Projeto 3 - Fundamentos em Sistemas Inteligentes

Renan Godoi de Medeiros
Matricula: 15/0146612

***Abstract*—O projeto consiste em aplicar o algoritmo SVM para classificar de forma supervisionada um conjunto de dados relacionados ao câncer de mama.**

I. Introdução

Este relatório tem por objetivo descrever o projeto da disciplina de Fundamentos em Sistemas Inteligentes e demonstrar as capacidades e limitações do algoritmo de SVM ou mais conhecido como Máquinas de Vetores de Suporte no âmbito de Sistemas Inteligentes.

Os tópicos foram divididos em 6 partes, o tópico II explica de forma clara os problemas que acarretaram no desenvolvimento do projeto assim como as dificuldades encontradas no mesmo.O tópico IV explica a razão pela qual tentei resolver os problemas citados em II.

Ainda no tópico III eu descrevo os materiais utilizados para a realização do projeto, assim como os métodos que eu utilizei no desenvolvimento do mesmo, em seguida, no tópico V é apresentada de forma clara como o projeto foi desenvolvido e também como ele foi dividido dentro do código do software. No tópico VI descrevo os resultados encontrados após uma série de testes e pesquisas a respeito.E por fim em VII apresento as minhas conclusões a respeito do trabalho, assim como as experiências que este trabalho me proporcionou.

II. Problema

Os problemas que acarretaram durante o projeto foram vários, desde a projeção de como seria feito o software até a sua implementação, pelo fato de existirem poucas informações sobre a utilização de validação cruzada em conjunto do algoritmo SVM, tive que fazer certas improvisações para realizar o trabalho da melhor maneira possível.

Tive problemas com o entendimento das metodologias requeridas na especificação do trabalho, pois era pedido que fosse feito um gráfico utilizando aparentemente um modelo svm e calcular vários erros em um intervalo de escala logaritmica, entretanto para um único modelo é difícil adiquirir uma variação significativa de resultados, por isso não obtive resultados tão variados, e também o outro problema foi medir os valores de C em escala logarítmica, pois para tal feito tive que realizar modificações nos parâmetros de eixo da função de geração de gráficos ao gerar os gráficos no software que construi.

A pesquisa também prejudicou o andamento do trabalho, a maioria dos exemplos que encontrei eram bastante simples, e as vezes variavam, pois na linguagem que utilizei, aparentemente existiam diversas formas e bibliotecas relacionadas ao algoritmo da SVM, por isso muitas vezes era um pouco complicado escolher qual era o mais adequado para se implementar este projeto.

Tive problemas também em relação a parametrização das SVMs, pois não ficou muito claro se os parâmetros precisavam ser variados ou não, por fim acabei optando por deixá-los em valores default com algumas exceções, como no caso da função svm do kernel de gauss em que mantive constante o valor de epsilon como havia previamente sido especificado na especificação do projeto.

III. Materiais e Métodos

Produzi este software utilizando a linguagem R, pois as funções e métodos já são disponibilizados para a utilização.Fiz uma pesquisa assídua sobre o assunto em vários fórums, sites e artigos disponíveis na internet, tanto em inglês quanto português.Utilizei uma máquina com Windows para a realização do trabalho, utilizei também a IDE RStudio para facilitar a construção do software, e modelei o programa com a ferramenta de modelagem Asta.

Também fiz uso do github para garantir caso ouvesse algum problema na minha máquina, utilizei a biblioteca e1071 e kernlab que dispõe das funções de SVM para classificação dos dados.

Dividi como foi requisitado, 70% para treino e 30% para teste, apesar de não ter encontrado exemplos para me basear de validação cruzada relacionados a este algoritmo, consiguir fazer o projeto baseando-me em projetos anteriores que já haviam sido construidos.

Utilizei a classe de diagnósticos dos pacientes, pois obtive melhores resultados comparando-se com outros atributos dentro do conjunto de dados da tabela, mesmo que o resultado não tendo sido tão satisfatório.

IV. Porque

As razões pelas quais quis resolver este problema foi meu interesse na implementação do projeto, pois de certa forma inteligência artificial implementada na prática é algo que não é visto muito no curso, e se formos analisar, praticamente inteligência artificial é utilizada em muitas áreas da tecnologia da informação, desde pequenos sites, até coisas mais complexas, como as novas arquiteturas de placas gráficas fabricadas pela empresa Nvidia a qual possui uma inteligência artificial responsável por ajudar a renderizar os gráficos de jogos e melhorar a imersão dos jogadores.

A existência de várias soluções para o problema deste projeto, foi um dos fatores que me incentivou em minhas escolhas, pois a Máquina de Suporte de Vetores possui uma

aplicabilidade imensa, logo é possível reunir muitas soluções para diferentes linguagens, inclusive linguagens que não são necessáriamente voltadas a estatística.

Apesar da Máquina de Suporte de Vetores possuir seus defeitos e talvez não ser a melhor escolha para a solução desse problema, já que existem muitos algoritmos capazes de resolver este problema de forma talvez até mais eficiente, a proposta deste trabalho é mostrar que as SVMs podem ser úteis de diversas formas.

V. Implementação

Para a implementação, fiz um reuso da arquitetura do projeto anterior, porém com algumas mudanças em algumas funcionalidades.

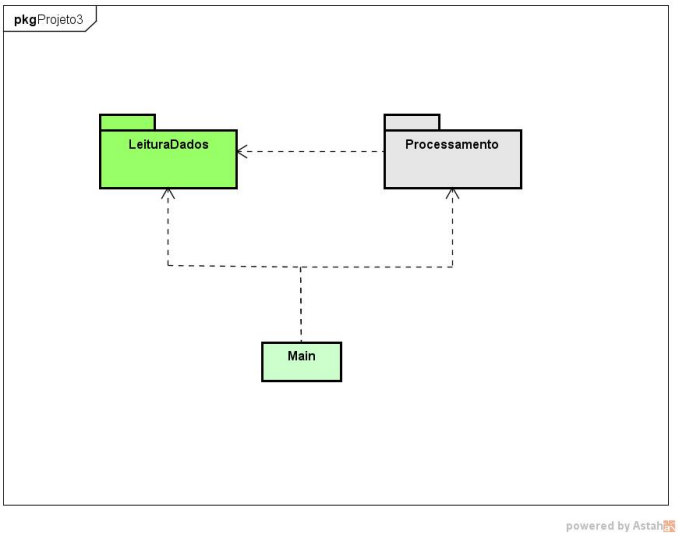

Fig. 1. Diagrama geral

A arquitetura deixei como estava no projeto anterior, a main depende do processamento e da leitura de dados, enquanto o processamento é dependente da leitura para que possa ser feita a análise de classificação do algoritmo.

Começando pela leitura representado na figura 2,assim como no projeto anterior o arquivo em formato .TXT contendo a tabela juntamente com os dados que serão lidos pelo leitor que fará a leitura de todos os dados contidos na tabela através da função *readfile()*, e retornará uma matriz correspondente com os dados lidos, contudo esta função retornará a matriz com as colunas nomeadas correspondentes aos atributos dos dados fornecidos que serão analisados pelo módulo de processamento.

Para o processamento dos dados na figura 3,ouve mudanças nas suas características eu descrevi três operações para os quais o algoritmo faria, que fora *ProcessarSVMKLinear(treino,teste)* basicamente responsável por processar o algoritmo SVM com Kernel Linear ,ou seja, o núcleo da máquina de vetores sendo linear, *ProcessarSVMKGauss(treino,teste)*, da mesma forma responsável por processar a máquina

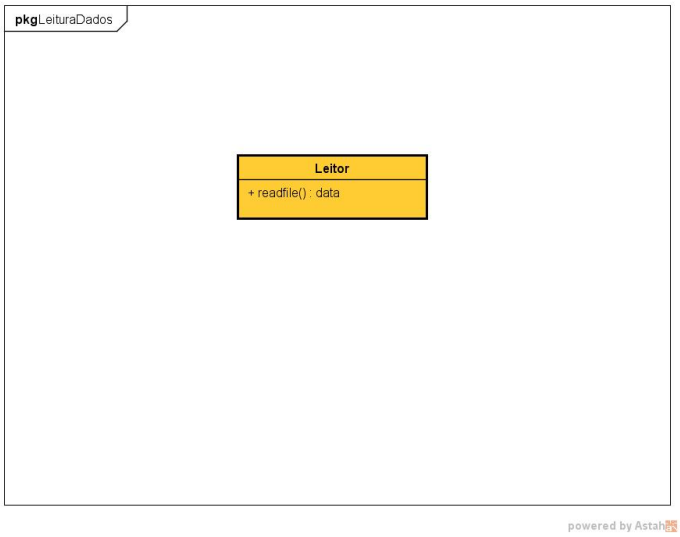

Fig. 2. Esquematização da leitura dos dados

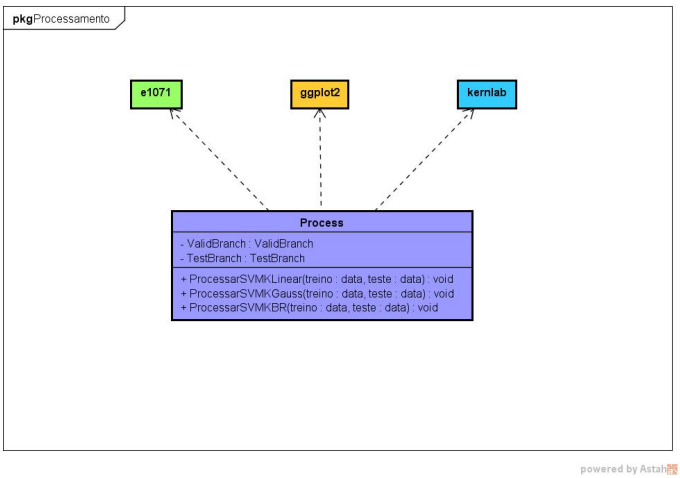

Fig. 3. Esquematização da processamento dos dados

no entanto com seu núcleo gaussiano e por fim *ProcessarSVMKFBR(treino,teste)*, que da mesma forma que as anteriores é responsável por processar a máquina de vetores entretanto com o núcleo FBR ou mais conhecido como Função de Base Radial, mas também pode ser chamado de RBF em inglês que seria Radial Basis Function que diferentemente das duas anteriores possui características para classificação diferenciadas,como por exemplo ao invés de observar os pontos do conjunto de dados apenas, este núcleo caracteriza os dados pela distância de vetores protótipos. Todos os métodos recebem um conjunto de dados para treino, e um conjunto de dados voltados aos testes nas SVMs.

Diferentemente do projeto anterior, este por sua vez herda 3 bibliotecas para a classe de processamento, aos quais possuem as funções *svm* e *ksvm* que foram essenciais para a construção do projeto.

VI. Resultados

Os resultados foram quase como o esperado, a taxa de erro produzida pelo algoritmo foi relativamente baixa para com núcleo linear, variando da casa dos 6% até 20% em quase todos os modelos criados para teste, como é possível observar na figura 4, onde mostra uma comparação entre os erros no teste dos modelos juntamente com os valores de C em escala logarítmica.

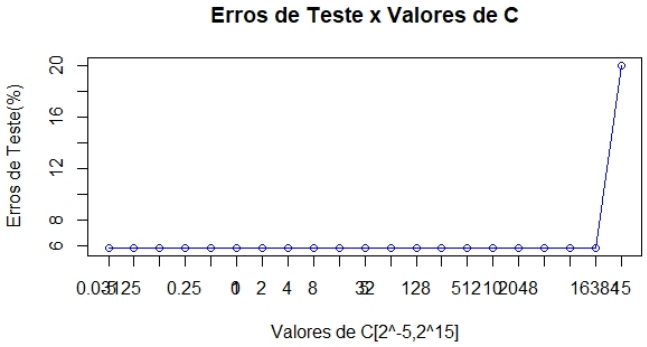

Fig. 4. Gráfico que representa a performance do algoritmo com kernel linear

o gráfico da figura 4, também mostra que ouve uma oscilação no modelo final e aparentemente o erro foi bem maior no último modelo testado que teve uma taxa de 20% aproximadamente, provavelmente devido a metodologia que utilizei para encontrar os erros de teste dos modelos criados.

Ao fazer o teste com kernel gaussiano, percebi uma diferença considerável na performance do algoritmo, e também nos resultados que acabaram sendo bem melhores que o kernel linear, chegando a uma taxa relativamente baixa, como pode ser observado na figura 5.

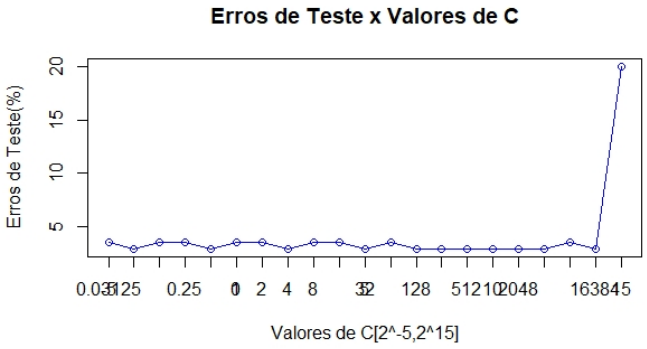

Fig. 5. Gráfico que representa a performance do algoritmo com kernel gaussiano

Entretanto diferentemente do gráfico com kernel linear, a menor taxa de erro que obtive foi menor que 5%, considerando que a menor taxa conquistada pela máquina de vetores com kernel linear foi de 6% podemos presumir que a precisão de acerto do algoritmo foi maior na maioria dos modelos testados, mesmo não tendo ficado com uma taxa tão constante quanto o kernel linear.

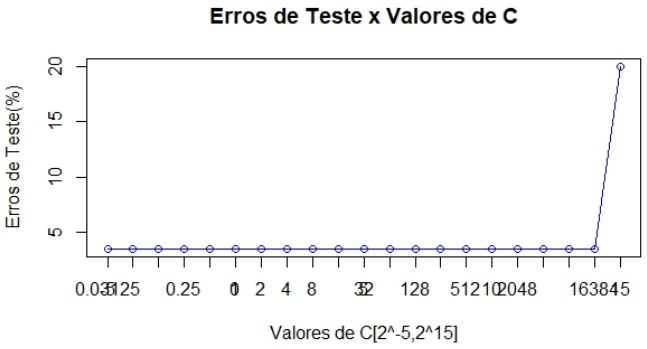

Fig. 6. Gráfico que representa a performance do algoritmo com kernel FBR

E finalmente no gráfico da figura 6, temos que em comparação com kernel gaussiano e o kernel linear, obteve aparentemente os melhores resultados, com uma taxa de erro abaixo de 5%,para a maioria dos modelos testados com exceção do último que teve o pior resultado assim como os outros, que inclusive teve o mesmo resultado para o último modelo testado, entretanto mesmo o último sendo o pior modelo, a média geral de erros de teste foi muito menor em comparação principalmente com o kernel linear que teve a pior média dos três métodos testados.

No entanto é possível observar que a svm do kernel linear esteve bem próximo de ser tão ótimo quanto o kernel FBR, sua taxa teve uma pequena variação, mas oscilou pouco em relação ao kernel FBR que obteve os melhores resultados possivelmente devido a função que utilizei para calcular a svm, pois utilizei uma função para melhorar a precisão da svm, oque pode ter resultado em uma taxa de acerto maior.

Apesar dos erros de teste com kernel gaussiano conseguir uma média de oscilações relativamente maiores do que os outros, foi oque mais se aproximou dos melhores resultados comparando-se com kernel FBR, pois o kernel gaussiano é teoricamente mais preciso que o kernel linear, entretanto pela minha classificação ser mais linear, pois eu utilizei o diagnóstico dos pacientes como fator classificatório, não faz muito sentido classificar a svm utilizando kernel FBR ou kernel gaussiano que são kernels não lineares, pois o custo de performance é relativamente pior para esses dois núcleos de svm em comparação com o kernel linear, mesmo que tenham uma precisão maior, sendo assim não há muita necessidade da utilização de um kernel não-linear para a realização desse tipo de classificação sendo que obtivemos resultados bastante semelhantes, logo o ideal para esta classificação seria a utilização do kernel linear, pois apesar de não ter alcançado os mesmos patamares dos outros kernels, obteve bons resultados e a melhor performance em comparação com os outros kernels testados, mesmo o kernel FBR tendo os melhores resultados.

VII. Conclusão

Neste trabalho abordei um assunto de grande importância em sistemas inteligentes, com o objetivo de mostrar o funcionamento do algoritmo de classificação de Máquinas de Suporte de Vetores, analisando os resultados obtidos, analisando a performance e se eram satisfatoriamente bons ou não.Também foi feita uma visão geral de como o projeto foi realizado,além dos softwares e IDE's que foram utilizados na sua construção.

Felizmente consegui cumprir todos os requisitos do trabalho, apesar das dificuldades encontradas em relação ao algoritmo, foi possível a sua realização graças não somente ao material disponibilizado na plataforma moodle da disciplina, mas também aos fórums e blogs de pesquisa e ajuda em relação ao assunto.

Este trabalho foi muito importante para o meu conhecimento, aprendi várias coisas em relação a máquinas de vetores de suporte .O projeto levou bastante tempo para ser construido, mas apesar das dificuldades foi interessante, a àrea de inteligência artificial é muito vasta principalmente quando pensamos em sistemas complexos.

References

[1] Gareth James, Daniela Witten, Trevor Hastle, Robert Tibshirani An Introduction to Statistical Learning with applications in R, 2014

[2] Machine Learning, Tom M. Michel, 1997

[3] Semisupervised learning. Encyclopedia of Machine Learning, Jerry Zhu, 2010.

[4] How To Select Support Vector Machine Kernels https://www.kdnuggets.com/2016/06/select-support-vector-machine-kernels.html