Sistemas Inteligentes - Regredir e Classificar

Tiago Gonçalves da Silva[1], **Guilherme Aguilar de Oliveira**[1]

[1]Departamento de Informática
Universidade Tecnológica Federal do Paraná (UTFPR) – Curitiba, PR – Brazil

tiagosilva.2019@alunos.utfpr.edu.br, guilhermeoliveira.2019@alunos.utfpr.edu.br

1. Introdução

Dentro do problema de classificação e regressão estabelecido consideramos estes subproblemas:

1. Como reconstruir o cálculo da gravidade dado os valores do histórico como treinamento?
2. Como fazer a classificação nas classes de gravidade dado os valores do histórico como treinamento?
3. Como contrastar os resultados obtidos com as diferentes técnicas?

Desta forma nota-se que o problema abordado trata-se de aprendizado supervisionado, dado que temos arquivos de histórico para treinamento, assim podemos resolver tanto com técnicas de classificação como de regressão, neste caso implementaremos duas de cada, para contrastarmos os resultados.

2. Fundamentação Teórica

Para resolver os subproblemas previamente citados, optou-se por buscar a implementação de algoritmos de classificação e regressão. Os algoritmos de classificação utilizados foram a *random forest* e árvores J48. E os algoritmos de regressão utilizados foram *random forest* e rede neural.

A comparação das técnicas será feita usando métricas usuais para treinamento/validação e testes, como raíz quadrada do erro quadrático médio, *f-measure*, acurácia e matriz de confusão.

3. Metodologia

3.1. Modelagem Ambiente

Os arquivos de inicialização contendo os dados para treinamento e testes possuíam diversas informações, mas para estas tarefas consideramos apenas três dados como entrada:

- **qPA**: qualidade da pressão arterial, resulta da avaliação da relação entre a pressão sistólica e a diastólica;
- **pulso**: pulsação ou Batimento por Minuto (pulso);
- **frm**: frequência da respiração por minuto (respiração).

Assim estes mesmos dados foram considerados para os dois tipos de problema, sendo usados para classificar entre as classes de gravidade no problema de classificação, e para reconstruir o cálculo da gravidade nos problemas de regressão.

Antes desses dados serem usados para o treinamento, foi feito uma normalização para deixar os valores na faixa de 0 e 1. A normalização utilizada consistiu em subtrair o

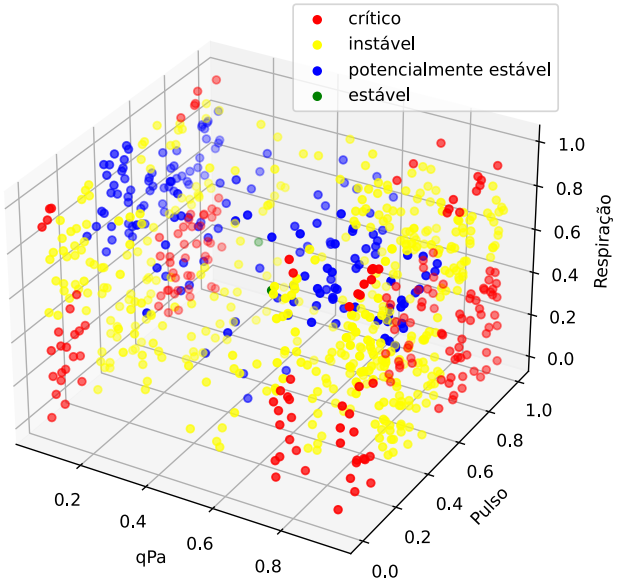

Figure 1. Dados históricos normalizados utilizados para treinar e testar os modelos

valor mínimo possível do dado e dividir pelo valor máximo possível subtraído pelo valor mínimo.

Além disso para fins de comparação foi decidido que 30% das vítimas seriam utilizadas para o treinamento dos modelos, e as outras 70% serviriam como os valores de teste de desempenho.

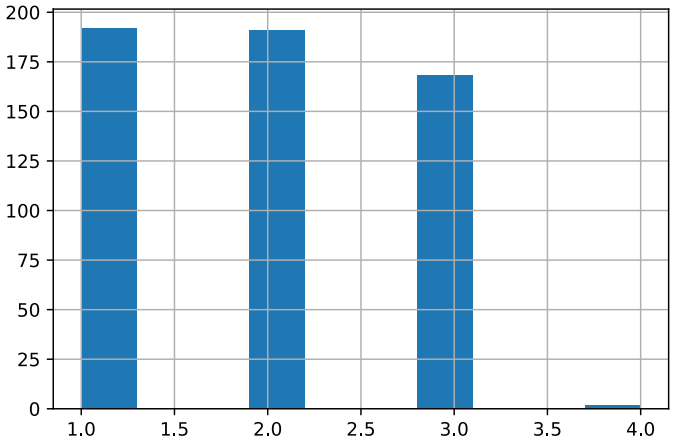

Figure 2. histograma de classes

Um problema observado nos dados para treinar os modelos foi a baixa quantidade de pontos com classe 4, no caso somente dois pontos de 800. Isso prejudicou a previsão dos modelos na qual a random forest e a rede neural para classificação não previam valores com classe 4.

3.2. Solução classificação - Random Forest

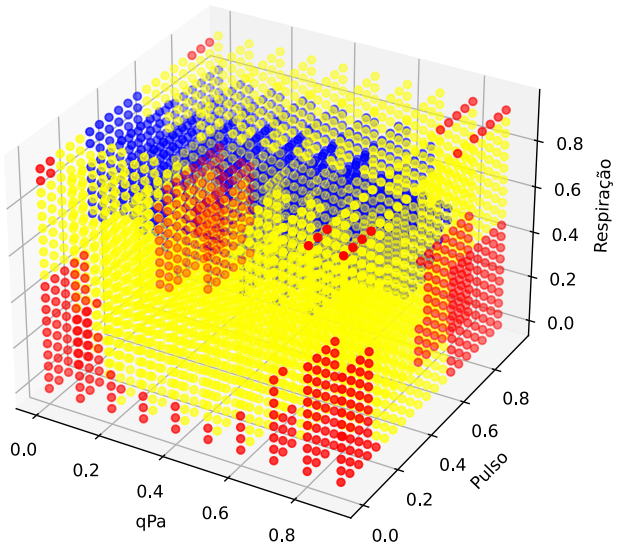

Figure 3. classificação da random forest

A classificação por random forest é feita pela previsão de várias árvores de decisão aleatórias. Cada árvore de decisão é formada por subconjuntos aleatórios dos dados. A função de agregação de todas as previsões é dado pelo valor que tem o maior número de ocorrências nas previsões das árvores de decisão. A implementação das árvores de decisão consiste em construir as melhores divisões dos dados para cada nó da árvore, que divide os dados em duas partes baseado no Coeficiente de Gini, este retorna um valor entre zero e um, onde zero significa a divisão perfeita e igual dos dados e um a divisão imperfeita. Sendo assim, testou-se todas as divisões dos dados para obter o menor Coeficiente de Gini possıvel. Na sequência é calculado o valor esperado encontrando a classe que mais aparece naquela divisão dos dados e é registrado este valor no nó. O número de divisões dependerá da altura, que é um valor fixo dado como parâmetro par a construção da árvore de decisão, parando assim a subdivisão da árvore quando este limite é atingido. Sabendo disso, a Random Forest é um conjunto de árvores de decisão treinadas com uma amostra aleatória dos dados e com alturas aleatórias.

A random forest teve um bom desempenho em classificação

3.3. Solução classificação - Árvore J48

O outro algoritmo de classificação escolhido foi a árvore J48, com o *Toolbox* Weka. A árvore J48 é uma extensão do algoritmo de árvores ID3, que possui características adicionais como *pruning*, consideração de valores perdidos, e, mais importante para este problema, consideração de valores contínuos por meio de faixas de valores.

A base do funcionamento do algoritmo J48 é a mesma do ID3, que é na construção de uma árvore de decisão, em que cada ramo constitui uma regra para a classificação, onde os nós são os atributos e a folha é a classe. Neste caso o tamanho da árvore e o conteúdo dos nós varia de acordo com a ordem que os atributos são testados e selecionados, já que é feita uma busca gulosa sobre qual atributo colocar de nó sem que haja *backtracing*.

O método usado para a seleção de qual atributo constituirá o nó é o conceito de entropia de informação, no qual se busca o atributo que causa a menor incerteza, e consequentemente exprime maior informação, sobre a seleção da classe. Assim para a construção da árvore são testados todos os atributos ainda não usados, e o que possuir a menor entropia é selecionado para ser o nó, então este processo é repetido até que todos os resultados do *dataset* sejam considerados.

Um conceito importante de se considerar ao implementar uma árvore de decisão é o *pruning*, que consiste em excluir da árvore os galhos que especificam demais as informações, a fim de construir um modelo com menor acurácia, porém mais generalizador.

Pruning é usado principalmente para impedir o *overfitting*, dado que caso a árvore tenha galhos muito específicos, a possibilidade destes galhos não classificarem corretamente os dados de teste é grande. As três formas mais comuns de *pruning* são: parar o algoritmo caso as amostras restantes sejam pequenas demais, limitar a profundidade da árvore para ela não ficar muito complexa, e não continuar com um ramo caso a diminuição do erro de classificação seja pequena demais.

3.4. Solução regressão - Random Forest

Este algoritmo tem em sua base muitas árvores de decisão aleatórias, na qual cada árvore é treinada com um subconjunto dos dados de treino escolhido aleatoriamente. Após todas as árvores serem criadas, o resultado da regressão é calculado pela média de todas as árvores.

Na solução do problema de regressão, este algoritmo teve um ótimo desempenho nas métricas de erro quadrado médio, além de ser muito mais eficiente no treinamento. Foram testados dois *Toolboxes* para treinar o algoritmo: o *Toolbox* do skitlearn e o *Toolbox* do Weka. Os dois produziram resultados semelhantes. Os parâmetros utilizados são de 2000 árvores de decisão e 100 iterações.

3.5. Solução regressão - Rede Neural

Para solucionar o problema, foi utilizado o modelo de rede neural. A rede construída que proveu uma boa capacidade de previsão (sem resultar em um *overfitting*) foi de $3x3$.

O algoritmo de aprendizado da rede é o *backpropagation*. A rede foi construída com 3 camadas escondidas, cada uma com três neurônios com função de ativação de sigmoide. A taxa de aprendizado utilizada foi de 0.1 com momento de 0.2. Foi Necessário um número de épocas bem considerável para os modelos encontrarem um ótimo local, por volta de 40 mil épocas. Mesmo assim, o ótimo local na maioria das vezes não é suficiente para superar as classificações da Random Forest.

O *Toolbox* utilizado para implementar a rede neural foi o *Toolbox* do Weka. Outros *Toolboxes* foram testados sendo o Tensorflow e o Pytorch, porém, o desempenho dos dois *Toolbox*s foi muito inferior ao Weka, entretanto estes possibilitam maior customização da rede.

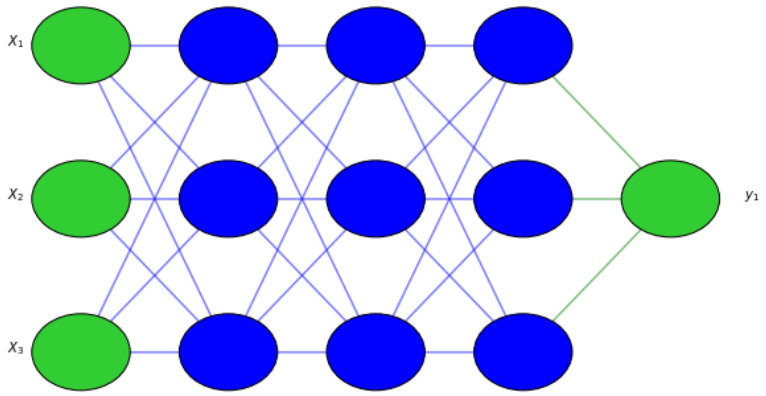

Figure 4. Rede neural utilizada

4. Resultados e análise

4.1. Resultados - Classificação

Para os algoritmos de classificação foram utilizadas as seguintes métricas para a comparação:

- *Precision*: a fração de elementos corretamente classificados como positivos sobre todos os que foram classificados como positivos;
- *Recall*: a fração de elementos corretamente classificados como positivos sobre todos os que são positivos;
- *F-measure*: calculado baseado nos dois anteriores: 2 * Precision * Recall / (Precision + Recall);
- *Acurácia*: porcentagem de acertos.

Para a obtenção destes resultados cada método foi executado 5 vezes e uma média aritimética foi executada sobre cada métrica de saída. Os resultados destas médias para os dois algoritmos estão no quadro abaixo:

Quadro 1: **Comparativo de resultado entre os métodos de classificação.**

Métodos	**Resultados**			
	Precision	Recall	F-measure	Acurácia
J48	0,877	0,875	0,874	0,875
Random Forest Classificação	0.9091	0.9091	0.9091	0.9091

4.2. Resultados - Regressão

Para os algoritmos de regressão foram utilizadas as seguintes métricas para a comparação:

- *RSME*: raiz do erro quadrado médio;
- *MSE*: erro quadrado médio;
- *MAE*: erro absoluto médio. Esta métrica cresce menos do que o erro quadrado se houver um erro muito grande em um único ponto;

Para a obtenção destes resultados cada método foi executado 5 vezes e uma média aritmética foi executada sobre cada métrica de saída. Os resultados destas médias para os dois algoritmos estão no quadro abaixo:

Quadro 2: **Comparativo de resultado entre os métodos de regressão.**

Métodos	**Resultados**		
	RSME	MSE	MAE
Rede neural	9.0326	73.0164	7.1260
Random Forest Regressão	5.5150	10.7855	2.1936

Para a regressão, o random forest se mostrou superior ao modelo de rede neural. Além disso, o random forest é um algoritmo mais rápido em complexidade de tempo do que a rede neural com *backpropagation*.

O modelo de rede neural teve dificuldade de aprender o padrão dos dados com a maioria das configurações testadas (de 1 a 5 camadas com de 3 a 5 neurônios em cada camada). A configuração encontrada que apresentou melhor desempenho com 40 mil épocas foi de $3x3$. Porém, faltou nesse trabalho, testar um número de épocas maior para os números maiores de camadas, pois estes demoram mais para descer o gradiente.

5. Conclusões

Consideramos que o trabalho atingiu o objetivo que era aplicar diferentes técnicas para fazer classificação e regressão, neste caso árvore J48 e *random forest* para classificação, e rede neural e *random forest* para regressão, a fim de fazer uma comparação dos resultados destes algoritmos.

Uma falha no nosso trabalho foi não encontrar uma otimização da rede neural que pudesse superar a *random forest*. Uma possibilidade seria treinar com um número grande de épocas, por volta de 150 mil, as redes com mais de 5 camadas.

Consideramos que uma próxima melhoria a fazer neste estudo seria a utilização de outros algoritmos, tanto de classificação como de regressão, por exemplo de sistema de inferência fuzzy, além de mais testes com outras parametrizações dos algoritmos já testados.

6. Referências bibliográficas

RUSSELL, Stuart J.; NORVIG, Peter. Inteligência artificial. Rio de Janeiro, RJ: Elsevier, 2004. 1021 p. ISBN 9788535211771.

7. Apêndice