Lista de abreviaturas

AVC	Acidente Vascular Cerebral
TC	Tomografia computadorizada
CNN	Rede Neural Convolucional
FN	Falso negativo
TN	Verdadeiro negativo
FP	Falso positivo
TP	Verdadeiro positivo
FPR	Taxa de falso positivo
TPR	Taxa de verdadeiro positivo
ROC	Característica de Operação do Receptor
AUC	Área abaixo da curva

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Acidente vascular cerebral – AVC

2.1.1 Fatores de risco da doença

2.1.2 Diagnóstico com tomografias computadorizadas (TC)

2.2 Aprendizado de Máquina

2.2.1 Aprendizado Supervisionado

2.2.1.1 Regressão Logística

2.2.1.2 Floresta Aleatória

2.3 Aprendizagem profunda

2.3.1 Neurônios artificiais

2.3.2 Redes Neurais Artificiais

2.3.3 Rede Neural Convolucional

2.5 Métricas para avaliação de desempenho

2.5.1 Acurácia

2.5.2 Tipos de erro

2.5.3 Matriz de confusão

2.5.4 Taxa de FP, taxa de TP e Curva ROC

3.METODOLOGIA

3.1 Base de dados para os fatores de risco

3.1.1 Variáveis categóricas Binárias: dois possíveis valores apenas

3.1.2 Variáveis categóricas não binárias: vários possíveis valores

3.1.3 Variáveis de valor contínuo

3.1.4 Variável ordinal

3.2 Base de dados para as imagens de tomografia computadorizada

3.3 Ferramentas utilizadas

3.3.1 Python

3.3.2 Bibliotecas da linguagem utilizada

2.1 Acidente vascular cerebral – AVC

O Acidente Vascular Cerebral (AVC) está entre as doenças de maior destaque que existem, sendo uma das principais causas de morte, incapacitação física e internação em todo o mundo. Tal doença é caracterizada pela alteração do fluxo sanguíneo que ocorre na região cerebral, impedindo que o sangue, e consequentemente o oxigênio, chegue às células da região, causando a morte das mesmas. A morte dessas células danifica o funcionamento do cérebro e pode deixar graves sequelas nas vítimas da doença.

A alteração do fluxo de sangue do cérebro pode se originar da obstrução total ou parcial de alguma artéria da região, o que é conhecido como acidente vascular cerebral isquêmico, ou simplesmente infarto cerebral. Esse tipo de AVC é o mais comum entre os casos da doença e pode ocorrer devido a um trombo (caso de trombose) ou êmbolo (caso de embolia) presente na vítima.

Uma outra origem da alteração do fluxo de sangue mencionada é o caso do rompimento de determinado vaso sanguíneo do cérebro, o que causa uma hemorragia na região. Esse caso caracteriza o acidente vascular cerebral hemorrágico. Tal rompimento de determinado vaso altera o nível da pressão intracraniana e pode dificultar também a chegada de sangue em outras áreas não afetadas. Embora seja menos comum, esse tipo de AVC é o mais grave, tendo maiores índices de mortalidade quando comparado ao outro tipo abordado (AVC isquêmico).

Com relação ao sintomas presentes nas vítimas, dor de cabeça forte sem causa aparente; fraqueza ou formigamento da face, braço ou perna (especialmente em apenas um dos lados do corpo); alteração da fala e/ou da compreensão; alteração da visão de um ou ambos os olhos; e dificuldade, ou incapacidade, de se movimentar são sintomas bastante comuns em ambos os tipos da doença.

Em específico para o caso de acidente vascular cerebral isquêmico, tem-se de forma comum os sintomas de tontura e perda de equilíbrio ou coordenação. Já para o caso de AVC hemorrágico pode ocorrer também náusea, vômito, confusão mental e, em alguns casos, perda de consciência. Em tal caso também podem aparecer os sintomas de sonolência exagerada, alterações na frequência cardíaca e respiratória, e até mesmo convulsões. Vale ressaltar que é de grande importância que um indivíduo com os sintomas apresentados procure assistência médica o mais rápido possível. Dessa forma, a doença pode ser diagnosticada e tratada rapidamente, possivelmente resultando em maiores chances de sobrevivência da vítima e menores sequelas.

2.1.1 Fatores de risco da doença

Existem alguns fatores que aumentam a probabilidade da ocorrência do acidente vascular cerebral, facilitando o desencadeamento da doença. Estes são denominados fatores de risco e podem ser inerentes à vida humana, maus hábitos, estilo de vida inadequado ou até mesmo questões genéticas. Os principais fatores são:

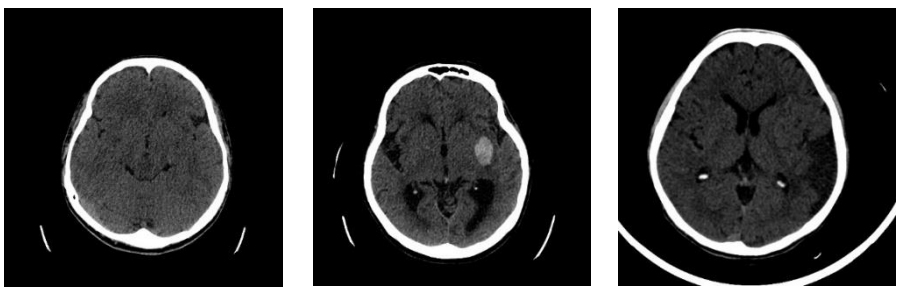
- Hipertensão
- Diabetes
- Obesidade
- Tabagismo
- Consumo excessivo e frequente de álcool e drogas
- Estresse
- Idade avançada (envelhecimento)
- Histórico familiar
- Sexo masculino
- Colesterol elevado
- Doenças cardiovasculares (principalmente as que produzem arritmia cardíaca)
- Sedentarismo
- Doenças do sangue

2.1.2 Diagnóstico com tomografias computadorizadas (TC)

Além da análise de fatores de risco e sintomas, outra forma de identificar o quadro de acidente vascular cerebral em vítimas da doença é através da análise de tomografias computadorizadas. Os quadros de AVC podem apresentar características aparentes quando analisados em tais tomografias, permitindo a identificação da doença e o nível de ocorrência da mesma em determinado cérebro.

Áreas hipodensas (mais escuras) que impedem a visualização de determinada região possivelmente indicando perda de tecido total ou parcial naquele local, por exemplo, é uma característica comum nos quadros da doença. A tomografia computadorizada vem sendo utilizada como principal forma de diagnosticar quadros de AVC para definir um possível tratamento. As figuras abaixo são exemplos de cada caso utilizado neste trabalho, sendo uma para caso de AVC hemorrágico, uma para isquêmico e uma para caso de normalidade (sem presença da doença):

Figura 01 – Exemplo de imagens de tomografia computadorizada para tipos diferentes de acidente vascular cerebral.



Condição normal AVC Hemorrágico AVC Isquêmico

Fonte:

2.2 Aprendizado de Máquina

O Aprendizado de Máquina, ou Aprendizagem de Máquina, é uma área da inteligência artificial que lida com o a criação de programas “inteligentes”, capazes de aprender determinados conceitos sem serem explicitamente programados para isso, capazes de adquirir conhecimento, de aprender e tomar decisões a partir desse aprendizado. Tal área explora também a capacidade de programas e sistemas conseguirem realizar previsões sobre determinadas situações, aprendendo através da análise de dados já existentes sobre tais situações.

Envolvendo conceitos estatísticos, reconhecimento de padrões e programação, a área de Aprendizado de Máquina possibilita que resultados sejam especulados de acordo com acontecimentos passados semelhantes, através de um processamento dos dados usando o reconhecimento de padrões. **Um sistema de aprendizado é um programa de computador que toma decisões baseadas em experiências acumuladas através da solução bem sucedida de problemas anteriores [1].**

Com relação à criação de uma solução utilizando conceitos de Aprendizado de Máquina, o processo geralmente envolve duas etapas, o treinamento e a avaliação. A etapa de treinamento é quando um determinado conjunto de dados é passado para o modelo processá-lo, aprendendo a reconhecer padrões e também a fazer previsões. A etapa de avaliação consiste em passar dados semelhantes ao determinado conjunto de dados utilizado no treino para que o desempenho do modelo seja medido, analisando os resultados esperados com os resultados obtidos.

No campo do Aprendizado de Máquina, existem três tipos diferentes de aprendizagem: aprendizado supervisionado, aprendizado não supervisionado e aprendizado por reforço. Os algoritmos de Aprendizado de Máquina utilizados neste trabalho pertencem à área de aprendizado supervisionado e, portanto, apenas esta será abordada com mais detalhes.

2.2.1 Aprendizado Supervisionado

O Aprendizado supervisionado utiliza dados rotulados no treinamento do modelo, o qual prevê uma variável dependente a partir de uma ou mais variáveis independentes. Existem diversos algoritmos desse tipo de aprendizado, sendo os principais: regressão linear, regressão logística, máquina de suporte vetorial, árvores de decisão, k-vizinhos mais próximos, floresta aleatória, entre outros. Os utilizados neste trabalho para desenvolver o classificador de fatores de risco foram apenas a Regressão Logística e o Classificador de Floresta Aleatória.

2.2.1.1 Regressão Logística

A Regressão Logística é um modelo estatístico capaz de modelar determinada probabilidade de um evento acontecer a partir da combinação linear de variáveis independentes entre si associadas a tal evento. Muito utilizado em situações de classificação binária, esse modelo combina variáveis independentes com respectivos pesos atribuídos a elas de modo a obter um valor a partir deste processo. Tal valor é passado para uma função logística que resulta na probabilidade de algum evento acontecer. No caso de uma classificação, tal resultado indica a probabilidade dos valores analisados pertencerem a determinada classe do problema em questão.

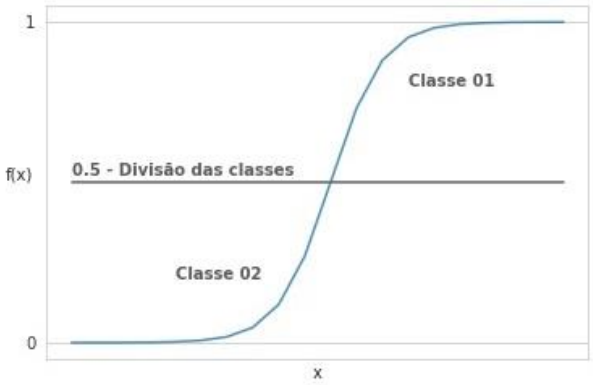
De forma simplificada, o algoritmo de Regressão Logística – especificamente para casos binários (utilizado neste trabalho) – ajusta uma curva, no caso usando uma função logística, que separa de forma eficiente os dados em duas classes distintas. Tal curva ajustada é obtida, de forma simplificada, através da junção das seguintes equações:

$$z = \sum_{i=0}^{n} x_i * \theta_i \text{ (equação I)}$$

$$\hat{y} = \frac{1}{1+e^{-z}} \text{ (equação II)}$$

Cada variável independente passada para o modelo é interpretada como x_i e cada peso respectivamente associado é interpretado como θ_i (i varia de 0 a n, sendo este o número total de variáveis passadas para o modelo). Através da combinação linear dos pontos $x * \theta$, é obtido o valor z (equação I) e este é passado para uma função logística (equação II), tendo como resultado disso um valor (comumente determinado de $\hat{y}$) entre 0 e 1. De forma padrão para casos binários, se $\hat{y}$ for maior ou igual a 0.5, então pode-se concluir que a entrada analisada pertence à classe 01, senão, tem-se o caso sendo pertencente a outra classe da classificação binária (por exemplo, classe 02).

Figura 02 – Gráfico de uma função logística f(x) sendo utilizada para divisão de duas classes a partir do valor 0.5 (limiar entre as classes)



Fonte: autoria própria

Cada peso da equação é identificado como θ_i e é gerado, inicialmente, de forma aleatória, sendo ajustado durante a fase de treinamento (ajuste) do modelo. Tal fase ocorre de forma iterativa e utiliza um conjunto de valores de dimensão m x n, em que m é o

número total de registros diferentes do conjunto de dados e n é o número total de variáveis de cada registro. Para cada iteração j (j varia de 0 a m), os valores θ são alterados para atingirem valores mais otimizados (que geram menores erros) tanto para casos de classe 01 quanto para casos de classe 02.

Erros extremamente baixos não necessariamente compõem um modelo eficiente. A medida que um classificador é composto por pesos muito ajustados (com erro muito baixo), ele fica exageradamente ajustado e, assim, com baixa capacidade de generalização. Esse caso é conhecido como sobreajuste e torna o modelo muito eficiente para classificar dados muito semelhantes com os que foi treinado, porém classificando de forma equivocada dados não semelhantes, o que se tem geralmente em uma situação real.

2.2.1.2 Floresta Aleatória

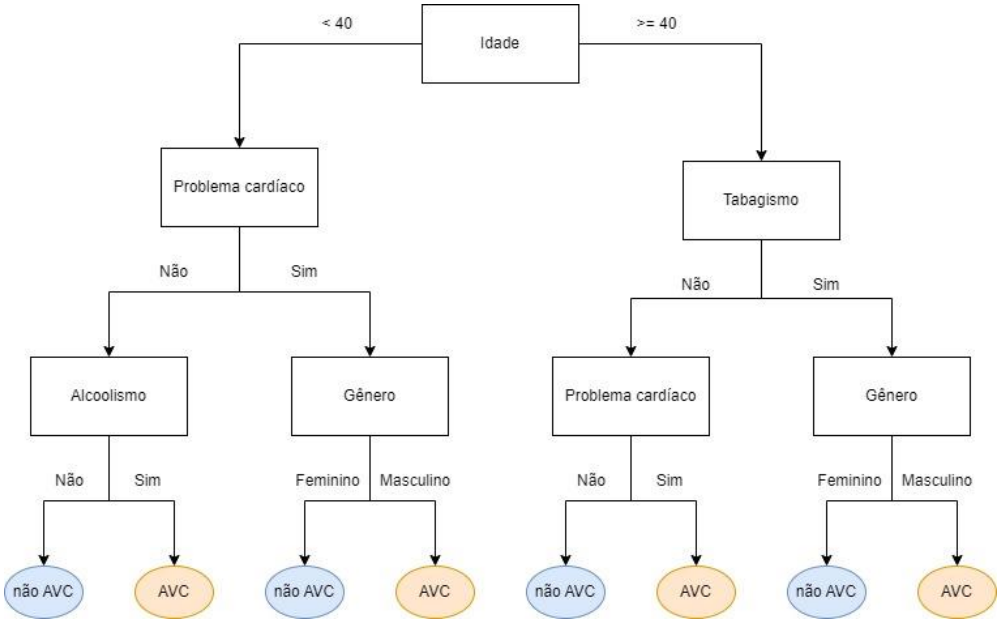
Floresta Aleatória é um modelo de aprendizado de máquina composto por um conjunto aleatório de Árvores de Decisão que pode ser usado em situações de regressão ou classificação, por exemplo. Especificamente para problemas de classificação, como o abordado neste trabalho, o modelo classifica de acordo com a classificação feita pelas árvores que o compõem. Dessa forma, se há um resultado classificativo mais frequente, isto é, se a maioria das árvores gerou determinada classificação, então esta será a que o modelo de floresta aleatória irá resultar em sua conclusão.

A estrutura de uma Árvore de Decisão é constituída por vários blocos de decisão comumente denominados de nós. Cada nó funciona como um bloco condicional que encaminha a entrada recebida para um de seus filhos de acordo com o valor o qual ele analisa. Tal processo é feito para todo nó pertencente à árvore até chegar em determinado nó folha que, por sua vez, representa uma das possíveis classes do problema. Portanto, em uma árvore de decisão, uma entrada percorre um dos diferentes caminhos possíveis até chegar em um nó folha que a classifica como uma das possíveis rotulações da situação. Tal caminho acabada sendo definido de acordo com os valores que pertencem a entrada em questão. A figura 03, localizada mais adiante, representa um exemplo que ilustra uma possível estrutura de uma Árvore de Decisão.

Para cada tipo de situação abordado, isto é, para cada problema e conjunto de dados, os caminhos, os nós e os critérios de divisão de cada nó mudam. A etapa de treinamento do modelo irá analisar quais divisões são mais eficientes, utilizando critérios como entropia, por exemplo, e assim definirá os melhores caminhos a serem construídos. Portanto, existem infinitas possibilidades de estruturas para árvores de decisão, sendo cada uma delas definidas de acordo com o conjunto de dados utilizado na etapa de treino do modelo.

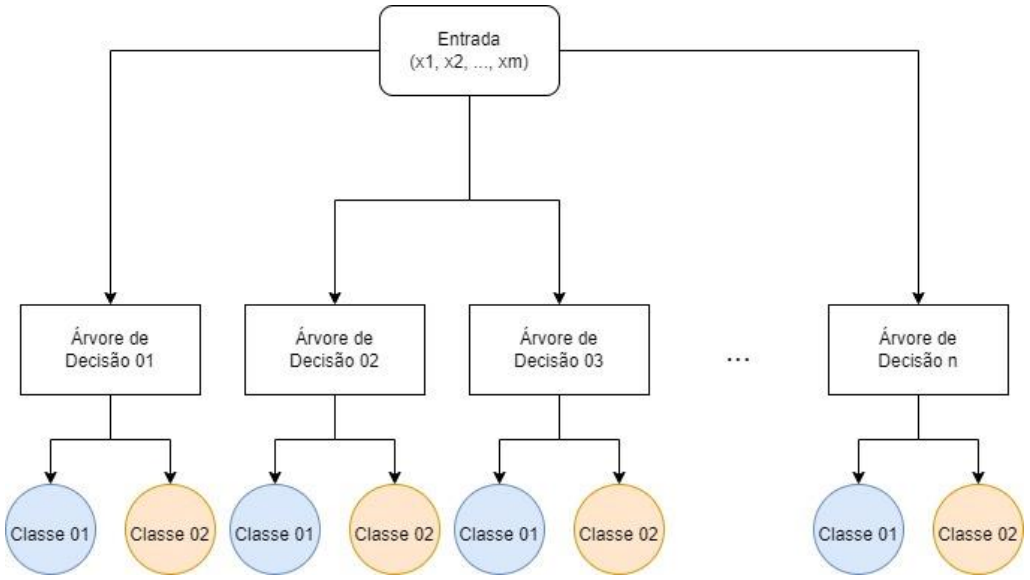
A ideia do algoritmo de Floresta Aleatória é justamente utilizar dessas diversas possibilidades de estrutura das Árvores de Decisão e criar assim uma floresta com várias árvores diferentes abordando o mesmo problema, classificando determinada entrada recebida de acordo com a classificação mais frequente das árvores que compõem o modelo.

Figura 03 – Exemplo de estrutura de uma Árvore de Decisão para um conjunto de dados que contenha as variáveis *Idade*, *Problema Cardíaco*, *Alcoolismo*, *Gênero* e *Tabagismo*, com as possíveis rotulações de classe *não AVC* e *AVC*.



Fonte: autoria própria

Figura 04 – Estrutura de um modelo de Floresta Aleatória para m variáveis, contendo n Árvores de Decisão com duas possibilidades de classificação: Classe 01 e Classe 02.



Fonte: autoria própria

2.3 Aprendizagem profunda

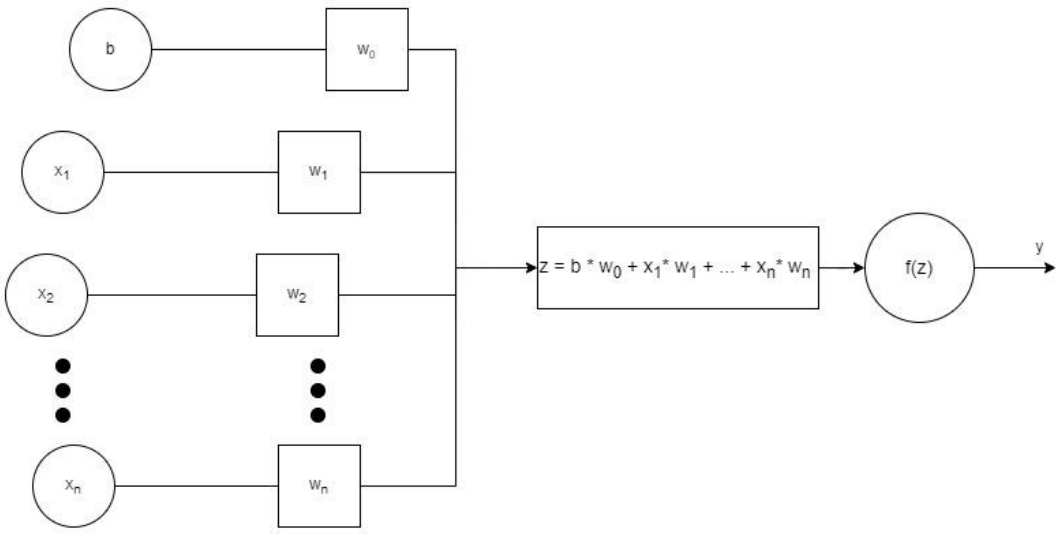
2.3.1 Neurônios artificiais

O neurônio artificial é o principal componente das redes neurais computacionais e seu funcionamento é baseado nos neurônios reais, que compõem o sistema nervoso. Os neurônios reais, base do sistema nervoso, são células que estabelecem conexões entre si para transmitir impulsos nervosos pela região cerebral. O neurônio artificial é uma estrutura com funcionamento que se assemelha do neurônio real, mantendo a principal característica dessas estruturas de criarem conexões entre si, recebendo informações provenientes de outros neurônios e passando elas adiante.

Os principais componentes de um neurônio artificial são as entradas, os pesos, a função de ativação e saída. As entradas, geralmente representadas pela letra *x*, recebem a informação que chega para o neurônio. Tal informação é multiplicada por um determinado peso, comumente representado pela letra *w*, e o resultado disso é passado para uma função de ativação que processa os valores recebidos e retorna um ou mais valores, os quais servirão de entrada para um outro neurônio artificial conectado à rede.

Outro componente muito comum nos neurônios das redes neurais artificiais é o viés (bias). Tal valor é processado junto com os dados *x*w* na função de ativação como tentativa de tornar o neurônio não muito tendencioso, isto é, não muito apegado aos dados que recebe. Isso é de grande utilidade para que a rede neural possa ficar não enviesada, tendo um melhor desempenho com relação a resultados corretos quando em contato com dados não conhecidos, graças à capacidade de generalização obtida nesse processo.

Figura 05 – Ilustração de uma possível estrutura para um neurônio artificial



Fonte: autoria própria

2.3.2 Redes Neurais Artificiais

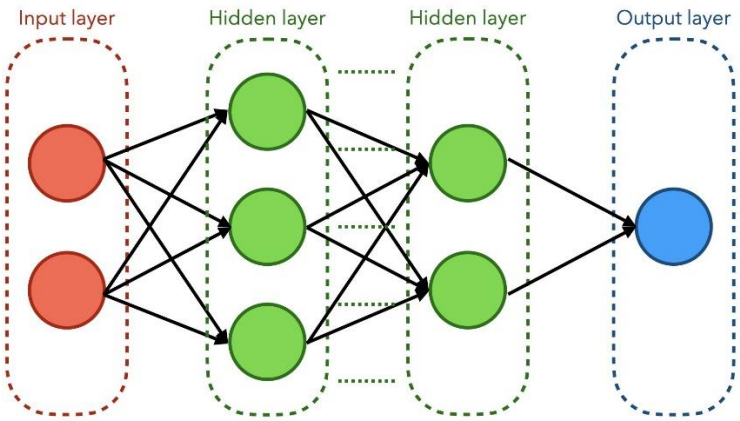
Redes neurais artificiais são estruturas do campo da inteligência artificial que possibilitam os computadores a processar dados se baseando no comportamento do cérebro humano. Ao receber um conjunto de dados específico, uma rede neural artificial é capaz de se ajustar a este conjunto, aprendendo a reconhecer padrões sobre estes dados e se tornando mais eficiente à medida que os processa. A estrutura da rede é composta por um conjunto de neurônios artificiais agrupados em camadas interconectadas que possibilitam o computador a aprender, reconhecer padrões e tomar decisões inteligentes.

Uma rede neural simples possui camadas de três tipos: camada de entrada, camada oculta e camada de saída. A camada de entrada é onde se encontram os neurônios responsáveis por receber os dados que serão processados pela rede. Tal camada faz o primeiro processamento desses dados e os encaminham para o próximo conjunto de neurônios. As camadas do tipo oculta recebem dados da camada de entrada ou de outras camadas ocultas, processando novamente os dados e passando o resultado para outra camada. Redes neurais podem ter uma ou mais camadas desse tipo.

Por fim, o conjunto de neurônios da camada de saída faz o último processamento dos dados, fornecendo o resultado final de todos os dados processados pela rede. Para problemas de classificação binária, tem-se apenas um nó na camada de saída, o qual indica a probabilidade de determinada entrada analisada pela rede ser ou não da classe abordada. Já para problemas de várias classes diferentes, tem-se vários nós nessa camada.

Além dessa estrutura de rede abordada, existem outros tipos mais específicos de redes neurais que apresentam melhores resultados quando aplicados em determinadas situações. No caso de problemas que envolvem a classificação de imagens e o reconhecimento de objetos, o uso de Redes Neurais Convolucionais é mais recorrente do que o uso da rede neural padrão, justamente pela boa capacidade que estas possuem de valorizar características da imagem.

Figura 06 – Ilustração simplificada de uma Rede Neural Artificial simples



Fonte: An Illustrated Guide to Artificial Neural Networks | by Fahmi Nurfikri | Towards Data Science

2.3.3 Redes Neurais Convolucionais

Comumente chamadas de Conv-Net ou simplesmente CNN, as redes neurais do tipo convolucional são muito utilizadas em tarefas que envolvem a identificação de objetos, reconhecimento de características e classificação de imagens. Esse tipo de rede tem a capacidade de valorizar detalhes da imagem que são relevantes para a classificação, realçando determinados agrupamentos de pixels através da aplicação de filtros matriciais na imagem, o que colabora para um processo de classificação mais eficiente.

As Redes Neurais Convolucionais apresentam três tipos principais de camadas em sua estrutura: camadas de convolução, camadas de agrupamento (*pooling*) e as camadas totalmente conectadas. Camadas de convolução, ou convolucionais, são as camadas responsáveis pelo processo de realce das características mencionado no parágrafo anterior.

Esse processo se dá pela aplicação de um filtro, comumente chamado de kernel, na imagem, vista como uma matriz de pixels, que está sendo processada pela rede, funcionando como um detector de recursos. Esse filtro tem formato matricial com tamanho típico de 3x3, podendo ser maior dependendo da rede, e o processo de aplicação se dá de forma iterativa via a realização do produto escalar entre seus valores e os da matriz da imagem processada, finalizando-se com a soma dos resultados dessa multiplicação.

Tal processo é iterativo, acontecendo até que o kernel multiplique toda a imagem. Alguns hiperparâmetros que podem variar o volume da imagem resultante são: o número de filtros que serão aplicados a imagem, o número de passos denominado *stride* que o kernel percorre para se deslocar de uma multiplicação para a outra e o *padding*, que adiciona zeros na matriz da imagem para chegar a determinado tamanho que favoreça a aplicação do kernel.

As camadas de agrupamento irão pegar o resultado da camada de convolução e reduzir o número de valores recebidos, mantendo as características realçadas pelos filtros da convolução, porém eliminando áreas vazias de informação relevante para a classificação. O processo de *pooling* destas camadas se dá por dois tipos diferentes que podem ser utilizados, o agrupamento máximo ou o agrupamento médio.

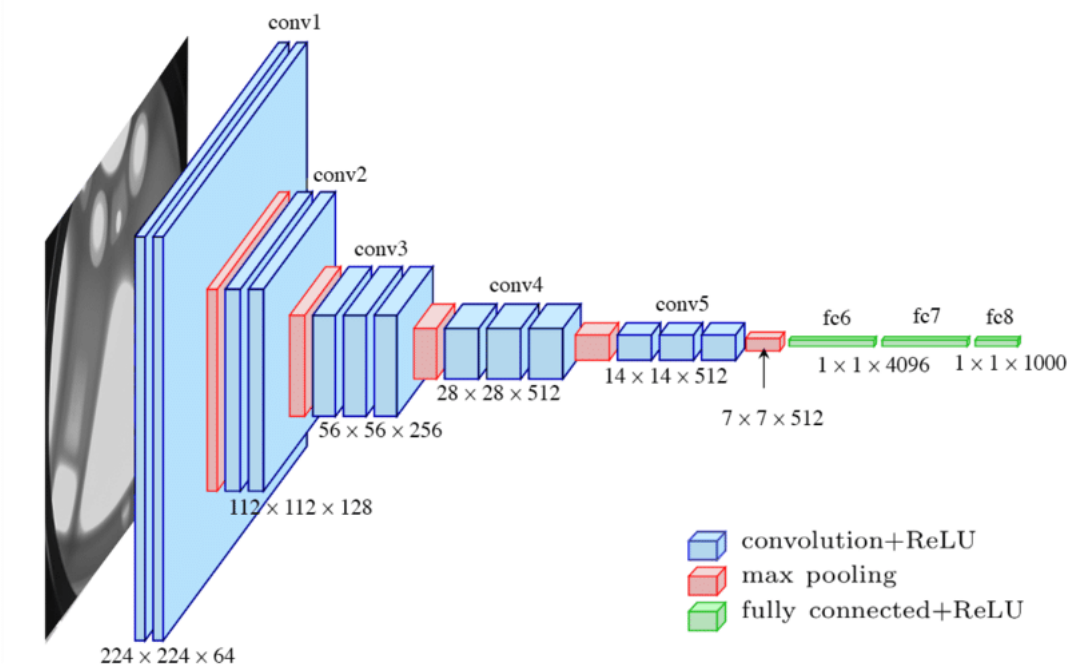
O agrupamento máximo é feito através da aplicação de um filtro que se desloca pela imagem que simplesmente pega o maior valor dentre os valores analisados na imagem e o coloca em uma matriz de saída, descartando os demais valores. O agrupamento médio se dá pela aplicação de um filtro que também se move pela imagem analisada, porém calculando o valor médio dos valores analisados pelo filtro, colocando-o em uma matriz de saída de forma semelhante ao outro agrupamento descrito. O agrupamento máximo tem maior frequência de utilização quando comparado com o agrupamento médio.

Por fim, nas camadas totalmente conectadas tem-se diversos neurônios totalmente conectados que irão processar os valores recebidos aplicando alguma função de ativação, gerando valores de saída para as próximas camadas ou para a camada de saída. Toda rede neural convolucional tem seu formato base composto por essas camadas, porém existem

diversos tipos de CNN diferentes que vão variar o número de instâncias dessas camadas mencionadas e os hiperparâmetros presentes nas mesmas.

A arquitetura de rede convolucional denominada de VGG-16, utilizada no desenvolvimento deste trabalho, apresenta treze camadas de convolução e cinco camadas de agrupamento, assim como três camadas totalmente conectadas. A imagem abaixo ilustra a distribuição dessas camadas, com as camadas de agrupamento entre grupos de camadas de convolução e as camadas totalmente conectadas localizadas no final da rede:

Figura 07 – Possível representação ilustrativa de uma rede da arquitetura de Rede Neural Convolucional VGG-16



Fonte: Fig. A1. The standard VGG-16 network architecture as proposed in [32].... | Download Scientific Diagram (researchgate.net)

2.5 Métricas para avaliação de desempenho

Com o modelo classificador ajustado de acordo com a situação abordada, torna-se fundamental a realização de uma boa avaliação de seu desempenho. **O objetivo da fase de avaliação é estimar os resultados de mineração de dados de forma rigorosa e obter a confiança de que são válidos e confiáveis antes de avançar [2].** Avaliar de forma adequada permite a identificação de problemas, como sobreajuste ou excesso de generalização, por exemplo, assim como visualizar de forma geral como seria a performance do modelo quando colocado em produção na prática.

Para realizar tal processo de avaliação nos modelos criados neste trabalho, foram definidas algumas métricas utilizadas na avaliação de modelos classificadores, as quais serão abordadas logo abaixo.

2.5.1 Acurácia

A acurácia é uma das métricas mais simples de se medir e justamente por isso sua utilização é muito popular na avaliação de classificadores. Porém, por ser demasiadamente simples, tal métrica pode esconder deficiências do modelo analisado e transparecer um resultado não muito confiável. O cálculo da acurácia é dado pela divisão do número de acertos do modelo pelo número total de casos analisados, ou seja, pelo número de acertos somados com o número de erros:

$$acurácia = \frac{acertos}{acertos + erros}$$

Como o cálculo dessa métrica não leva em conta fatores além de acertos e erros, como o tipo de erro, por exemplo, é bastante interessante utilizar a Acurácia em conjunto com outras métricas para fazer uma avalição mais eficiente, levando em conta também os tipos de valores falsos que o modelo gera.

2.5.2 Tipos de erro

Na fase de avaliação de um modelo também pode ser interessante olhar para o desempenho do modelo com relação a erros específicos. Analisar a taxa de falso positivo, erro do tipo I, e de falso negativo, erro do tipo II, permite visualizar melhor qual tipo de erro é predominante na questão, assim como ter uma ideia percentual probabilística de como o modelo em questão reagiria em situações reais com relação aos tipos de erros.

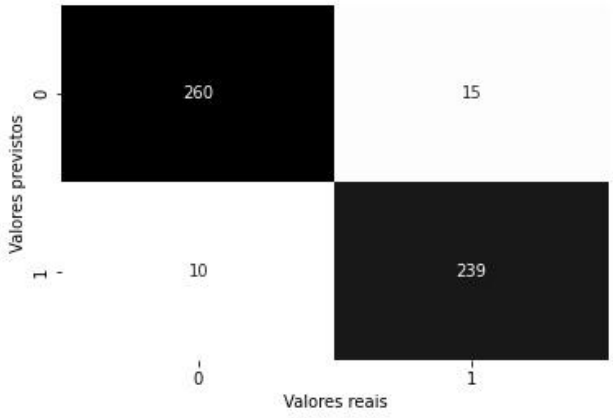
Além dos tipos de erros (falsos), tem-se também dois diferentes casos de acertos, sendo estes os verdadeiros positivos e verdadeiros negativos. É comumente utilizado para simplificar a nomenclatura dessas métricas as seguintes siglas: FN para falso negativo, TP para verdadeiro positivo, TN para verdadeiro negativo e FP para falso positivo.

2.5.3 Matriz de confusão

A matriz de confusão é uma métrica bastante utilizada na avaliação de classificadores pois ela fornece uma visão sobre a confusão entre as classes que um determinado classificar realizou na análise de casos passados para ele. Uma matriz de confusão é uma matriz de dimensão *n* x *n* sendo que *n* é o número de classes do problema abordado.

Uma matriz de confusão separa as decisões tomadas pelo classificador, tornando explícito como uma classe está sendo confundida com outra. Desta forma, diferentes tipos de erros podem ser tratados separadamente [2]. Na matriz de confusão temos dois tipos de acertos e erros, os verdadeiros positivos (TP) e negativos (TN), e os falsos positivos (FP) e negativos (FN), respectivamente. Valores verdadeiros ocorrem quando o modelo classifica corretamente valores negativos (0) ou positivos (1) e valores falsos ocorrem quando se tem a classificação incorreta desses mesmos valores, negativos ou positivos.

Figura 06 – Exemplo de matriz de confusão para um classificador binário em que as classes são rotuladas como 0 (negativo) e 1 (positivo). No eixo vertical tem-se a rotulação 0 ou 1 para os valores previstos pelo modelo e no eixo horizontal, a rotulação dos valores reais



Fonte: autoria própria

2.5.4 Taxa de FP, taxa de TP e Curva ROC

A taxa de FP, ou FPR, mede a fração dos valores negativos que foram previstos incorretamente, julgados como positivos pelo classificador em questão. Para realizar o cálculo dessa métrica, a seguinte equação é utilizada:

$$FPR = \frac{FP}{TN + FP}$$

A taxa de TP, ou TPR, mede a fração dos valores positivos que foram previstos de forma correta, julgados também como positivos pelo modelo. O cálculo da TPR é feito da seguinte forma:

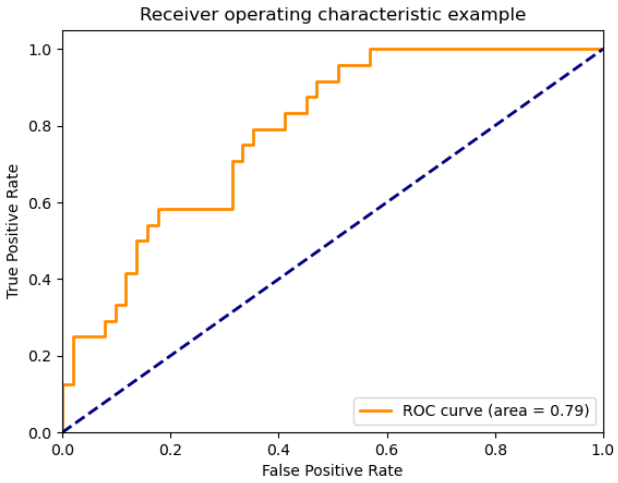
$$TPR = \frac{TP}{TP + FN}$$

É de grande importância que sejam analisadas ambas as métricas descritas acima em conjunto, pois assim torna-se possível ter um diagnóstico confiável e saber se o modelo está extremamente tendencioso ou não. Por exemplo, em uma situação em que determinado classificador rotula sempre todos os dados como sendo de uma classe positiva, sua TPR terá valor máximo (1.0), pois para todo valor positivo recebido, ele acertará todos. Porém, este mesmo modelo, como consequência, errará todos os casos negativos, e isso só poderá ser enxergado se a FPR for analisada, pois seu valor também será 1.0.

Nesse cenário descrito, analisar apenas a TPR indicaria um modelo muito eficiente, já que ele não errou nenhum caso positivo que recebeu. Dessa forma, estaria sendo omitido o alto viés e a baixa capacidade de generalização do classificador em questão. Um modelo tendencioso assim possivelmente apresentaria péssimos resultados na prática.

Para facilitar a análise em conjunto de ambas as métricas, existe uma outra métrica muito poderosa denominada de curva ROC (Característica de Operação do Receptor, em português). Ela é uma ferramenta gráfica que permite verificar de forma conjunta TPR e FPR a medida que o limiar entre as classes (probabilidade que separa as classes) do modelo varia. Tal variação é feita para diferentes valores do limiar que geram diferentes valores de TPR e FPR como resultado. Estes valores formam pontos em um gráfico TPR x FPR e a junção desses pontos formam a curva ROC.

Figura 07 – Visualização gráfica de uma curva ROC (em laranja) constituída por pontos obtidos a partir de diferentes valores para o limiar entre as classes de determinado modelo preditivo. É possível visualizar também no gráfico uma curva $x = y$ (em azul) de referência que indica pontos de TPR e FPR de valores iguais.



Fonte: 3.3. Métricas e pontuação: quantificando a qualidade das previsões — documentação do scikit-learn 1.1.3

Outra forma de utilizar a curva ROC para avaliação de determinado modelo é calculando sua área. A partir deste processo, tem-se um número que pode variar entre 0 e 1.0, sendo que, quanto mais próximo de 1.0, mais eficiente o modelo seria, com máximo TPR e mínimo FPR. A área abaixo da curva ROC é uma métrica denominada de Pontuação AUC (área abaixo da curva, em português) ROC.

3. METODOLOGIA

A metodologia deste trabalho se consistiu no primeiro momento a entender mais sobre os assuntos abordados, através de um levantamento bibliográfico sobre os temas de Aprendizado de Máquina e Aprendizado Profundo, estudando os principais modelos e técnicas dessas áreas para criar uma solução correta e eficiente do problema abordado. Para ambos os modelos criados aqui, foram seguidas as etapas: levantamento e pré-processamento, aplicação de técnicas das áreas, dos dados que envolvem o problema (fatores de risco e imagens de tomografia computadorizada), processamento desses dados para criação do modelo classificador e fase final de avaliação do mesmo, para entender o desempenho do modelo criado.

3.1 Base de dados para os fatores de risco

Os dados utilizados na construção do modelo de Aprendizado de Máquina para fatores de risco foram retirados do site *Kaggle*, sendo um dos conjuntos de dados disponibilizados para propósitos educacionais. O conjunto original conta ao todo com 5110 registros, em que cada um contém informações de uma pessoa diferente, rotulada como vítima de AVC ou não. O conjunto original estava desbalanceado, com a grande maioria dos valores sendo de pessoas rotuladas como não AVC. As características presentes no conjunto de dados são:

3.1.1 Variáveis categóricas Binárias: dois possíveis valores apenas

- Sexo: Masculino ou feminino, em que o sexo masculino é um fator de risco
- Hipertensão: Indivíduo é ou não hipertenso. Hipertensão é um fator de risco.
- Doença cardíaca: Indivíduo tem ou não alguma doença cardíaca, o que é um fator de risco.
- Estado civil de ser ou já ter sido casado: Sim ou não, tal variável pode estar ligada a estilo de vida e estresse, o qual é um fator de risco da doença.
- Tipo de residência: Urbana ou rural, tal variável esta ligada a estilo de vida, que pode ser um fator considerável para a doença
- Vítima de AVC: Sim ou não, essa é a variável alvo do trabalho, sendo utilizada para rotular os dados para previsão.

3.1.2 Variáveis categóricas não binárias: vários possíveis valores

- Tipo de emprego: empregado privado, funcionário público (governo), empreendedor ou não trabalha.
- Status com relação a tabagismo: indivíduo nunca fumou, fuma formalmente, fuma regularmente ou situação desconhecida.

3.1.3 Variáveis de valor contínuo:

- Idade: idade avançada é um fator de risco para a doença
- Nível de glicose: diabetes é um fator de risco para a doença
- BMI: Índice de massa corpórea no padrão americano, utilizado para identificar obesidade.

3.1.4 Variável ordinal:

- Id: Identificação de cada registro. Tal variável foi desconsiderada por não ter impacto na variável alvo do trabalho que rotula os casos de AVC.

3.2 Base de dados para as imagens de tomografia computadorizada

Para criação do modelo classificador de imagens de TC, foi utilizado um conjunto de dados com 475 imagens no total, divididas em três classes diferentes: AVC Isquêmico, AVC Hemorrágico ou não AVC. Entretanto, como o objetivo deste trabalho é de apenas identificar a ocorrência de Acidente Vascular Cerebral em determinada vítima, sem especificar o tipo de AVC de fato, este conjunto foi redistribuído para apenas duas classes.

Dessa forma, o conjunto original passou a ter apenas as classes AVC e não AVC, com as seguintes distribuições:

- AVC possui 301 imagens
- Não AVC possui 174 imagens

3.3 Ferramentas utilizadas

3.3.1 Python

Python é uma linguagem de programação de alto nível, gratuita e muito utilizada no desenvolvimento de aplicações modernas. Criada por volta de 1990, tal linguagem apresenta um uso mais simples quando comparado com o uso de outras linguagens do mercado, trazendo mais facilidade para quem não é necessariamente do ramo da programação. Ela prioriza a legibilidade do código, combinando uma sintaxe clara com recursos poderosos de sua biblioteca padrão (recursos nativos) e de outras bibliotecas desenvolvidas pela comunidade.

Por ser acessível e amigável, a linguagem de programação Python se tornou bastante popular no mercado, com muitos frameworks e módulos prontos desenvolvidos e disponibilizados para os usuários da ferramenta. Bibliotecas de áreas como Ciência de Dados, Automação, Aprendizado de Máquina, Aprendizado profundo, Redes Neurais, são bastante famosas e utilizadas pelos usuários da linguagem, podendo ser facilmente importadas para que não haja a necessidade de implementar funções complexas e componentes desde o início.

A alta disponibilidade de bibliotecas e recursos para as áreas de Aprendizado de Máquina e Aprendizado profundo, assim como a facilidade de uso da linguagem foram os principais critérios para escolha do uso dessa linguagem para o desenvolvimento deste trabalho. Abaixo estão listadas as principais bibliotecas Python criadas por terceiros e utilizadas para o desenvolvimento da solução proposta.

3.3.2 Bibliotecas da linguagem utilizada

- *Numpy*: utilizada para trabalhar com vetores e matrizes de forma eficiente e facilitada, possibilitando operações matemáticas velozes com essas estruturas em grandes dimensões e complexidades. Otimizada com a linguagem de programação C, Numpy traz velocidade e performance bastante agradável para essas operações no Python.

- *Pandas:* biblioteca Python para análise e manipulação de dados. Permite gerenciar dados em grandes volumes, aplicando operações matemáticas de diferentes graus de complexidade de uma forma bastante flexível, fácil, rápida e poderosa. Diversas operações presentes nesta biblioteca são construídas em cima de componentes *Numpy*.

- *Matplotlib*: biblioteca base para muitas bibliotecas de visualização de dados gráfica, compreende a criação de gráficos estáticos, animados e interativos em Python.

- *Seaborn:* biblioteca Python para visualização de dados em alto nível. Baseada na biblioteca Matplotlib, Seaborn traz muitas implementações de gráficos prontas como: gráfico de linhas, gráfico de barras, gráfico de pontos de calor, etc.

- *Scikit-learn*: também conhecida como sk learn, é uma das principais ferramentas para se trabalhar com Aprendizado de Máquina na linguagem Python. Tal biblioteca traz diversas implementações de modelos de regressão, classificação e agrupamento, assim como a implementação de várias métricas de avaliação bastante conhecidas como: acurácia, precisão, matriz de confusão, erro quadrado médio, curva ROC, etc.

  Construída a partir da junção de outras bibliotecas como Numpy, Matplotlib e SciPy (biblioteca que implementa métodos estatísticos em Python), a Scikit-learn traz consigo também diversos componentes para otimização/ajuste de hiperparâmetros, possibilitando que modelos eficientes sejam criados e aperfeiçoados para contextos mais específicos. Sk Learn traz também algumas implementações de componentes para divisão de dados inteligente, como algoritmos para validação cruzada de dados, e algoritmos para normalização dos dados, como normalização padrão, normalização min-max, entre outras.

- *Tensor Flow*: plataforma de código aberta que facilita a criação de modelos de Aprendizagem de Máquina e Aprendizagem profunda. Tensor Flow traz possibilidades para que o usuário do framework possa trabalhar com modelos prontos ou cria-los de acordo com as necessidades. Com esta biblioteca é possível facilmente estruturar redes neurais simples e complexas, alterando o número de neurônios, função de ativação, adicionando diferentes tipos de camadas, camadas de pré-processamento, entre outros.

4. DESENVOLVIMENTO

Referências

AVC

https://www.gov.br/saude/pt-br/assuntos/saude-de-a-a-z/a/avc

https://bvsms.saude.gov.br/avc-acidente-vascular-cerebral/

http://departamentos.cardiol.br/dha/revista/8-3/acidente.pdf

https://dco-unesp-bauru.github.io/tcc-bcc-2020-2/BrunaLT/thesis-BrunaLT.pdf#page=39&zoom=100,113,261

https://scielo.br/j/rbepid/a/KFNpCf4NCd8mhcsT4FqJsHP/?lang=pt#:~:text=Segundo%20os%20registros%20no%20Sistema,tratamento%20do%20AVC1-4.

https://pebmed.com.br/como-identificar-o-avc-isquemico-na-tomografia-computadorizada/

https://www.medway.com.br/conteudos/como-identificar-os-sinais-precoces-do-avc-na-tc/

ML

https://profs.info.uaic.ro/~ciortuz/SLIDES/2017s/ml0.pdf

[1] https://dcm.ffclrp.usp.br/~augusto/publications/2003-sistemas-inteligentes-cap4.pdf

https://lamfo-unb.github.io/2017/07/27/tres-tipos-am/

https://www.ibm.com/topics/logistic-regression

https://scikit-learn.org/stable/modules/linear_model.html

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

https://www.datacamp.com/tutorial/random-forests-classifier-python

https://en.wikipedia.org/wiki/Random_forest

https://en.wikipedia.org/wiki/Decision_tree

DP

https://drauziovarella.uol.com.br/corpo-humano/neuronio/#:~:text=Compartilhar-,Neur%C3%B4nios%20s%C3%A3o%20as%20c%C3%A9lulas%20que%20caracterizam%20o%20sistema%20nervoso%2C%20respons%C3%A1veis,externo%20ou%20do%20pr%C3%B3prio%20organismo.

https://www.gsigma.ufsc.br/~popov/aulas/rna/neuronio_artificial/index.html#:~:text=O%20neur%C3%B4nio%20artificial%20%C3%A9%20um,Esquema%20do%20neur%C3%B4nio%20biol%C3%B3gico.&text=Viu%2Dse%20o%20c%C3%A9rebro%20como%20um%20sistema%20computacional.

https://aws.amazon.com/pt/what-is/neural-network/#:~:text=Uma%20rede%20neural%20%C3%A9%20um,camadas%2C%20semelhante%20ao%20c%C3%A9rebro%20humano.

https://towardsdatascience.com/why-we-need-bias-in-neural-networks-db8f7e07cb98

https://arxiv.org/abs/1511.08458

https://www.upgrad.com/blog/basic-cnn-architecture/#:~:text=other%20advanced%20tasks.-,What%20is%20the%20architecture%20of%20CNN%3F,the%20main%20responsibility%20for%20computation.

https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939

https://www.jeremyjordan.me/convnet-architectures/

https://www.ibm.com/cloud/learn/convolutional-neural-networks

Métricas de avaliação:

Livro data Science para negócios [2]

sklearn.metrics.accuracy_score — scikit-learn 1.1.3 documentation

sklearn.metrics.confusion_matrix — scikit-learn 1.1.3 documentation

ROC Curve, a Complete Introduction | by Reza Bagheri | Towards Data Science

3.3. Métricas e pontuação: quantificando a qualidade das previsões — documentação do scikit-learn 1.1.3

Erro do tipo I – Wikipédia, a enciclopédia livre (wikipedia.org)

Metodologia – base de dados para fatores de risco

Stroke Prediction Dataset | Kaggle

Metodologia – base de dados para imagens

Ferramentas

Python: https://pt.wikipedia.org/wiki/Python

Numpy: https://numpy.org/

Seaborn https://seaborn.pydata.org/

Matplotlib https://matplotlib.org/

Pandas: https://pandas.pydata.org/

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

Tensor flow: https://www.tensorflow.org/overview