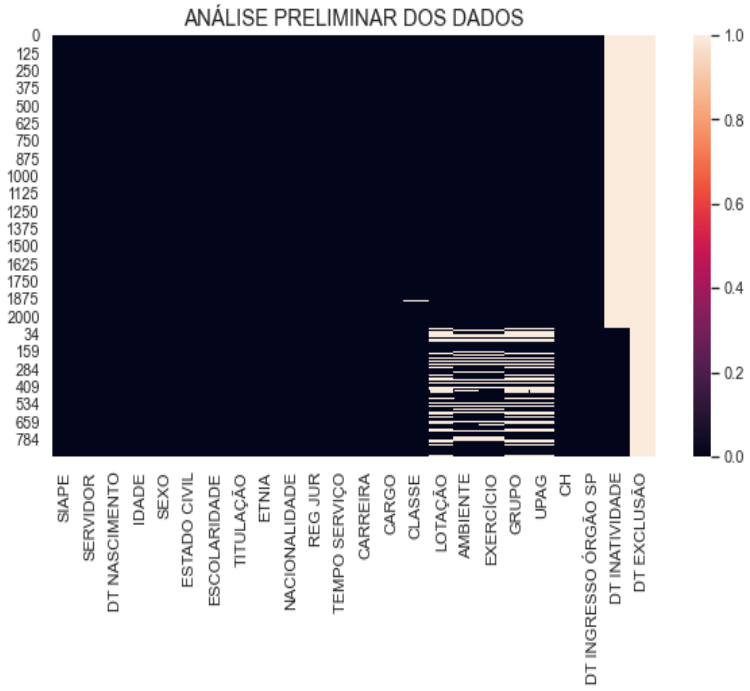
Sendo assim, não é objetivo deste trabalho o estudo detalhado legislação para aposentadoria, mas sim, o estudo dos atributos dos aposentados: idade, tempo de serviço, sexo, carreira, cargo, etc …, para que o nosso modelo de Machine Learning, possa aprender com os dados e possa fazer as previsões.

Para isso, foi solicitado a UFTM, por meio do e-mail, uma lista dos servidores ativos e inativos, em 2020-07, contendo os seguintes dados: Nome do Servidor, Data de Nascimento, Idade, Sexo, Estado Civil, Escolaridade, Titulação, Etnia, Nacionalidade, Regime Jurídico, Tempo Serviço, Carreira, Cargo, Classe, Lotação, Grupo de Lotação, Carga Horária, Data Ingresso no Órgão, Data de Aposentadoria (Inatividade), Data de Exclusão.

3. ANÁLISE PRELIMINAR DOS DADOS

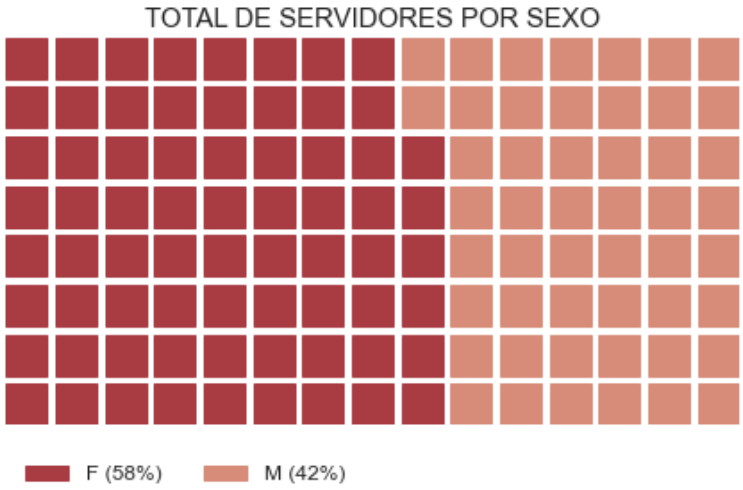
Primeiramente foi feito uma análise para identificar os dados nulos ou em branco (cor branca no gráfico) e identificamos que aparentemente a base de dados não apresenta anomalias que prejudicaria o modelo ou que deveriam ser tratadas.

Sendo bem visível os dois grupos de servidores: os ativos (“DATA DE INATIVIDADE” em branco) e os inativos (“DATA DE INATIVIDADE” em preto).



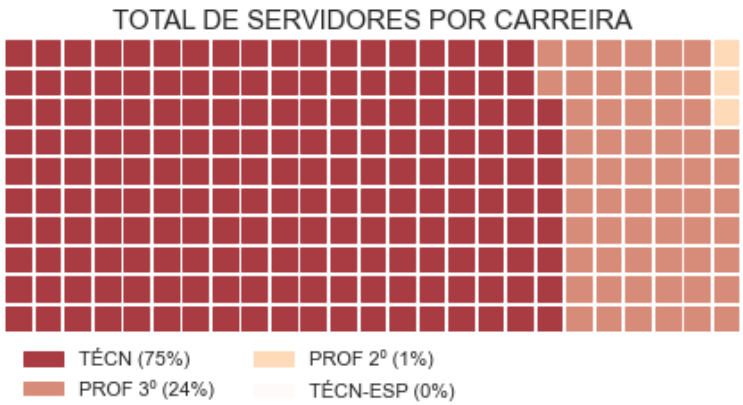
Fonte: UFTM

A população estudada, a maioria é do sexo feminino, mais de 50%.



Fonte: UFTM

Entre as carreiras: A maioria, 75% é da carreira de Técnico Administrativo em Educação (TAE), 24 % são da carreira de Professores de 3º grau (PROF 3º) e 1% são da carreira de Professores de 2º (PROF 2º). Também existe nesta base de dados servidores que estão trabalhando na UFTM, mas são servidores de outros órgãos (TÉCN-ESP).

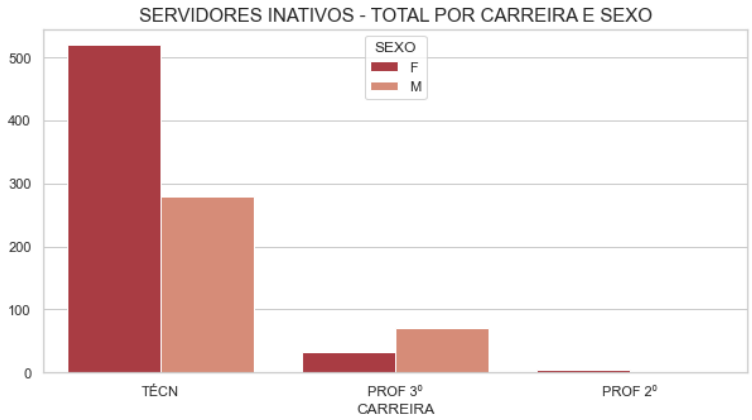


Fonte: UFTM

Tendo em vista uma análise descritiva da nossa população, ela será divida em dois grupos: Servidores Ativos e Servidores Inativos, para que possamos identificar características específicas de cada um.

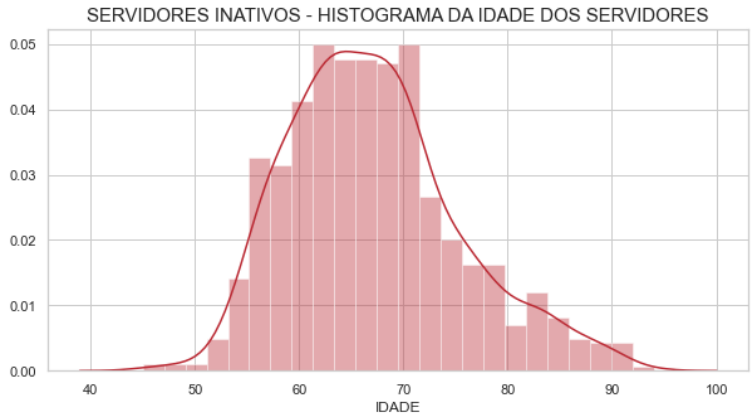
SERVIDORES INATIVOS

Identificamos que a grande maioria dos servidores inativos é da carreira de Técnico Administrativo em Educação e do sexo feminino.



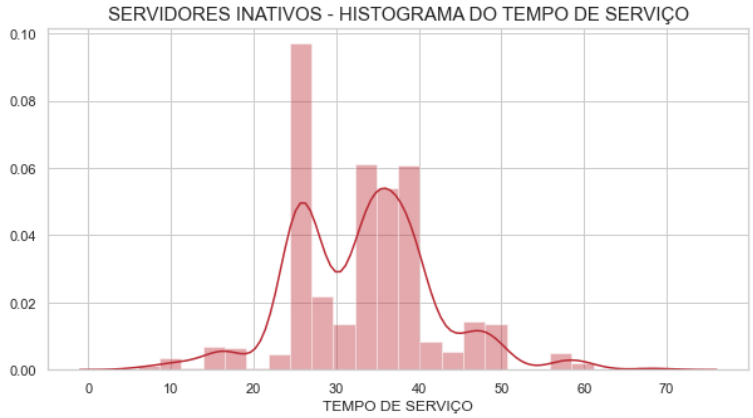
Fonte: UFTM

A idade dos servidores inativos está variando entre 50 e 90 anos com a maior frequência entre 60 e 70 anos.



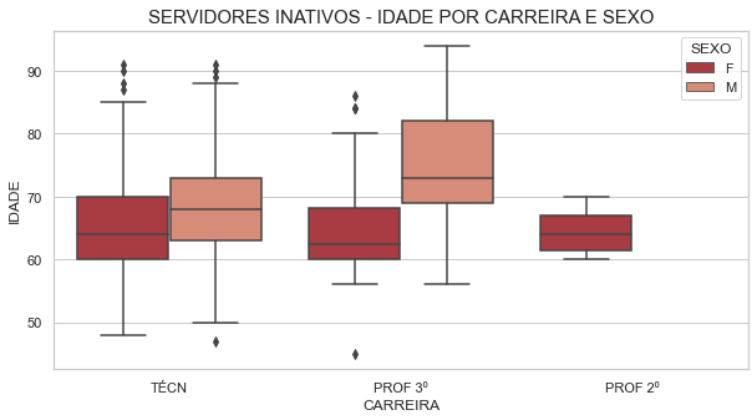
Fonte: UFTM

E o tempo de serviço dos servidores inativos está variando: de 10 até 60 anos com a maior frequência: 25 anos e de 32 até 40 anos.



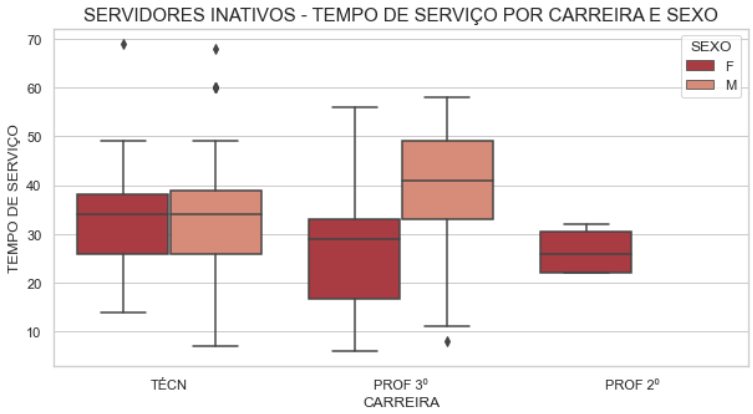
Fonte: UFTM

Também verificamos que a média das idades dos servidores inativos é maior que 60 anos em todas as carreiras e para todos os sexos.



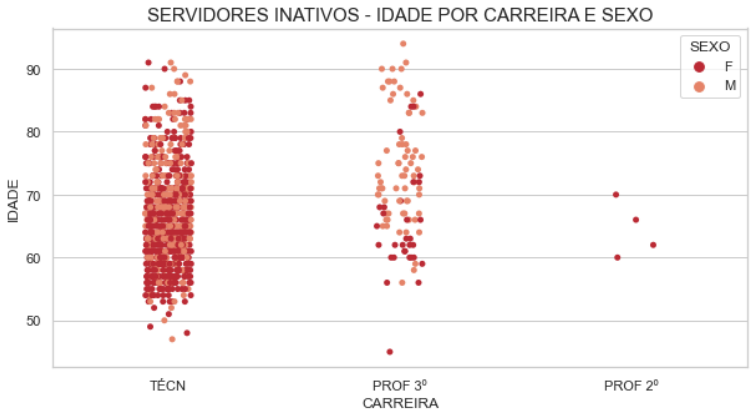
Fonte: UFTM

Sobre a média do tempo de serviço dos servidores inativos, percebemos que sobre as carreiras: Técnico Administrativos em Educação a média é maior que 30 anos, Professor 3º a média está entre 25 e 42 anos e Professor 2º a média é maior que 25 anos.



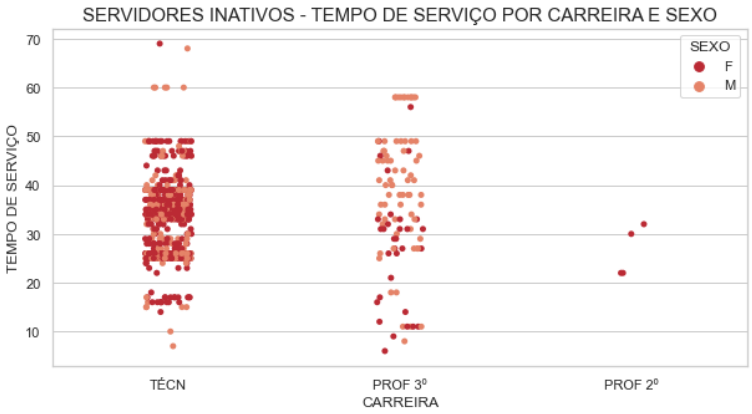
Fonte: UFTM

Verificamos que agrupando os servidores inativos por carreira, os servidores da carreira de Técnico Administrativos em Educação, há uma homogeneidade entre as idades e sexo, o que se mostra diferente para Professores, onde a maioria é do sexo feminino.



Fonte: UFTM

Já para o tempo de serviço, a observação anterior se mantêm, há uma homogeneidade entre o tempo de serviço e sexo para os servidores Técnico Administrativo em Educação, e para os Professores a maioria é do sexo feminino.



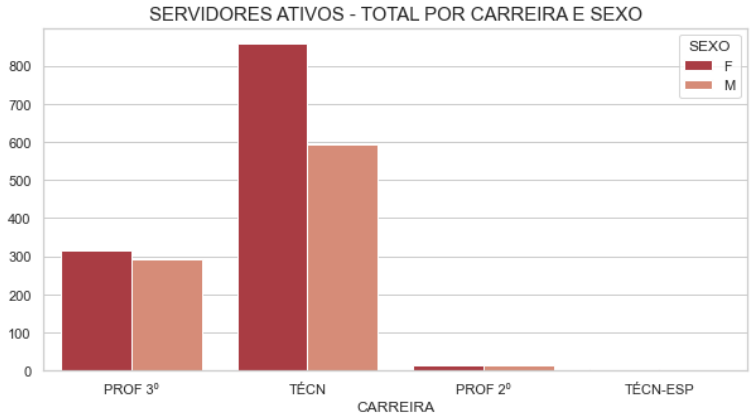
Fonte: UFTM

Analisando a idade e o tempo de serviço e agrupando por sexo, observamos que também há uma homogeneidade entre essas duas variáveis.

Fonte: UFTM

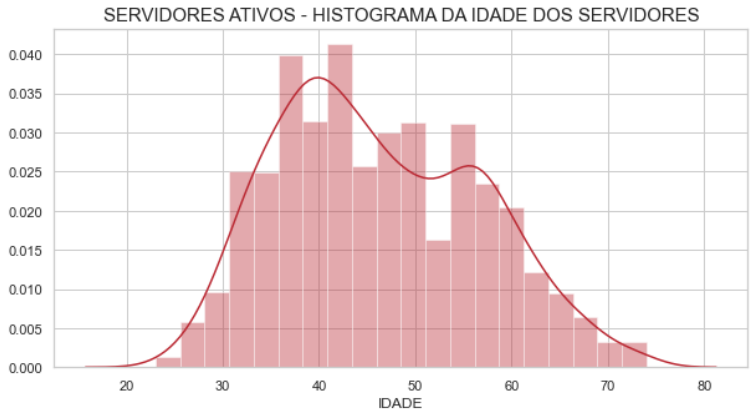
SERVIDORES ATIVOS

Sobre os servidores ativos, a maioria é o sexo feminino, sendo maior essa observação nos servidores da carreira de Técnico Administrativo em Educação, e para os Professores há quase uma igualdade entre os sexos.



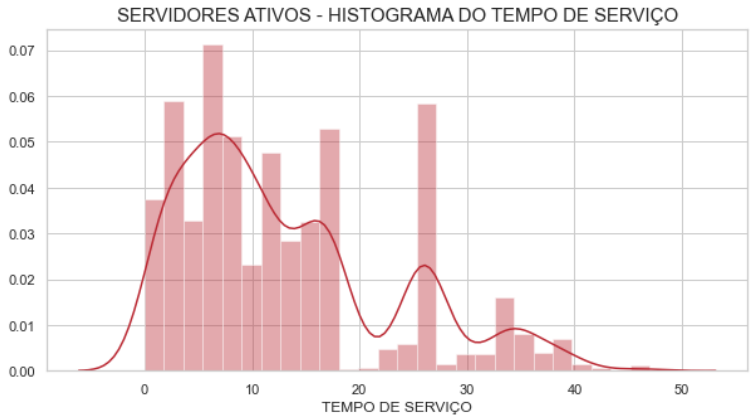
Fonte: UFTM

A idade dos servidores ativos está variando entre 20 e 70 anos com a maior frequência entre 36 e 44 anos.



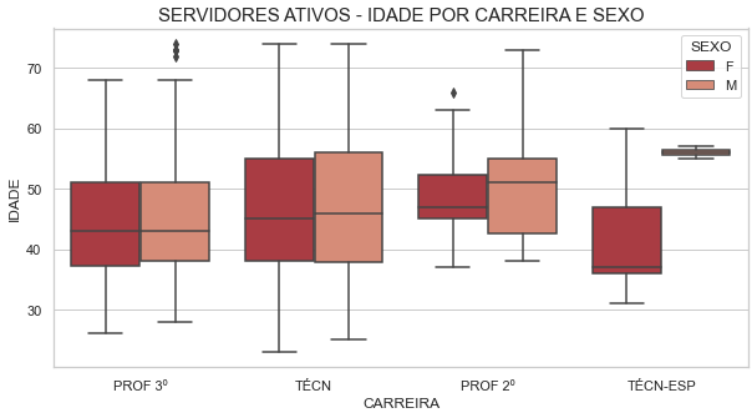
Fonte: UFTM

Sobre o tempo de serviço dos servidores ativos a maioria está entre: 0 até 20 anos e que estes dados não estão bem distribuídos.



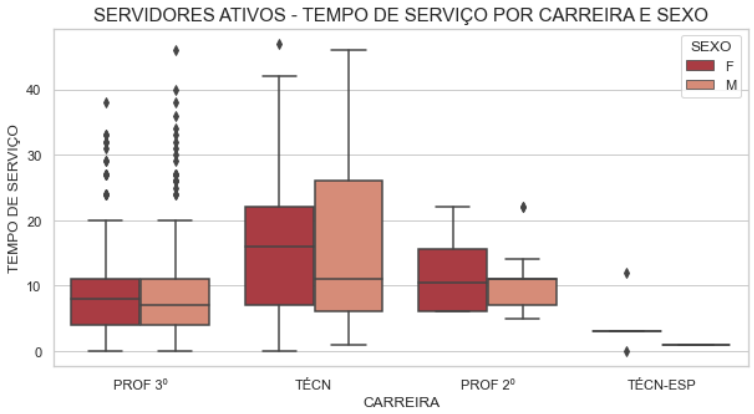
Fonte: UFTM

Também verificamos que a média das idades dos servidores ativos é maior que 40 anos em todas as carreiras e para todos os sexos.



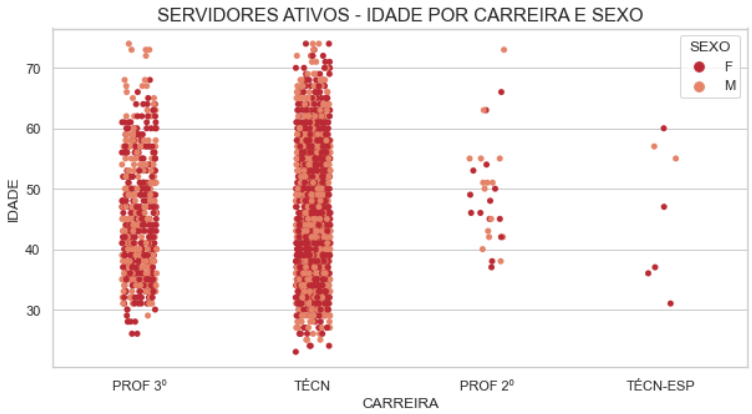
Fonte: UFTM

Sobre a média do tempo de serviço dos servidores ativos, percebemos que sobre as carreiras: Técnico Administrativos em Educação e os Professores de 2º a média é maior que 10 anos, Professor 3º a média é menos que 10 anos.



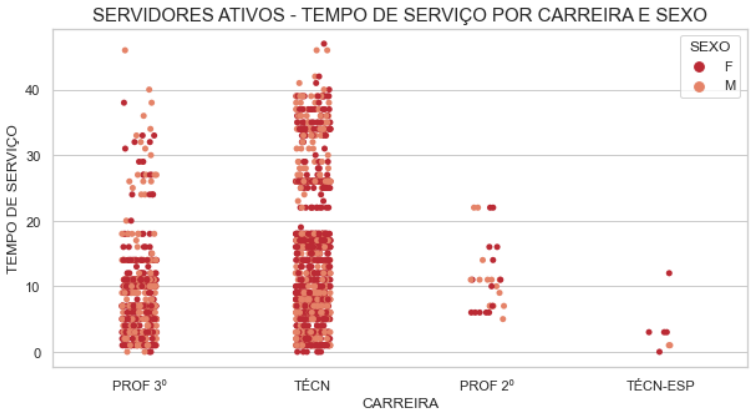
Fonte: UFTM

A distribuição por idade se mantém bem homogêa quando comparada por sexo, menos a maioria dos servidores ativos sendo do sexo feminino.



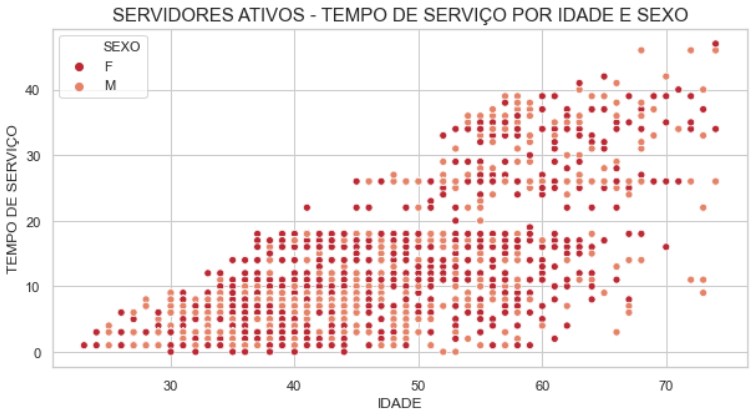
Fonte: UFTM

Por tempo de serviço a distribuição se mantém bem homogênea quando comparada por sexo.



Fonte: UFTM

Examinando a idade e o tempo de serviço dos servidores ativos e agrupando por sexo, observamos que também há uma uniformidade entre essas duas variáveis.



Fonte: UFTM

4. ESCOLHA DO MODELO DE MACHINE LEARNING

Após a análise exploratória dos dados, identificamos características do grupo que queremos prever e percebemos que poderíamos usar a técnica de Regressão Logística para criar o nosso modelo.

Também, após análise dos dados, foi necessário fazer o tratamento de alguns dados, pois como o modelo de Regressão Logística necessita que todos os dados que serão processados sejam numéricos, e assim os campos que tinham os valores textos foram transformados em números.

Em seguida foi criado um código de Marchine Learning (em anexo) que:

- Receberá todos os dados dos servidores
- Fará o tratamento da informação
- Dividirá os dados em dois conjuntos: treinamento e teste
- Fará o treinamento dos modelo
- Fará o teste de precisão do modelo
- Receberá os dados dos servidores ativos para prever
- Devolverá os dados previsto para aposentadoria

5. CONCLUSÃO

Executando o modelo de previsão e analisando o teste de precisão do modelo, resultado abaixo, percebemos que o modelo obteve em média um percentual de 86 % de acerto. É um percentual muito bom, se considerarmos a quantidade de variáveis e o tipo de aprendizagem de máquina.

Também podemos confirmar a perfomance do modelo através do resultado na matriz de confusão.

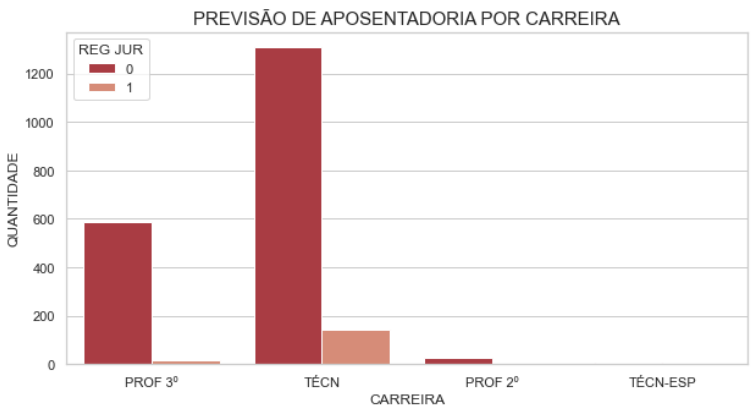
```
--------------------
CLASSIFICATION REPORTE

              precision    recall  f1-score   support

           0       0.91      0.92      0.92       630
           1       0.81      0.80      0.80       269

   micro avg       0.88      0.88      0.88       899
   macro avg       0.86      0.86      0.86       899
weighted avg       0.88      0.88      0.88       899

--------------------
CONFUSION MATRIX

[[578  52]
 [ 54 215]]
--------------------
```

Assim, tendo a UFTM um total de 2091 servidores ativos, no ano de 2020, o modelo fez uma previsão 191 aposentadorias. Que indica 9,13 % de egresso com aposentadoria no ano, um percentual bem expressivo.

Sobretudo, mesmo o modelo fazendo uma previsão tão elevada, não é uma certeza, que todos os servidores solicitarão aposentadoria, pois há outras variáveis que interferem nesta escolha.

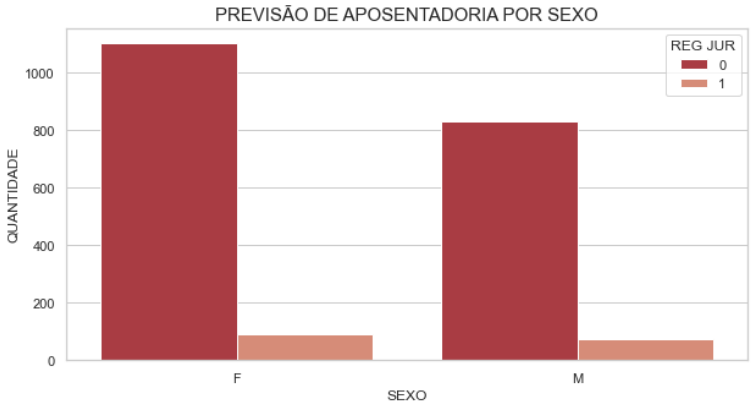
Mas, mesmo assim, órgão com posse desses dados, poderá fazer uma intervenção estratégica para diminuir o impacto dos egressos, por cargo e nas áreas que serão mais afetadas.

Está é uma o cenário da situação atual, sendo “0” são os servidores ativos, “1” é a previsão de aposentadoria.



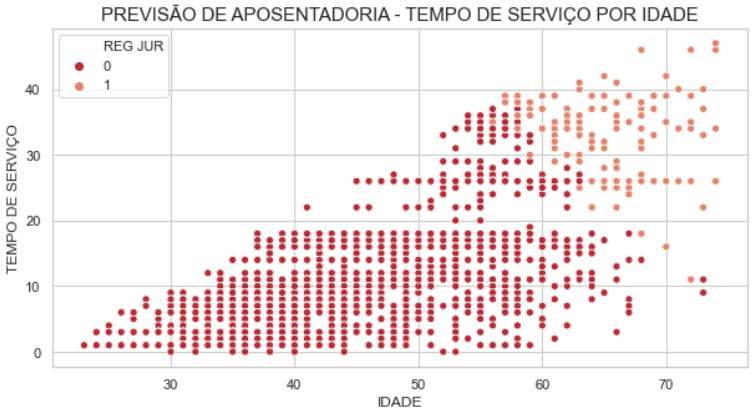
Fonte: Previsão do modelo

A previsão de aposentadoria por sexo ficou bem equilibrada a diferença.



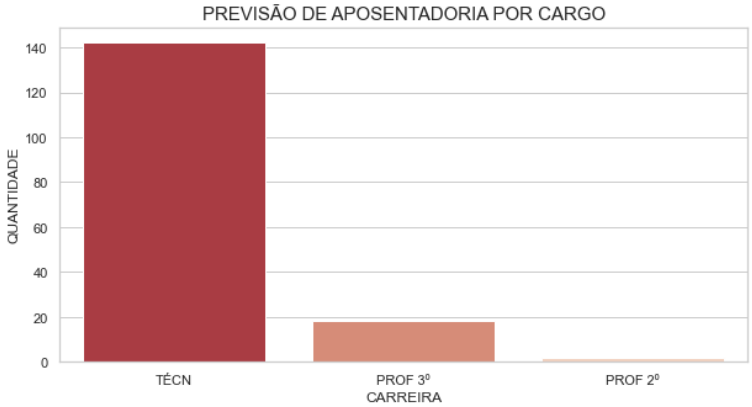
Fonte: Previsão do modelo

Comparando o tempo de serviço e idade, podemos observar que o modelo conseguiu separar bem as duas variáveis, quanto maior o tempo de serviço e maior a idade o modelos separou os ativos dos que podem aposentar.



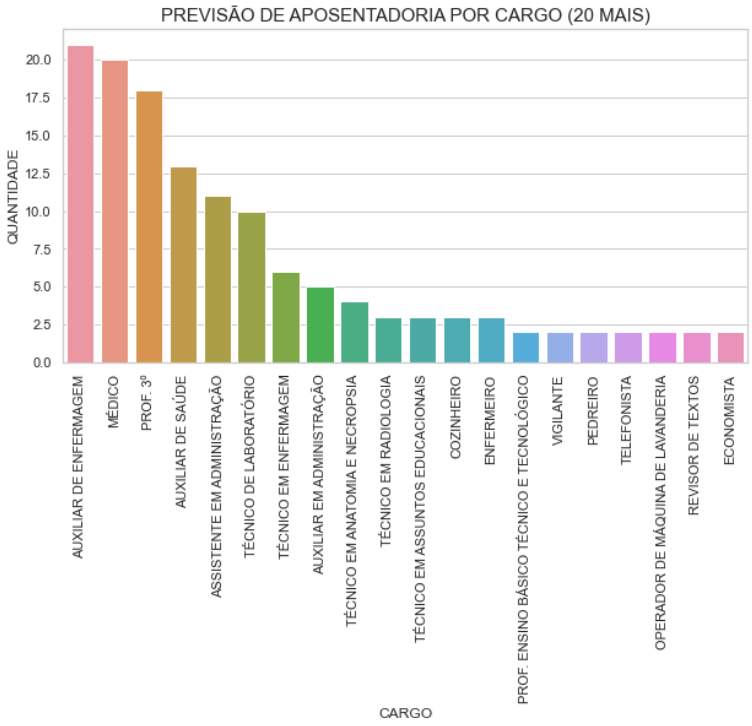
Fonte: Previsão do modelo

A maior parte do egresso será na carreira de técnicos administrativos em educação.



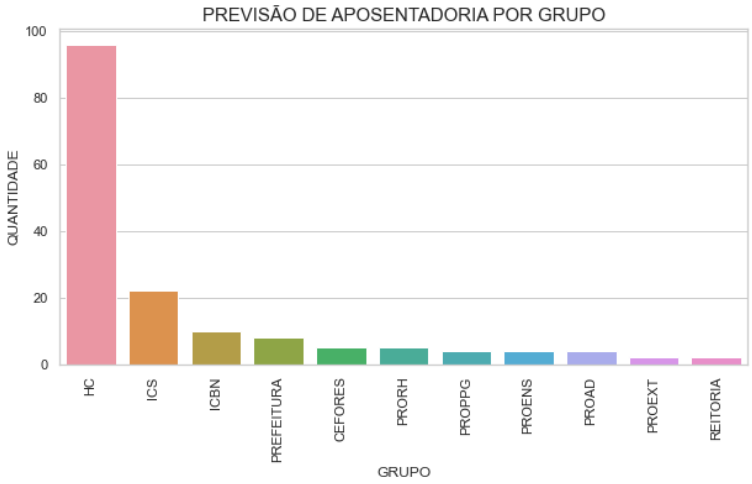
Fonte: Previsão do modelo

Os cargos da UFTM que deverão ter egressos.



Fonte: Previsão do modelo

Os grandes áreas da UFTM que deverão ter egressos.



Fonte: Previsão do modelo

Tendo essa previsão em mãos, a Universidade, por meio da Pró-Reitoria de Recursos Humanos e dos gestores das áreas, poderão planejar melhor sua força de trabalho para que não sejam pegos de surpresa com pedidos de aposentadoria que poderiam paralisar atividades importantes para o desenvolvimento da missão da universidade, e o atendimento aos seus diversos públicos.

6. REFERÊNCIAS

WHEELAN, Charles, Estatística – O que é, para que serve, como funciona. Rio de Janeiro: ed. Zahar, 2013.

Bruce, Peter; **Bruce**, Andrew, Estatística Prática para Cientistas de dados. Rio de Janeiro: ed. Alfa Books, 2019.

Provost, Foster; **Fawcett**, Tom, Data Science para Negócios. Rio de Janeiro: ed. Alfa Books, 2016.

Mckinney, Wes, Python para Análise de Dados. São Paulo, ed. Novatec 2018.

Grus, Joel, Data Science do Zero – Primeiras Regras com o Python. Rio deJaneiro: ed. Alfa Books, 2019.

7. ANEXOS

SOLICITAÇÃO DA INFORMAÇÃO

Gmail **Tiago Touso <tiagotouso@gmail.com>**

Solicitação de informação
1 mensagem

Tiago Touso <tiagotouso@gmail.com> 9 de julho de 2020 10:10
Para: "assessoria.prorh" <assessoria.prorh@uftm.edu.br>

RESPOSTA DA SOLICITAÇÃO

Gmail **Tiago Touso <tiagotouso@gmail.com>**

Re: Solicitação de informação

1 mensagem

Assessoria em Gestão de RH <assessoria.prorh@uftm.edu.br> 9 de julho de 2020 10:13
Para: Tiago Touso <tiagotouso@gmail.com>

tiagotouso@gmail.com> escreveu:

> Bom dia!
>
> Gostaria de receber uma lista atualizada dos servidores ativos e inativos da UFTM, contendo os seguintes dados:

CÓDIGO DO MODELO

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.heatmap(dados.isnull(), )
ax.set_xlabel('')
ax.set_ylabel('')
ax.set_title('ANÁLISE PRELIMINAR DOS DADOS', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
dd = dados['SEXO'].value_counts()
total = len(dados)

fig = plt.figure(
    FigureClass = Waffle,
    rows = 8,
    columns = 15,
    values = dd,
    title={
        'label': 'TOTAL DE SERVIDORES POR SEXO',
        'loc': 'center',
        'fontdict': { 'fontsize': 15 }
    },
    labels=["{0} ({1}%)".format(k, round(v / total * 100)) for k, v in dd.items()],
    legend={
#         'labels': ["{0} ({1}%)".format(k, v) for k, v in data.items()],  # lebel
        'loc': 'lower left',
        'bbox_to_anchor': (0, -0.2),
        'ncol': len(dd),
        'framealpha': 0,
        'fontsize': 12
    },
    colors = ('#a93c43', '#d68c78')
)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
dd = dados['CARREIRA'].value_counts()
total = len(dados)

fig = plt.figure(
    FigureClass = Waffle,
    rows = 14,
    columns = 25,
    values = dd,
    title={ 'label': 'TOTAL DE SERVIDORES POR CARREIRA',
           'loc': 'center',
            'fontdict': { 'fontsize': 15 }  },
    labels=["{0} ({1}%)".format(k, round(v / total * 100)) for k, v in dd.items()],
    legend={'loc': 'lower left',
            'bbox_to_anchor': (0, -0.25),
            'ncol': 2,
            'framealpha': 0},
    colors = ('#a93c43', '#d68c78', '#FFDAB9', '#FFFAFA')
)

num += 1
# fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

ANALISE DOS SERVIDORES INATIVOS

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.countplot(x='CARREIRA', data=aposentados, hue='SEXO')
ax.set_xlabel('CARREIRA')
ax.set_ylabel('')
ax.set_title('SERVIDORES INATIVOS - TOTAL POR CARREIRA E SEXO', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.distplot(aposentados['IDADE'])
ax.set_xlabel('IDADE')
ax.set_ylabel('')
ax.set_title('SERVIDORES INATIVOS - HISTOGRAMA DA IDADE DOS SERVIDORES', fontsize=

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.distplot(aposentados['TEMPO SERVIÇO'])
ax.set_xlabel('TEMPO DE SERVIÇO')
ax.set_ylabel('')
ax.set_title('SERVIDORES INATIVOS - HISTOGRAMA DO TEMPO DE SERVIÇO', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.boxplot(x='CARREIRA', y= 'IDADE', data=aposentados, hue='SEXO')
ax.set_xlabel('CARREIRA')
ax.set_ylabel('IDADE')
ax.set_title('SERVIDORES INATIVOS - IDADE POR CARREIRA E SEXO', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.boxplot(x='CARREIRA', y= 'TEMPO SERVIÇO', data=aposentados, hue='SEXO')
ax.set_xlabel('CARREIRA')
ax.set_ylabel('TEMPO DE SERVIÇO')
ax.set_title('SERVIDORES INATIVOS - TEMPO DE SERVIÇO POR CARREIRA E SEXO', fontsiz

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.stripplot(x='CARREIRA', y= 'IDADE', data=aposentados, hue='SEXO')
ax.set_xlabel('CARREIRA')
ax.set_ylabel('IDADE')
ax.set_title('SERVIDORES INATIVOS - IDADE POR CARREIRA E SEXO', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.stripplot(x='CARREIRA', y= 'TEMPO SERVIÇO', data=aposentados, hue='SEXO')
ax.set_xlabel('CARREIRA')
ax.set_ylabel('TEMPO DE SERVIÇO')
ax.set_title('SERVIDORES INATIVOS - TEMPO DE SERVIÇO POR CARREIRA E SEXO', fontsiz

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.scatterplot(x='IDADE', y='TEMPO SERVIÇO', data=aposentados, hue='SEXO')
ax.set_xlabel('IDADE')
ax.set_ylabel('TEMPO DE SERVIÇO')
ax.set_title('SERVIDORES INATIVOS - TEMPO DE SERVIÇO POR IDADE E SEXO', fontsize=1

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

ANALISE DOS SERVIDORES ATIVOS

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.countplot(x='CARREIRA', data=ativos, hue='SEXO')
ax.set_xlabel('CARREIRA')
ax.set_ylabel('')
ax.set_title('SERVIDORES ATIVOS - TOTAL POR CARREIRA E SEXO', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.distplot(ativos['IDADE'])
ax.set_xlabel('IDADE')
ax.set_ylabel('')
ax.set_title('SERVIDORES ATIVOS - HISTOGRAMA DA IDADE DOS SERVIDORES', fontsize=15

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.distplot(ativos['TEMPO SERVIÇO'])
ax.set_xlabel('TEMPO DE SERVIÇO')
ax.set_ylabel('')
ax.set_title('SERVIDORES ATIVOS - HISTOGRAMA DO TEMPO DE SERVIÇO', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.boxplot(x='CARREIRA', y= 'IDADE', data=ativos, hue='SEXO')
ax.set_xlabel('CARREIRA')
ax.set_ylabel('IDADE')
ax.set_title('SERVIDORES ATIVOS - IDADE POR CARREIRA E SEXO', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.boxplot(x='CARREIRA', y= 'TEMPO SERVIÇO', data=ativos, hue='SEXO')
ax.set_xlabel('CARREIRA')
ax.set_ylabel('TEMPO DE SERVIÇO')
ax.set_title('SERVIDORES ATIVOS - TEMPO DE SERVIÇO POR CARREIRA E SEXO', fontsize=

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.stripplot(x='CARREIRA', y= 'IDADE', data=ativos, hue='SEXO')
ax.set_xlabel('CARREIRA')
ax.set_ylabel('IDADE')
ax.set_title('SERVIDORES ATIVOS - IDADE POR CARREIRA E SEXO', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.stripplot(x='CARREIRA', y= 'TEMPO SERVIÇO', data=ativos, hue='SEXO')
ax.set_xlabel('CARREIRA')
ax.set_ylabel('TEMPO DE SERVIÇO')
ax.set_title('SERVIDORES ATIVOS - TEMPO DE SERVIÇO POR CARREIRA E SEXO', fontsize=

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.scatterplot(x='IDADE', y='TEMPO SERVIÇO', data=ativos, hue='SEXO')
ax.set_xlabel('IDADE')
ax.set_ylabel('TEMPO DE SERVIÇO')
ax.set_title('SERVIDORES ATIVOS - TEMPO DE SERVIÇO POR IDADE E SEXO', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

MODELO DE IA

In [ ]:

```
class previsaoAposentadoria():

    def __init__(self):

        self.dadosModelo = None
        self.dadosPredicao = None

        self.dadostratadosModelo = None
        self.dadostratadosPredicao = None

        self.resultado = None

        self.modelo = None

        from sklearn.linear_model import LogisticRegression
        self.modelo = LogisticRegression()

        self.dicEstadoCivil = {}
        self.dicCarreira = {}
        self.dicEscolaridade = {}
        self.dicTitulacao = {}
        self.dicEtnia = {}
        self.dicCH = {}

    def CriarDicionarios(self):

        dd = self.dadosModelo

        tabela = [('ESTADO CIVIL', self.dicEstadoCivil),
                  ('CARREIRA', self.dicCarreira),
                  ('ESCOLARIDADE', self.dicEscolaridade),
                  ('TITULAÇÃO', self.dicTitulacao),
                  ('ETNIA', self.dicEtnia),
                  ('CH', self.dicCH)]

        for campo, dicionario in tabela:
            contador = 0
            for vl in dd[campo].unique():
                dicionario[vl] = contador
                contador += 1


    def Tratamento(self, opcao):

        if opcao == 0:
            dd = self.dadosModelo
        else:
            dd = self.dadosPredicao


        dd = dd[['DT NASCIMENTO', 'IDADE', 'SEXO', 'ESTADO CIVIL', 'ESCOLARIDADE',
                    'ETNIA', 'REG JUR', 'TEMPO SERVIÇO', 'CARREIRA',  'CH', 'DT IN

        SEXO_M = pd.get_dummies(dd['SEXO'], drop_first=True)

        dd['SEXO'] = SEXO_M

        dd['DT NASCIMENTO'] = dd['DT NASCIMENTO'].apply(lambda x: int(x[:4]))
        dd['DT INGRESSO ÓRGÃO SP'] = dd['DT INGRESSO ÓRGÃO SP'].apply(lambda x: ir

        dd['REG JUR'] = dd['REG JUR'].apply(lambda x: 1 if x == "APOSENTADO" else
```

```
        dd['ESTADO CIVIL'] = dd['ESTADO CIVIL'].apply(lambda x: self.dicEstadoCivi
        dd['CARREIRA'] = dd['CARREIRA'].apply(lambda x: self.dicCarreira[x])
        dd['ESCOLARIDADE'] = dd['ESCOLARIDADE'].apply(lambda x: self.dicEscolarida
        dd['TITULAÇÃO'] = dd['TITULAÇÃO'].apply(lambda x: self.dicTitulacao[x])
        dd['ETNIA'] = dd['ETNIA'].apply(lambda x: self.dicEtnia[x])
        dd['CH'] = dd['CH'].apply(lambda x: self.dicCH[x])

        if opcao == 0:
            self.dadostratadosModelo = dd
        else:
            self.dadostratadosPredicao = dd



    def Treinamento(self, opcao):

        from sklearn.model_selection import train_test_split
        from sklearn.metrics import classification_report
        from sklearn.metrics import confusion_matrix

        if opcao == 0:
            dd = self.dadostratadosModelo

            x = dd.drop('REG JUR', axis=1)
            y = dd['REG JUR']

            xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.3)

            self.modelo.fit(xtrain, ytrain)
            predict = self.modelo.predict(xtest)

            print('-' * 20)
            print('CLASSIFICATION REPORTE')
            print()
            print(classification_report(ytest, predict))
            print('-' * 20)

            print('CONFUSION MATRIX')
            print()
            print(confusion_matrix(ytest, predict))
            print('-' * 20)

        else:
            dd = self.dadostratadosPredicao
            x = dd.drop('REG JUR', axis=1)

            predict = self.modelo.predict(x)
            self.resultado = predict
```

TREINANDO O MODELO

In [ ]:
```
previsao = previsaoAposentadoria()
previsao.dadosModelo = dados
previsao.CriarDicionarios()
previsao.Tratamento(0)
previsao.Treinamento(0)
```

CRIANDO A PREVISÃO

In [ ]:
```
previsao.dadosPredicao = ativos
previsao.Tratamento(1)
previsao.Treinamento(1)
```

RESULADO DA PREVISÃO

In [ ]:
```
previsao.resultado
```

In [ ]:
```
ativos['REG JUR'] = previsao.resultado
ativos[ativos['REG JUR'] == 1].to_csv('PREVISÃO DE APOSENTADORIA.csv', sep='\t', e
```

In [ ]:
```
len(ativos)
```

In [ ]:
```
ativos['REG JUR'].value_counts()
```

In [ ]:
```
191 / 2091
```

ANÁLISE DA PREVISÃO

In [ ]:
```
ativos.columns
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.countplot(x='CARREIRA', data=ativos, hue='REG JUR')
ax.set_xlabel('CARREIRA')
ax.set_ylabel('QUANTIDADE')
ax.set_title('PREVISÃO DE APOSENTADORIA POR CARREIRA', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.countplot(x='SEXO', data=ativos, hue='REG JUR')
ax.set_xlabel('SEXO')
ax.set_ylabel('QUANTIDADE')
ax.set_title('PREVISÃO DE APOSENTADORIA POR SEXO', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
fig, ax = plt.subplots(figsize=figt)
ax = sn.scatterplot(x='IDADE', y='TEMPO SERVIÇO', data=ativos, hue='REG JUR')
ax.set_xlabel('IDADE')
ax.set_ylabel('TEMPO DE SERVIÇO')
ax.set_title('PREVISÃO DE APOSENTADORIA - TEMPO DE SERVIÇO POR IDADE', fontsize=15

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
a, b, = zip(*ativos[ativos['REG JUR'] == 1]['CARREIRA'].value_counts().items())
dd = pd.DataFrame({'CARREIRA': a, 'TOTAL': b})

fig, ax = plt.subplots(figsize=figt)
ax = sn.barplot(x='CARREIRA', y='TOTAL', data=dd)
ax.set_xlabel('CARREIRA')
ax.set_ylabel('QUANTIDADE')
ax.set_title('PREVISÃO DE APOSENTADORIA POR CARGO', fontsize=15)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
a, b, = zip(*ativos[ativos['REG JUR'] == 1]['CARGO'].value_counts().items())
dd = pd.DataFrame({'CARGO': a, 'TOTAL': b})

fig, ax = plt.subplots(figsize=figt)
ax = sn.barplot(x='CARGO', y='TOTAL', data=dd[:20])
ax.set_xlabel('CARGO')
ax.set_ylabel('QUANTIDADE')
ax.set_title('PREVISÃO DE APOSENTADORIA POR CARGO (20 MAIS)', fontsize=15)
plt.xticks(rotation=90)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

In [ ]:
```
a, b, = zip(*ativos[ativos['REG JUR'] == 1]['GRUPO'].value_counts().items())
dd = pd.DataFrame({'GRUPO': a, 'TOTAL': b})

fig, ax = plt.subplots(figsize=figt)
ax = sn.barplot(x='GRUPO', y='TOTAL', data=dd[:20])
ax.set_xlabel('GRUPO')
ax.set_ylabel('QUANTIDADE')
ax.set_title('PREVISÃO DE APOSENTADORIA POR GRUPO (20 MAIS)', fontsize=15)
plt.xticks(rotation=90)

num += 1
fig.savefig('GRAF' + str(num).rjust(3,'0'), bbox_inches='tight')
```

FIM