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

Andreia Prudenciano Penedo

ANÁLISE DE DEVEDORES DO FGTS DO ESTADO DE MINAS GERAIS POR MEIO DE BASE PÚBLICA

São Paulo
2023

Andreia Prudenciano Penedo

ANÁLISE DE DEVEDORES DO FGTS DO ESTADO DE MINAS GERAIS POR MEIO DE BASE PÚBLICA

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.

São Paulo

2023

SUMÁRIO

1. Introdução

1.1. Contextualização

O FGTS, Fundo de Garantia por Tempo de Serviço, é um direito garantido aos trabalhadores registrados em carteira de trabalho. Ele foi criado pela Lei nº 5.107, de 13 de setembro de 1966 e vigente a partir de 01 de janeiro de 1967, para proteger o trabalhador demitido sem justa causa.

Porém os valores depositados nas contas dos trabalhadores são utilizados pelo Governo Federal para financiar programas de habitação, saneamento e infraestrutura.

No portal do governo é disponibilizado bases de consulta às dívidas ativas da União e do FGTS, separados por estado.

No trabalho proposto será utilizado os dados para classificar, extrair e analisar as dívidas de empresas no FGTS do estado de Minas Gerais.

1.2. O problema proposto

Seguindo a técnica proposta dos 5-Ws, será esclarecida as dúvidas relevantes:

(Why?) Por que esse problema é importante?

Conforme informado na seção anterior, o FGTS é utilizado para financiar áreas vitais do desenvolvimento do nosso País. Quanto mais empresas devem ao governo, menos recursos há para a utilização nessas áreas.

Analisar essas dívidas pode trazer um mapa geral para aplicar um plano de correção, a fim de obter maiores empresas saldando as suas respectivas dívidas. Isso aumentaria a arrecadação dos valores sem necessidade de criar, por exemplo, um imposto.

(Who?) De quem são os dados analisados? De um governo? Um ministério ou secretaria? Dados de clientes?

Os dados são disponibilizados pelo portal do Governo Federal.

(What?): Quais os objetivos com essa análise? O que iremos analisar?

O objetivo é identificar quais regiões do estado de Minas Gerais tem o maior valor em dívida no FGTS. Assim, será possível tomar ações segmentadas por áreas.

(Where?): Trata dos aspectos geográficos e logísticos de sua análise.

Os dados utilizados serão da base do Estado de Minas Gerais.

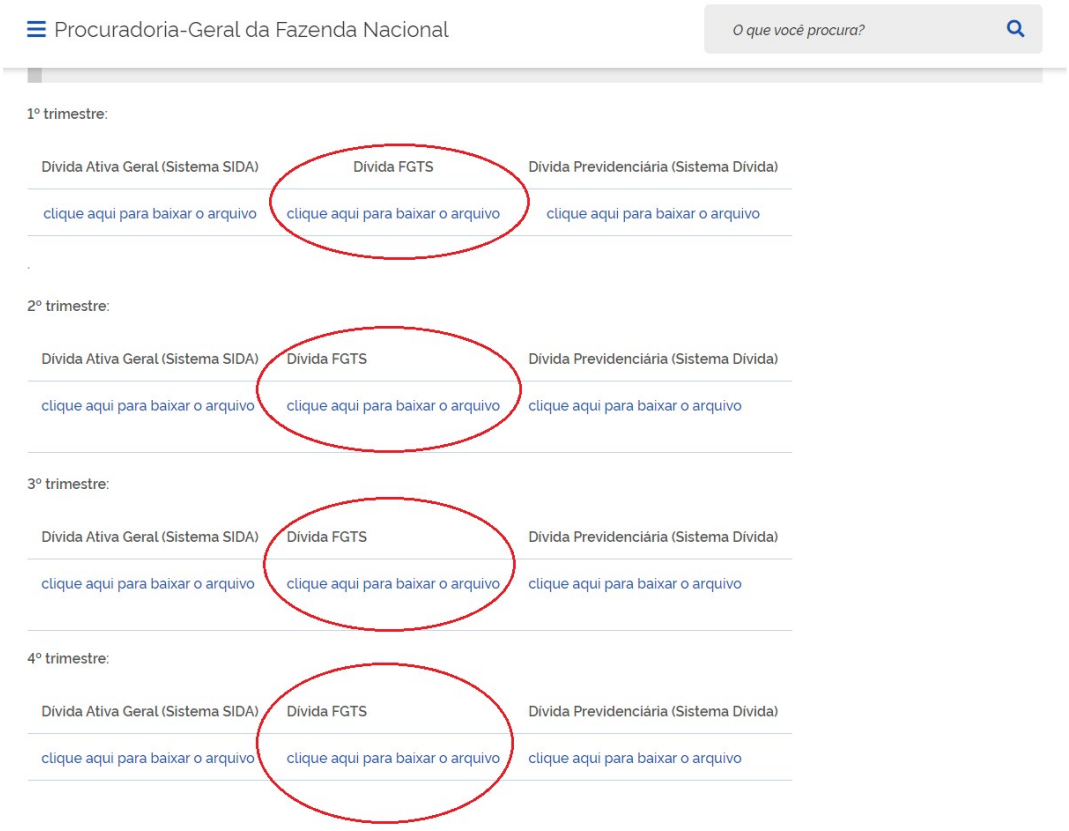
(When?): Qual o período está sendo analisado? A última semana? Os últimos 6 meses? O ano passado?

Utilizaremos a base do ano de 2022.

2. Coleta de Dados

Os dados foram obtidos por meio do portal do Governo Federal, e pode ser acessado através deste link.

Para a nossa análise de dados, utilizaremos a base de dívidas de FGTS do ano de 2022 do estado de Minas Gerais. O site do governo disponibiliza os dados por trimestre, então será necessário baixar os quatro períodos, conforme print abaixo:



O arquivo baixado está em formato compactado (.zip) e separado por estado na extensão csv. O arquivo referente ao estado de Minas Gerais está disponível no notebook para download.

O dicionário de campos é detalhado abaixo, conforme o próprio portal dispõe através deste link:

Nome da coluna/campo	Descrição	Tipo
CPF_CNPJ	Número identificador do contribuinte no cadastro de pessoas físicas ou no cadastro nacional de pessoas jurídicas	string
TIPO_PESSOA	Indica se é uma pessoa física ou jurídica	string
TIPO_DEVEDOR	Indica se o devedor é principal (titular original da dívida) ou corresponsável (foi vinculado posteriormente à dívida)	string
NOME_DEVEDOR	Nome do devedor	string
UF_UNIDADE_RESPONSAVEL	Unidade federativa da unidade da PGFN responsável pela cobrança do devedor	string
UNIDADE_RESPONSAVEL	Unidade da PGFN responsável pelo acompanhamento do devedor	string
ENTIDADE_RESPONSAVEL	Indica se o débito de FGTS está sendo cobrado pela PGFN ou pela Caixa Econômica Federal	string
UNIDADE_INSCRICAO	Indica a unidade da PGFN que realizou a inscrição em dívida ativa	string
NUMERO_INSCRICAO	Número da inscrição em dívida ativa	string
TIPO_SITUACAO_INSCRICAO	Indica se a inscrição está em cobrança (situação irregular), em benefício fiscal (em parcelamento ou moratória), em negociação, suspenso por decisão judicial, garantia (integralmente garantida)	string
SITUACAO_INSCRICAO	Situação da inscrição no sistema de controle de créditos	string
RECEITA_PRINCIPAL	Receita do crédito que está sendo cobrado	string
DATA_INSCRICAO	Data em que o crédito foi inscrito em dívida ativa	datetime (dd/mm/yyyy)
INDICADOR_AJUIZADO	Indica se o crédito está sendo cobrado judicialmente	string
VALOR_CONSOLIDADO	Valor do débito na data de extração, com acréscimos legais	float

3. Processamento/Tratamento de Dados

Como os dados estão separados por trimestre, agruparemos as informações para tratamento único e consolidado.

Para cada período será efetuado os seguintes passos para tratamento:

- Importar base do trimestre
- Remover registros duplicados
- Adicionar coluna com a informação do período importado

Por fim consolidaremos os registros em um único arquivo.

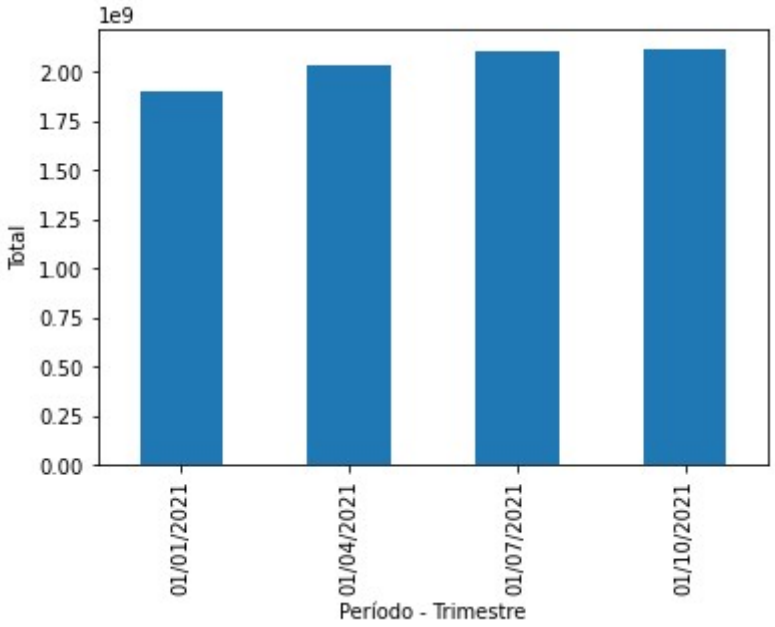
4. Análise e Exploração dos Dados

A análise dos dados será feita com a linguagem Phyton através da ferramenta Jupyter Notebook e todos os arquivos criados serão disponibilizados em links.

Após a unificação das informações em um único arquivo, poderemos segmentar as informações por diversas categorias, conforme disposto no arquivo 2_Analises_Base_Consolidada.ipynb:

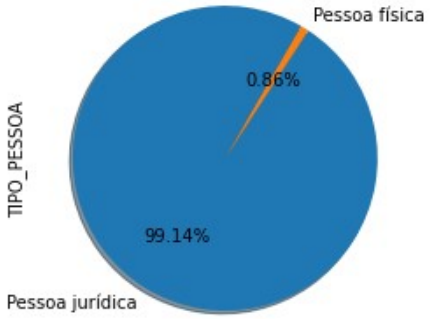
4.1.1 Por Período

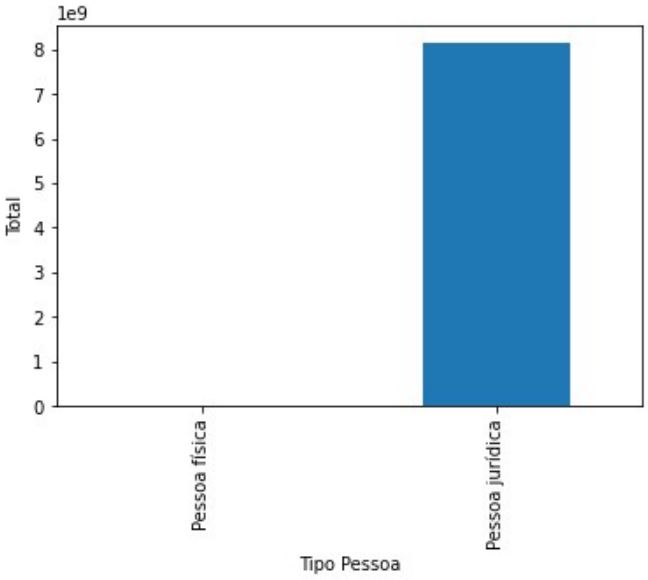
No gráfico abaixo conseguimos visualizar que todos os períodos estão com valores sem muita discrepância porém é igualmente crescente.



4.1.2 Por tipo de pessoa

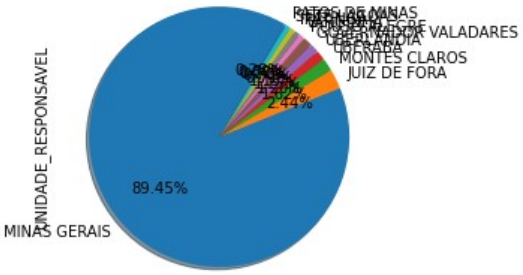
Nos próximos gráficos conseguimos separar o total de dívida por tipo de pessoa e seu respectivo percentual. Desta forma identificamos que a grande maioria dos devedores são de pessoas jurídicas.

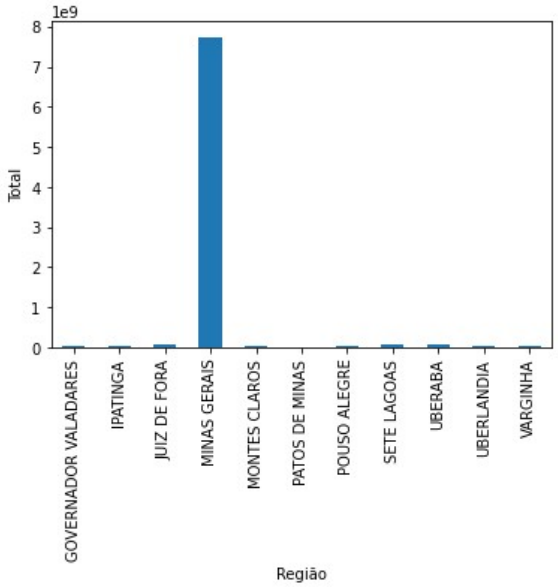




4.1.3 Por Região

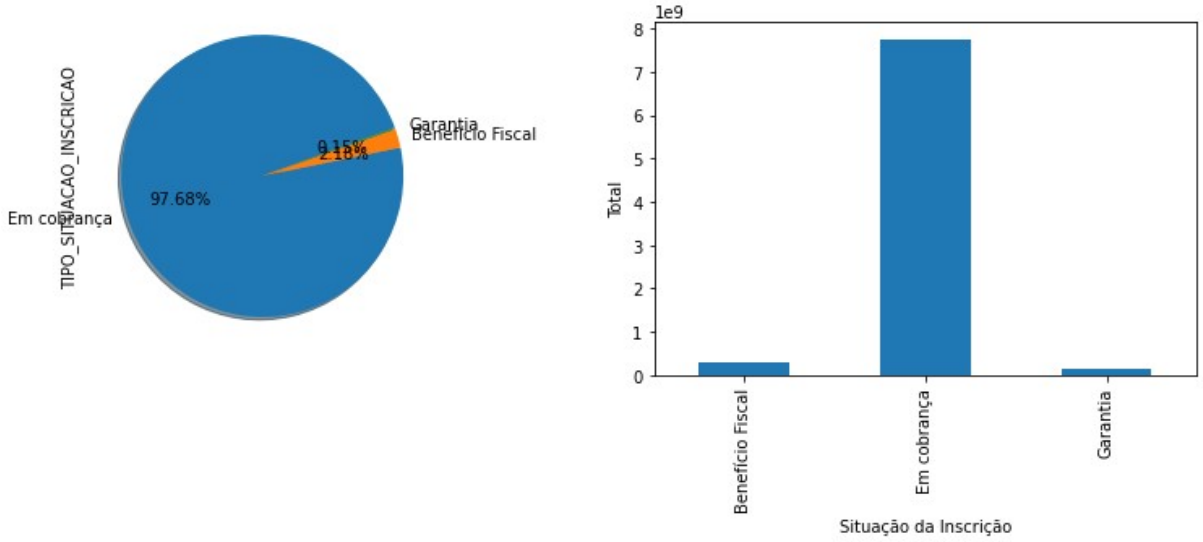
Nos gráficos a seguir temos o percentual e total por região. Notamos então que a região Minas Gerais representa a maior parte das dívidas de FGTS.





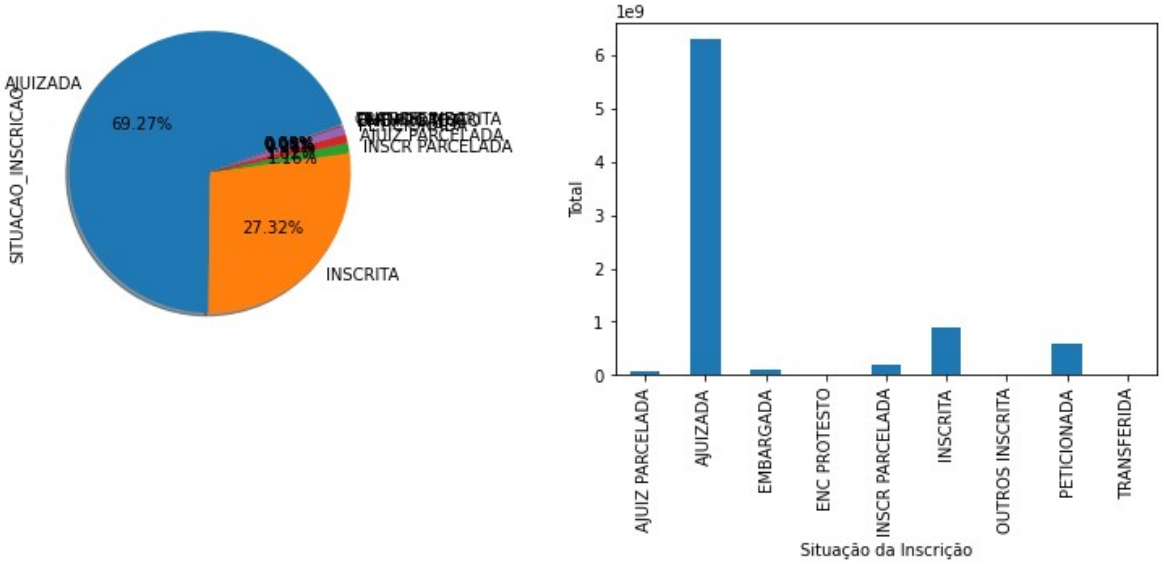
4.1.4 Por tipo da situação de inscrição

Nestes gráficos também temos o percentual e total por tipo da situação da inscrição. O tipo Em cobrança, ou situação irregular, representa a maior parte da dívida.



4.1.5 Por situação da inscrição

Os gráficos a seguir representam o percentual e total da situação da inscrição no sistema de controle de créditos. A situação Ajuizada representa a maior parte das dívidas.



5. Criação de Modelos de Machine Learning

Usaremos para nossa análise de dados os modelos Scatter Plot, Multiple Regression e Line Regression.

Seguiremos os passos, detalhados no arquivo 3_Machine_Learning.ipynb, para geração dos nossos modelos. As primeiras etapas serão destinadas à configuração básica do ambiente:

1- Importação dos pacotes e classes necessários

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings('ignore')

from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_absolute_percentage_error
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import RandomizedSearchCV

pd.options.display.max_columns = 100
pd.options.display.max_rows = 500

from pandas.tseries.offsets import DateOffset
from scipy.stats import f_oneway
from scipy import stats
```

2- Importação do dataset consolidado

```
## Importando a base Consolidada
dividasMG = pd.read_csv(r'O:\OneDrive\ANDREIA\MATERIAL_ESTUDOS\PUC\TCC\Datasets\Consolidado2021.csv', sep=';', parse_dates=True,
```

3- Exibição dos dados importados

```
dividasMG.head()
```

	PERIODO	CPF_CNPJ	TIPO_PESSOA	TIPO_DEVEDOR	NOME_DEVEDOR	UF_UNIDADE_RESPONSAVEL	UNIDADE_RESPONSAVEL	ENTIDADE_RESPONS
0	01/01/2021	06.286.147/0001-76	Pessoa jurídica	Principal	ART BRINDES INDUSTRIA E COMERCIO EIRELI	MG	UBERABA	
1	01/01/2021	06.286.147/0001-76	Pessoa jurídica	Principal	ART BRINDES INDUSTRIA E COMERCIO EIRELI	MG	UBERABA	
2	01/01/2021	17.268.863/0001-47	Pessoa jurídica	Principal	JOAQUIM GERVASIO COUTINHO	MG	MINAS GERAIS	
3	01/01/2021	19.376.161/0001-02	Pessoa jurídica	Principal	HMAXX INDUSTRIA E COMERCIO DE ALIMENTOS E ADOC...	MG	MINAS GERAIS	
4	01/01/2021	05.517.165/0001-59	Pessoa jurídica	Principal	ANDREA DA SILVA VIEIRA	MG	MINAS GERAIS	

4- Definição do dado utilizado para nossa análise

```
df = dividasMG[['VALOR_CONSOLIDADO']]
```

```
agrupPeriodo.first()
```

PERIODO	CPF_CNPJ	TIPO_PESSOA	TIPO_DEVEDOR	NOME_DEVEDOR	UF_UNIDADE_RESPONSAVEL	UNIDADE_RESPONSAVEL	ENTIDADE_RESPONSAV
01/01/2021	06.286.147/0001-76	Pessoa jurídica	Principal	ART BRINDES INDUSTRIA E COMERCIO EIRELI	MG	UBERABA	PG
01/04/2021	22.905.919/0001-67	Pessoa jurídica	Principal	FUJI LANCHES LTDA	MG	MINAS GERAIS	PG
01/07/2021	03.200.004/0001-01	Pessoa jurídica	Principal	CONDOMINIO ESPECIAL BAHIA SHOPPING	MG	MINAS GERAIS	PG
01/10/2021	05.667.429/0001-50	Pessoa jurídica	Principal	CEPAC CENTRAL PAULISTA DE COUROS LTDA	MG	MINAS GERAIS	PG

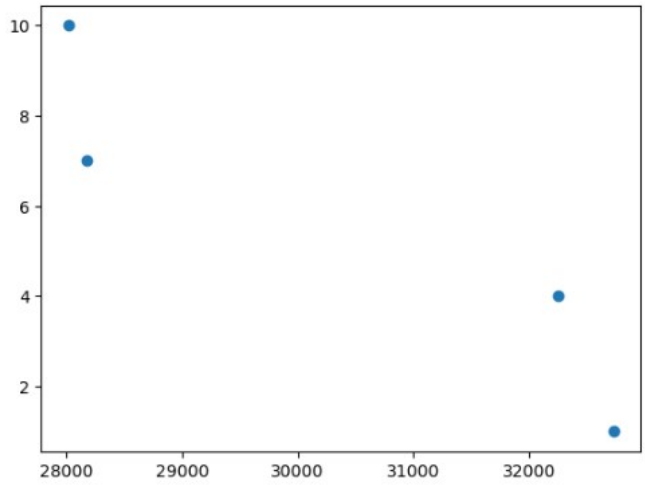
```
## Definindo as variaveis a serem utilizadas
x = dividasMG['PERIODO'].value_counts().tolist()
y = pd.DatetimeIndex(dividasMG.groupby(['PERIODO']).groups.keys()).day
```

```
##Simplificando Dataframe com campos mais importantes para análise
data = { 'Periodo': y,
         'QtdePorPeriodo': x,
         'TotalPorPeriodo': totalPorPeriodo
       }

newDf = pd.DataFrame(data)
print(newDf)
```

```
   Periodo  QtdePorPeriodo  TotalPorPeriodo
0        1           32728     1.902615e+09
1        4           32244     2.028318e+09
2        7           28181     2.101908e+09
3       10           28022     2.115273e+09
```

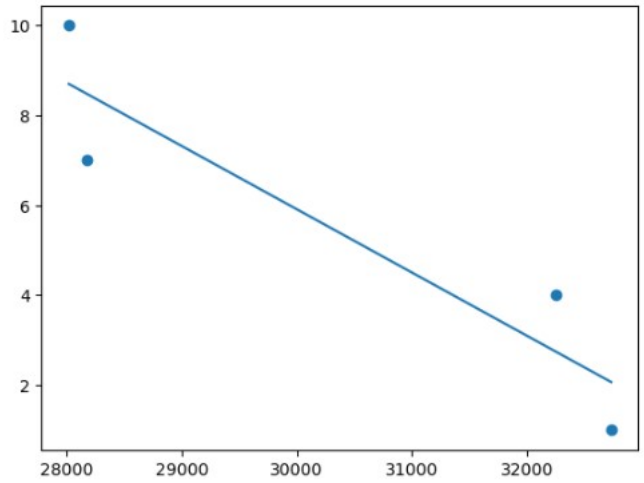
5.1. Scatter Plot

```
##Scatter Plot
plt.scatter(x, y)
plt.show()
```



5.2. Linear Regression

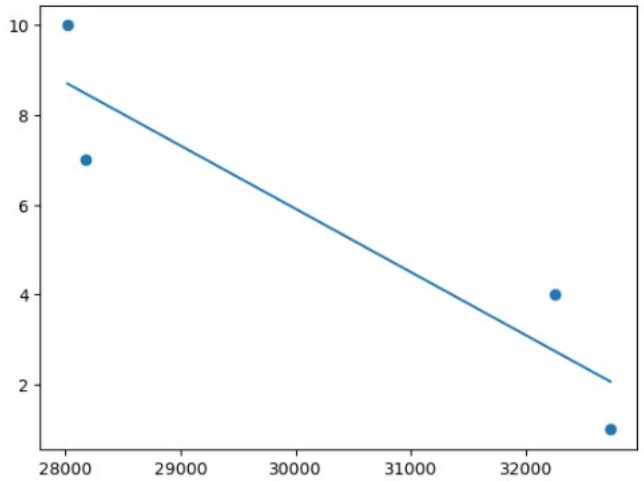
```
##Linear Regression
slope, intercept, r, p, std_err = stats.linregress(x, y)

def myfunc(x):
  return slope * x + intercept

mymodel = list(map(myfunc, x))

plt.scatter(x, y)
plt.plot(x, mymodel)
plt.show()
```




```
##Linear Regression - Prediction
##Quantidade de devedores por periodo em 5 anos
slope, intercept, r, p, std_err = stats.linregress(x, y)

def myfunc(x):
  return slope * x + intercept

speed = myfunc(5)

print(speed)
```

```
48.18039394074245
```

```
slope, intercept, r, p, std_err = stats.linregress(x, y)

def myfunc(x):
  return slope * x + intercept

mymodel = list(map(myfunc, x))

plt.scatter(x, y)
plt.plot(x, mymodel)
plt.show()
```



```
slope, intercept, r, p, std_err = stats.linregress(x, y)

print(r)
```

```
-0.9241060008021352
```

5.3. Multiple Regression

```
##Multiple Regression
totalPorPeriodo = round(dividasMG.groupby(['PERIODO'])['VALOR_CONSOLIDADO'].agg('sum'), 2).tolist()
```

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

```
data_pred = 30;

df['Prediction'] = df[['VALOR_CONSOLIDADO']]
```

```
X = np.array(df.drop(['Prediction'], axis=1))
X = X[:-data_pred]
```

```
y = np.array(df['Prediction'])
y = y[:-data_pred]
```

```
##Teste de 30%
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)
```

```
x_pred = np.array(df.drop(['Prediction'], 1))[-data_pred: ]
```

```
lr = LinearRegression()
lr.fit(X_train, y_train)
```

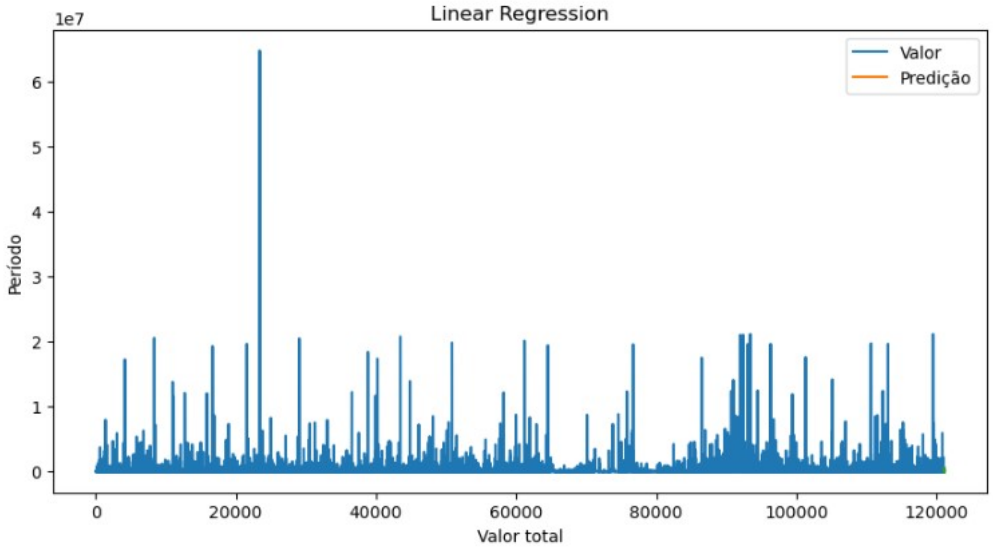
```
▾ LinearRegression
LinearRegression()
```

```
teste_modelo = lr.score(X_test, y_test)
print(teste_modelo)
```

```
1.0
```

```
lr_pred = lr.predict(x_pred)
```

```
predictions = lr_pred

valid = df[X.shape[0]:]
valid['Predictions'] = predictions
plt.figure(figsize=(10,5))
plt.title('Linear Regression')
plt.xlabel('Valor total')
plt.ylabel('Período')
plt.plot(df['VALOR_CONSOLIDADO'][0:])
plt.plot(valid[['VALOR_CONSOLIDADO', 'Predictions']])
plt.legend(['Valor', 'Predição'])
plt.show()
```

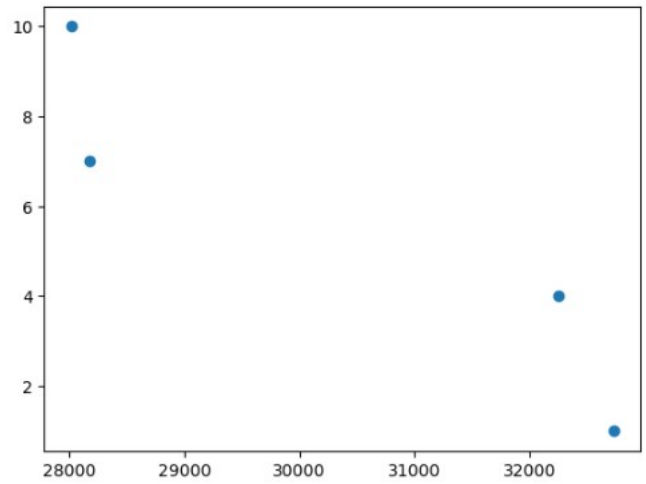


6. Interpretação dos Resultados

A primeira observação que foi possível foi a necessidade de aplicar mais variáveis de tempo em nossas análises. A utilização apenas do período de 12 meses não foi suficiente para obtermos gráficos mais apurados, devido ao tratamento em períodos das bases disponibilizadas.
Entretanto a utilização de mais de 1 ano de registros acarretaria em um processamento mais extenso e oneroso visto que a quantidade de registros seria demasiadamente grande.

No nosso primeiro gráfico, do modelo Scatter Plot, verificamos o eixo x que representa a quantidade total de registros por período e o eixo y que lista os quatros períodos do ano.

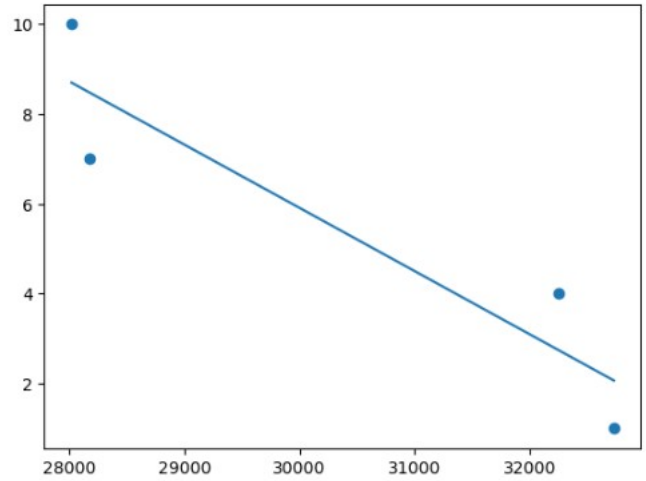
```
##Scatter Plot
plt.scatter(x, y)
plt.show()
```



Entendemos que os últimos períodos do ano foram os que tiveram menor quantidade de registros. Entretanto, relembrando o gráfico do item 4.1.1, onde verificamos que o valor de débito é crescente, chegamos a conclusão que há menos registros nos últimos períodos mas com totais superiores.

No modelo Linear Regression usamos uma linha entre os pontos encontrados para prever os valores futuros. Usando os métodos slope e intercept teremos novos eixos x e y para gerarmos uma nova matriz de valores.

```
##Linear Regression
slope, intercept, r, p, std_err = stats.linregress(x, y)

def myfunc(x):
  return slope * x + intercept

mymodel = list(map(myfunc, x))

plt.scatter(x, y)
plt.plot(x, mymodel)
plt.show()
```

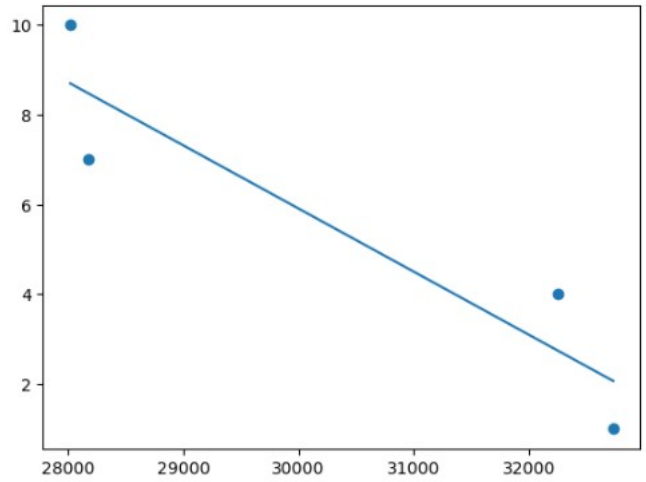


Utilizando o modelo abaixo, fazemos a predição da dívida em 5 anos:

```
##Linear Regression - Prediction
##Quantidade de devedores por periodo em 5 anos
slope, intercept, r, p, std_err = stats.linregress(x, y)

def myfunc(x):
  return slope * x + intercept

speed = myfunc(5)

print(speed)
```

```
48.18039394074245
```

```
slope, intercept, r, p, std_err = stats.linregress(x, y)

def myfunc(x):
  return slope * x + intercept

mymodel = list(map(myfunc, x))

plt.scatter(x, y)
plt.plot(x, mymodel)
plt.show()
```

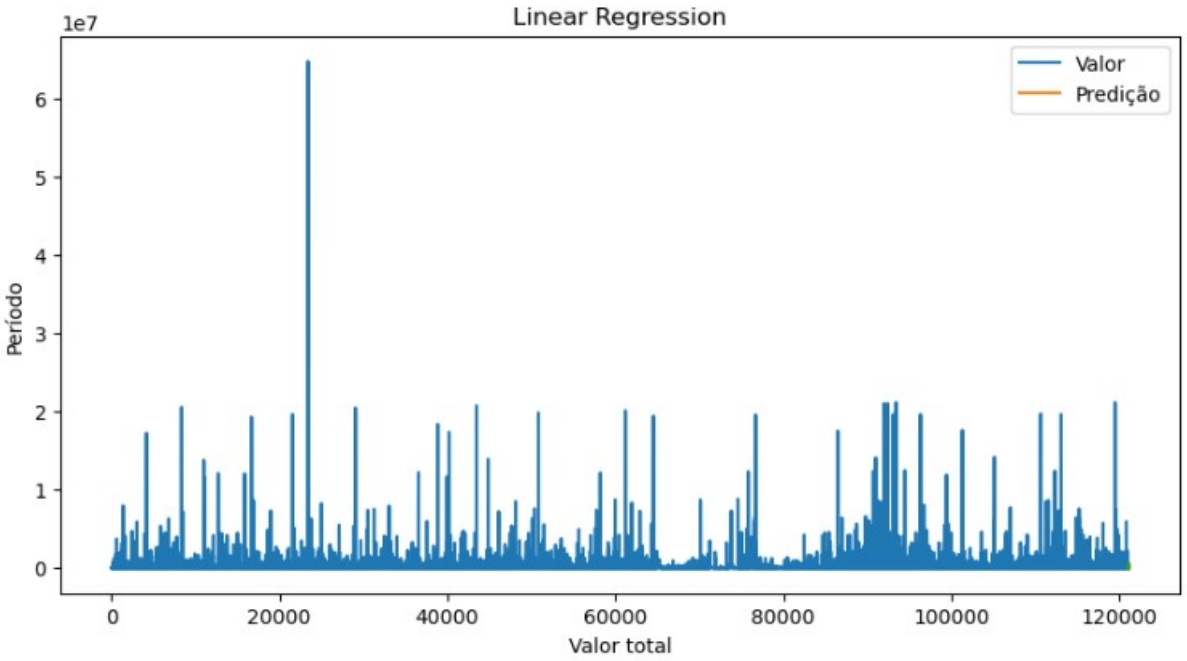


Através dos eixos x e y podemos identificar a relação linear de valores. Se retornar valor 0 desta relação, chamada de coeficiente de correlação (r), significa nenhum relacionamento e nada pode ser previsto. Já o retorno entre 1 e -1 significa 100% relacionado, conforme podemos constatar abaixo:

```
slope, intercept, r, p, std_err = stats.linregress(x, y)

print(r)
```

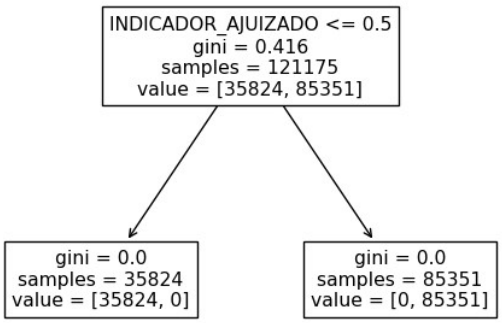
```
-0.9241060008021352
```

No grafico abaixo apresentamos o resultado da Linear Regression, onde podemos observar que a predição é praticamente imperceptível ao final do total, no eixo x. Novamente ressaltando que apenas o periodo de 12 meses não foi suficiente para uma análise mais apurada.



No arquivo 4_Arvore_Decisao.ipynb está brevemente exemplificado o modelo Árvore de Decisão, utilizando os dados TIPO_SITUACAO_INSCRICAO e INDICADOR_AJUIZADO.

```
%matplotlib inline

d = {'Benefício Fiscal': 0, 'Em cobrança': 1, 'Garantia': 2}
df['TIPO_SITUACAO_INSCRICAO'] = df['TIPO_SITUACAO_INSCRICAO'].map(d)
d = {'SIM': 1, 'NAO': 0}
df['INDICADOR_AJUIZADO'] = df['INDICADOR_AJUIZADO'].map(d)

features = ['TIPO_SITUACAO_INSCRICAO', 'INDICADOR_AJUIZADO']

X = df[features]
y = df['INDICADOR_AJUIZADO']

dtree = DecisionTreeClassifier()
dtree = dtree.fit(X, y)

tree.plot_tree(dtree, feature_names=features)
sys.stdout.flush()
```



7. Apresentação dos Resultados

Para este capítulo, devemos escolher um modelo de workflow para nossa apresentação dos resultados. E foi eleito o modelo proposto por Vasandani:

Título: Análise de devedores do FGTS por meio de base pública	
Definição do problema	Analisar a base de devedores do FGTS do estado de Minas Gerais.
Resultados e previsões	Identificar tipos de devedores, períodos de maior valor de dívida e respectivas regiões.
Aquisição de dados	Os datasets foram obtidos pelo portal do Governo Federal, disposta de forma pública e gratuita.
Modelagem	Aplicamos alguns tratamentos utilizando algumas bibliotecas como por exemplo o Sklearn, para tratar alguns dados e facilitar a consulta e análise dos registros.
Avaliação do modelo	Os resultados foram obtidos e analisados por relatórios de matriz de confusão.
Preparação dos dados	Alguns dados não foram relevantes para a nossa análise, como por exemplo o CNPJ, pois são dados extremamente específicos. Nestes casos ignoramos e apenas utilizamos as informações que se mostraram mais interessantes.

8. Links

Link para o vídeo: https://youtu.be/LZUm6t1VWmk

Link para o repositório: https://github.com/andreiappenedo/TCC_PUC

REFERÊNCIAS

Dados abertos do Governo Federal: https://www.gov.br/pgfn/pt-br/assuntos/divida-ativa-da-uniao/dados-abertos/dados-abertos

W3Schools: https://www.w3schools.com/python

Scikit-Learn: https://scikit-learn.org/

Jasmine Vasandani: https://towardsdatascience.com/a-data-science-workflow-canvas-to-kickstart-your-projects-db62556be4d0

Geeksforgeeks: https://www.geeksforgeeks.org/python-programming-language/

Realpython: https://realpython.com/

Shanelynn: https://www.shanelynn.ie/

Python: https://www.python.org/

Kaggle: https://www.kaggle.com/learn/python

APÊNDICE

Programação/Scripts

1_Importando_Bases_Exportando_Consolidado.ipynb

```
import pandas as pd
import numpy as np

## Importar a base do Primeiro Trimestre de 2021 ##
primeiroTri2021 =
pd.read_csv(r'O:\OneDrive\ANDREIA\MATERIAL_ESTUDOS\PUC\TCC\Datasets\arquivo_lai_FGTS_MG_202103.csv
', sep=';', parse_dates=True, encoding='iso-8859-1')
primeiroTri2021.shape

# Remove duplicadas
primeiroTri2021 = primeiroTri2021.drop_duplicates()

## Adiciona coluna para identificar o período do dataframe ##
primeiroTri2021.insert(0, "PERIODO", '01/01/2021', True)
primeiroTri2021.info()
primeiroTri2021.head()
## Importar a base do Segundo Trimestre de 2021 ##
segundoTri2021 =
pd.read_csv(r'O:\OneDrive\ANDREIA\MATERIAL_ESTUDOS\PUC\TCC\Datasets\arquivo_lai_FGTS_MG_202106.csv
', sep=';', parse_dates=True, encoding='iso-8859-1')
segundoTri2021.shape
segundoTri2021 = segundoTri2021.drop_duplicates()

## Adiciona coluna para identificar o período do dataframe ##
segundoTri2021.insert(0, "PERIODO", '01/04/2021', True)
segundoTri2021.info()
segundoTri2021.head()

## Importar a base do Terceiro Trimestre de 2021 ##
terceiroTri2021 =
pd.read_csv(r'O:\OneDrive\ANDREIA\MATERIAL_ESTUDOS\PUC\TCC\Datasets\arquivo_lai_FGTS_MG_202109.csv
', sep=';', parse_dates=True, encoding='iso-8859-1')
terceiroTri2021.shape
terceiroTri2021 = terceiroTri2021.drop_duplicates()

## Adiciona coluna para identificar o período do dataframe ##
```

```
terceiroTri2021.insert(0, "PERIODO", '01/07/2021', True)
terceiroTri2021.info()
terceiroTri2021.head()

## Importar a base do Quarto Trimestre de 2021 ##
quartoTri2021 =
pd.read_csv(r'O:\OneDrive\ANDREIA\MATERIAL_ESTUDOS\PUC\TCC\Datasets\arquivo_lai_FGTS_MG_202112.csv
', sep=';', parse_dates=True, encoding='iso-8859-1')
quartoTri2021.shape
quartoTri2021 = quartoTri2021.drop_duplicates()

## Adiciona coluna para identificar o período do dataframe ##
quartoTri2021.insert(0, "PERIODO", '01/10/2021', True)
quartoTri2021.info()
quartoTri2021.head()

## Concatena os 4 períodos em um único dataframe ##
consolidado = pd.concat([primeiroTri2021, segundoTri2021, terceiroTri2021, quartoTri2021])

## Exporta para csv para trabalharmos a análise dos dados ##
consolidado.to_csv(r'O:\OneDrive\ANDREIA\MATERIAL_ESTUDOS\PUC\TCC\Datasets\Consolidado2021.cs
v', sep=';', index=False, encoding='iso-8859-1')
```

2_Analises_Base_Consolidada.ipynb

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

## Importando a base Consolidada
dividasMG =
pd.read_csv(r'O:\OneDrive\ANDREIA\MATERIAL_ESTUDOS\PUC\TCC\Datasets\Consolidado2021.csv', sep=';',
parse_dates=True, encoding='iso-8859-1')

dividasMG.info()

## Somar as dividas por periodo
dividasMG.groupby('PERIODO')['VALOR_CONSOLIDADO'].sum().plot.bar()

plt.xlabel('Período - Trimestre')
plt.ylabel('Total')

## Os valores são representados na ordem de milhão
```

```
print(dividasMG.groupby('PERIODO')['VALOR_CONSOLIDADO'].sum())

## Total por TIPO_PESSOA
dividasMG.TIPO_PESSOA.value_counts().plot(kind='pie', autopct='%1.2f%%', shadow=True, startangle=60)

## Total da divida por TIPO_PESSOA, para saber se pessoa juridica ou fisica deve mais
dividasMG.groupby('TIPO_PESSOA')['VALOR_CONSOLIDADO'].sum().plot.bar()
plt.xlabel('Tipo Pessoa')
plt.ylabel('Total')

## Total por UNIDADE_RESPONSAVEL
dividasMG.UNIDADE_RESPONSAVEL.value_counts().plot(kind='pie', autopct='%1.2f%%', shadow=True,
startangle=60)

## Total da divida por UNIDADE_RESPONSAVEL
dividasMG.groupby('UNIDADE_RESPONSAVEL')['VALOR_CONSOLIDADO'].sum().plot.bar()

plt.xlabel('Região')
plt.ylabel('Total')

## Total por TIPO_SITUACAO_INSCRICAO
dividasMG.TIPO_SITUACAO_INSCRICAO.value_counts().plot(kind='pie', autopct='%1.2f%%', shadow=True,
startangle=20)

## Total da divida por TIPO_SITUACAO_INSCRICAO
dividasMG.groupby('TIPO_SITUACAO_INSCRICAO')['VALOR_CONSOLIDADO'].sum().plot.bar()

plt.xlabel('Tipo da Situação da Inscrição')
plt.ylabel('Total')

## Total por SITUACAO_INSCRICAO
dividasMG.SITUACAO_INSCRICAO.value_counts().plot(kind='pie', autopct='%1.2f%%', shadow=True,
startangle=20)

## Total da divida por SITUACAO_INSCRICAO
dividasMG.groupby('SITUACAO_INSCRICAO')['VALOR_CONSOLIDADO'].sum().plot.bar()

plt.xlabel('Situação da Inscrição')
plt.ylabel('Total')

## Total por RECEITA_PRINCIPAL
```

```
dividasMG.RECEITA_PRINCIPAL.value_counts().plot(kind='pie', autopct='%1.2f%%', shadow=True,
startangle=20)

## Total por INDICADOR_AJUIZADO
dividasMG.groupby('INDICADOR_AJUIZADO')['VALOR_CONSOLIDADO'].sum().plot.bar()

plt.xlabel('Indicador Ajuizado')
plt.ylabel('Total')
```

3_Machine_Learning.ipynb

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings('ignore')

from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_absolute_percentage_error
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import RandomizedSearchCV

pd.options.display.max_columns = 100
pd.options.display.max_rows = 500

from pandas.tseries.offsets import DateOffset
from scipy.stats import f_oneway
from scipy import stats

## Importando a base Consolidada
  dividasMG =
pd.read_csv(r'O:\OneDrive\ANDREIA\MATERIAL_ESTUDOS\PUC\TCC\Datasets\Consolidado2021.csv',
sep=';', parse_dates=True, encoding='iso-8859-1')

dividasMG.head()
df = dividasMG[['VALOR_CONSOLIDADO']]
agrupPeriodo.first()
```

```
## Definindo as variaveis a serem utilizadas
x = dividasMG['PERIODO'].value_counts().tolist()
y = pd.DatetimeIndex(dividasMG.groupby(['PERIODO']).groups.keys()).day
print(x)

##Simplificando Dataframe com campos mais importantes para análise
data = { 'Periodo': y,
       'QtdePorPeriodo': x,
       'TotalPorPeriodo': totalPorPeriodo
     }

newDf = pd.DataFrame(data)
print(newDf)

##Scatter Plot
plt.scatter(x, y)
plt.show()

##Linear Regression
slope, intercept, r, p, std_err = stats.linregress(x, y)

def myfunc(x):
  return slope * x + intercept

mymodel = list(map(myfunc, x))

plt.scatter(x, y)
plt.plot(x, mymodel)
plt.show()

##Linear Regression - Prediction
##Quantidade de devedores por periodo em 5 anos
slope, intercept, r, p, std_err = stats.linregress(x, y)

def myfunc(x):
  return slope * x + intercept

speed = myfunc(5)
print(speed)

slope, intercept, r, p, std_err = stats.linregress(x, y)
```

```
def myfunc(x):
  return slope * x + intercept

mymodel = list(map(myfunc, x))

plt.scatter(x, y)
plt.plot(x, mymodel)
plt.show()

slope, intercept, r, p, std_err = stats.linregress(x, y)
print(r)

##Multiple Regression
totalPorPeriodo = round(dividasMG.groupby(['PERIODO'])['VALOR_CONSOLIDADO'].agg('sum'), 2).tolist()

from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

data_pred = 30;

df['Prediction'] = df[['VALOR_CONSOLIDADO']]

X = np.array(df.drop(['Prediction'], axis=1))
X = X[:-data_pred]

y = np.array(df['Prediction'])
y = y[:-data_pred]

##Teste de 30%
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)
x_pred = np.array(df.drop(['Prediction'], 1))[-data_pred: ]

lr = LinearRegression()
lr.fit(X_train, y_train)

teste_modelo = lr.score(X_test, y_test)
print(teste_modelo)

lr_pred = lr.predict(x_pred)
predictions = lr_pred

valid = df[X.shape[0]:]
```

```
valid['Predictions'] = predictions
plt.figure(figsize=(10,5))
plt.title('Linear Regression')
plt.xlabel('Valor total')
plt.ylabel('Período')
plt.plot(df['VALOR_CONSOLIDADO'][0:])
plt.plot(valid[['VALOR_CONSOLIDADO', 'Predictions']])
plt.legend(['Valor', 'Predição'])
plt.show()
```

4_Arvore_Decisao.ipynb

```
import pandas as pd
import numpy as np
import sys
import matplotlib
matplotlib.use('Agg')

import pandas
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
import matplotlib.pyplot as plt

## Importando a base Consolidada
  df =
pd.read_csv(r'O:\OneDrive\ANDREIA\MATERIAL_ESTUDOS\PUC\TCC\Datasets\Consolidado2021.csv',
sep=';', parse_dates=True, encoding='iso-8859-1')

df = df.replace([np.inf, -np.inf], np.nan)
df = df.dropna()
df = df.reset_index()

%matplotlib inline

d = {'Benefício Fiscal': 0, 'Em cobrança': 1, 'Garantia': 2}
df['TIPO_SITUACAO_INSCRICAO'] = df['TIPO_SITUACAO_INSCRICAO'].map(d)
d = {'SIM': 1, 'NAO': 0}
df['INDICADOR_AJUIZADO'] = df['INDICADOR_AJUIZADO'].map(d)

features = ['TIPO_SITUACAO_INSCRICAO', 'INDICADOR_AJUIZADO']

X = df[features]
```

```
y = df['INDICADOR_AJUIZADO']

dtree = DecisionTreeClassifier()
dtree = dtree.fit(X, y)

tree.plot_tree(dtree, feature_names=features)
sys.stdout.flush()
```