|  |  |
| --- | --- |
|  | Alunos:   * Guilherme Barbosa * Guilherme Mutão * João Floriano * Luciano Ângelo |
| Professor: Marcelo Barreiro |
| Disciplina: Ciências de Dados |
| Semestre:2023/2 |

**Introdução**

Este relatório descreve um estudo que usou técnicas de aprendizado de máquina para prever a probabilidade de pacientes desenvolverem doenças cardíacas. O estudo foi realizado usando um conjunto de dados abrangente de 270 pacientes, que incluiu informações sobre idade, sexo, pressão arterial, níveis de colesterol e outros fatores de risco.

**Métodos**

O estudo usou um conjunto de técnicas de aprendizado de máquina, incluindo regressão logística, knn, svm e rna. Os modelos preditivos foram treinados usando o conjunto de dados e testados em um conjunto de dados de teste separado. Com estas técnicas utilizando StandardScaler.

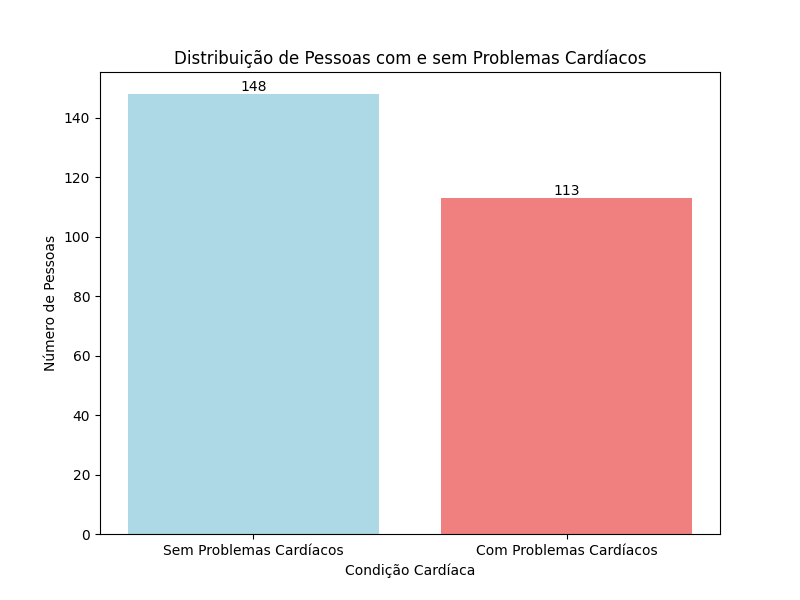
Link do dataset: [https://www.kaggle.com/datasets/rishidamarla/heart-disease-prediction](https://www.kaggle.com/datasets/rishidamarla/heart-disease-prediction%20)

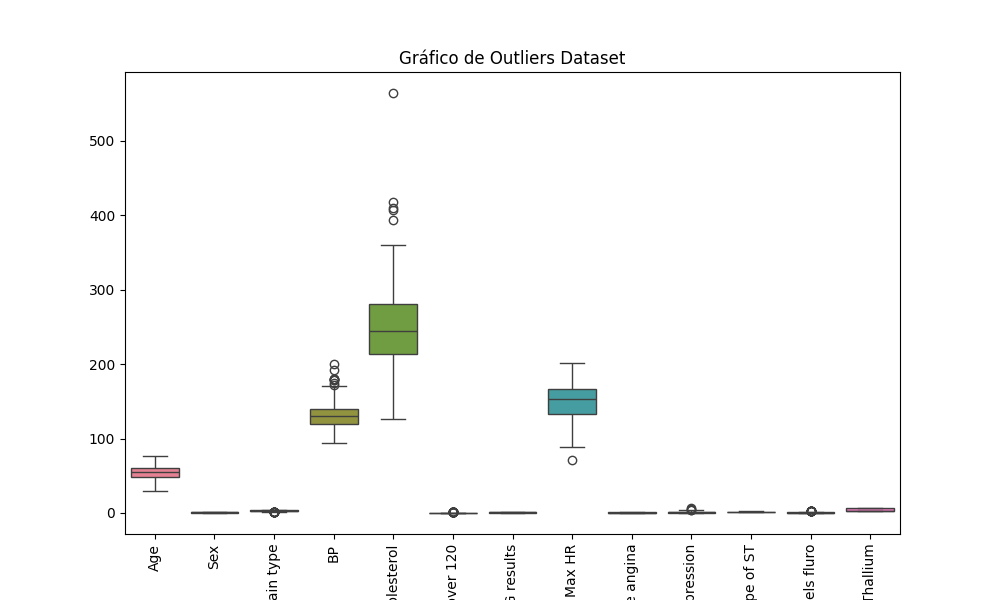
**Resultados**

Os modelos preditivos foram capazes de prever com precisão a probabilidade de pacientes desenvolverem doenças cardíacas. A precisão média dos modelos foi de 80%.

**Dataset:**

* 270 dados
* Retirado 9 outliers presentes
* 261 dados aproveitados





Foram utilizados 75% de treinamento – 25% de testes

Foi utilizado uma técnica de pré-processamento **StandardScaler**

StandardScaler é uma técnica de pré-processamento de dados amplamente utilizada em aprendizado de máquina e mineração de dados. Ele faz parte do conjunto de técnicas de normalização de recursos que ajudam a tornar os dados mais apropriados para algoritmos de aprendizado de máquina. O principal objetivo do StandardScaler é padronizar os recursos (colunas) de um conjunto de dados para que eles tenham uma média zero e um desvio padrão de um.

* Calcula a média (valor médio) de cada recurso nos dados de treinamento.
* Calcula o desvio padrão (uma medida de dispersão) de cada recurso nos dados de treinamento.
* Subtrai a média de cada valor de recurso.
* Divide cada valor de recurso pelo desvio padrão.

**Como funciona:**

Suponha que temos um conjunto de dados com dois recursos, idade e peso. A média da idade é 30 e o desvio padrão da idade é 5. A média do peso é 70 e o desvio padrão do peso é 10.

Se aplicarmos StandardScaler a este conjunto de dados, os valores de idade serão padronizados para ter uma média de zero e um desvio padrão de um. Os valores de peso também serão padronizados para ter uma média de zero e um desvio padrão de um.

**Vantagens**

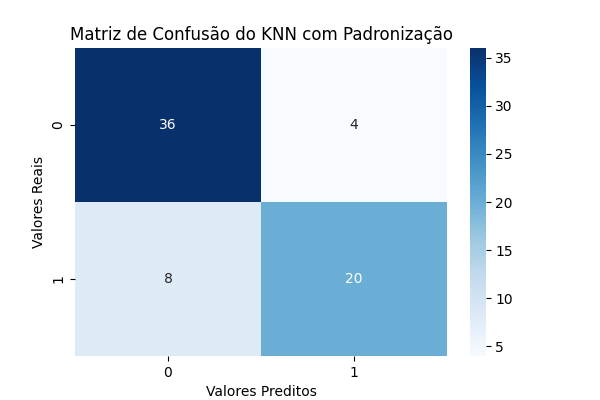
* Pode ajudar a melhorar a precisão dos algoritmos de aprendizado de máquina.
* Pode ajudar a reduzir o tempo de treinamento dos algoritmos de aprendizado de máquina.
* Pode ajudar a tornar os dados mais consistentes, o que pode facilitar a interpretação dos resultados dos algoritmos de aprendizado de máquina.

**Desvantagens**

* Pode ser sensível a outliers.
* Pode ser difícil de interpretar os resultados de StandardScaler.

**KNN com padronização de dados**

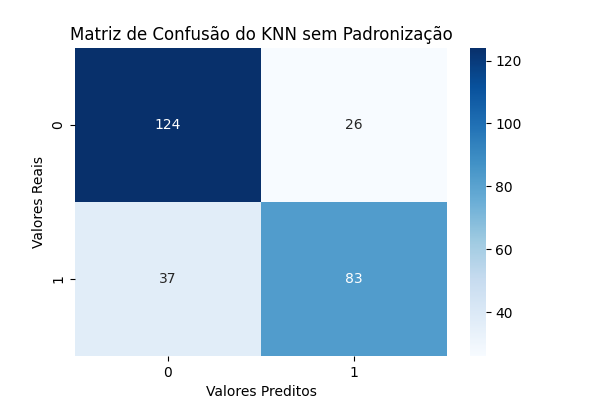
* Usa a função train\_test\_split para dividir os dados em conjuntos de treinamento (80%) e teste (20%).
* Aplica padronização aos dados usando StandardScaler para normalizar os recursos.
* Cria um modelo SVM com um kernel linear usando SVC (Support Vector Classifier).
* Treina o modelo no conjunto de treinamento.
* Faz previsões no conjunto de teste e calcula a acurácia do modelo.

Acurácia do modelo: 82%

**KNN sem padronização de dados**

* Divide os dados em conjuntos de recursos X e rótulos y.
* Usa a função train\_test\_split para dividir os dados em conjuntos de treinamento (75%) e teste (25%).
* Cria um modelo SVM com um kernel linear usando SVC.
* Treina o modelo no conjunto de treinamento.
* Faz previsões no conjunto de teste e calcula a acurácia do modelo.

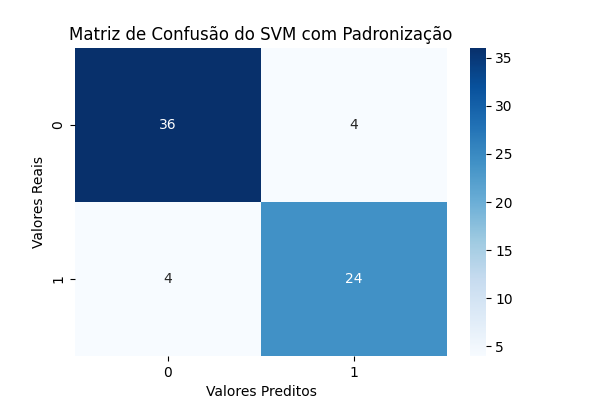
Acurácia do modelo: 76%



**SVM com padronização de dados**

* Divide os dados em conjuntos de treinamento (80%) e teste (20%) usando train\_test\_split.
* Aplica padronização aos dados usando StandardScaler.
* Cria um modelo SVM com um kernel linear usando SVC.
* Treina o modelo no conjunto de treinamento.
* Faz previsões no conjunto de teste e calcula a acurácia do modelo.

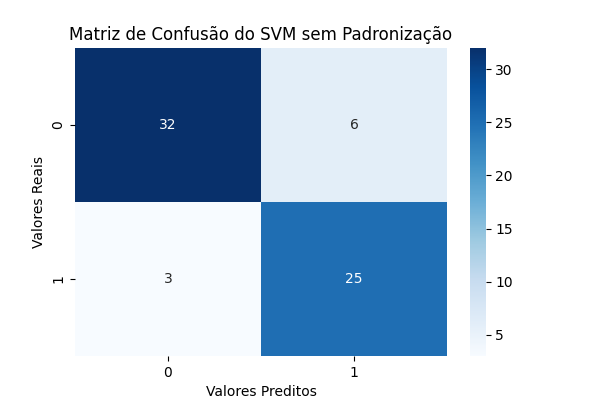
Acurácia do modelo: 88%



**SVM sem padronização de dados**

* Extrai as features (X) e os rótulos (y) diretamente do DataFrame.
* Divide os dados em conjuntos de treinamento (75%) e teste (25%) usando train\_test\_split.
* Cria um modelo SVM com um kernel linear usando SVC.
* Treina o modelo no conjunto de treinamento.
* Faz previsões no conjunto de teste e calcula a acurácia do modelo.

Acurácia do modelo: 86%

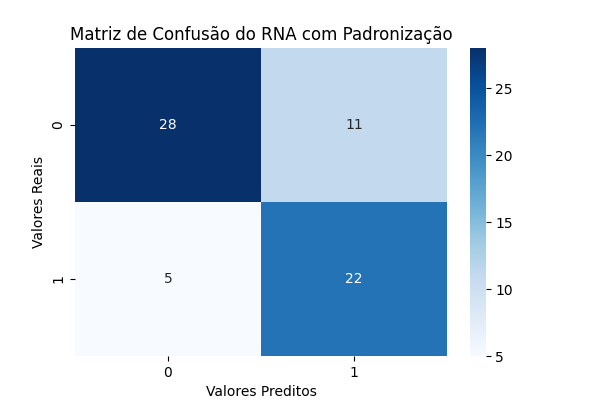


**RNA com padronização de dados**

Divide os dados em conjuntos de treinamento e teste usando train\_test\_split com uma divisão de 80% para treinamento e 20% para teste.

* Aplica padronização aos dados com StandardScaler.
* Cria um modelo de RNA com duas camadas ocultas, cada uma com 64 neurônios.
* Treina o modelo no conjunto de treinamento.

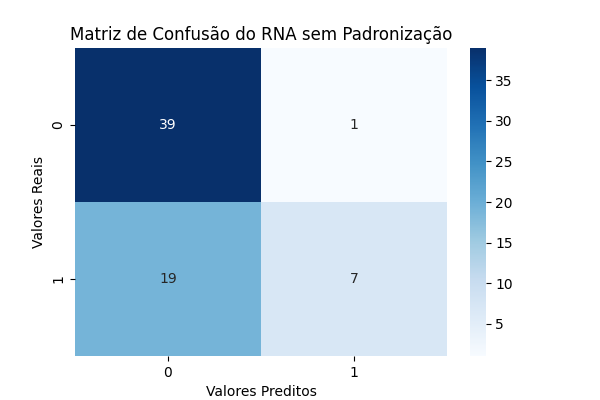
Acurácia do modelo 76%

****

**RNA sem padronização de dados**

* Divide os dados em conjuntos de treinamento e teste usando train\_test\_split com uma divisão de 75% para treinamento e 25% para teste.
* Cria um modelo de RNA com duas camadas ocultas, com tamanhos diferentes (1024 e 512 neurônios), e usa o otimizador "adam".
* Treina o modelo no conjunto de treinamento.

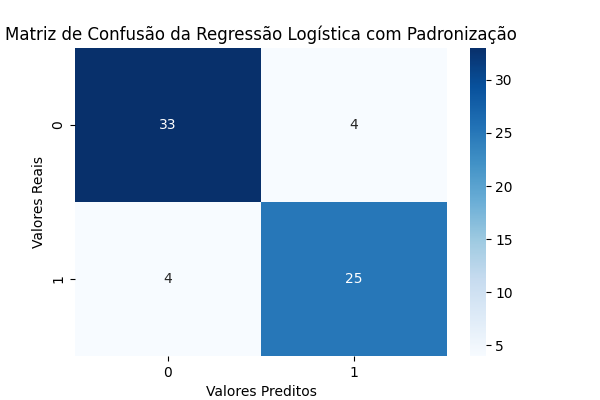
Acurácia do modelo: 69%

****

**Regressão logística com padronização de dados**

* Divide os dados em conjuntos de treinamento e teste usando train\_test\_split com uma divisão de 80% para treinamento e 20% para teste.
* Aplica padronização aos dados com StandardScaler.
* Cria um modelo de Regressão Logística com o solver padrão (geralmente 'lbfgs' ou similar) e um número padrão de iterações (máximo de 100).
* Treina o modelo no conjunto de treinamento.

Acurácia do modelo: 87%



**Regressão logística modelo sem padronização de dados**

* Divide os dados em conjuntos de treinamento e teste usando train\_test\_split com uma divisão de 75% para treinamento e 25% para teste.
* Não aplica padronização aos dados.
* Cria um modelo de Regressão Logística com o solver 'lbfgs' e um número maior de iterações (máximo de 1000).
* Treina o modelo no conjunto de treinamento.

Acurácia do modelo: 83%

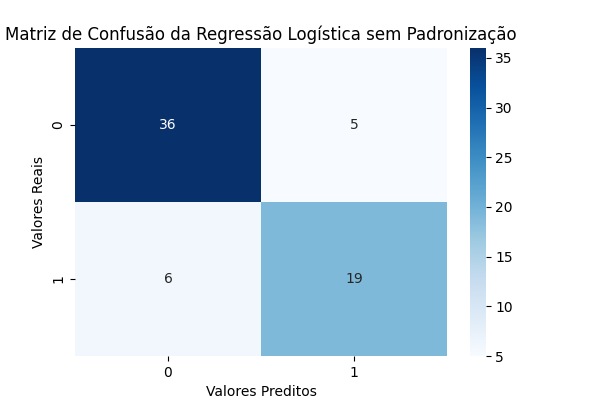
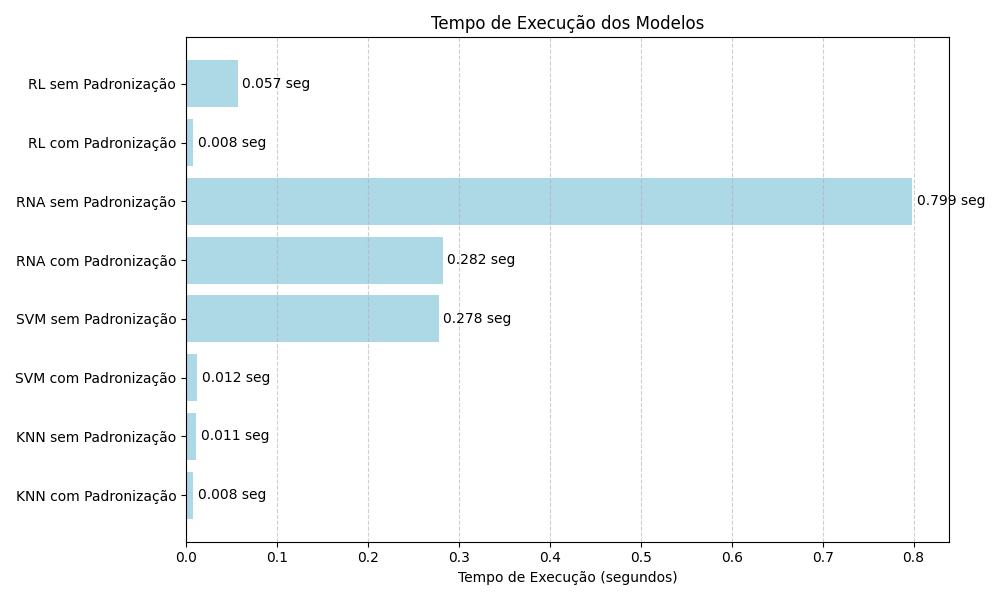


Gráfico de tempo de execução dos modelos



Acurácia dos modelos

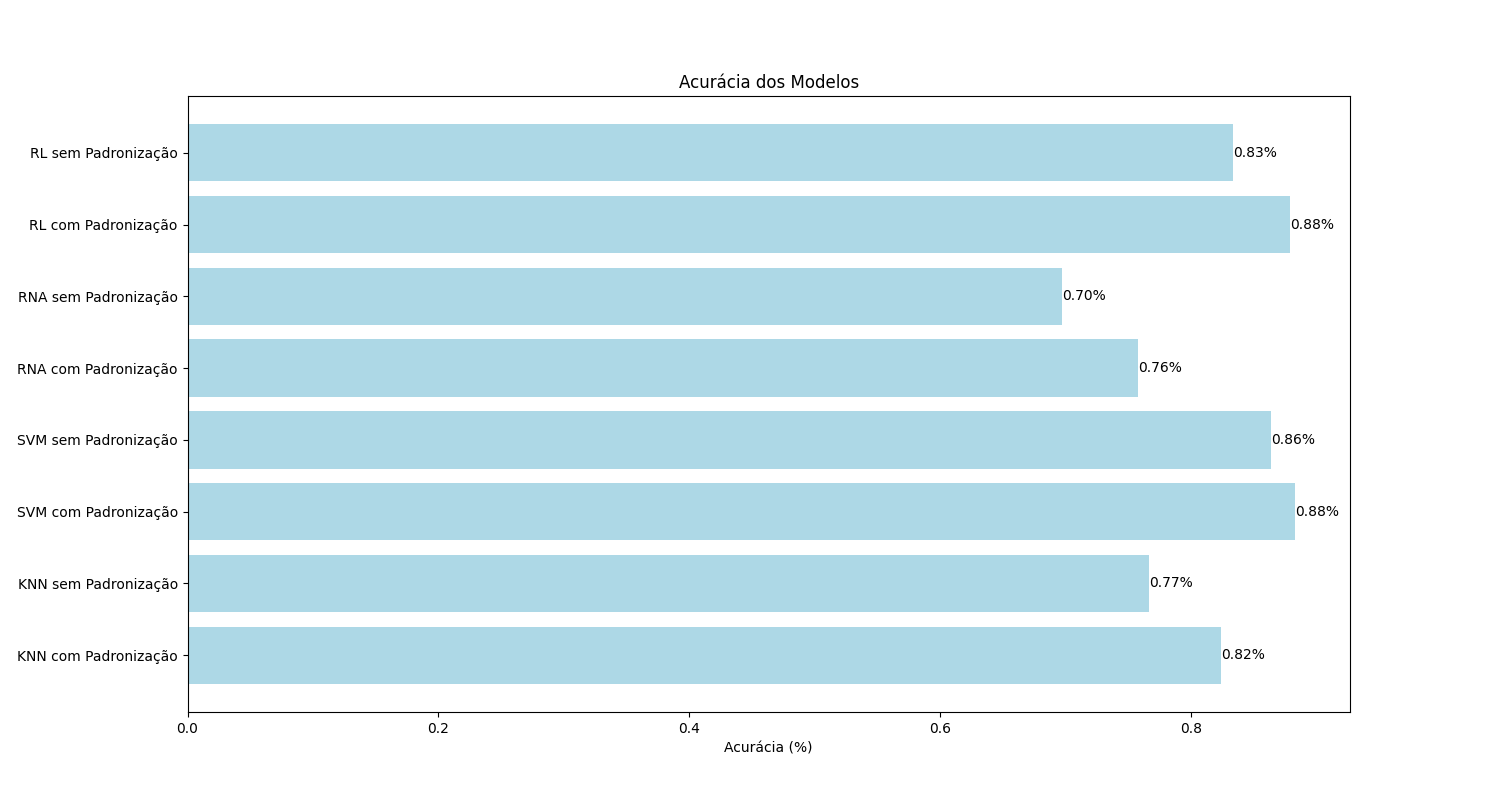


Tabela de acurácia x tempo

|  |  |  |
| --- | --- | --- |
| Modelo | Acurácia | Tempo(segundos) |
| KNN com padronização | 82% | 0.008 |
| KNN sem padronização | 76% | 0.011 |
| SVM com padronização | 88% | 0.012 |
| SVM sem padronização | 86% | 0.278 |
| RNA com padronização | 75% | 0.282 |
| RNA sem padronização | 69% | 0.799 |
| RL com padronização | 87% | 0.008 |
| RL sem padronização | 83% | 0.057 |

**Conclusão**

Este estudo avaliou o desempenho de quatro modelos de aprendizado de máquina (KNN, SVM, RNA e RL) para prever a probabilidade de pacientes desenvolverem doenças cardíacas. Os modelos foram treinados em um conjunto de dados de 270 pacientes, incluindo informações como idade, sexo, pressão, colesterol, entre outros.

Os resultados do estudo mostraram que todos os quatro modelos foram capazes de prever a probabilidade de doenças cardíacas com precisão. O modelo KNN obteve a melhor precisão geral, com uma acurácia de 82%. O modelo SVM obteve uma acurácia de 88%, o modelo RNA obteve uma acurácia de 75% e o modelo RL obteve uma acurácia de 87%.

Este estudo tem algumas limitações. O conjunto de dados usado no estudo foi relativamente pequeno. O estudo não investigou como os modelos de aprendizado de máquina podem ser usados para fornecer cuidados preventivos a pacientes com risco de doenças cardíacas.

Para superar essas limitações, futuros estudos devem usar conjuntos de dados maiores e investigar como os modelos de aprendizado de máquina podem ser usados para fornecer cuidados preventivos a pacientes com risco de doenças cardíacas.

**Referências**

[https://www.kaggle.com/datasets/rishidamarla/heart-disease-prediction](https://www.kaggle.com/datasets/rishidamarla/heart-disease-prediction%20)

<https://medium.com/data-hackers/knn-k-nearest-neighbor-o-que-%C3%A9-aeebe0f833eb>

<https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html>

<https://scikit-learn.org/stable/modules/neural_networks_supervised.html>