

**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS**  
**NÚCLEO DE EDUCAÇÃO A DISTÂNCIA**  
**Pós-graduação *Lato Sensu* em Inteligência Artificial e Aprendizado de Máquina**

**Luciano Barbosa Da Silva**

**PREDIÇÃO DE ACIDENTE VASCULAR CEREBRAL COM BASE EM FATORES  
DE RISCO.**

Guarulhos  
Abril de 2023

**Luciano Barbosa Da Silva**

**PREDIÇÃO DE ACIDENTE VASCULAR CEREBRAL COM BASE EM FATORES  
DE RISCO.**

Trabalho de Conclusão de Curso apresentado  
ao Curso de Especialização em Inteligência  
Artificial e Aprendizado de Máquina, como  
requisito parcial à obtenção do título de  
*Especialista*.

Guarulhos  
Abril de 2023

# Sumário

<b>1. INTRODUÇÃO .....</b>	<b>5</b>
1.1. Contextualização.....	5
<b>2. DESCRIÇÃO DO PROBLEMA E DA SOLUÇÃO PROPOSTA .....</b>	<b>6</b>
<b>3. COLETA DE DADOS .....</b>	<b>7</b>
3.1. Dataset.....	7
3.1.1. Listagem descritiva das colunas:.....	7
<b>4. PROCESSAMENTO/TRATAMENTO DE DADOS.....</b>	<b>8</b>
4.1. Ferramentas.....	8
4.2. Bibliotecas utilizadas.....	10
4.2.1. Leitura do Dataset: Stroke Prediction Dataset.csv.....	12
4.2.2. Informações do DataFrame.....	13
4.3. Tratamento dos dados.....	14
4.3.1. Renomeando colunas.....	14
4.3.2. Removendo colunas.....	15
4.3.3. Padronizando Tipos das Colunas.....	15
4.3.4. Tratamento de Valores Ausentes.....	16
4.3.5. Informações do Dataframe Final.....	19
<b>5. ANÁLISE E EXPLORAÇÃO DOS DADOS.....</b>	<b>20</b>
5.1. Estatística descritiva dos dados.....	20
5.1.1. Análise de pacientes que tiveram AVC no conjunto de dados.....	22
5.1.2. Análise de AVC por Gênero.....	24
5.1.3. Análise de distribuição dos atributos.....	24
5.1.3.1. Distribuição de AVC por Nível Médio de Glicose.....	26
5.1.3.2. Distribuição de AVC por IMC.....	27
5.1.3.3. Distribuição de AVC por Idade.....	29
5.2. Histórico de Doença Cardíaca e Hipertensão no conjunto de dados.....	30
5.3. Análise visual dos atributos Categóricos.....	31
5.3.1. Matriz de Correlação.....	32
5.3.1.1. Gráfico da Matriz de Correlação.....	33
5.3.1.2. Classificando a Matriz de Correlação.....	34
5.3.1.3. Correlação Negativa.....	37
5.3.1.4. Seleção de Correlação Forte.....	38
5.3.2. Assimetria dos atributos.....	39
5.3.2.1. Representação Gráfica da Assimetria dos Atributos.....	41
<b>6. PREPARAÇÃO DOS DADOS PARA OS MODELOS DE APRENDIZADO DE MÁQUINA.....</b>	<b>43</b>
<b>7. APLICAÇÃO DE MODELOS DE APRENDIZADO DE MÁQUINA .....</b>	<b>43</b>

8. AVALIAÇÃO DOS MODELOS DE APRENDIZADO DE MÁQUINA E DISCUSSÃO DOS RESULTADOS.....	43
9. CONCLUSÃO.....	43
10. LINKS.....	44
11. REFERÊNCIAS.....	44

## **1. Introdução**

### **1.1. Contextualização**

O acidente vascular cerebral (AVC) é uma das principais causas de morte e incapacidade no mundo, afetando cerca de 15 milhões de pessoas a cada ano. De acordo com a Organização Mundial da Saúde (WORLD HEALTH ORGANIZATION, 2020), a medicação adequada e modificação do estilo de vida podem prevenir pelo menos 50% dos eventos de AVC.

Acidente vascular cerebral (AVC), ou popularmente conhecido entre as pessoas como derrame, está entre as principais causas de morte no mundo, e quando não leva a óbito o AVC pode deixar terríveis sequelas cognitivas e até mesmo físicas.

Atualmente existem diversos fatores de risco que podem potencializar consideravelmente a probabilidade de uma pessoa sofrer um AVC, sendo que alguns deles podem ser administrados para ajudar a prevenir ou reduzir o risco de AVC. Entre esses fatores de risco, podemos citar o estilo de vida, histórico familiar, fatores genéticos, pressão arterial elevada, níveis elevados de colesterol, tabagismo, diabetes, obesidade, sedentarismo e outras condições médicas relevantes.

Neste contexto, o objetivo deste trabalho é desenvolver um modelo de aprendizado de máquina supervisionado para prever o risco de acidente vascular cerebral com base em fatores de risco relevantes. Serão abordadas técnicas de modelagem e preparação de dados, bem como análises exploratórias para identificar atributos que potencializem positivamente um modelo capaz de prever o risco de uma pessoa sofrer AVC.

## 2. Descrição do Problema e da Solução Proposta

O objetivo deste trabalho é fazer uma junção de análises exploratórias e modelagem preditiva para obter informações que sejam relevantes para prever o risco de acidente vascular cerebral (AVC), o mesmo não foi elaborado para atender finalidades clínicas ou médicas, e sim apresentar como a inteligência artificial e aprendizado de máquina podem contribuir positivamente na detecção de doenças ou até mesmo apoiar no direcionamento de esforços clínicos.

A construção de um modelo de predição de AVC implica em desafios como processamento de dados clínicos e fatores consideramos como de risco, assim o presente trabalho de conclusão de curso tem como objetivo trazer e avaliar alguns modelos e sua eficácia na identificação precoce de pacientes com risco de AVC.

O dataset selecionado foi obtido através do Kaggle. O Kaggle é uma plataforma para aprendizado de ciência de dados, é também uma comunidade, uma das maiores da internet, para assunto relacionados com ciência de dados.

- Stroke Prediction Dataset: Neste dataset é apresentado características clínicas que podem ser utilizadas para prever esses eventos de acidente vascular cerebral. O mesmo contém informações como idade, hipertensão, doença cardíaca, estado civil, índice de massa corporal (IMC), hábito de fumar, tipo de trabalho, tipo de residência, nível de glicose no sangue e se o paciente já teve um AVC prévio.

### 3. Coleta de Dados

- Site: [www.kaggle.com](http://www.kaggle.com)
- Linguagem de programação utilizada para análise de dados foi Python 3.9.13, com interface de desenvolvimento Jupyter Notebook e editor de código Visual Studio Code (VS Code) versão 1.76.1.

#### 3.1. Dataset

- Stroke Prediction Dataset:  
<https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset>

##### 3.1.1. Listagem descritiva das colunas:

**Tabela 1:** Stroke Prediction Dataset.csv

Nome do Atributo	Tradução	Descrição	Tipo
Id	Identificação	Identificador único do paciente	Integer
Gender	Gênero	Contém o sexo biológico da pessoa, masculino ou feminino.	Object
Age	Idade	Contém a idade do paciente	Float
Hypertension	Hipertensão	Informação se o paciente tem Hipertensão (0=Não / 1=Sim)	Integer
Heart_disease	Doença Cardíaca	Informação se o paciente tem Doença Cardíaca (0=Não / 1=Sim)	Integer
Ever_married	Já Casado	Informação se o paciente é Casado (Yes / No)	Object
Work_type	Tipo de Trabalho	Informação se o paciente exerce algum tipo de trabalho	Object
Residence_type	Tipo de Residência	Informações de tipo da residência	Object
Avg_glucose_level	Nível médio de Glicose	Contém informação de nível médio de glicose no sangue	Float
Bmi	IMC	Informação de Índice de Massa corporal do paciente	Float
Smoking_status	Condição de Fumante	Informação sobre histórico do paciente quanto ao tabagismo	Object
Stroke	AVC	Informação se o paciente teve AVC (0=Não / 1=Sim)	Integer

Fonte: Autor

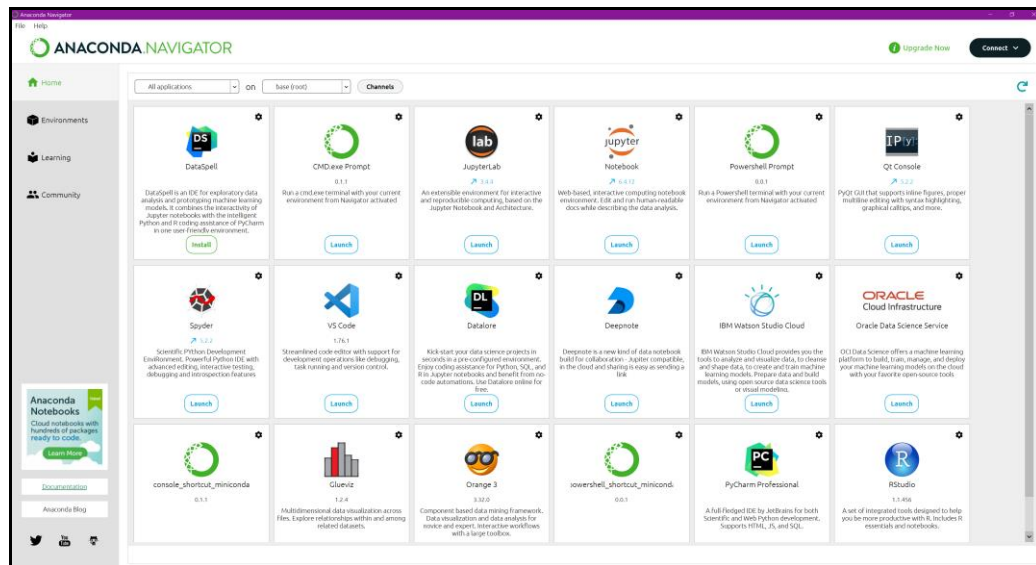
## 4. Processamento/Tratamento de Dados

Nesta seção será explorada todas as ferramentas e bibliotecas utilizadas para o processamento e tratamento dos dados.

### 4.1. Ferramentas

A ferramenta utilizada para desenvolvimento foi a distribuição do Anaconda, a plataforma inclui diversos editores de código-fonte além de pacotes e bibliotecas já pré-instaladas. Versão 2022.10 (figura 1), disponível em <https://www.anaconda.com/distribution/>.

**Figura 1:** Screenshot do Anaconda Navigator, da distribuição Anaconda.

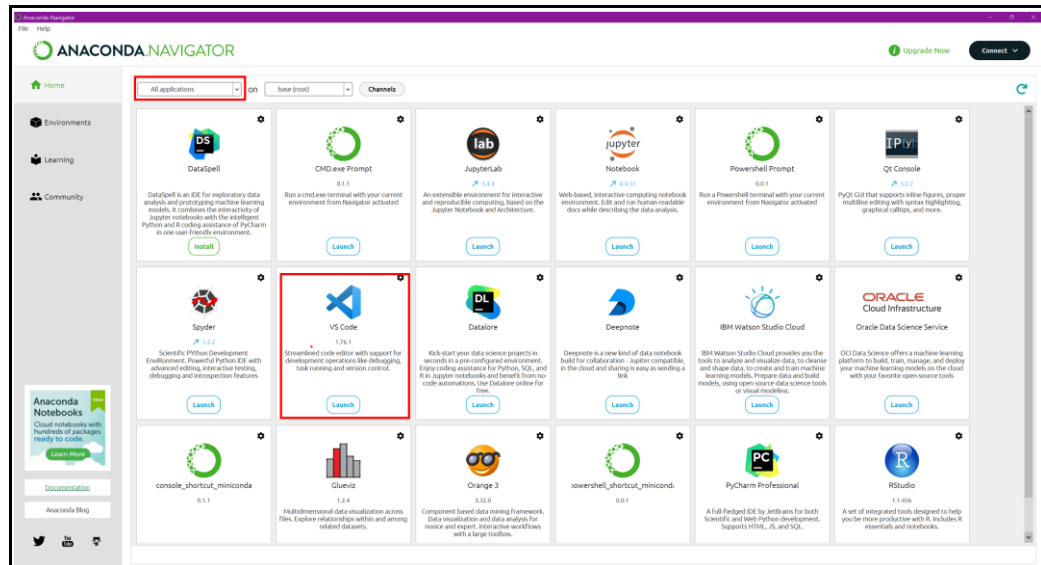


Fonte: Autor

Foi utilizado como ambiente integrado de desenvolvimento ou (IDE), o Visual Studio Code (VS Code) que também fornece interface ao formato Jupyter notebook. O mesmo se encontra nas aplicações disponíveis para download do Anaconda Navigator.

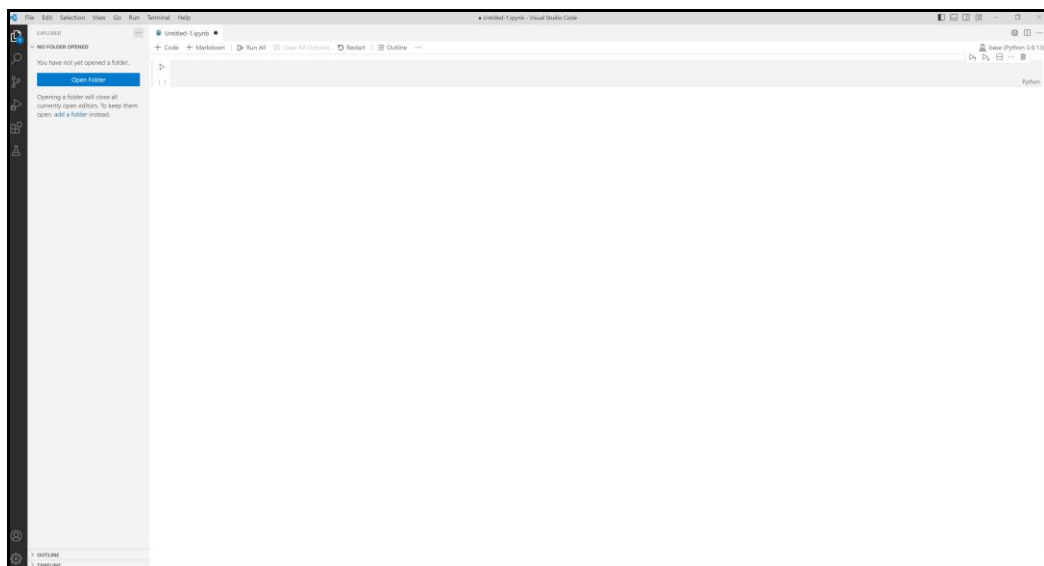


**Figura 2:** Screenshot do (VS Code) e todas as demais aplicações disponíveis para download na plataforma do Anaconda Navigator.



Fonte: Autor

**Figura 3:** Screenshot do editor Visual Studio Code (VS Code).



Fonte: Autor

## 4.2. Bibliotecas utilizadas

**Figura 4:** Screenshot importação das bibliotecas

```
import pandas as pd          # lib para manipulação de dados
import numpy as np          # lib para processamento numérico.
import matplotlib.pyplot as plt # lib para utilização dos gráficos
import matplotlib.ticker as mtick # lib para formatar os valores exibidos nos eixos de um gráfico gerado pela biblioteca Matplotlib.
import seaborn as sns       # lib para exibir dados estatísticos
import datetime             # lib para manipulação de datas
import os                   # lib para manipulação de pastas e diretórios

from sklearn.model_selection import train_test_split # lib para dividir um conjunto de dados em treino e teste.
from sklearn.linear_model import LinearRegression   # lib para ajuste de um modelo de regressão linear
from sklearn.pipeline import Pipeline               # lib para construção de um pipeline que encapsula vários passos de processamento de dados e ajuste de modelo.
from sklearn.preprocessing import StandardScaler    # lib para normalização de atributos dos dados, para que eles tenham média zero e desvio padrão 1.
from sklearn.tree import DecisionTreeRegressor      # lib para ajuste de um modelo de regressão com uma árvore de decisão.
```

Fonte: Autor

A tabela 2 mostra de forma um pouco mais detalhada as bibliotecas importadas.

**Tabela 2:** Descrição das bibliotecas utilizadas

Biblioteca	Descrição	Comando(s)
<i>Pandas</i>	<p>Pacote de ferramentas para análise de dados e manipulação, construída sobre a base da linguagem de programação python.</p> <p>Informações: <a href="https://pandas.pydata.org/">https://pandas.pydata.org/</a></p>	<code>import pandas as pd</code>
<i>Numpy</i>	<p>Pacote de ferramentas utilizada para realizar cálculos em Arrays Multidimensionais, construída sobre a base da linguagem de programação python.</p> <p>Informações: <a href="https://numpy.org/">https://numpy.org/</a></p>	<code>import numpy as np</code>
<i>Datetime</i>	<p>Pacote de ferramenta que fornece classes para manipulação de datas, construída sobre a base da linguagem de programação python.</p> <p>Informações: <a href="https://docs.python.org/pt-br/3/library/datetime.html">https://docs.python.org/pt-br/3/library/datetime.html</a></p>	<code>import Datetime</code>

<i>Matplotlib</i>	<p>Pacote de ferramentas utilizada para criação de gráficos e visualização de dados, construída sobre a base da linguagem de programação python.</p> <p>Informações:  <a href="https://matplotlib.org/">https://matplotlib.org/</a></p>	<pre>import matplotlib.pyplot as plt</pre>
<i>Seaborn</i>	<p>Pacote de ferramentas utilizadas para criação de gráficos e visualização de dados de alto nível baseada na lib Matplotlib.</p> <p>Informações:  <a href="https://seaborn.pydata.org/">https://seaborn.pydata.org/</a></p>	<pre>import seaborn as sns</pre>
<i>OS</i>	<p>Pacote de ferramentas utilizadas para manipular arquivos, pastas e diretórios do sistema.</p> <p>Informações:  <a href="https://docs.python.org/3/library/os.html">https://docs.python.org/3/library/os.html</a></p>	<pre>Import OS</pre>
<i>Sklearn</i>	<p>Pacote de ferramentas utilizadas para trabalhar com Machine Learning, com ela são importados diversos métodos, algoritmos e técnicas para facilitar a codificação.</p> <p>Informações:  <a href="https://scikit-learn.org/stable/">https://scikit-learn.org/stable/</a></p>	<pre>From sklearn.model_selection import train_test_split  -----  from sklearn.neighbors import KNeighborsClassifier  from sklearn.metrics import accuracy_score, classification_report, confusion_matrix  -----  from sklearn.preprocessing import StandardScaler  from sklearn.tree import</pre>

		<pre> DecisionTreeRegressor  -----  from sklearn.model_selection import GridSearchCV  -----  from sklearn.pipeline import Pipeline  -----  from sklearn.ensemble import RandomForestClassifier </pre>
--	--	---

Fonte: Autor

#### 4.2.1. Leitura do Dataset: Stroke Prediction Dataset.csv

Na Figura abaixo é apresentado código utilizado para obter os dados salvos em formato csv e transformá-los em um formato Data Frame para iniciarmos análise.

**Figura 5:** Screenshot lendo o dataset e armazenando em um dataframe

```

# Lendo o dataset e armazenando em um dataframe chamado df_stroke
path =r'data\healthcare-dataset-stroke-data.csv'
df_stroke = pd.read_csv(path)

```

✓ 0.1s

Fonte: Autor

A seguir é apresentado o resultado em Data Frame dos dados coletados, selecionando apenas os 5 últimos registros encontrados através do comando `df_stroke.tail()`.

**Figura 6:** Screenshot exibindo dataset Stroke data

```
# Selecionando os ultimos registros do dataframe para conferencia
df_stroke.tail()
```

✓ 0.1s

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
5105	18234	Female	80.0	1	0	Yes	Private	Urban	83.75	NaN	never smoked	0
5106	44873	Female	81.0	0	0	Yes	Self-employed	Urban	125.20	40.0	never smoked	0
5107	19723	Female	35.0	0	0	Yes	Self-employed	Rural	82.99	30.6	never smoked	0
5108	37544	Male	51.0	0	0	Yes	Private	Rural	166.29	25.6	formerly smoked	0
5109	44679	Female	44.0	0	0	Yes	Govt_job	Urban	85.28	26.2	Unknown	0

Fonte: Autor

#### 4.2.2. Informações do DataFrame

Os códigos abaixo imprimem informações sobre o conjunto de dados "Stroke Prediction". As informações incluem a dimensão do conjunto de dados, que mostra o número de linhas e colunas, uma lista de todas as colunas presentes no conjunto de dados e os tipos de dados de cada coluna. Essas informações são importantes para entender a estrutura do conjunto de dados e selecionar as variáveis relevantes para análise.

**Figura 7:** Screenshot exibindo informações sobre o dataset Stroke data

```
print("\nDimensões de Stroke Prediction Dataset:\n{}\n".format(df_stroke.shape))
print("\nCampos de Stroke Prediction Dataset:\n{}\n".format(list(df_stroke.keys())))
print("\nTipos dos dados:\n{}\n".format(df_stroke.dtypes))
```

Dimensões de Stroke Prediction Dataset:  
(5110, 12)

Campos de Stroke Prediction Dataset:  
['id', 'gender', 'age', 'hypertension', 'heart\_disease', 'ever\_married', 'work\_type', 'Residence\_type', 'avg\_glucose\_level', 'bmi', 'smoking\_status', 'stroke']

Tipos dos dados:	
id	int64
gender	object
age	float64
hypertension	int64
heart_disease	int64
ever_married	object
work_type	object
Residence_type	object
avg_glucose_level	float64
bmi	float64
smoking_status	object
stroke	int64
dtype: object	

Fonte: Autor

### 4.3. Tratamento dos dados

Nesta seção será detalhado cada tratamento realizado. A formatação dos dados será cuidadosamente verificada e serão aplicados filtros se necessário, para assegurar a sua qualidade. Qualquer dado considerado desnecessário será devidamente excluído.

#### 4.3.1. Renomeando colunas

As colunas foram renomeadas com a intenção de padronizar os dados e facilitar na execução das análises que serão executadas na sequência deste trabalho. Conforme indicado Figura 8 abaixo, usamos a função *rename()* para efetuar esse ajuste.

**Figura 8:** Screenshot Renomeando colunas Dataframe: df\_stroke

```
df_stroke.rename(  
    {'gender': 'Genero',  
     'age': 'Idade',  
     'hypertension': 'Hipertensao',  
     'heart_disease': 'Doença_cardiaca',  
     'ever_married': 'Ja_Casado',  
     'work_type': 'Tipo_de_trabalho',  
     'Residence_type': 'Tipo_de_residencia',  
     'avg_glucose_level': 'Nivel_medio_glicose',  
     'bmi': 'IMC',  
     'smoking_status': 'Condição_de_fumante',  
     'stroke': 'AVC',  
    },  
    axis=1, inplace=True)
```

Fonte: Autor

### 4.3.2. Removendo colunas

O atributo *ID* será removido, é apenas uma chave primária para identificar um o paciente, não é relevante para o trabalho.

**Figura 9:** Screenshot Remoção coluna ID Dataframe: df\_stroke

```
del df_stroke['id'] # Coluna Id - Identificador unico do paciente
```

Fonte: Autor

### 4.3.3. Padronizando Tipos das Colunas

Usamos o método *astype()* para ajustar a coluna Idade para formato Inteiro.

**Figura 10:** Screenshot ajuste de formato da coluna Idade com *astype()*

```
df_stroke["Idade"] = df_stroke["Idade"].astype(int)
```

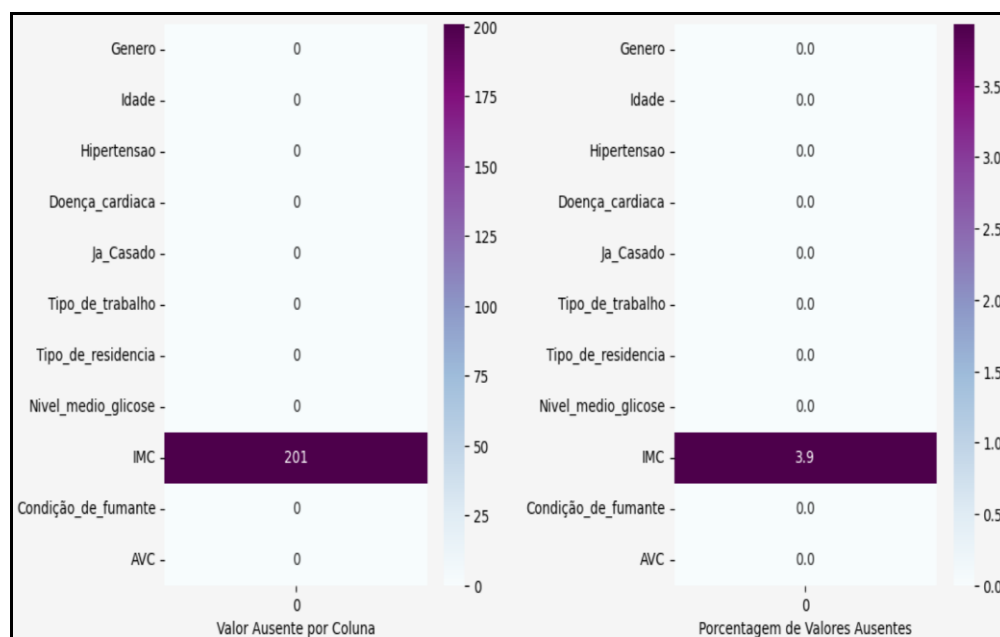
Fonte: Autor

#### 4.3.4. Tratamento de Valores Ausentes

Como podemos observar na Figura 11, o atributo *IMC* contém 201 valores ausentes, no qual representa 3,9% do total da base.

Por se tratar de uma porcentagem baixa de dados ausentes em comparação com o total poderíamos eliminar essas 201 linhas, ou até mesmo substituir os dados ausentes por valores padrões ou médios, porém, como estamos tratando de dados clínicos talvez no "*mundo real*" essas práticas represente um enorme risco de erro em um diagnóstico prevenido.

**Figura 11:** Screenshot – Gráfico Heatmap de valores Ausentes por coluna



Fonte: Autor

Para desenvolver os dados em forma gráfica, foi utilizado o seguinte código, exibido nas Figuras 12.



**Figura 12:** Screenshot Código para plotar gráficos heatmap dados ausentes

```

#Parametrização graficos
fig = plt.figure(figsize=(12, 6), dpi=100)
gs = fig.add_gridspec(1, 2)
gs.update(wspace=0.55, hspace=0.5)

cor_fundo = '#f5f5f5'

fig.patch.set_facecolor(cor_fundo)
ax0 = fig.add_subplot(gs[0, 0])
ax0.set_facecolor(cor_fundo)
ax1 = fig.add_subplot(gs[0, 1])
ax1.set_facecolor(cor_fundo)

#grafico 1
ax = sns.heatmap(ax=ax0, data=df_stroke.isna().sum().to_frame(), annot=True, fmt='d', cmap='BuPu')
ax.set_xlabel('Valor Ausente por Coluna')

#grafico 2
# calcular a porcentagem de valores ausentes em cada coluna
missing_perc = df_stroke.isna().mean() * 100

# plotar o gráfico
ax = sns.heatmap(ax=ax1, data = missing_perc.to_frame(), annot=True, fmt='.1f', cmap='BuPu')
ax.set_xlabel('Porcentagem de Valores Ausentes')

plt.show()

```

Fonte: Autor

Como alternativa para esse trabalho vamos optar por utilizar um modelo de árvore de decisão simples que, com base na idade e no sexo de todas as outras amostras dará uma previsão justa para os valores ausentes.

**Figura 13:** Screenshot - Código para modelo de árvore de decisão simples

```
# Cria um pipeline para padronizar os recursos e ajustar um modelo de árvore de decisão
df_stroke_imc_pipe = Pipeline(steps=[
    ('scale', StandardScaler()),
    ('lr', DecisionTreeRegressor(random_state=42))
])

# Seleciona as colunas relevantes do dataframe e codifica o Genero como valores numéricos
X = df_stroke[['Idade', 'Genero', 'IMC']].copy()
X['Genero'] = X['Genero'].replace({'Male': 0, 'Female': 1, 'Other': -1}).astype(int)

# Separa as amostras com valores de IMC faltantes
missing = X[X['IMC'].isna()]

# Ajusta o modelo com as amostras que têm valores de IMC
X = X[~X['IMC'].isna()]
Y = X.pop('IMC')
df_stroke_imc_pipe.fit(X, Y)

# Faz as previsões de IMC faltantes e preenche os valores previstos no dataframe original
predicted_imc = pd.Series(df_stroke_imc_pipe.predict(missing[['Idade', 'Genero']]), index=missing.index)
df_stroke.loc[missing.index, 'IMC'] = predicted_imc
```

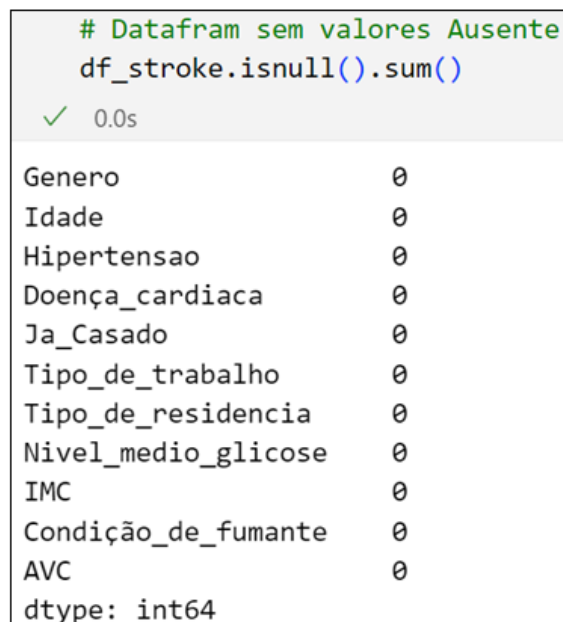
Fonte: Autor

#### 4.3.5. Informações do Dataframe Final

Para garantir que os dados estão padronizados e que não existem valores nulos no DataFrame, foi executado a função *isnull()* onde recupera todos os registros que estão com valores nulo e logo em seguida aplicado a função *sum()*, pois somamos a quantidade de registros nulos da coluna correspondente, conforme a Figura 14.

Neste caso não existe nenhuma coluna com registro nulo, por isso os valores aparecem zerados.

**Figura 14:** Screenshot - Conferindo valores nulos no Dataframe df\_stroke.



```
# Datafram sem valores Ausente
df_stroke.isnull().sum()
```

✓ 0.0s

Genero	0
Idade	0
Hipertensao	0
Doença_cardiaca	0
Ja_Casado	0
Tipo_de_trabalho	0
Tipo_de_residencia	0
Nivel_medio_glicose	0
IMC	0
Condição_de_fumante	0
AVC	0
dtype: int64	

Fonte: Autor

## 5. Análise e Exploração dos Dados

Nesta etapa serão realizadas análises exploratórias para identificar correlações e associações entre as variáveis do conjunto de dados, permitindo a identificação de tendências, padrões e insights importantes. Adicionalmente, serão aplicadas técnicas de estatística descritiva para resumir as características do conjunto de dados, incluindo medidas de centralidade, dispersão e simetria.

Além disso, as informações obtidas através do pré-processamento dos dados, como a limpeza e transformação de dados, serão levadas em conta para garantir a qualidade dos resultados obtidos e minimizar o impacto de possíveis erros.

Por fim, serão apresentadas as principais descobertas e conclusões derivadas das análises realizadas, fornecendo uma visão geral das principais informações encontradas no conjunto de dados.

### 5.1. Estatística descritiva dos dados

Ao utilizar o comando *describe()* no conjunto de dados, podemos obter algumas informações importantes que auxiliam na análise inicial dos dados fornecidos. Ele fornece informações importantes sobre as variáveis numéricas presentes no conjunto de dados, como a contagem (count), a média (mean), o desvio padrão (std), os valores mínimo (min), máximo (max) e os quartis (25%, 50% e 75%).

Essas informações permitem obter uma visão geral sobre as características dos dados, como a variação, a distribuição e a centralidade. Além disso, o *describe()* também pode fornecer informações sobre as variáveis categóricas, como a contagem, o número de valores exclusivos (unique), o valor mais frequente (top) e a frequência do valor mais frequente (freq). O comando *describe()* exibe prioritariamente os campos numéricos. Os campos categóricos são exibidos isoladamente.

**Figura 15:** Screenshot – Estatística descritiva dos dados com *describe()*.

Exibindo apenas os campos numéricos:				
	Idade	Hipertensao	Doença_cardiaca	Nivel_medio_glicose \
count	5110.000000	5110.000000	5110.000000	5110.000000
mean	43.215264	0.097456	0.054012	106.147677
std	22.633866	0.296607	0.226063	45.283560
min	0.000000	0.000000	0.000000	55.120000
25%	25.000000	0.000000	0.000000	77.245000
50%	45.000000	0.000000	0.000000	91.885000
75%	61.000000	0.000000	0.000000	114.090000
max	82.000000	1.000000	1.000000	271.740000

	IMC	AVC
count	5110.000000	5110.000000
mean	28.919073	0.048728
std	7.730623	0.215320
min	10.300000	0.000000
25%	23.700000	0.000000
50%	28.300000	0.000000
75%	32.900000	0.000000
max	97.600000	1.000000

Exibindo apenas os campos categóricos:				
	Genero	Ja_Casado	Tipo_de_trabalho	Tipo_de_residencia \
count	5110	5110	5110	5110
unique	3	2	5	2
top	Female	Yes	Private	Urban
freq	2994	3353	2925	2596

Condição_de_fumante	
count	5110
unique	4
top	never smoked
freq	1892

**Figura 16:** Código para comando *describe()*, figura 15.

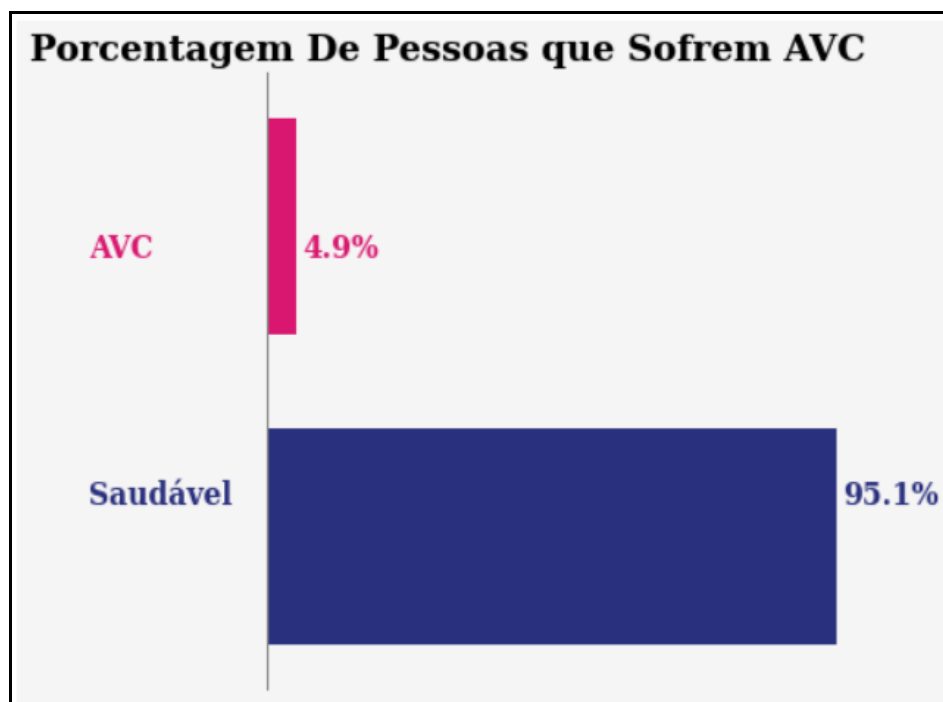
```
# Campos Numericos
print("\nExibindo apenas os campos numéricos:\n{0}\n".format(df_stroke.describe()))

# Campos Categóricos
categ = df_stroke.dtypes[df_stroke.dtypes == "object"].index
print("\nExibindo apenas os campos categóricos:\n{0}\n".format(df_stroke[categ].describe(), sep='\n'))
```

### 5.1.1. Análise de pacientes que tiveram AVC no conjunto de dados.

Encontra-se um enorme desequilíbrio das classes! Seria uma amostra excessiva da classe minoritária. No caso do conjunto de dados em questão, apenas cerca de 5% das pessoas sofreram um AVC, enquanto as outras 95% não tiveram essa condição. Isso significa que a classe "AVC" é uma classe minoritária em relação à classe "Não AVC". Essa distribuição desigual pode afetar a análise e os resultados obtidos a partir desses dados, especialmente em modelos de aprendizado de máquina que podem ser treinados de maneira enviesada devido à falta de representação da classe minoritária. Existem algumas técnicas que podem ajudar a lidar com dados desbalanceados e minimizar o viés na análise e nos resultados, algumas delas serão abordadas ao longo do trabalho.

**Figura 17:** Screenshot – Gráfico Porcentagem de Pessoas que sofrem AVC



Fonte: Autor

Para desenvolver os dados em forma gráfica, foi utilizado o seguinte código exibido na Figura 18. Foi utilizado a biblioteca *Matplotlib*, além de customização do gráfico, alteração das cores e título.

**Figura 18:** Código - Gráfico Porcentagem de Pessoas que sofrem AVC

```
# calcular a porcentagem de observações com e sem AVC
avc_count = df_stroke['AVC'].value_counts()
avc_percent = 100 * avc_count / len(df_stroke)

# criar a figura e os eixos
fig, ax = plt.subplots(figsize=(6, 6), dpi=70)

# criar fundo
fig.patch.set_facecolor('#f6f5f5')
ax.set_facecolor('#f6f5f5')

# plotar as barras
ax.barh([0], avc_percent[0], height=0.7, color='#29307d')
ax.barh([1], avc_percent[1], height=0.7, color='#d91770')

# adicionar os rótulos
plt.text(-30, 0.1, 'Saudável', {'font': 'Serif', 'weight': 'bold', 'size': '16', 'color': '#29307d'})
plt.text(-30, 0.9, 'AVC', {'font': 'Serif', 'weight': 'bold', 'size': '16', 'color': '#d91770'})
plt.text(avc_percent[0] + 1, 0.1, f'{avc_percent[0]:.1f}%', {'font': 'Serif', 'weight': 'bold', 'size': '16', 'color': '#29307d'})
plt.text(avc_percent[1] + 1, 0.9, f'{avc_percent[1]:.1f}%', {'font': 'Serif', 'weight': 'bold', 'size': '16', 'color': '#d91770'})

# configurar os eixos
ax.set_xlim(0, 100)
ax.set_ylim(-0.5, 1.5)
ax.axvline(x=0, color='gray', linewidth=1)
ax.set_axis_off()

# adicionar o título
ax.set_title('Porcentagem De Pessoas que Sofrem AVC', font= 'Serif', fontsize=20, fontweight='bold', loc='right')

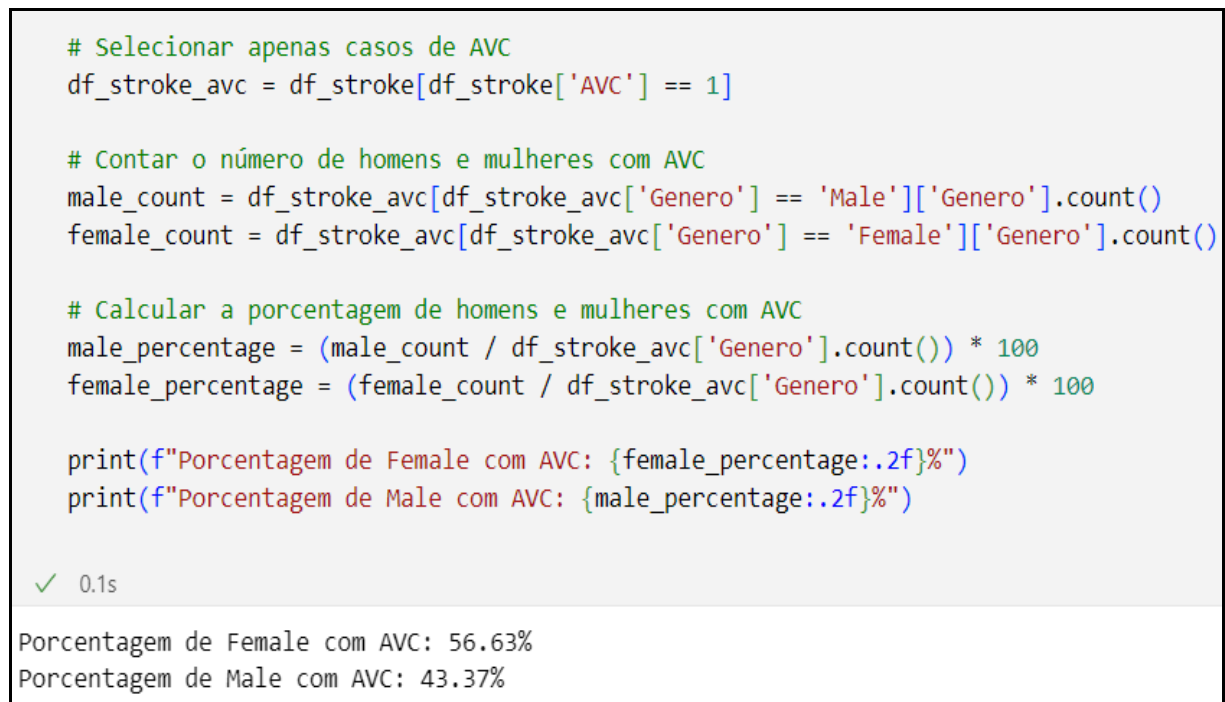
plt.show()
```

Fonte: Autor

### 5.1.2. Análise de AVC por Gênero

Com base na análise realizada no código fornecido abaixo, foi observado que 56,63% dos casos de AVC ocorreram em mulheres, enquanto 43,37% ocorreram em homens. Isso indica que as mulheres têm uma maior probabilidade de ter AVC do que os homens, com uma diferença de cerca de 13%.

**Figura 19:** Screenshot – Resultado em porcentagem de AVC por gênero



Fonte: Autor

### 5.1.3. Análise de distribuição dos atributos

A figura abaixo é um gráfico de densidade de probabilidade que mostra a distribuição da idade para homens e mulheres no conjunto de dados. O gráfico está dividido em duas cores, roxo para homens e rosa para mulheres, permitindo a fácil visualização da diferença na distribuição de idade entre os gêneros.

Podemos observar que há uma sobreposição significativa entre as distribuições de idade dos homens e mulheres, sugerindo que a idade pode não ser um fator determinante para o risco de AVC.

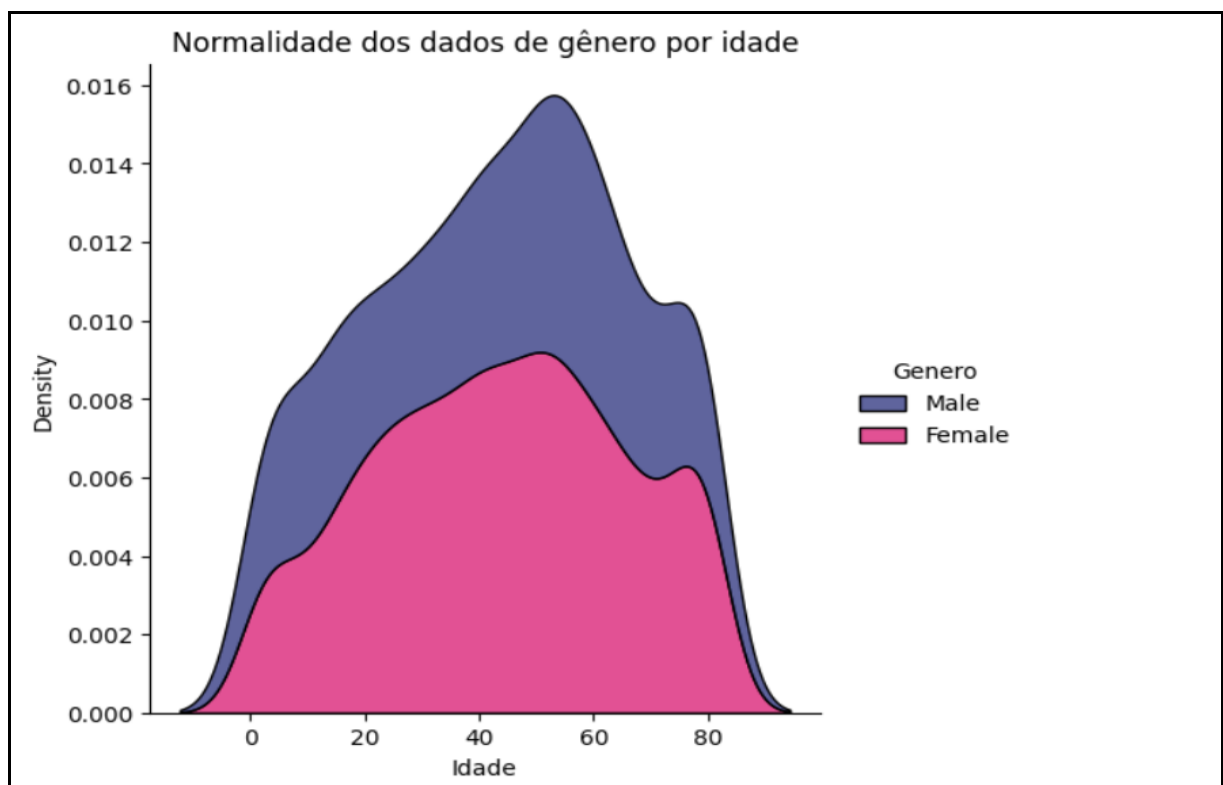
No entanto, a distribuição de idade para mulheres é mais ampla, o que pode



indicar que há uma maior variabilidade no risco de AVC entre as mulheres, e isso pode ser correlacionado com outros fatores como a gravidez, menopausa e outros fatores de risco específicos do sexo feminino.

Em resumo, o gráfico indica que a idade pode não ser um fator determinante para o risco de AVC, mas é necessário uma análise mais aprofundada de outros fatores de risco específicos do sexo feminino para entender a maior variabilidade na distribuição de idade das mulheres.

**Figura 20:** Screenshot – Normalidade dos dados de Gênero por Idade



Fonte: Autor

Para desenvolver os dados distribuição em forma gráfica, foi utilizado o seguinte código exibido na Figura 21. Foi utilizado a biblioteca *Seaborn* com a versão de gráfico *displot*, além de customização do gráfico, alteração das cores e título.

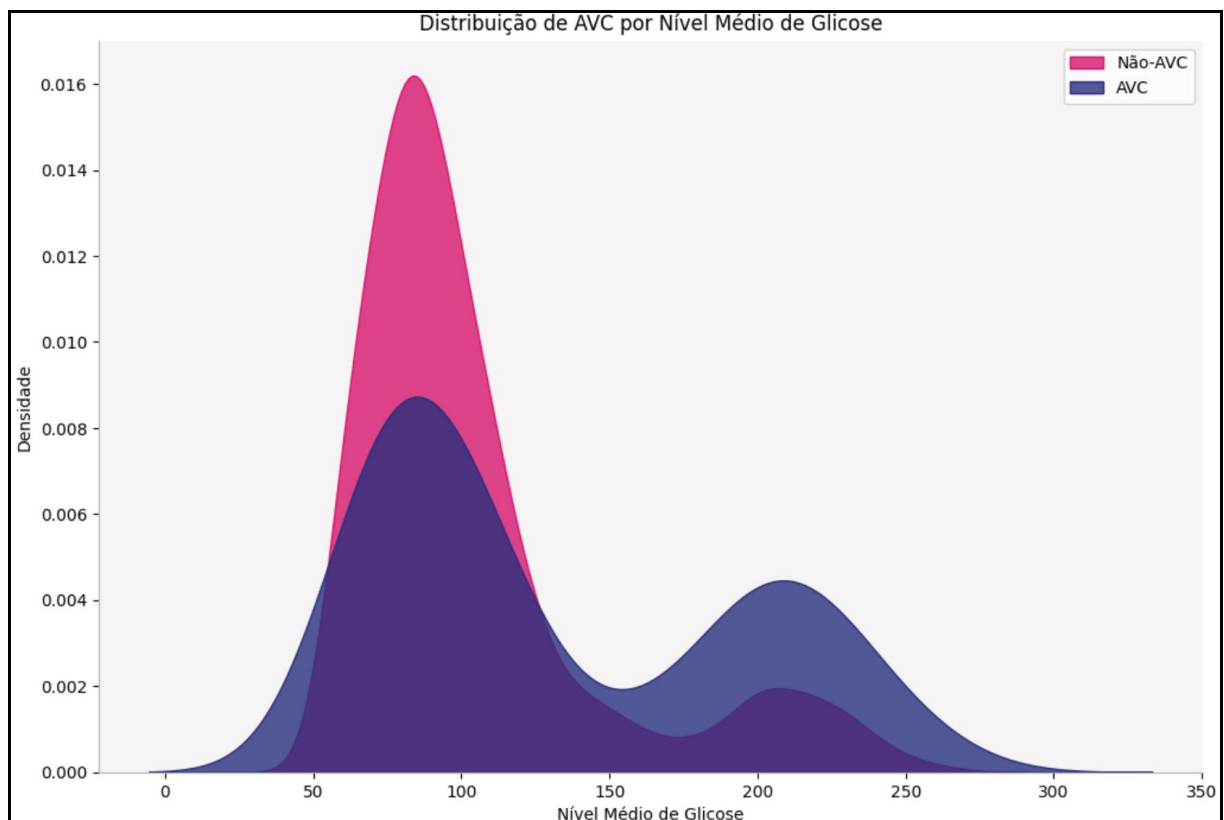
**Figura 21:** Código – Normalidade dos dados de Gênero por Idade

```
df_stroke_gender = df_stroke.loc[df_stroke['Genero'].isin(['Male', 'Female'])]
sns.displot(df_stroke_gender, x="Idade", hue="Genero", kind="kde", multiple='stack', palette=['#29307d', '#d91770'])
plt.title("Normalidade dos dados de gênero por idade")
plt.show()
```

Fonte: Autor

### 5.1.3.1. Distribuição de AVC por Nível Médio de Glicose

O gráfico de densidade de probabilidade exibido mostra a distribuição do nível médio de glicose no sangue para indivíduos com e sem AVC. A distribuição de nível médio de glicose para indivíduos com AVC é ligeiramente mais alta em relação aos indivíduos sem AVC, o que sugere uma possível associação entre o nível médio de glicose e AVC. O gráfico permite uma fácil visualização da diferença na distribuição de nível médio de glicose entre os dois grupos.

**Figura 22:** Screenshot – Gráfico de AVC por Nível Médio de Glicose

Fonte: Autor

**Figura 23:** Código – Gráfico de AVC por Nível Médio de Glicose

```

sns.set_palette(sns.color_palette(['#d91770', '#29307d']))

fig, ax = plt.subplots(figsize=(12, 8))

sns.kdeplot(x='Nivel_medio_glicose', data=df_stroke[df_stroke['AVC'] == 0], shade=True, ax=ax, alpha=0.8, label='Não-AVC')
sns.kdeplot(x='Nivel_medio_glicose', data=df_stroke[df_stroke['AVC'] == 1], shade=True, ax=ax, alpha=0.8, label='AVC')

plt.xlabel('Nível Médio de Glicose')
plt.ylabel('Densidade')
plt.title('Distribuição de AVC por Nível Médio de Glicose')

ax.set_facecolor('#f6f5f5')
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_color('#b0b0b0')
ax.spines['left'].set_color('#b0b0b0')

plt.legend()
plt.show()

```

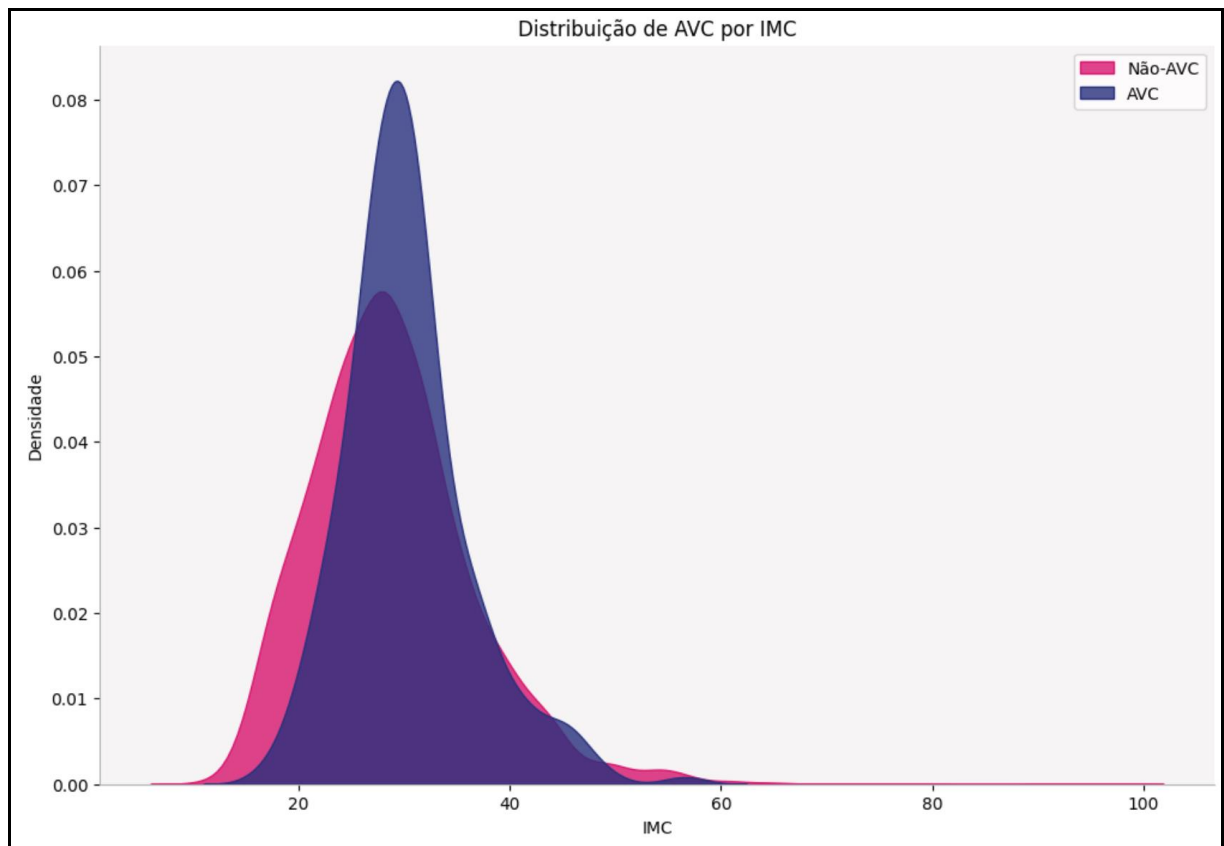
Fonte: Autor

Para desenvolver os dados distribuição em forma gráfica de AVC por Nível Médio de Glicose, foi utilizado o seguinte código exibido na Figura 23. Foi utilizado a biblioteca Seaborn com a versão de gráfico *kdeplot*,

### 5.1.3.2. Distribuição de AVC por IMC

Podemos observar que não há uma diferença significativa na distribuição do IMC entre indivíduos com e sem AVC, sugerindo que o IMC pode não ser um fator determinante para o risco de AVC. No entanto, é importante destacar que o IMC é apenas uma medida de peso em relação à altura e não leva em consideração a composição corporal, como a proporção de massa muscular e gordura, que pode estar relacionada ao risco de AVC.

Em resumo, o gráfico indica que o IMC pode não ser um fator determinante para o risco de AVC, mas é necessário uma análise mais aprofundada de outros fatores relacionados à composição corporal para entender a associação entre peso e risco de AVC.

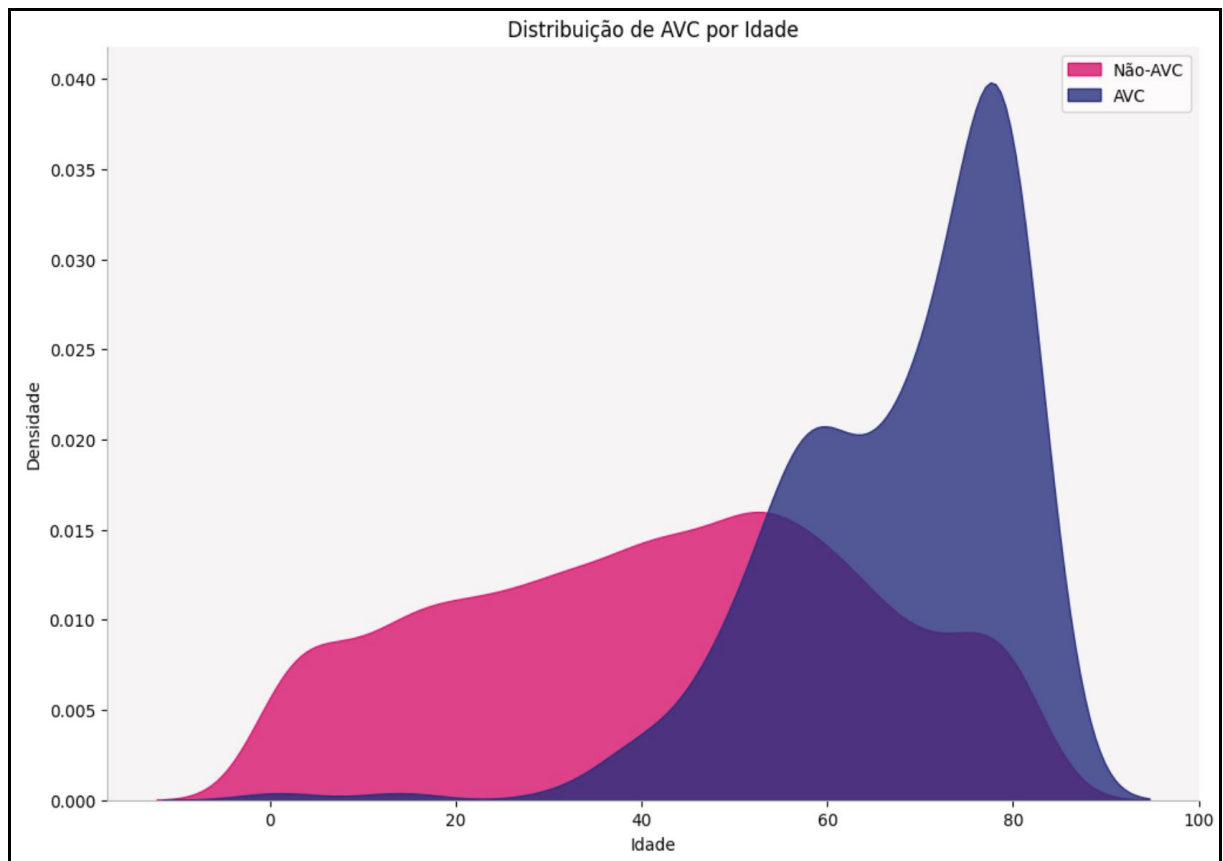
**Figura 24:** Screenshot – Gráfico de AVC por IMC

Fonte: Autor

### 5.1.3.3. Distribuição de AVC por Idade

É possível observar no gráfico de densidade de probabilidade que a distribuição da idade para indivíduos com AVC está deslocada para a direita em relação aos indivíduos sem AVC, o que sugere que o AVC é mais comum em pessoas mais velhas. Os valores para a distribuição de idade de indivíduos sem AVC variam principalmente entre 40 e 70 anos, enquanto a distribuição para indivíduos com AVC tem a maior parte de sua densidade entre 60 e 80 anos.

**Figura 25:** Screenshot – Gráfico de AVC por Idade



Fonte: Autor

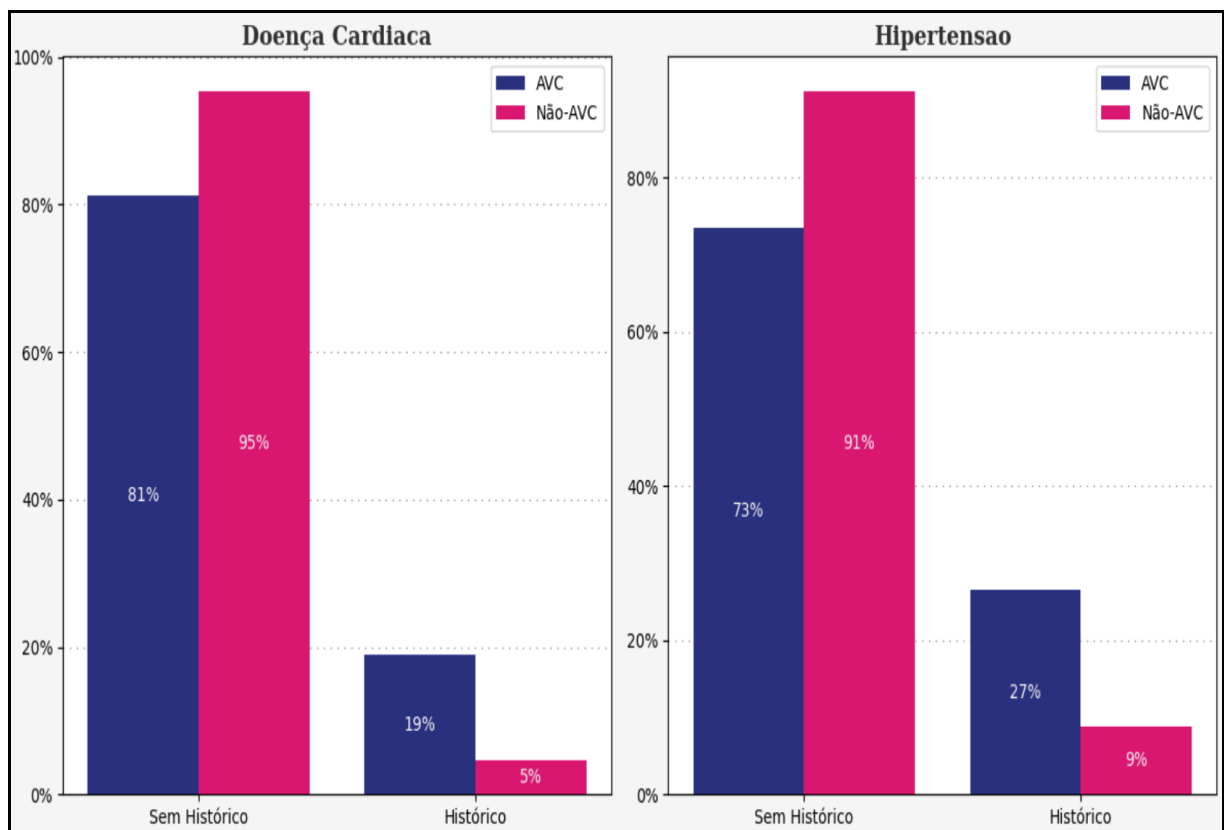
## 5.2. Histórico de Doença Cardíaca e Hipertensão no conjunto de dados.

Na figura 26 temos dois gráficos de barras, compostos por duas barras empilhadas, que representam a porcentagem de indivíduos com e sem AVC que possuem histórico ou não de doença cardíaca e hipertensão.

Na primeira barra, observa-se que indivíduos com AVC têm maior probabilidade de ter histórico de doença cardíaca do que aqueles sem AVC. Na segunda barra, observa-se que a porcentagem de indivíduos com AVC e hipertensão é significativamente maior do que a porcentagem de indivíduos sem AVC e hipertensão.

Isso sugere que tanto a doença cardíaca quanto a hipertensão podem ser fatores de risco para AVC.

**Figura 26:** Gráfico de barras de Doença Cardíaca e Hipertensão



Fonte: Autor

**Figura 27:** Código para criar função e plotar gráfico de barras da figura 26

```
# Função para calcular a porcentagem de Doença Cardíaca e Hipertensão
def stacked_bar(df, attr, label_attr, label1, label2, title, ax):
    positivo = pd.DataFrame(df[df[label_attr] == 1][attr].value_counts())
    positivo["Percentage"] = positivo[attr].apply(lambda x: x/sum(positivo[attr])*100)
    negativo = pd.DataFrame(df[df[label_attr] == 0][attr].value_counts())
    negativo["Percentage"] = negativo[attr].apply(lambda x: x/sum(negativo[attr])*100)

    # Definir configuração e cor das barras.
    x = np.arange(len(positivo))
    ax.grid(color='gray', linestyle=':', axis='y', zorder=0, dashes=(1, 5))
    ax.bar(x, height=positivo["Percentage"], zorder=3, color="#29307d", width=0.4)
    ax.bar(x + 0.4, height=negativo["Percentage"], zorder=3, color="#d91770", width=0.4)
    ax.set_xticks(x + 0.2)
    ax.set_xticklabels([label1, label2])
    ax.yaxis.set_major_formatter(mtick.PercentFormatter())
    ax.yaxis.set_major_locator(mtick.MultipleLocator(20))
    for i, j in zip(x, positivo["Percentage"]):
        ax.annotate(f'{j:0.0f}%', xy=(i, j/2), color='#f6f5f5', horizontalalignment='center', verticalalignment='center')
    for i, j in zip(x, negativo["Percentage"]):
        ax.annotate(f'{j:0.0f}%', xy=(i+0.4, j/2), color='#f6f5f5', horizontalalignment='center', verticalalignment='center')
    ax.set_title(title, fontsize=14, fontweight='bold', fontfamily='serif', color="#323232")
```

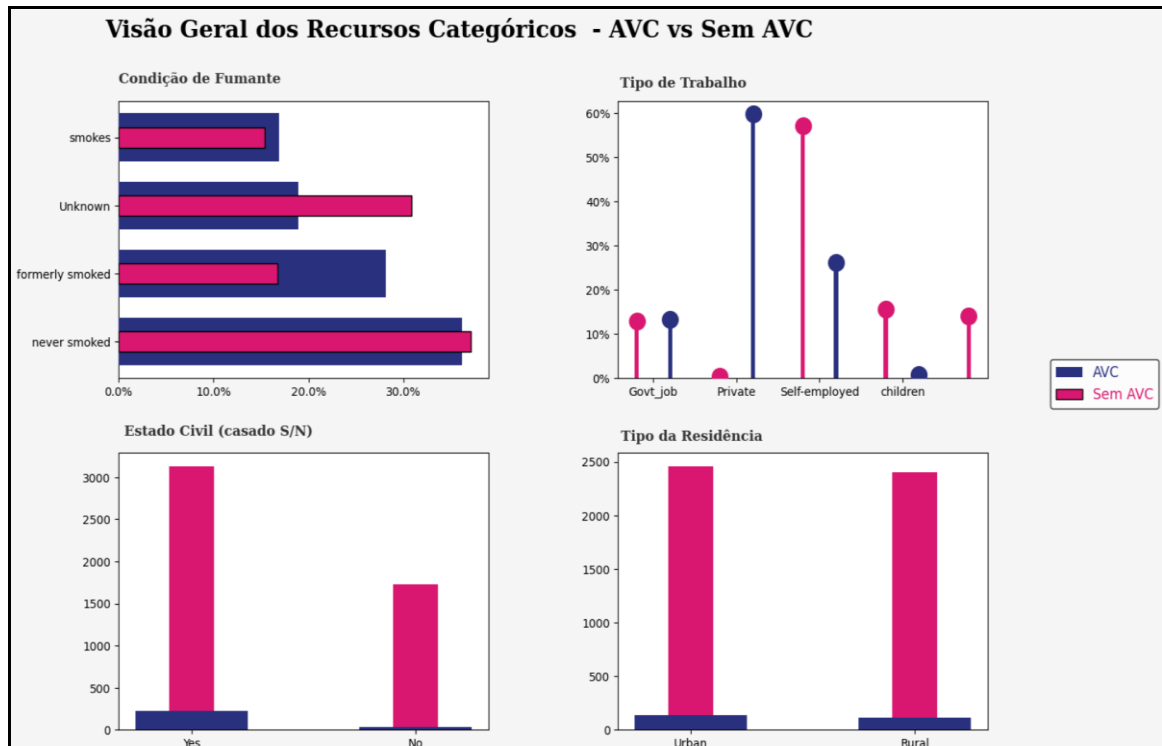
Fonte: Autor

### 5.3. Análise visual dos atributos Categóricos

No painel abaixo, podemos visualizar um gráfico de barras que apresenta os resultados em porcentagem de algumas variáveis categóricas em relação aos casos de AVC. Observando os resultados, podemos inferir que os atributos *Estado Civil* e *Tipo de Residência* não parecem ter grande influência no surgimento de AVCs.

No entanto, em relação à *condição de fumante*, podemos notar que a maioria dos casos de AVC ocorreram em indivíduos que nunca fumaram (37,03%), seguido de ex-fumantes (17,32%) e fumantes atuais (15,44%) e dados desconhecidos (30,22%). Quanto ao tipo de trabalho, a maioria dos casos de AVC ocorreram em pessoas empregadas no setor privado (57,24%), seguido de trabalhadores autônomos (16,03%), crianças (13,44%), funcionários públicos (12,85%) e aqueles que nunca trabalharam (0,43%). Por fim, em relação ao tipo de residência, os casos de AVCs foram praticamente igualmente distribuídos entre áreas urbanas (50,80%) e rurais (49,20%).

Os dados apresentados são importantes para a análise exploratória inicial, mas ainda são insuficientes para decidir quais variáveis devem ser incluídas ou excluídas em um modelo.

**Figura 28:** Screenshot – Gráfico Geral dos Recursos Categóricos

Fonte: Autor

### 5.3.1. Matriz de Correlação

Antes de criar a matriz de correlação, é necessário converter as variáveis categóricas em numéricas para que possam ser incluídas na análise. Nesse sentido, utilizamos a função *map()* para atribuir valores numéricos a cada categoria das variáveis categóricas presentes em nossa base de dados de AVC. Com as variáveis ajustadas, podemos criar a matriz de correlação e avaliar a relação entre as variáveis quantitativas e categóricas.

**Figura 29:** Código para converter dados categóricos em numéricos

```
df_stroke['Genero'] = df_stroke['Genero'].map({'Male':0, 'Female':1})
df_stroke['Tipo_de_residencia'] = df_stroke['Tipo_de_residencia'].map({'Urban':0, 'Rural':1})
df_stroke['Condição_de_fumante'] = df_stroke['Condição_de_fumante'].map({'formerly smoked':0, 'never smoked':1, 'smokes':2, 'Unknown':3})
df_stroke['Ja_Casado'] = df_stroke['Ja_Casado'].map({'Yes':0, 'No':1})
df_stroke['Tipo_de_trabalho'] = df_stroke['Tipo_de_trabalho'].map({'Private':0, 'Self-employed':1, 'Govt_job':2, 'children':3, 'Never_worked':4})
```

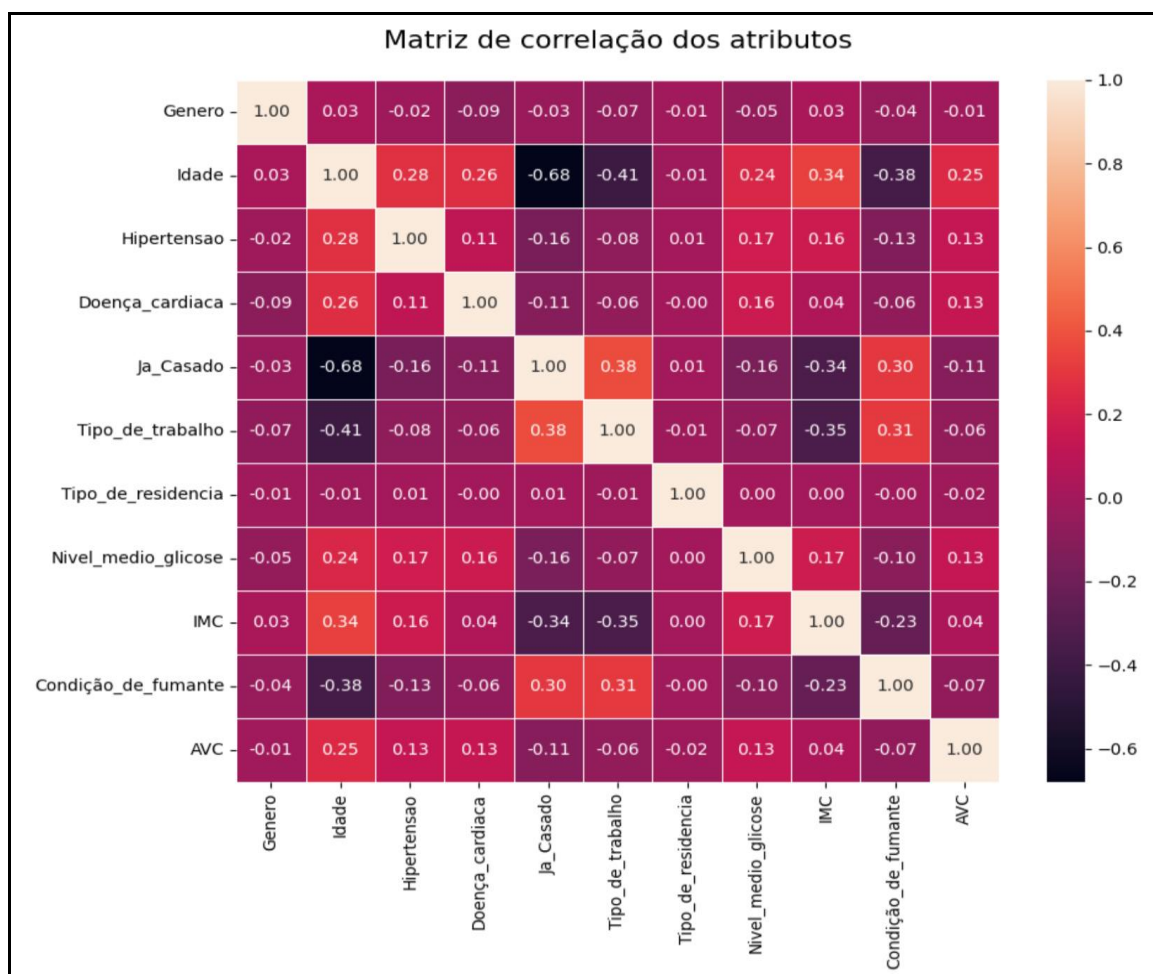
Fonte: Autor



### 5.3.1.1. Gráfico da Matriz de Correlação

A matriz de correlação é uma ferramenta gráfica usada para analisar as relações entre variáveis em um conjunto de dados. Cada célula na grade representa o valor do coeficiente de correlação entre duas variáveis, que indica a força e a direção da relação entre elas. É uma matriz quadrada, onde cada linha representa uma variável e todas as colunas representam as mesmas variáveis que as linhas, portanto, o número de linhas é igual ao número de colunas. A matriz é simétrica, o que significa que a correlação entre a e b será a mesma que entre b e a. Os elementos diagonais são sempre 1, já que representam a correlação de cada variável consigo mesma. Os pontos dos eixos denotam o recurso que cada um deles representa.

**Figura 30:** Screenshot Gráfico Matrix de Correlação dos Atributos



Fonte: Autor

Um grande valor positivo (próximo a 1,0) indica uma forte correlação positiva, ou seja, se o valor de uma das variáveis aumenta, o valor da outra variável também aumenta. Um grande valor negativo (próximo a -1,0) indica uma forte correlação negativa, ou seja, o valor de uma variável diminui com o aumento da outra e vice-versa. Um valor próximo a 0 (positivo ou negativo) indica a ausência de qualquer correlação entre as duas variáveis e, portanto, essas variáveis são independentes uma da outra.

Além disso, cada célula na matriz é representada por tons de uma cor, onde os tons mais escuros indicam valores menores, enquanto os tons mais brilhantes correspondem a valores maiores (perto de 1). Essa escala é dada com a ajuda de uma barra colorida no lado direito do gráfico. Em resumo, a matriz de correlação é uma ferramenta útil para visualizar as relações entre variáveis em um conjunto de dados e pode ajudar a identificar padrões e tendências importantes.

**Figura 31:** Código para gerar Gráfico Matrix de Correlação da figura 30

```
# Correlação dos dados
plt.figure(figsize=(10,8))
ax = sns.heatmap(df_stroke.corr(), annot=True, linewidth=0.5, fmt='0.2f')
ax.set_title('Matriz de correlação dos atributos', fontsize=16, pad=20)
plt.show()
```

Fonte: Autor

#### 5.3.1.2. Classificando a Matriz de Correlação

Uma forma útil de interpretar a correlação entre variáveis é gerar um dataframe e classificar os valores, o que pode ajudar na análise de casos com vários atributos. Essa prática facilita visualmente a identificação de correlação entre as variáveis, e para isso, vamos usar o método *unstack()* no Pandas que retorna uma série com multiIndex, ou seja, cada valor da Série é representado por mais de um índice

A análise dos pares de correlação permite visualizar as relações entre as variáveis e compreender a influência que uma variável pode ter sobre a outra. Caso haja uma correlação positiva forte entre duas variáveis, o aumento em uma delas tende a levar a um aumento proporcional na outra. Por outro lado, quando a

correlação é negativa, um aumento em uma das variáveis tende a levar a uma diminuição na outra.

**Figura 32:** Classificando a Matriz de Correlação

Genero	Genero	1.000000
	Idade	0.027855
	Hipertensao	-0.021223
	Doença_cardiaca	-0.085685
	Ja_Casado	-0.030171
	...	
AVC	Tipo_de_residencia	-0.015458
	Nivel_medio_glicose	0.131945
	IMC	0.041458
	Condição_de_fumante	-0.066393
	AVC	1.000000
Length: 121, dtype: float64		

Fonte: Autor

**Figura 33:** Código para Classificar a Matriz de Correlação da figura 32

```
# Gerando um dataframe para Classificação de Matrix de correlação
mat_corr = df_stroke.corr(method='pearson')
corr_par = mat_corr.unstack()
corr_par
```

Fonte: Autor

O resultado abaixo mostra que o conjunto de dados possui uma matriz de correlação simétrica, com cada valor sendo repetido duas vezes na saída classificada. Isso ocorre porque cada par de características aparece duas vezes na matriz de correlação. A classificação dos coeficientes de correlação agora permite tomar decisões baseadas nas relações entre as variáveis, identificando quais possuem forte correlação positiva ou negativa, e quais possuem correlação fraca ou nenhuma. Para classificar esses valores, foi utilizado a função a `sort_values()`.

**Figura 34:** Ordenando valores da Classificando a Matriz de Correlação

Ja_Casado	Idade	-0.679181
Idade	Ja_Casado	-0.679181
Tipo_de_trabalho	Idade	-0.413852
Idade	Tipo_de_trabalho	-0.413852
	Condição_de_fumante	-0.376142
	...	
Doença_cardiaca	Doença_cardiaca	1.000000
Hipertensao	Hipertensao	1.000000
Idade	Idade	1.000000
Condição_de_fumante	Condição_de_fumante	1.000000
AVC	AVC	1.000000
Length: 121, dtype: float64		

Fonte: Autor

**Figura 35:** Código para Ordenar a Matriz de Correlação da figura 34

```
# Ordenando valores da Matrix de correlação
par_sorted = corr_par.sort_values(kind="quicksort")
print(par_sorted)
```

Fonte: Autor

### 5.3.1.3. Correlação Negativa

O valor negativo indica uma correlação inversa entre as variáveis, ou seja, quando uma variável aumenta, a outra tende a diminuir. Podem ser útil para entender a relação entre variáveis e possivelmente ajudar na identificação de padrões ou insights relevantes para nossos modelos.

**Figura 36:** Correlação com valores Negativos

Ja_Casado	Idade	-0.679181
Idade	Ja_Casado	-0.679181
Tipo_de_trabalho	Idade	-0.413852
Idade	Tipo_de_trabalho	-0.413852
	Condição_de_fumante	-0.376142
	...	
Tipo_de_residencia	Genero	-0.006105
	Condição_de_fumante	-0.004656
Condição_de_fumante	Tipo_de_residencia	-0.004656
Doença_cardiaca	Tipo_de_residencia	-0.003092
Tipo_de_residencia	Doença_cardiaca	-0.003092
Length: 62, dtype: float64		

Fonte: Autor

**Figura 37:** Código Correlação com valores Negativos da figura 36

```
# Imprimindo somente correlações negativas
par_corr_negativos = par_sorted[par_sorted < 0]
print(par_corr_negativos)
```

Fonte: Autor

#### 5.3.1.4. Seleção de Correlação Forte

Nesta seção vamos abordar os recursos fortemente relacionados, ou seja, tentaremos filtrar os pares de recursos cujos valores de coeficiente de correlação são maiores que 0,5 ou menores que -0,5.

Na saída apresentada na figura 38, apenas a correlação entre as variáveis "Ja\_Casado" e "Idade" apresenta um valor de correlação forte negativa (-0.679181), enquanto todas as outras apresentam uma correlação forte positiva (1.0), o que indica uma correlação perfeita entre essas variáveis. Ou seja, todas as outras variáveis são altamente correlacionadas com elas mesmas, o que não é muito informativo para a análise.

**Figura 38:** Seleccionando pares da correlação fortes

Ja_Casado	Idade	-0.679181
Idade	Ja_Casado	-0.679181
Genero	Genero	1.000000
Tipo_de_trabalho	Tipo_de_trabalho	1.000000
IMC	IMC	1.000000
Nivel_medio_glicose	Nivel_medio_glicose	1.000000
Tipo_de_residencia	Tipo_de_residencia	1.000000
Ja_Casado	Ja_Casado	1.000000
Doença_cardiaca	Doença_cardiaca	1.000000
Hipertensao	Hipertensao	1.000000
Idade	Idade	1.000000
Condição_de_fumante	Condição_de_fumante	1.000000
AVC	AVC	1.000000
dtype: float64		

Fonte: Autor

**Figura 39:** Código Correlação variáveis "Ja\_Casado" e "Idade" da figura 38

```
par_corr_forte = par_sorted[abs(par_sorted) > 0.5]
print(par_corr_forte)
```

Fonte: Autor

Por fim, gerou-se um dataframe de correlação entre AVC e todas as outras variáveis, mas constatou-se uma correlação perfeita entre a variável "AVC" e ela mesma. Essa ocorrência é comum, uma vez que uma variável tem correlação perfeita consigo mesma. Entretanto, essa correlação não possibilita a identificação de relações entre "AVC" e as demais variáveis do conjunto de dados. Essa análise é relevante para a etapa seguinte do projeto, que é a criação de modelos de aprendizado de máquina. Para aprimorar o conjunto de dados e melhorar o desempenho dos modelos, na próxima sessão serão apresentadas algumas técnicas que podem contribuir para a melhoria da precisão do modelo de predição de acidente vascular cerebral com base em fatores de risco.

**Figura 40:** Correlação entre AVC e demais variáveis

```

avc_corr_forte = mat_corr["AVC"][abs(mat_corr["AVC"]) > 0.5]
print(avc_corr_forte)

```

AVC 1.0  
Name: AVC, dtype: float64

Fonte: Autor

### 5.3.2. Assimetria dos atributos

A avaliação da assimetria de atributos é uma etapa importante na análise de dados, pois uma distribuição assimétrica pode levar a uma interpretação incorreta dos resultados.

Na exploração apresentada foi constatado que alguns atributos apresentaram valores de coeficiente de assimetria maiores do que 1, como o Nível médio de glicose (1.57), Hipertensão (2.72), Doença cardíaca (3.95) e AVC (4.19). Isso indica que essas variáveis têm uma cauda mais longa à direita, o que sugere que a maioria dos indivíduos tem valores mais baixos nesses atributos, mas há uma minoria que possui valores muito mais altos, resultando em uma assimetria positiva. Essa informação é importante para modelagem, pois devemos estar cientes dessas distribuições e aplicar técnicas que considerem essa assimetria, como transformação de variáveis ou uso de modelos específicos. Já outras variáveis

apresentaram coeficientes de assimetria menores, como o Gênero (-0.35) e a Idade (-0.14), indicando uma assimetria negativa. Em geral, a avaliação da assimetria é importante para garantir a qualidade da análise estatística e aumentar a precisão dos modelos preditivos.

**Figura 41:** Verificando assimetria dos atributos

Genero	-0.349410
Idade	-0.140425
Tipo_de_residencia	0.032107
Condição_de_fumante	0.077863
Ja_Casado	0.657745
Tipo_de_trabalho	0.972287
IMC	1.048490
Nivel_medio_glicose	1.572284
Hipertensao	2.715392
Doença_cardiaca	3.947244
AVC	4.193284
dtype:	float64

Fonte: Autor

**Figura 42:** Código - Verificando assimetria dos atributos, figura 41.

```
# Assimetria e ordenação dos resultados
df_stroke.skew().sort_values(kind="quicksort")
```

Fonte: Autor

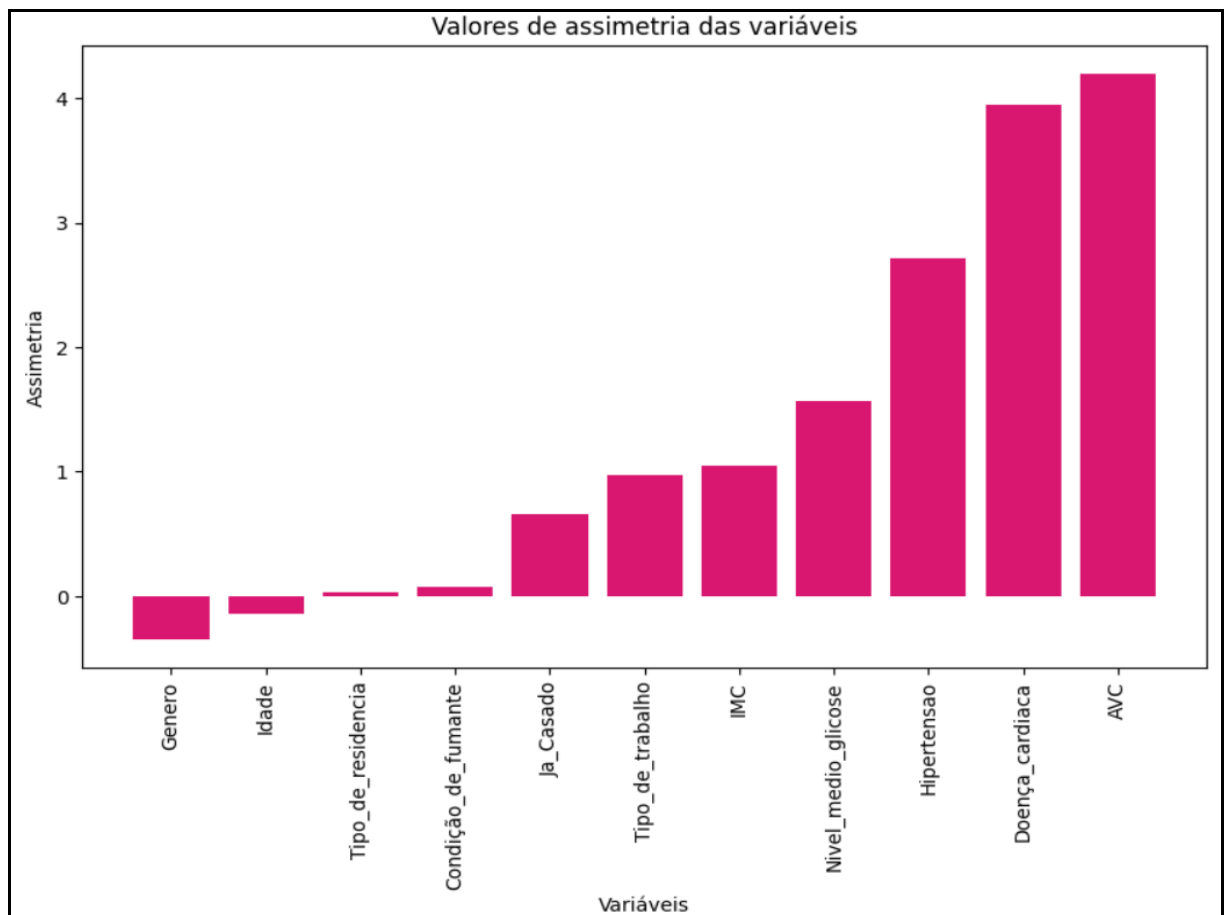
É importante lembrar que a presença de assimetria nos dados pode ser influenciada pela presença de variáveis binárias, uma vez que essas variáveis podem assumir apenas dois valores, e isso pode criar um viés na distribuição dos dados. Por exemplo, na análise dos dados do AVC, a presença da variável binária "AVC" pode ter impactado no alto valor de assimetria observado. Portanto, ao avaliar a assimetria em um conjunto de dados que contém variáveis binárias, é importante ter em mente que a presença dessas variáveis pode distorcer a distribuição dos dados e, portanto, é necessário avaliar a assimetria de maneira cuidadosa,



considerando a natureza das variáveis envolvidas. É necessário tomar cuidado ao interpretar a assimetria em atributos que contêm variáveis binárias, pois a distribuição dos dados pode não ser representativa da realidade, podendo levar a decisões equivocadas. Por isso, é recomendado o uso de técnicas que levem em consideração a natureza dessas variáveis e possíveis distorções que elas possam causar.

### 5.3.2.1. Representação Gráfica da Assimetria dos Atributos

**Figura 43:** Representação Gráfica da Assimetria dos Atributos



Fonte: Autor

**Figura 41:** Código - Gráfico para Assimetria dos Atributos, figura 40.

```
# Calcular os valores de assimetria das variáveis
skewness = df_stroke.skew().sort_values(kind="quicksort")

# Criar um gráfico de barras verticais
fig, ax = plt.subplots(figsize=(10,6))
ax.bar(skewness.index, skewness.values)

# Adicionar rótulos e título
ax.set_xlabel('Variáveis')
ax.set_ylabel('Assimetria')
ax.set_title('Valores de assimetria das variáveis')
plt.xticks(rotation=90)

# Mostrar o gráfico
plt.show()
```

Fonte: Autor

## **6. Preparação dos Dados para os Modelos de Aprendizado de Máquina**

Nesta etapa você deve descrever os tratamentos realizados especificamente para o(s) modelo(s) de Aprendizado de Máquina escolhido(s), como por exemplo a criação de atributos, o balanceamento da base de dados (*undersampling* ou *oversampling*), divisão da base em treino, validação e teste, entre outros.

## **7. Aplicação de Modelos de Aprendizado de Máquina**

Nesta seção você deve apresentar o(s) modelo(s) de Aprendizado de Máquina desenvolvido(s) no trabalho. Mostre partes do código-fonte para ilustrar a implementação de cada modelo. A escolha do(s) modelo(s) deve ser adequada e justificada ao problema proposto. Embora possa ser considerado o uso de ferramentas como Weka, Knime e Orange, por exemplo, encoraja-se a implementação com linguagens como Python ou R. Não é obrigatório, mas sugere-se testar mais de um tipo de algoritmo, para que resultados distintos possam ser comparados. Por exemplo, se o trabalho trata de uma classificação, modelos como Árvores de Decisão, Redes Neurais Artificiais e Support Vector Machine poderiam ser utilizados. Além disso, devem ser escolhidas e implementadas as métricas adequadas ao problema proposto, bem como os seus resultados apresentados.

## **8. Avaliação dos Modelos de Aprendizado de Máquina e Discussão dos Resultados**

Nesta seção você deve relatar os resultados alcançados ao final do trabalho. Mostre os resultados das métricas adotadas, seja através de gráficos, tabelas, dentre outros, que permitam a validação do seu trabalho.

## **9. Conclusão**

Nesta seção você deve apresentar um fechamento para o trabalho. É importante apresentar um breve resumo do trabalho, resgatando o problema, como foi tratado e os resultados obtidos, bem como as limitações e perspectivas (trabalhos futuros).

## 10. Links

Todos os códigos gerados e a documentação desenvolvida são disponibilizados no repositório do GitHub.

Link GitHub: [https://github.com/LucianobSilva/TCC\\_PUC\\_Minas\\_IA\\_AM](https://github.com/LucianobSilva/TCC_PUC_Minas_IA_AM)

## 11. Referências

WHO. **AVC, Acidente Vascular Encefálico** Disponível em <https://www.emro.who.int/health-topics/stroke-cerebrovascular-accident/index.html>. Acesso em: 03/02/2023.

PIEPER, Brett Slatkin. **Python Eficaz: 59 maneiras de escrever melhores códigos em Python**. São Paulo: Novatec, 2016.

MORETTIN, Pedro A.; SINGER, Julio M. **Estatística e Ciência de Dados**. 1ª ed. Rio de Janeiro: LTC, 2022.

GERON, Aurélien. **Mãos à Obra Aprendizado de Máquina com Scikit-Learn, Keras e TensorFlow: Conceitos, ferramentas e técnicas para a construção de sistemas inteligentes**. 2ª ed. São Paulo: Novatec, 2020.