# Previsão de Estágio da Infecção por COVID-19 – Dados do HC

Alberto da Rocha Miranda - 12542498

Pedro Andrade Pereira - 14602007

Instituto de Matemática e Estatística, Universidade de São Paulo - SP

30/06/2025

**Resumo**

***Neste trabalho buscou-se construir uma rede neural que pudesse prever o estágio de infecção de um paciente de COVID-19 ao classificar se ele possui ou não a imunoglobulina G a partir de exames médicos diversos. Utilizando dados do Hospital das Clínicas da USP disponibilizados foi feito o pré-processamento destes dados a fim de poder utilizá-los asseguradamente na rede neural e garantir um melhor desempenho da rede. A construção da rede se deu visando a quantidade de exames e como o processo de aprendizado da rede neural se daria. O estágio de treinamento e teste da rede aconteceu pelo método k-fold, dando como retorno o valor de medidas estatísticas de 5 modelos de rede neural obtidos, dos quais um deles foi escolhido e analisado em função do objetivo principal do trabalho.***

## I. Introdução

### I. Motivação para o trabalho

Durante uma dada infecção de COVID-19 o corpo de um paciente deve produzir anticorpos a fim de combater a doença. Um destes anticorpos produzidos é do tipo IgG (imunoglobulina G) sendo ele o principal responsável por garantir que infecções futuras sejam combatidas muito mais rapidamente, de forma até imperceptível. Este tipo de anticorpo geralmente só é gerado em etapas posteriores do combate à infecção e é percebido no corpo após o paciente estar curado.

A informação de que uma pessoa possui este tipo de anticorpo contra o Covid-19 em seu sistema garante que seu corpo já teve uma infecção passada e, caso ela seja recente, mostra que já está numa fase posterior da infecção. Isto pode ser valioso para determinar a imunidade da pessoa e também o estágio atual da doença.

### II. Objetivos do trabalho

Este trabalho tem como objetivo buscar a criação e treinamento de uma rede neural capaz de determinar, através de diversos exames médicos relacionados ou não com a infecção de Covid-19, a presença de anticorpos IGg contra o SARS-CoV-2. O treinamento deve ocorrer utilizando os dados dos exames médicos do ano de 2020 de pacientes do Hospital das Clínicas da Faculdade de Medicina da USP (HC), e como objetivo secundário, deve ser discutida a viabilidade de um modelo utilizando este fluxo de dados.

### III. Estrutura do Relatório

O relatório a seguir contém uma boa descrição dos dados utilizados, o pré-processamento feito com eles para torna-los viáveis para entrada na rede neural, a arquitetura desta rede feita de forma a maximizar a acurácia das previsões, a descrição dos experimentos feitos para treinar e testar a rede, os resultados obtidos e, por fim, a discussão sobre os resultados e o alcance do objetivo principal.

## II. Metodologia

### I. Dados

Os dados utilizados vieram do HC, disponibilizados em um repositório montado pela Universidade de São Paulo (USP) e a Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), e estão separados em três partes:

1. O dicionário de colunas, indicando o tipo de cada coluna e campos das outras partes, com uma breve descrição deles.

2. Os dados dos pacientes, contendo os IDs dos pacientes, sexo/gênero, ano de nascimento e outras informações não relevantes para nosso trabalho.

3. Os dados dos exames, separados por tipos de exames, contendo dados dos atendimentos (separados por datas) de cada paciente, os nomes dos exames, o tipo de análise do exame (de analito), o resultado da análise, seu indicador da unidade de medida e um valor de referência do resultado.

Destes dados, o que buscamos usar são os resultados dos exames e análises médicas, sexo/gênero e ano de nascimento do paciente e, por fim, a data e o ID do atendimento. Como os pacientes possuem registros de mais de um atendimento, a separação de dados deve ser feita indexando os atendimentos, sem manter um registro do histórico dos pacientes.

Os dados utilizados, fora o índice do atendimento, serão tratados apenas como dois tipos, numéricos e binários, mesmo que não estejam dispostos obrigatóriamente desta forma. A disponibilização dos dados foi feita em três planilhas para cada parte tratada, sendo a do dicionário no formato XLSX e as outras duas como CSV, utilizando o caractere "como separador.

## II. Pré-processamento dos dados

O pré-processamento de dados é uma etapa extremamente importante no desenvolvimento de um modelo de rede neural funcional. Nesta etapa devemos juntar tudo numa única planilha ou dataframe, melhorar a qualidade dos dados, selecionar apenas os que podem ser usados e fazer com que eles estejam em formatos mais aceitáveis para a rede a fim de otimizá-la.

Os dados do HC não vieram ordenados por atendimento, além de que os dados dos pacientes vieram separados, portanto, o primeiro passo a ser seguido foi o de juntar as duas planilhas em uma usando o ID do paciente e então reorganizar a planilha de exames em função dos atendimentos. Para isso foi utilizada a função de "pivotagem"da tabela, alterando o formato dos dados seguindo um dado índice (ID de atendimento, gênero e ano de nascimento) e os valores de uma das colunas para serem as novas colunas da tabela pivot. Como nosso objetivo é utilizar os dados das análises dos exames, foi necessário juntar as colunas de exames e de análises em uma única, e só então realizar o pivot da tabela selecionando os valores desta nova coluna conjunta para serem as novas colunas da nossa tabela.

### II.1 Organização das respostas

Os dados de resposta da rede neural devem ser os da coluna de exames "COVID-19 - PESQUISA DE ANTICORPOS IgG COVID-19 IgG", que nos dá exatamente as informações que procuramos sobre a existência ou não da imunoglobulina G em formato binário, representado por "Reagente"(existente) e "Não reagente"(não existente), os dados classificados como "indeterminados"foram descartados. Como o exame possui dois tipos de análises, o binário e o numérico, e nossa saída esperada da rede é binária, descartamos a segunda coluna registrada como "COVID-19 - PESQUISA DE ANTICORPOS IgG COVID-19 índice"para não causar algum viés nos dados de treinamento.

Além disso, dos dados de respostas, foi preciso descartar todos os que não possuíam o resultado do exame de Covid-19 IgG, porém, com o objetivo de aumentar nossa quantidade de dados de forma pouco invasiva e também melhorar a distribuição das classes da resposta, foi seguida uma hipótese coerente nesta situação de que se o paciente houvesse realizado algum outro teste relacionado à detecção de covid naquele atendimento, o resultado do exame de IgG poderia ser considerado como "Não Reagente". Como o objetivo é minimizar o viés nos dados, foram inclusas neste procedimento apenas as colunas de testes de covid-19 com uma quantidade menor do que a dos dados de resposta, então o exame "Coronavírus 2019-nCoV", com cerca de 2000 resultados, não foi considerado nesta etapa para que as classes de resposta não fossem extremamente desbalanceadas e enviésadas.

### II.2 Melhoria da qualidade dos dados

Após esta etapa, é necessário focar na melhoria da qualidade de dados e conversão deles para formatos que podem ser usados como entrada de uma rede neural. Neste caso, os dados vieram em diferentes formatos e sem uma padronização clara, sendo que muitos deles são apenas numéricos, com algumas exceções como alguns representados por frações e outros por desigualdades (ex: $> 1$). Estes foram todos aproximados por floats em nossa planilha final, sendo que as desigualdades foram escolhidas para serem exatamente o número que aparece na declaração, mesmo claramente não sendo uma representação fiel, pode ser uma aproximação aceitável para não precisar diminuir o volume de dados.

O outro tipo de dado que existia na planilha era os binário, porém este também tinha algumas exceções, com vários deles sendo intermediários como "Raro", "Normal", "Ligeiramente", etc. ou representado resultados indeterminados. Além disso, os resultados binários eram representados fora de padrão como palavras com aspecto "positivo"ou "negativo"no ambiente médico. Estas palavras foram identificadas com o uso de algoritmos separa-

dos. Palavras que indicavam algo positivo neste ambiente foram classificadas como 1 e as negativas como 0, porém caso não fosse achada nenhuma destas palavras com estes aspectos, também havia a possibilidade dos resultados binários serem indicados de outra forma e, por este motivo, também foram atribuídos a valores binários os resultados que se encaixavam como um dos dois valores mais utilizados naquela coluna utilizando o método de "Label Encoding".

### II.3 Limpeza de colunas

Foram identificados e descartados os dados com resultados que fugiam do esperado, como colunas apenas com datas, nomes de pessoas e horários, pois não indicavam ligação alguma com nossa resposta, poderiam apenas atrapalhar nosso treinamento com maior dimensionalidade e confusão no momento de conversão para dados numéricos ou binários. Por um motivo parecido, colunas com menos de 10% de dados não nulos foram descartadas com o objetivo de diminuir a quantia de dimensões trabalhadas, diminuindo a chance de overfitting no modelo e visando que o número de dados seja menor que a quantidade de dimensões.

Para valores nulos restantes na planilha final, foi escolhido trocá-los, para valores numéricos, por zero e, para valores binários, por 0.5. O sexo/gênero do paciente foi classificado como binário (0 e 1) e a idade foi calculada a partir da diferença do ano de nascimento com o ano de 2020 (ano de todos os atendimentos da planilha), sendo que idades nulas foram substituídas pela mediana geral das idades, que deve representar bem esta população dos exames. No fim, ao todo, restaram 766 linhas de dados com 172 colunas, com exceção da coluna índice.

### II.4 Normalização

O pré-processamento dos dados abrange apenas até esta melhoria da qualidade dos dados feita, porém, além disso, também foi realizada uma normalização de dados logo após o momento de separação entre os dados de treinamento e teste, visando não contaminar os dados e não enviesar os resultados. A normalização foi realizada seguindo o método Min-Max, buscando otimizar o treinamento da rede, já que a normalização faz com que os dados fiquem num intervalo entre 0 e 1, e estando numa escala melhor, a rede neural pode convergir mais rápido com a distribuição menos extrema dos valores dos pesos. A relação da normalização segue a descrita na equação 1:

$$x_i' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{1}$$

Segundo [de Amorim et al., 2023], a técnica Min-Max para normalização é muito útil para quando não temos dados atípicos, ou seja, dados muito maiores do que a média. Neste caso nos serve bem, pois como já foi visto, dados muito extremos nestes resultados médicos são geralmente representados como desigualdades, portanto não fogem muito da média. Além disso, é seguido também o passo de que a normalização ocorre apenas nos dados de treinamento, e os dados de teste apenas são transformados seguindo esta normalização do treino.

## III. Arquitetura da rede neural

A arquitetura da rede neural seguiu um objeto da classe MPLClassifier, liberado pela biblioteca aberta "Scikit-Learn"descrita em [Pedregosa et al., 2011]. As propriedades definidas para a rede foram:

- 172 entradas
- 200 batches
- Taxa de aprendizado constante de 0,001
- Saída binária (0 ou 1)
- 2 camadas escondidas
    1. Primeira camada com 100 neurônios
    2. Segunda camada com 50 neurônios

A função de ativação escolhida foi a função ReLU, boa para classificadores, sendo definida pela fórmula $R(x) = \max(0, x)$. Já a função que está sendo otimizada pela rede, ou seja, a função de perda, foi escolhida para ser a função Log Loss, definida pela equação 2. Esta equação serve bem para classificações binárias e aplica uma penalidade maior, com crescimento exponencial, para o quanto mais confiante o modelo está sobre uma previsão errada. Neste caso, $y$ é a resposta real, $N$ é a quantidade de amostras e $p$ é a saída prevista pelo modelo.

$$LogLoss(y, p) = -\frac{1}{N}\sum_{i=1}^{N}(y_i \log p_i + (1 - y_i)\log(1 - p_i)) \tag{2}$$

No nosso caso, todas estas propriedades definidas foram pensadas tendo em vista o tamanho das entradas (176), visando que o modelo reconheça padrões gerais de associação dos resultados de cada exame com o vírus do covid-19 na primeira camada escondida, e, na segunda camada, já sejam reconhecidos padrões mais específicos relacionados à imunoglobulina G, prevendo, então, o estágio

da doença do paciente. Para a otimização da rede neural foi executado apenas o processo de normalização dos dados, como descrito na seção 2.

## IV. Descrição dos experimentos

Os treinamentos foram feitos utilizando o método "k-fold"com $k = 5$, que separa os dados em 5 partes, sendo 4 partes para treinamento e uma para testes (neste caso então são 20% dos dados para teste), rodando o treinamento 5 vezes e cada vez mudando em qual parte se dá o conjunto de testes. O modelo de "k-fold"utilizado foi o descrito pela biblioteca do Scikit-learn, com a ordem dos dados sendo sempre aleatorizada antes da divisão dos batches e a divisão sendo obtida de forma automatizada pelo método "split". Este modelo é muito útil para nosso caso pois não temos muitos dados disponíveis, então com ele é possível obter diferentes modelos que podem performar melhor ou pior separando menos dados do que normalmente seria necessário.

A métrica utilizada para avaliar os modelos foi a acurácia, dada pela equação 3, em que $TN$ e $TP$ são, respectivamente, os números de valores negativos e positivos classificados corretamente e $FN$ e $FP$ os números de valores negativos e positivos classificados erroneamente.

$$\text{Acurácia} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{TP + TN}{n} \quad (3)$$

Após os 5 treinamentos do k-fold, o modelo com a melhor acurácia foi selecionado e então foi realizado um teste final com os 20% dos dados pré-processados iniciais que não foram tocados por nenhum dos treinamentos, procurando reafirmar a efetividade do modelo.

Outras medidas utilizadas para avaliação do modelo foram:

$$\begin{cases} \text{Precisão} & = \frac{TP}{TP+FP} \\ \text{Cobertura} & = \frac{TP}{TP+FN} \\ \text{Medida-F} & = \frac{2 \cdot \text{Precisão} \cdot \text{Cobertura}}{\text{Precisão} + \text{Cobertura}} \end{cases}$$

## III. Resultados

Para verificar o balanceamento dos dados das respostas e um possível viés nos dados referente a isso, abaixo encontra-se o histograma com o valor binário de todos os dados:

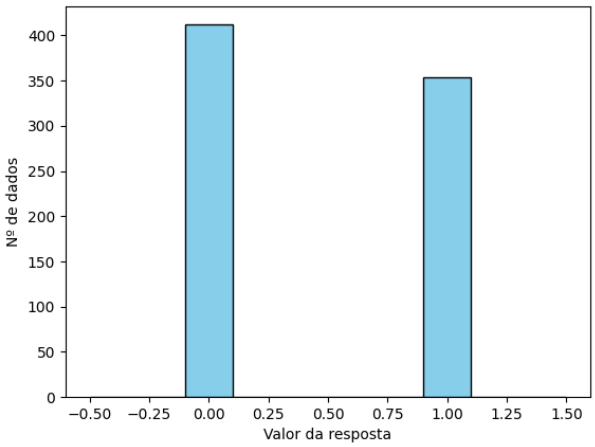


**Figura 1:** *Histograma mostrando a quantidade de dados 0 e 1 nos dados de respostas separados para o treinamento com o k-fold.*

Dos resultados obtidos dos experimentos, os valores das medidas para avaliação de cada modelo encontram-se na tabela 1. Foram agrupados os piores, melhores, a média e o desvio padrão de cada tipo de medida para todos os modelos do k-fold.

| Medida | Pior valor | Melhor valor | Média | Desvio Padrão |
|---|---|---|---|---|
| Acurácia | 0.5752 | 0.7190 | 0.6605 | 0.0509 |
| Precisão | 0.5270 | 0.7000 | 0.6311 | 0.0599 |
| Cobertura | 0.5429 | 0.7027 | 0.6373 | 0.0686 |
| Medida-F | 0.5455 | 0.6950 | 0.6334 | 0.0612 |

**Tabela 1:** *Valores das medidas para avaliações dos modelos para todos os modelos dos k-treinamentos.*

Para uma melhor comparação, na figura 2 é possível ter uma ideia do desempenho de cada modelo em seus testes, avaliando apenas pela acurácia.

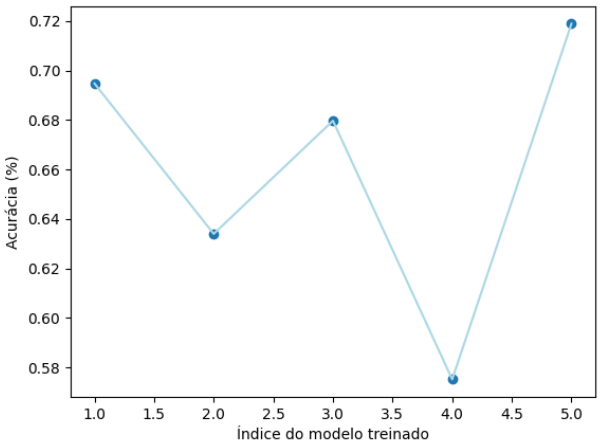


**Figura 2:** *Desempenho da acurácia de cada um dos 5 modelos treinados.*

Selecionando o modelo com a melhor acurácia, a matriz de confusão deste está representada na figura 3, dando uma boa ideia de se há algum viés nos dados com estes resultados.

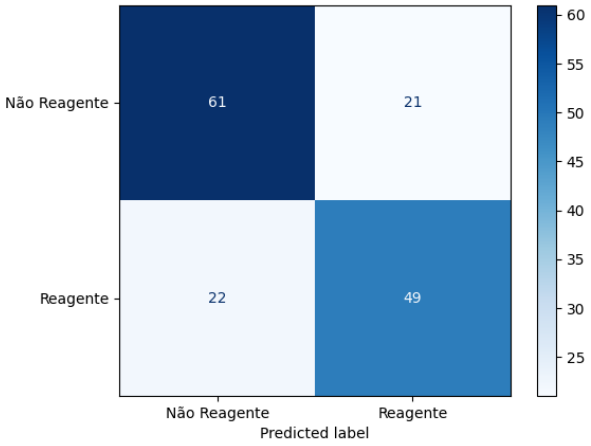


**Figura 3:** *Gráfico heatmap da matriz de confusão do modelo com maior acurácia (neste caso o 5º modelo).*

## IV. Discussão

Analisando-se os resultados obtidos é possível ver que há um leve desbalanceamento nos dados (figura 1) com um volume um pouco maior de dados negativos (não reagente) para o IgG, o que acarretou em resultados mais favoráveis do melhor modelo para identificar pessoas com resultado negativo (figura 3). Porém, no geral, os resultados das medidas feitas nos mostram que o modelo teve sucesso em identificar padrões para realizar a classificação da fase do vírus apenas com exames médicos variados, com a média de acurácia dos modelos sendo de 66,05% e precisão média de 63,11%. Além disso, os resultados de desvio-padrão foram bem pequenos, mostrando que os modelos são estáveis independentemente dos dados de treinamento e teste selecionados.

O real problema é que, visando a viabilidade e utilidade deste modelo, os resultados não são nem um pouco otimistas. Num contexto de aplicação para a medicina, nosso modelo da rede neural com maior acurácia (5º modelo do k-fold) não obteve resultados bons, ficando num limite próximo à separação da aleatoriedade e prevendo somente 70% dos casos corretamente. Além disso, por conta da falta de dados, não foram feitos mais testes com este modelo de rede para assegurar este resultado, que pode ser ainda pior com um maior volume de dados não tocado pelo treinamento. Dessa forma, vemos que nossos resultados com a rede neural não são realmente satisfatórios na prática, e isto provavelmente se deve por conta da baixa quantidade de dados disponíveis para treiná-la. Apenas 612 dados foram usados no treino, sendo que alguns possuiam certo viés por terem sido adicionados com uma hipótese que pode conter algumas falhas.

Porém, tendo em vista que a performance da rede neural foi razoavelmente boa (média), há de se perceber que os resultados obtidos mostram um grande potencial nesta rede neural construída para criar e treinar um ótimo modelo, com uso viável no meio profissional com pacientes, caso haja um volume útil de dados (contendo os resultados do exame de Covid-19 IgG) consideravelmente maior do que o disponibilizado pelo HC.

## Referências

[de Amorim et al., 2023] de Amorim, L. B., Cavalcanti, G. D., and Cruz, R. M. (2023). The choice of scaling technique matters for classification performance. *Applied Soft Computing*, 133:109924.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.