

**PROJETO:** PREVENDO DIABETES EM MULHERES

**DISCENTE:** GUILHERME ALVES ()

**DISCENTE:** VINICIUS DE SOUZA SANTOS(BI3008061)

**DOCENTE:** DR. MURILO VARGES DA SILVA

**DISCIPLINA:** MDAEL8 – MINERAÇÃO DE DADOS

# LISTA DE FIGURAS

[Figura 1: Base de dados de prevenção de diabete em mulheres 4](#_Toc130424055)

[Figura 2: Saída do DataCleaning do projeto. 7](#_Toc130424056)

[Figura 3: Saida do algoritmo de Normalização 9](#_Toc130424057)

[Figura 4: Gráficos de Dispersão entre o Resultado e a Espessura da Pele 15](#_Toc130424058)

[Figura 5: Gráficos de setores entre o Resultado e o número de gestações 16](#_Toc130424059)

[Figura 6: Gráficos de Histograma entre o resultado e o IMC 17](#_Toc130424060)

Sumário

[LISTA DE FIGURAS 2](#_Toc130424020)

[1 Objetivo 2](#_Toc130424021)

[1.1 Sobre a base de dados 3](#_Toc130424022)

[2 Pré-processamento 4](#_Toc130424023)

[3 Normalização e Redução de Dados 8](#_Toc130424024)

[3.1 Normalização dos dados: 8](#_Toc130424025)

[3.2 Redução dos dados: 10](#_Toc130424026)

[4 Análise descritiva de dados – Visualização 12](#_Toc130424027)

[4.1 Gráficos em Histograma, Dispersão e Setores 13](#_Toc130424028)

[5 Análise descritiva de dados – Medidas 18](#_Toc130424029)

[Referências Bibliográficas 19](#_Toc130424030)

# 1 Objetivo

Estimular o aluno a aplicar os conhecimentos apresentados no decorrer da disciplina em problemas reais de mineração de dados, utilizando as técnicas de seleção, pré-processamento e transformação de dados, técnicas de visualização de dados, análise descritiva, análise de grupos, classificação e estimação/regressão.

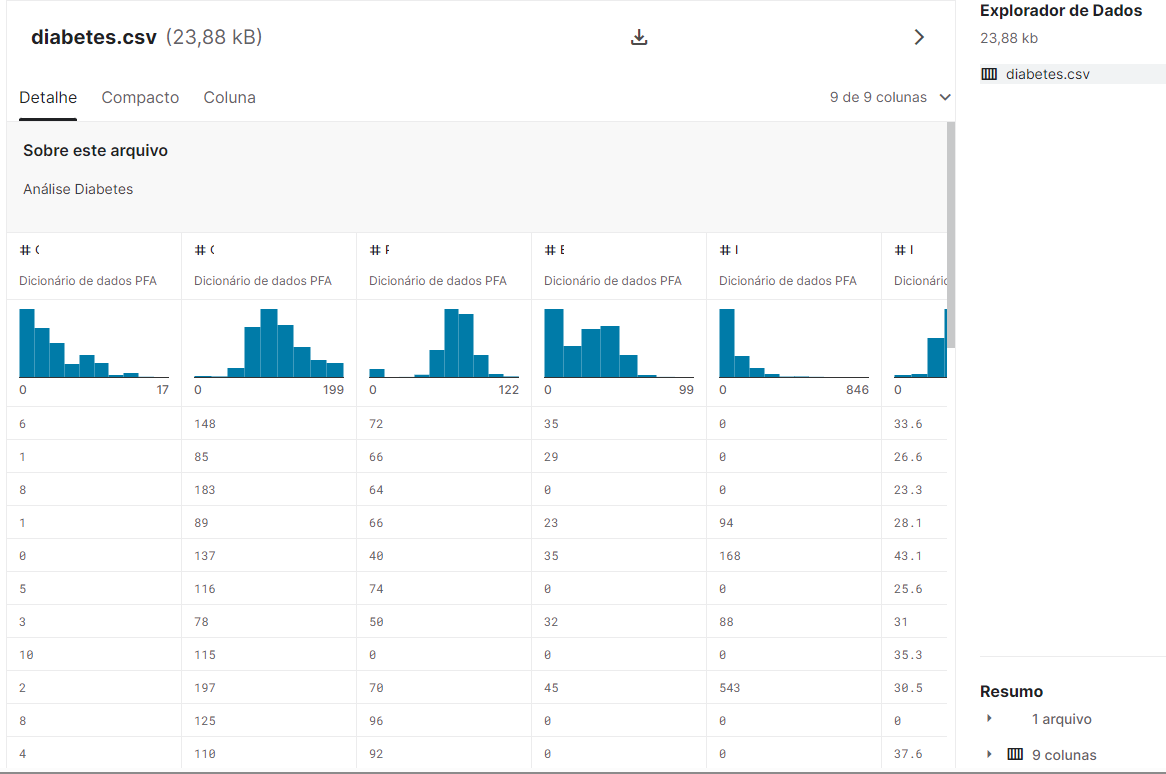
## 1.1 Sobre a base de dados

Este conjunto de dados é originalmente do Instituto Nacional de Diabetes e Doenças Digestivas e Renais. O objetivo do conjunto de dados é prever de forma diagnóstica se um paciente tem diabetes, com base em certas medidas de diagnóstico incluídas no conjunto de dados. Várias restrições foram colocadas na seleção dessas instâncias de um banco de dados maior. Em particular, todos os pacientes aqui são mulheres com pelo menos 21 anos de idade, descendentes dos índios Pima.2 A partir dos dados do arquivo (.csv) Podemos encontrar várias variáveis, algumas delas são independentes (várias variáveis reditivas médicas) e apenas uma variável dependente de destino (resultado).

# 2 Pré-processamento

Nessa primeira etapa do projeto foi feito o pré-processamento dos dados da base de prevenção a diabetes, essa base de dados foi disponibilizada pelo site da Kaggle nela foi abstraída informações relevantes como: Número de gestações, insulina, IMC, espessura da pele entre outros, com essa base de dados vamos fazer a preparação desses dados, conforme observado na figura 1 apresentando a base de dados analisada.

Figura 1: Base de dados de prevenção de diabete em mulheres



fonte: CHAUHAN, 2022

Na aula foi apresentado através de um repositório do professor vários algoritmos de mineração de dados e para esse capítulo usamos apenas o do diretório de processamento.

Para fazer o pré-processamento usamos o algoritmo DataCleaning.py.

import pandas as pd

import numpy as np

def main():

    # Faz a leitura do arquivo

    names = ['Número Gestações','Glucose','pressao Arterial','Expessura da Pele','Insulina','IMC','Função Pedigree Diabete','Idade','Resultado']

    features = ['Número Gestações','Glucose','pressao Arterial','Expessura da Pele','Insulina','IMC','Função Pedigree Diabete','Idade','Resultado']

    output\_file = '0-Datasets/diabetesClear.data'

    input\_file = '0-Datasets/diabetes.data'

    df = pd.read\_csv(input\_file,         # Nome do arquivo com dados

                     names = names,      # Nome das colunas

                     usecols = features, # Define as colunas que serão  utilizadas

                     na\_values='?')      # Define que ? será considerado valores ausentes

    df\_original = df.copy()

    # Imprime as 20 primeiras linhas do arquivo

    print("PRIMEIRAS 20 LINHAS\n")

    print(df.head(20))

    print("\n")

    # Imprime informações sobre dos dados

    print("INFORMAÇÕES GERAIS DOS DADOS\n")

    print(df.info())

    print("\n")

    # Imprime uma analise descritiva sobre dos dados

    print("DESCRIÇÃO DOS DADOS\n")

    print(df.describe())

    print("\n")

    # Imprime a quantidade de valores faltantes por coluna

    print("VALORES FALTANTES\n")

    print(df.isnull().sum())

    print("\n")

    columns\_missing\_value = df.columns[df.isnull().any()]

    print(columns\_missing\_value)

    method = 'mean' # number or median or mean or mode

    for c in columns\_missing\_value:

        UpdateMissingValues(df, c, method)

    print(df.describe())

    print("\n")

    print(df.head(15))

    print(df\_original.head(15))

    print("\n")

    # Salva arquivo com o tratamento para dados faltantes

    df.to\_csv(output\_file, header=False, index=False)

def UpdateMissingValues(df, column, method="mean", number=0):

    if method == 'number':

        # Substituindo valores ausentes por um número

        df[column].fillna(number, inplace=True)

    elif method == 'median':

        # Substituindo valores ausentes pela mediana

        median = df['Density'].median()

        df[column].fillna(median, inplace=True)

    elif method == 'mean':

        # Substituindo valores ausentes pela média

        mean = round(df[column].mean(), 2)  # Modificação na linha da média

        df[column].fillna(mean, inplace=True)

    elif method == 'mode':

        # Substituindo valores ausentes pela moda

        mode = df[column].mode()[0]

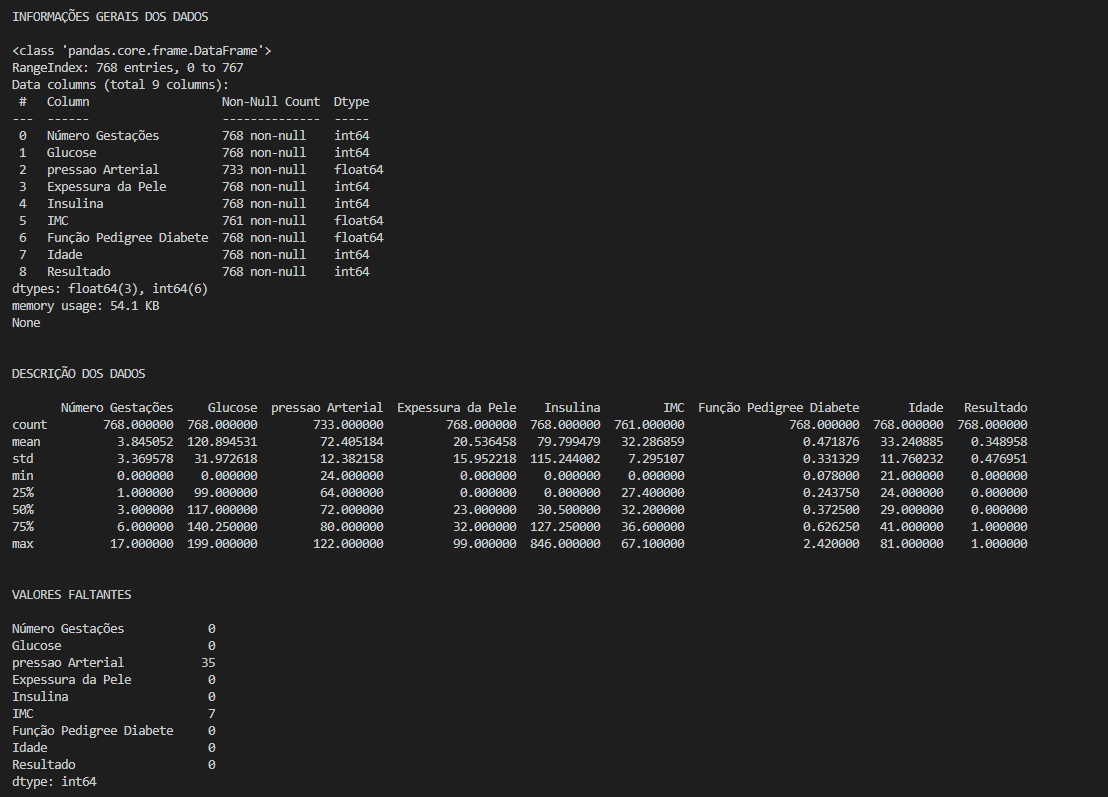
        df[column].fillna(mode, inplace=True)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Ao adaptar esse algoritmo a nossa base de dados obtivemos as seguintes saídas.

Figura 2: Saída do DataCleaning do projeto.



fonte: vscode, 2023

Foi observado duas colunas com dados faltantes e nela foi substituído em uma nova base pela mediana dos valores, foi escolhido a mediada devido os dados da coluna ser assimétrico, ou seja, em um histograma os dados da direita ou esquerda são desiguais conforme verificamos abaixo:

Figura 3: Histograma dos dados de Pressão Arterial

Gráfico, Histograma

Descrição gerada automaticamente

fonte: vscode, 2023

Figura 4: Histograma dos Dados de IMC

Gráfico, Histograma

Descrição gerada automaticamente

fonte: vscode, 2023

# 3 Normalização e Redução de Dados

## 3.1 Normalização dos dados:

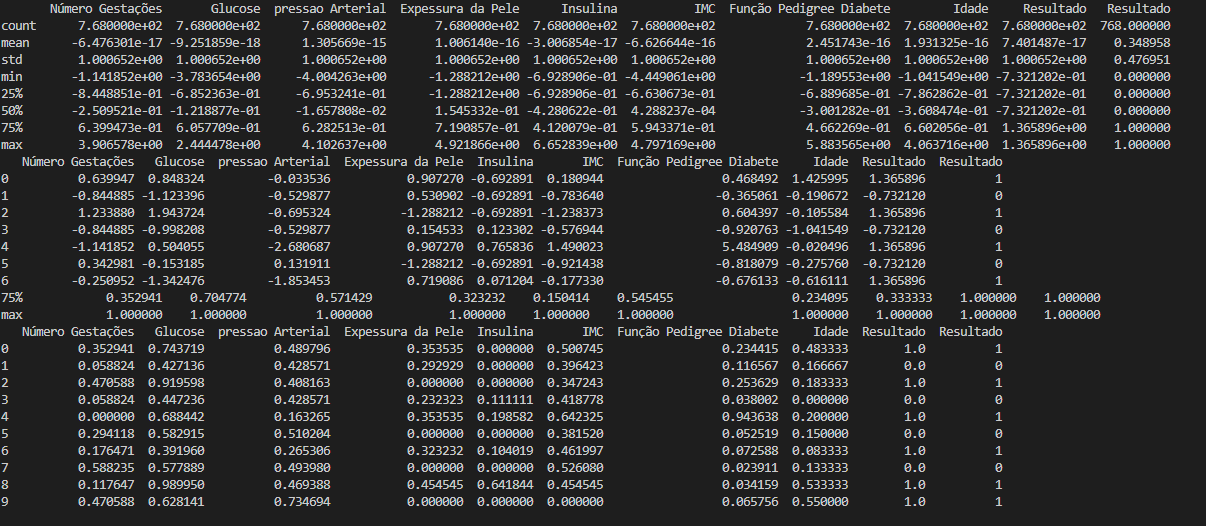
A normalização de dados é o processo de escalar os dados para que todos estejam na mesma escala. Isso é importante porque algumas variáveis podem ter uma escala muito maior do que outras e podem ter um impacto desproporcional na análise. Existem diferentes métodos de normalização, mas um dos mais comuns é a normalização Min-Max, que ajusta os valores para um intervalo entre 0 e 1. Para aplicar a normalização Min-Max à base de dados de prevê diabetes, seguimos esses passos:

* Importar a biblioteca Scikit-learn para a normalização dos dados.
* Dividir a base de dados em duas partes, uma para as variáveis dependentes e outra para a variável de saída.
* Aplicar a normalização Min-Max às variáveis dependentes usando a função MinMaxScaler().
* Concatenar as variáveis dependentes normalizadas com a variável de saída para formar a base de dados normalizada.
* import pandas as pd
* from sklearn.preprocessing import MinMaxScaler
* # Faz a leitura do arquivo
* input\_file = '0-Datasets/diabetesClear.data'
* names = ['Número Gestações','Glucose','pressao Arterial','Expessura da Pele','Insulina','IMC','Função Pedigree Diabete','Idade','Resultado']
* features = ['Número Gestações','Glucose','pressao Arterial','Expessura da Pele','Insulina','IMC','Função Pedigree Diabete','Idade','Resultado']
* target = 'Resultado'
* df = pd.read\_csv(input\_file,    # Nome do arquivo com dados
* names = names) # Nome das colunas
* # Separar as variáveis dependentes da variável de saída
* X = df.drop('Resultado', axis=1)
* y = df['Resultado']
* # Aplicar a normalização Min-Max às variáveis dependentes
* scaler = MinMaxScaler()
* X\_norm = scaler.fit\_transform(X)
* # Criar um novo dataframe com as variáveis dependentes normalizadas e a variável de saída
* df\_norm = pd.DataFrame(X\_norm, columns=X.columns)
* df\_norm['Resultado'] = y
* # Salvar a base de dados normalizada em um novo arquivo csv
* df\_norm.to\_csv('dados\_diabetes\_normalizados.csv', index=False)

O arquivo csv com a base de dados original é lido usando a biblioteca pandas, e as variáveis dependentes (todas as colunas exceto a última) são separadas da variável de saída (a última coluna). Em seguida, a normalização Min-Max é aplicada às variáveis dependentes usando a classe MinMaxScaler da biblioteca Scikit-learn.

O resultado da normalização é armazenado em um novo dataframe chamado df\_norm, que contém as variáveis dependentes normalizadas e a variável de saída original. Finalmente, o dataframe normalizado é salvo em um novo arquivo csv usando a função to\_csv() da biblioteca pandas.

Figura 5: Saida do algoritmo de Normalização



fonte: vscode, 2023

## 3.2 Redução dos dados:

A redução de dados é o processo de reduzir a dimensionalidade dos dados para eliminar variáveis desnecessárias ou redundantes. Isso pode ajudar a reduzir o tempo de processamento e melhorar a precisão do modelo. Uma técnica comum de redução de dados é a Análise de Componentes Principais (PCA), que reduz as variáveis para um número menor de componentes que capturam a maior parte da variação dos dados. Para aplicar a PCA à base de dados fornecida, você pode seguir os seguintes passos:

* Importar a biblioteca Scikit-learn para a redução dos dados.
* Dividir a base de dados em duas partes, uma para as variáveis dependentes e outra para a variável de saída.
* Aplicar a PCA às variáveis dependentes usando a função PCA().
* Selecionar o número de componentes que capturam a maior parte da variação dos dados.
* Transformar as variáveis dependentes reduzidas em uma matriz e concatenar com a variável de saída para formar a base de dados reduzida.
* import pandas as pd
* from sklearn.decomposition import PCA
* # Ler o arquivo csv com a base de dados normalizada
* df\_norm = pd.read\_csv('dados\_diabetes\_normalizados.csv')
* # Separar as variáveis dependentes da variável de saída
* X\_norm = df\_norm.drop('Resultado', axis=1)
* y\_norm = df\_norm['Resultado']
* # Aplicar a PCA às variáveis dependentes normalizadas
* pca = PCA(n\_components=3)
* X\_pca = pca.fit\_transform(X\_norm)
* # Criar um novo dataframe com as variáveis dependentes reduzidas e a variável de saída
* df\_pca = pd.DataFrame(X\_pca, columns=['PCA1', 'PCA2', 'PCA3'])
* df\_pca['Resultado'] = y\_norm
* # Salvar a base de dados reduzida em um novo arquivo csv
* df\_pca.to\_csv('dados\_diabetes\_reduzidos.csv', index=False)

O arquivo csv com a base de dados normalizada é lido usando a biblioteca pandas, e as variáveis dependentes (todas as colunas exceto a última) são separadas da variável de saída (a última coluna).

Em seguida, a PCA é aplicada às variáveis dependentes normalizadas usando a classe PCA da biblioteca Scikit-learn, especificando o número de componentes que se deseja manter. Neste exemplo, estamos mantendo os três primeiros componentes principais.

O resultado da redução é armazenado em um novo dataframe chamado df\_pca, que contém as variáveis dependentes reduzidas e a variável de saída original. Finalmente, o dataframe reduzido é salvo em um novo arquivo csv usando a função to\_csv() da biblioteca pandas.

Logo abaixo temos o resultado da saída do PCA e percebemos algo positivo que os dados estão bastante dispersos uns dos outros.

Gráfico, Gráfico de dispersão

Descrição gerada automaticamente

Em resumo, a normalização e redução de dados são técnicas importantes para pré-processar a base de dados utilizada para diabetes antes de aplicar algoritmos para prever diabetes. A normalização ajuda a garantir que todas as variáveis estejam na mesma escala, enquanto a redução ajuda a reduzir a dimensionalidade dos dados para melhorar a precisão do modelo.

# 4 Análise descritiva de dados – Visualização

A análise descritiva de dados é uma técnica muito utilizada em mineração de dados, que tem como objetivo analisar e entender os dados de um conjunto de dados para obter informações relevantes. Essa técnica permite entender melhor as características dos dados, identificar padrões, tendências e anomalias, além de verificar a qualidade dos dados.

A visualização de dados é uma ferramenta essencial para a análise descritiva de dados, pois permite a representação gráfica dos dados de uma forma que facilita a compreensão e a interpretação das informações contidas no conjunto de dados. Com a visualização, é possível identificar padrões e tendências que não seriam detectados com apenas a análise dos dados em formato tabular.

Entre as principais técnicas de visualização utilizadas na análise descritiva de dados estão o histograma, o gráfico de setores, o gráfico de barras, o gráfico de dispersão e a matriz de correlação. Cada uma dessas técnicas é utilizada para representar diferentes tipos de dados e para identificar diferentes tipos de padrões e tendências.

Em resumo, a análise descritiva de dados e a visualização de dados são técnicas fundamentais em mineração de dados para compreender e extrair informações úteis de grandes conjuntos de dados.

## 4.1 Gráficos em Histograma, Dispersão e Setores

Os gráficos de histograma, dispersão e setores são comumente utilizados na análise descritiva de dados por suas capacidades de visualização eficaz e fácil interpretação dos dados.

O gráfico de histograma é útil para visualizar a distribuição dos dados e pode ajudar a identificar padrões, como picos e caudas longas, bem como a presença de valores atípicos. É particularmente útil quando se trabalha com dados contínuos ou discretos.

O gráfico de dispersão é utilizado para analisar a relação entre duas variáveis. É capaz de identificar padrões, correlações e relacionamentos entre variáveis e pode ser usado para identificar possíveis causas e efeitos.

Já o gráfico de setores é comumente usado para mostrar a proporção de cada categoria em um conjunto de dados. É útil quando se trabalha com dados categóricos e pode ajudar a identificar padrões ou desequilíbrios entre as categorias.

Esses gráficos são considerados eficazes porque são fáceis de entender e interpretar. Eles permitem que os usuários vejam rapidamente as tendências, padrões e correlações nos dados, sem precisar olhar para tabelas de números ou dados brutos. Além disso, esses gráficos podem ser usados para identificar problemas nos dados, como valores discrepantes ou ausentes, o que pode levar a análises mais precisas e confiáveis.

Em resumo, esses gráficos são valiosos para a análise descritiva de dados porque oferecem uma maneira visualmente atraente e intuitiva de explorar os dados e comunicar os resultados de maneira clara e concisa.

Abaixo segue o algoritmo em Python utilizado para a plotagem desses gráficos conforme a base de dados de Prever Diabetes.

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

df = pd.read\_csv('0-Datasets/diabetesClear.data')

for coluna in df.columns:

    # Histograma

    plt.figure()

    sns.histplot(data=df, x=coluna)

    plt.title(f'Histograma da coluna {coluna}')

    plt.savefig(f'3-Analise Descritiva - Visualização/Graficos/Histograma/historgrama\_{coluna}.png')

    plt.close()

    # Gráfico de setores

    plt.figure()

    df[coluna].value\_counts().plot(kind='pie')

    plt.title(f'Gráfico de setores da coluna {coluna}')

    plt.savefig(f'3-Analise Descritiva - Visualização/Graficos/Grafico de Setores/grafico\_setores\_{coluna}.png')

    plt.close()

    # Dispersão

    plt.figure()

    sns.scatterplot(data=df, x=coluna, y='Outcome')

    plt.title(f'Dispersão da coluna {coluna} em relação à coluna alvo')

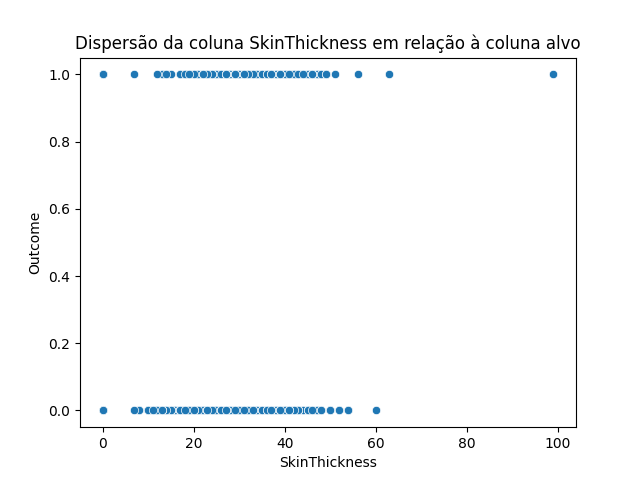
    plt.savefig(f'3-Analise Descritiva - Visualização/Graficos/Dispersao/dispersao\_{coluna}.png')

    plt.close()

Nesse código, vemos todas correlação além da coluna de resultado com as demais temos uma relação de todas as colunas entre sim e podemos ver mais detalhadamente a relação de cada coluna com a outra, ao total foi gerado 71 gráficos pois temos pois temos 8 colunas em 8 sendo relaciona e por último cada coluna relacionada com nossa coluna alvo e com esses gráficos conseguimos ter um entendimento melhor

Abaixo temos alguns exemplos de saída desses gráficos em histograma.

Figura 6: Gráficos de Dispersão entre o Resultado e a Espessura da Pele



fonte: elaborado pelo autor (2023)

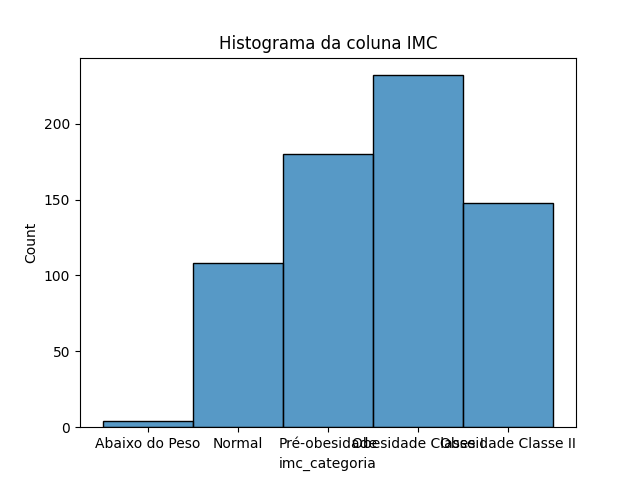
Figura 7: Gráficos de setores entre o Resultado e o IMC categorizado

Gráfico, Gráfico de pizza

Descrição gerada automaticamente

fonte: elaborado pelo autor (2023)

Figura 8: Gráficos de Histograma entre o resultado e o IMC Categorizado



fonte: elaborado pelo autor (2023)

# 5 Análise descritiva de dados – Medidas

Análise descritiva de dados é uma técnica de mineração de dados que busca descrever as características de um conjunto de dados, sem inferir relações causais ou preditivas. Ela é utilizada para explorar os dados, obter insights e compreender melhor as informações disponíveis.

As medidas estatísticas são ferramentas fundamentais na análise descritiva de dados. As medidas de tendência central, dispersão, posição relativa e associação são exemplos de medidas estatísticas utilizadas na análise descritiva de dados.

Medidas de tendência central: São medidas que indicam o ponto central de um conjunto de dados. As três medidas de tendência central mais comuns são a média, a mediana e a moda.

Medidas de dispersão: São medidas que indicam o grau de variação dos dados em relação a uma medida central. As medidas de dispersão mais comuns são o desvio padrão, a variância e o coeficiente de variação.

Medidas de posição relativa: São medidas que indicam a posição de um valor em relação aos demais valores de um conjunto de dados. As medidas de posição relativa mais comuns são o percentil e o quartil.

Medidas de associação: São medidas que indicam a relação entre duas ou mais variáveis. As medidas de associação mais comuns são a correlação e a covariância.

Em resumo, a análise descritiva de dados utiliza medidas estatísticas para resumir e descrever os dados, de forma a obter insights e compreender melhor as informações disponíveis. As medidas de tendência central, dispersão, posição relativa e associação são algumas das medidas estatísticas mais comuns utilizadas na análise descritiva de dados.

## 5.1 Medidas de tendência central

Medidas de tendência central são utilizadas para resumir a distribuição dos dados de uma variável quantitativa, indicando onde a maioria dos valores estão concentrados. As três medidas de tendência central mais comuns são a média, a mediana e a moda.

A média é a soma de todos os valores dividida pelo número total de valores e é frequentemente utilizada como uma medida padrão de tendência central. A mediana é o valor que divide a distribuição ao meio, de modo que metade dos valores estão acima dela e metade estão abaixo. Já a moda é o valor mais frequente na distribuição, ou seja, o valor que aparece com maior frequência.

Cada uma dessas medidas de tendência central pode ser útil em diferentes situações. A média é sensível a valores extremos e pode não ser representativa de uma distribuição assimétrica ou com outliers. A mediana é uma medida mais robusta que a média, pois não é afetada por valores extremos e é frequentemente utilizada quando a distribuição não é normal. Já a moda é útil para identificar o valor mais comum em uma distribuição e é frequentemente usada em distribuições simétricas ou com uma única moda clara.

Em resumo, as medidas de tendência central são uma maneira de resumir e descrever a distribuição dos dados de uma variável quantitativa, ajudando a entender onde a maioria dos valores estão concentrados e como se comportam em relação aos outros valores.

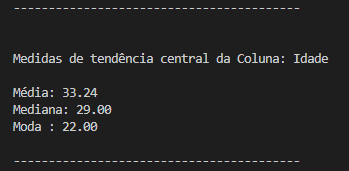
    # Medidas de tendência central

    media = df[target].mean()

    mediana = df[target].median()

    moda = df[target].mode()[0]

Figura 9 Saida do Algoritmo de Medida de Tendencia Central



Oque podemos observar nessa saída e que média é de 33.24 anos, mas a mediana é de apenas 29 anos, o que sugere que há alguns pacientes com idades mais altas que estão aumentando a média. A moda de 22 anos indica que este é o valor mais comum para a idade dos pacientes na base de dados.

## 5.2 Medidas de dispersão

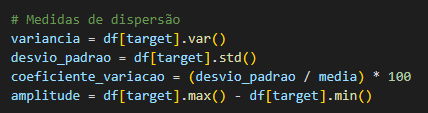
As medidas de dispersão, em mineração de dados, são usadas para descrever a variabilidade ou a dispersão dos dados de uma variável quantitativa em torno de uma medida de tendência central, como a média ou a mediana.

As medidas de dispersão mais comuns incluem a amplitude, a variância e o desvio padrão. A amplitude é a diferença entre o maior e o menor valor na distribuição. Já a variância é uma medida de quanto os valores variam em relação à média, calculada como a média dos quadrados das diferenças entre cada valor e a média. O desvio padrão é a raiz quadrada da variância e indica quanto os valores tendem a se desviar da média.

As medidas de dispersão são importantes porque fornecem informações sobre a variabilidade dos dados em torno da medida de tendência central. Uma distribuição com baixa dispersão terá a maioria dos valores próximos à medida de tendência central, enquanto uma distribuição com alta dispersão terá valores mais espalhados ao redor da medida de tendência central.

Assim como as medidas de tendência central, as medidas de dispersão podem ser úteis em diferentes situações. A amplitude é uma medida simples e direta de dispersão, mas pode ser influenciada por valores extremos. A variância e o desvio padrão são mais robustos, mas podem ser difíceis de interpretar, especialmente em unidades não familiares.

Em resumo, as medidas de dispersão são usadas para descrever a variabilidade dos dados em torno de uma medida de tendência central, e são importantes para entender a forma e a dispersão da distribuição dos dados.



Texto

Descrição gerada automaticamente

Os resultados das medidas de dispersão indicam que há uma variação considerável na idade dos pacientes na base de dados de Prever Diabetes. A variância de 138.30 indica que os valores de idade estão espalhados em torno da média de 33.24 anos, enquanto o desvio padrão de 11.76 indica que há uma variação relativamente alta em relação à média. Isso sugere que a idade dos pacientes está distribuída de forma mais ampla do que seria esperado se fosse distribuída normalmente.

O coeficiente de variação de 35.38% indica que a variação relativa na idade dos pacientes na base de dados é relativamente alta em relação à média, o que sugere que a idade dos pacientes varia bastante. A amplitude de 60.00 indica que a diferença entre a idade mais jovem e a mais velha na base de dados é de 60 anos, o que indica uma grande variabilidade na idade dos pacientes.

Esses resultados indicam que a base de dados de Prever Diabetes pode ter uma distribuição não normal em relação à idade dos pacientes, com uma variação considerável em torno da média. Isso sugere que a idade dos pacientes pode ser um fator importante a ser considerado ao prever o diabetes nessa base de dados, uma vez que há uma grande variação na idade dos pacientes.

Os resultados das medidas de tendência central e de dispersão da idade dos pacientes na base de dados de Prever Diabetes são importantes para entender a distribuição da idade e a variação dos valores na amostra. Esses resultados podem afetar a análise da base de dados, dependendo do objetivo da análise.

Por exemplo, se o objetivo da análise for avaliar a associação entre a idade dos pacientes e a ocorrência de diabetes, a grande variação na idade pode indicar a necessidade de uma análise mais detalhada, como o uso de técnicas de estratificação por idade, para avaliar como diferentes faixas etárias podem estar associadas a diferentes riscos de diabetes.

Além disso, a distribuição não normal dos valores de idade pode afetar a seleção e a validação de modelos de previsão de diabetes, pois os modelos de regressão e classificação geralmente assumem uma distribuição normal das variáveis de entrada. É importante considerar esses fatores ao interpretar os resultados da base de dados e ao selecionar as técnicas de análise mais adequadas para alcançar os objetivos da análise.

## 5.3 Medidas de posição relativa

As medidas de posição relativa, em mineração de dados, são usadas para descrever a posição de um valor dentro de uma distribuição em relação a outras medidas, como a média ou a mediana. Essas medidas fornecem informações sobre a posição relativa de um valor em relação à distribuição como um todo.

As medidas de posição relativa mais comuns são o percentil e o quartil. O percentil é o valor abaixo do qual uma porcentagem especificada de observações em uma distribuição está contida. Por exemplo, o percentil 50 é a mediana, ou o valor abaixo do qual 50% dos valores estão contidos. Já os quartis dividem a distribuição em quartos iguais, com o primeiro quartil (Q1) sendo o valor abaixo do qual 25% dos valores estão contidos, o segundo quartil (Q2) sendo a mediana e o terceiro quartil (Q3) sendo o valor abaixo do qual 75% dos valores estão contidos.

As medidas de posição relativa são úteis para comparar valores individuais com a distribuição como um todo e avaliar a posição relativa de um valor em relação aos outros valores. Por exemplo, um valor acima do terceiro quartil indica que está acima da maioria dos valores na distribuição, enquanto um valor abaixo do primeiro quartil indica que está abaixo da maioria dos valores.

Em resumo, as medidas de posição relativa são usadas para descrever a posição de um valor em relação à distribuição como um todo, e são importantes para entender a posição relativa de um valor em relação aos outros valores. As medidas de posição relativa podem ser usadas em conjunto com as medidas de tendência central e dispersão para fornecer uma descrição completa da distribuição dos dados.

    print(f"\nMedidas de posição relativa da Coluna: {target}\n")

    print('Z Score:\n{}\n'.format((df[target] - df[target].mean())/df[target].std())) # Z Score

    print("Primeiro quartil: {:.2f}".format(Q1))

    print("Segundo quartil (Mediana): {:.2f}".format(Q2))

    print("Terceiro quartil: {:.2f}".format(Q3))

    print("\n-----------------------------------------\n")

Texto

Descrição gerada automaticamente

Os resultados das medidas de posição relativa, como o Z-score e os quartis, fornecem informações importantes sobre como a idade dos pacientes na base de dados está distribuída em relação à média e aos limites inferiores e superiores.

O Z-score indica quantos desvios padrão cada valor de idade está acima ou abaixo da média da amostra. Por exemplo, um Z-score de 1.4 indica que um paciente tem uma idade 1.4 desvios padrão acima da média da amostra. Isso pode ser útil para identificar valores extremos na amostra, que podem afetar a análise da base de dados.

Os quartis fornecem informações sobre a distribuição da idade dos pacientes na base de dados. O primeiro quartil (Q1) de 24 anos indica que 25% dos pacientes têm uma idade menor ou igual a 24 anos. O segundo quartil (Q2), que é a mediana, de 29 anos indica que 50% dos pacientes têm uma idade menor ou igual a 29 anos. O terceiro quartil (Q3) de 41 anos indica que 75% dos pacientes têm uma idade menor ou igual a 41 anos. Isso sugere que a maioria dos pacientes na base de dados tem idades abaixo de 41 anos, mas há alguns pacientes mais velhos que estão aumentando a média.

Esses resultados indicam que a idade dos pacientes na base de dados de Prever Diabetes está distribuída de forma bastante ampla, com valores extremos que podem afetar a análise. Além disso, a maioria dos pacientes tem idades abaixo de 41 anos, o que pode ser útil para identificar grupos de pacientes com idades semelhantes que possam ter riscos de diabetes semelhantes.

## 5.4 Medidas de Associação

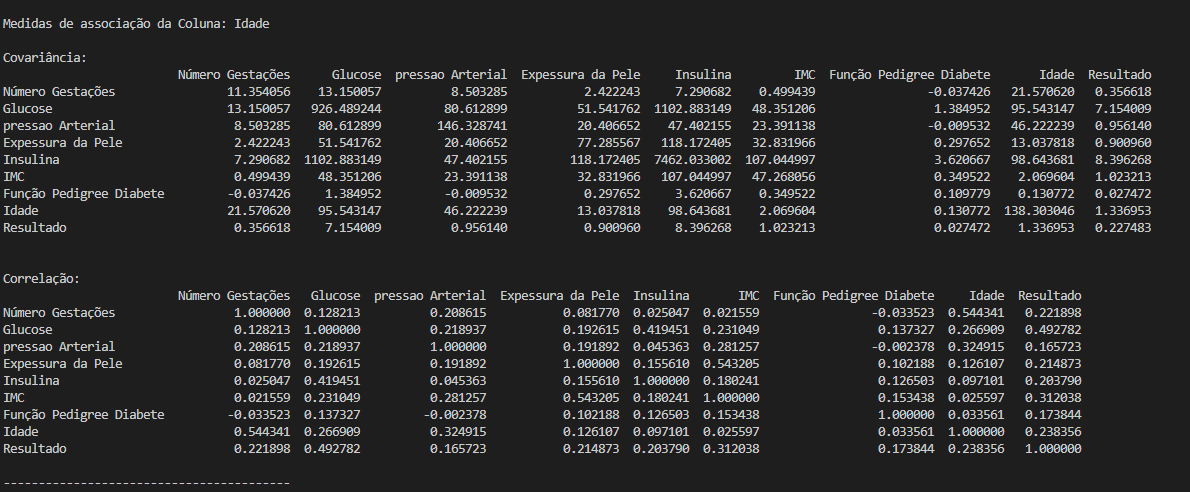
As medidas de associação, em mineração de dados, são usadas para avaliar a relação entre duas variáveis. Elas fornecem informações sobre a direção e a força da associação entre as variáveis, e são usadas para identificar padrões e relações nos dados.

As medidas de associação mais comuns incluem a correlação e a covariância. A correlação mede a força da relação linear entre duas variáveis, variando de -1 (correlação negativa perfeita) a +1 (correlação positiva perfeita), com 0 indicando ausência de correlação. A covariância mede a relação entre duas variáveis, mas é afetada pela escala das variáveis e pode ser difícil de interpretar.

As medidas de associação são importantes porque ajudam a entender como as variáveis se relacionam entre si. Por exemplo, a correlação pode ser usada para determinar se existe uma relação positiva ou negativa entre duas variáveis, e se essa relação é forte ou fraca.

No entanto, é importante ter cuidado ao interpretar as medidas de associação, pois elas podem indicar uma relação aparente entre duas variáveis, mas não necessariamente uma relação causal. Além disso, outras variáveis ou fatores podem influenciar a associação observada entre as variáveis, o que pode levar a interpretações equivocadas.

Em resumo, as medidas de associação são usadas para avaliar a relação entre duas variáveis e fornecer informações sobre a direção e a força da associação. Elas são úteis para identificar padrões e relações nos dados, mas devem ser interpretadas com cuidado e considerando outros fatores que possam influenciar a associação observada.



Com base nas medidas de associação da coluna "Idade", podemos notar que a covariância entre a idade e o número de gestações é de 21.57, indicando uma relação positiva, ou seja, quanto maior a idade, maior o número de gestações.

Além disso, a correlação entre idade e resultado do teste de diabetes é de 0,238, indicando uma relação positiva fraca, ou seja, há uma leve tendência de que pessoas mais velhas tenham maior probabilidade de apresentar resultado positivo no teste de diabetes.

No entanto, é importante ressaltar que a idade é apenas uma das variáveis que podem estar relacionadas ao resultado do teste de diabetes, e é necessário analisar outras variáveis para uma avaliação mais completa.

É possível observar que a idade tem correlação positiva com o resultado do teste de diabetes, indicando que pessoas mais velhas tendem a ter maior probabilidade de ter diabetes. Além disso, a idade tem correlação moderada com o número de gestações e forte correlação com a idade gestacional. No entanto, para uma análise mais detalhada, é necessário considerar os demais resultados das medidas de associação e aprofundar no contexto e na interpretação clínica dos dados.

# 5 Análise de Grupo

As análises de grupos, também conhecidas como análises de clusters, são técnicas de mineração de dados que têm como objetivo encontrar grupos de objetos similares em um conjunto de dados.

A fundamentação teórica das análises de grupos está relacionada com técnicas de aprendizado não supervisionado, que buscam identificar estruturas subjacentes em dados sem a necessidade de um conjunto de dados rotulados.

Existem diferentes métodos de análises de grupos, como a análise hierárquica, a análise k-means e a análise de mistura de Gaussianas. Em geral, esses métodos utilizam uma medida de distância para calcular a similaridade entre objetos e, a partir disso, agrupá-los em clusters.

A escolha da medida de distância depende do tipo de dados sendo analisados e da natureza do problema em questão. Além disso, é importante definir critérios para avaliar a qualidade dos clusters obtidos, como a coerência interna dos grupos e a separação entre eles.

As análises de grupos são amplamente utilizadas em diversas áreas, como biologia, marketing, finanças e ciência da computação, para identificar padrões e estruturas em dados não rotulados, possibilitando a geração de insights e tomada de decisões.

Existem diversas ferramentas e técnicas que podem ser utilizadas para realizar análises de grupos. Algumas das principais são:

Análise hierárquica: essa técnica permite criar uma hierarquia de clusters, na qual grupos menores são agrupados em grupos maiores, e assim por diante. Existem dois tipos principais de análise hierárquica: aglomerativa e divisiva.

Análise k-means: essa técnica agrupa os dados em k clusters, em que k é definido pelo usuário. O algoritmo calcula centróides para cada grupo e ajusta os clusters de forma iterativa, até que a convergência seja alcançada.

Análise de mistura de Gaussianas: essa técnica é utilizada para modelar dados em que a distribuição dos grupos é gaussiana. O algoritmo estima os parâmetros da distribuição de cada grupo, como a média e a variância, e utiliza esses parâmetros para atribuir objetos a clusters.

Análise de densidade: essa técnica utiliza uma medida de densidade para encontrar regiões de alta densidade em um conjunto de dados. Os objetos dentro dessas regiões são agrupados em clusters.

Redução de dimensionalidade: essa técnica é utilizada para reduzir a dimensionalidade do conjunto de dados, de forma a facilitar a análise. Algumas técnicas comuns de redução de dimensionalidade incluem a Análise de Componentes Principais (PCA) e a Análise de Fatores.

Além dessas técnicas, existem diversas ferramentas e softwares que podem ser utilizados para realizar análises de grupos, como o Python com as bibliotecas Scikit-Learn e Pandas, o R com os pacotes cluster e factoextra, e o Weka. A escolha da ferramenta depende das necessidades do projeto e da familiaridade do usuário com cada ferramenta.

## 5.1 K-Means

O método K-means é uma técnica de análise de grupos amplamente utilizada em mineração de dados e aprendizado de máquina. Ele é utilizado para agrupar um conjunto de dados em K clusters, em que K é um número pré-definido de grupos.

O algoritmo K-means funciona de forma iterativa e envolve os seguintes passos:

Inicialização: o número K de clusters é definido pelo usuário e o algoritmo seleciona aleatoriamente K objetos do conjunto de dados como centróides iniciais.

Atribuição de objetos a clusters: cada objeto do conjunto de dados é atribuído ao cluster cujo centróide está mais próximo. A distância entre os objetos e os centróides é calculada utilizando uma medida de distância, geralmente a distância euclidiana.

Atualização dos centróides: os centróides de cada cluster são atualizados para refletir a posição dos objetos que foram atribuídos a ele.

Repetição: os passos 2 e 3 são repetidos até que os centróides deixem de se mover ou até que o número máximo de iterações seja alcançado.

Convergência: quando a iteração termina, o algoritmo tem como resultado um conjunto de K clusters, onde cada objeto pertence a um dos clusters.

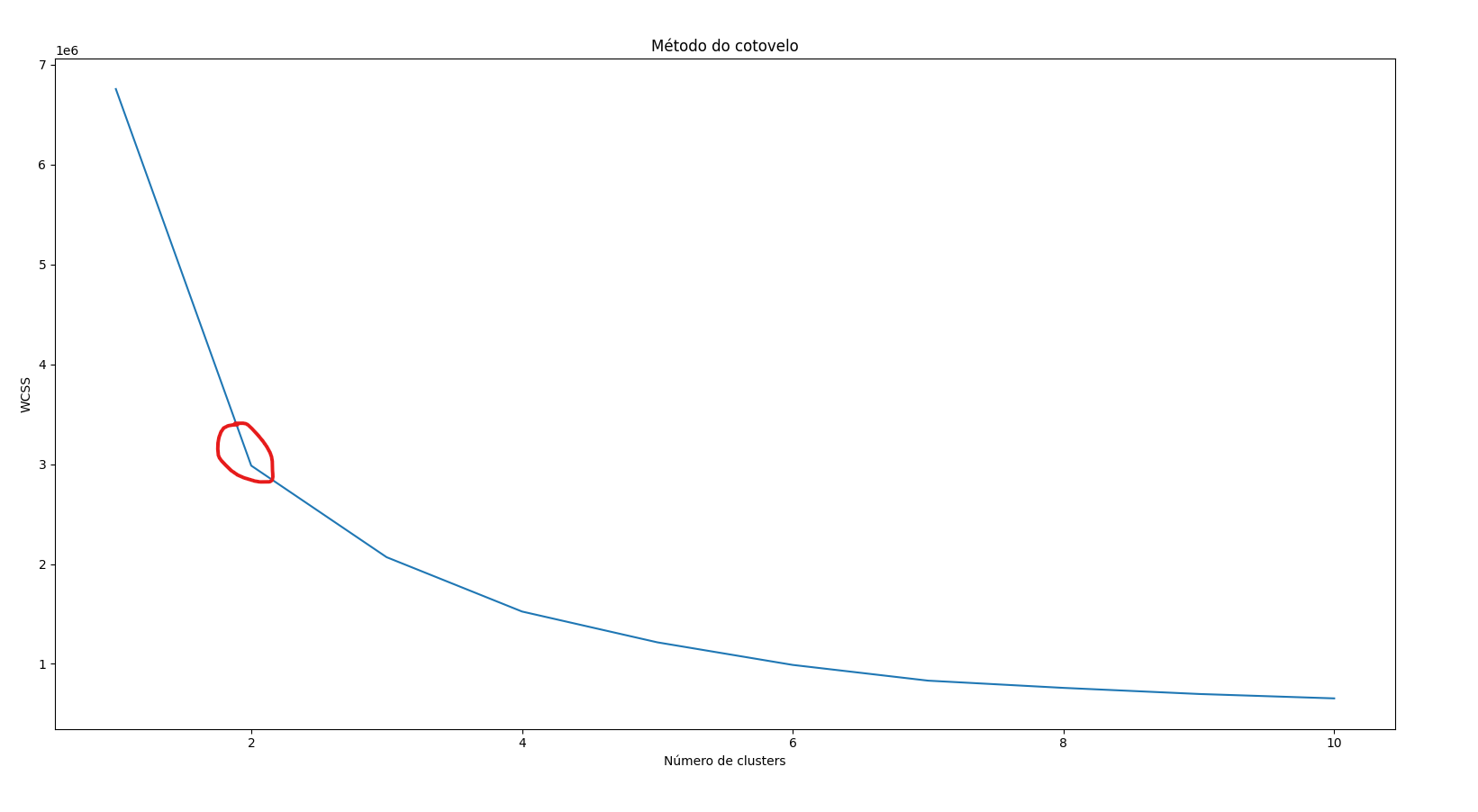
Uma das principais vantagens do método K-means é a sua eficiência computacional, sendo capaz de lidar com grandes conjuntos de dados. No entanto, a qualidade dos clusters obtidos pode ser afetada pela escolha dos centróides iniciais, que pode levar a soluções subótimas. Para lidar com esse problema, é comum executar o algoritmo várias vezes com diferentes valores iniciais de centróides e escolher a solução com o menor valor de função objetivo.

O método K-means é amplamente utilizado em diversas áreas, como marketing, biologia, finanças e ciência da computação, para segmentar clientes, identificar grupos de proteínas com funções semelhantes, agrupar empresas com perfis financeiros semelhantes, entre outras aplicações.

Para podemos iniciar usando o kmeans precisamos saber antes quantas centroids ou quantos Clusters vamos utilizar na plotagem, existem alguns algoritmos que já fazem esse cálculo, porem foi desejado realizamos o método do cotovelo que por definição seria: (elbow method) é uma técnica utilizada para determinar o número ideal de clusters a serem usados em um algoritmo de clustering, como o K-means. Ele consiste em plotar a soma dos quadrados intra-cluster (WCSS) em função do número de clusters e identificar o ponto em que a curva apresenta uma inflexão, que se assemelha a um cotovelo, indicando que a adição de mais clusters não melhora significativamente a qualidade da segmentação.

O WCSS é uma medida da variação dos pontos dentro de cada cluster, e é calculado somando-se as distâncias euclidianas entre cada ponto e o centroide do seu cluster. Quanto menor o WCSS, mais compactos são os clusters e maior é a sua separação.

Veja abaixo o exemplo do cotovelo para a nossa base de dados.



fonte: elaborado pelo autor (2023)

Com esse resultamos concluímos que o número de Clusters ideal para essa base seria 2. Com esse entendimento podemos gerar os gráficos do Kmeans pelo Código abaixo.

*#Implementation of Kmeans from scratch and using sklearn*

*#Loading the required modules*

import numpy as np

from scipy.spatial.distance import cdist

from sklearn.datasets import load\_digits

from sklearn.decomposition import PCA

from sklearn.cluster import KMeans

from sklearn.metrics import silhouette\_score

from sklearn.metrics import silhouette\_samples

import matplotlib.pyplot as plt

import pandas as pd

*#Defining our kmeans function from scratch*

def KMeans\_scratch(*x*,*k*, *no\_of\_iterations*):

    idx = np.random.choice(len(*x*), *k*, *replace*=False)

*#Randomly choosing Centroids*

    centroids = *x*[idx, :] *#Step 1*

*#finding the distance between centroids and all the data points*

    distances = cdist(*x*, centroids ,'euclidean') *#Step 2*

*#Centroid with the minimum Distance*

    points = np.array([np.argmin(i) for i in distances]) *#Step 3*

*#Repeating the above steps for a defined number of iterations*

*#Step 4*

    for \_ in range(*no\_of\_iterations*):

        centroids = []

        for idx in range(*k*):

*#Updating Centroids by taking mean of Cluster it belongs to*

            temp\_cent = *x*[points==idx].mean(*axis*=0)

            centroids.append(temp\_cent)

        centroids = np.vstack(centroids) *#Updated Centroids*

        distances = cdist(*x*, centroids ,'euclidean')

        points = np.array([np.argmin(i) for i in distances])

    return points

def show\_digitsdataset(*digits*):

    fig = plt.figure(*figsize*=(6, 6))  *# figure size in inches*

    fig.subplots\_adjust(*left*=0, *right*=1, *bottom*=0, *top*=1, *hspace*=0.05, *wspace*=0.05)

    for i in range(64):

        ax = fig.add\_subplot(8, 8, i + 1, *xticks*=[], *yticks*=[])

*# ax.imshow(digits.axes[i], cmap=plt.cm.binary, interpolation='nearest')*

*# label the image with the target value*

        ax.text(0, 7, str(*digits*.resultado[i]))

*#fig.show()*

def plot\_samples(*projected*, *labels*, *title*):

    fig = plt.figure()

    u\_labels = np.unique(*labels*)

    for i in u\_labels:

        plt.scatter(*projected*[*labels* == i , 0] , *projected*[*labels* == i , 1] , *label* = i,

*edgecolor*='none', *alpha*=0.5, *cmap*=plt.cm.get\_cmap('tab10', 10))

    plt.xlabel('component 1')

    plt.ylabel('component 2')

    plt.legend()

    plt.title(*title*)

def main():

    input\_file = '0-Datasets/diabetesClear.data'

    names = ['Número Gestações','Glucose','pressao Arterial','Expessura da Pele','Insulina','IMC','Função Pedigree Diabete','Idade','Resultado']

    features = ['Número Gestações','Glucose','pressao Arterial','Expessura da Pele','Insulina','IMC','Função Pedigree Diabete','Idade','Resultado']

    target = 'Resultado'

    digits = pd.read\_csv(input\_file,         *# Nome do arquivo com dados #df =  data framing*

*names* = names,      *# Nome das colunas*

*usecols* = features, *# Define as colunas que serão  utilizadas*

*na\_values*='?')      *# Define que ? será considerado valores ausentes*

    show\_digitsdataset(digits)

*#Transform the data using PCA*

    pca = PCA(2)

    projected = pca.fit\_transform(digits)

    print(pca.explained\_variance\_ratio\_)

*# print(digits.data.shape)*

    print(projected.shape)

    plot\_samples(projected, digits.resultado, 'Original Labels')

*#Applying our kmeans function from scratch*

    labels = KMeans\_scratch(projected,6,5)

*#Visualize the results*

    plot\_samples(projected, digits.resultado, 'Clusters Sexo KMeans from scratch')

*#Applying sklearn kemans function*

    kmeans = KMeans(*n\_clusters*=6).fit(projected)

    print("teste")

    print(kmeans.inertia\_)

    print(projected)

    centers = kmeans.cluster\_centers\_

    score = silhouette\_score(projected, kmeans.labels\_)

    print("For n\_clusters = {}, silhouette score is {})".format(10, score))

*#Visualize the results sklearn*

    plot\_samples(projected, kmeans.labels\_, 'TESTE Clusters Labels KMeans from sklearn')

    plt.show()

if \_name\_ == "\_main\_":

    main()

Com esse código foi gerado esse gráfico do Kmeans

# Referências Bibliográficas

CHAUHAN, Aman. Predict Diabetes. Kaggle.com. Disponível em: <https://www.kaggle.com/datasets/whenamancodes/predict-diabities>. Acesso em: 22 fev. 2023.

LABVW. O que fazer em relação à insulina alta - Laboratório Verner Willrich. Laboratório Verner Willrich. Disponível em: <https://labvw.com.br/blog/o-que-fazer-em-relacao-a-insulina-alta/>. Acesso em: 29 mar. 2023.

‌