Universidade Federal de Pernambuco – UFPE

Centro de Informática – CIn

Aprendizagem de Máquina

Lista de Exercícios 2 - Relatório

Recife – 2019.1

Aprendizagem de Máquina

Lista de Exercícios 2 - Relatório

Relatório dos resultados obtidos realizando os experimentos definidos na lista

Escrito por:

Alexsandro Vítor Serafim de Carvalho (avsc@cin.ufpe.br).

Entrega do relatório: 21/04/2019

Sumário

1. Introdução

Este trabalho trata-se de um relatório dos resultados das atividades definidas na 2ª lista de exercícios da cadeira de Aprendizagem de Máquina. Ela consiste em avaliar o funcionamento de 3 versões do algoritmo de seleção de protótipos LVQ

As versões utilizadas são: LVQ1, LVQ2.1 e LVQ3. Cada algoritmo foi executado para gerar diferentes quantidades de protótipos (3, 5, 10, 15, 20, 25, 30, 35, 40). Os protótipos selecionados foram testados no classificador KNN, sem peso e com os valores de k = {1, 3}. Suas taxas de acerto, tempos de treinamento e tempos de teste foram coletadas para avaliar qual seria o mais adequado para cada dataset.

Neste exercício, foi usada a implementação do KNN da biblioteca scikit-learn no lugar da implementação da lista anterior.

2. Metodologia

Para avaliar a eficiência das variações do LVQ foi realizada uma validação cruzada (k-fold cross-validation). Foram formados 5 grupos com quantidades aproximadamente iguais de amostras de cada classe.

Cada versão do LVQ foi então executada uma vez para cada valor de k citado no tópico anterior x nº de protótipos a ser gerado x nº de grupos, com cada um dos grupos sendo usado como conjunto de teste enquanto o resto foi usado para treinamento. Foram calculadas médias das 3 estatísticas extraídas de cada iteração e elas foram plotadas em gráficos para observar o seu comportamento.

3. Tratamentos

Inicialmente, os protótipos do LVQ1 são amostras selecionadas aleatoriamente do conjunto de treinamento, com a garantia de pelo menos uma amostra de cada classe e de a mesma amostra não ser selecionada duas vezes como protótipo. No caso dos algoritmos LVQ2.1 e LVQ3, eles foram executados sobre os protótipos gerados no LVQ1.

As entradas de um dos datasets (JM1) foram tratadas substituindo valores ausentes pela média daquele atributo. Em ambos os datasets também houve um processo de normalização dos atributos para todos eles se manterem no intervalo de [0, 1]. As proporções de cada classe foram mantidas em cada grupo do k-folds.

4. Datasets

Os datasets utilizados são ambos do repositório promise, datasets KC1 e JM1.

O dataset KC1 contém 2 classes: true e false, com 1783 amostras false (84,54%) e 326 amostras true (15,46%), totalizando 2109 amostras.

O dataset JM1 contém 2 classes: true e false, com 8779 amostras false (80,65%) e 2106 amostras true (19,35%), totalizando 10885 amostras.

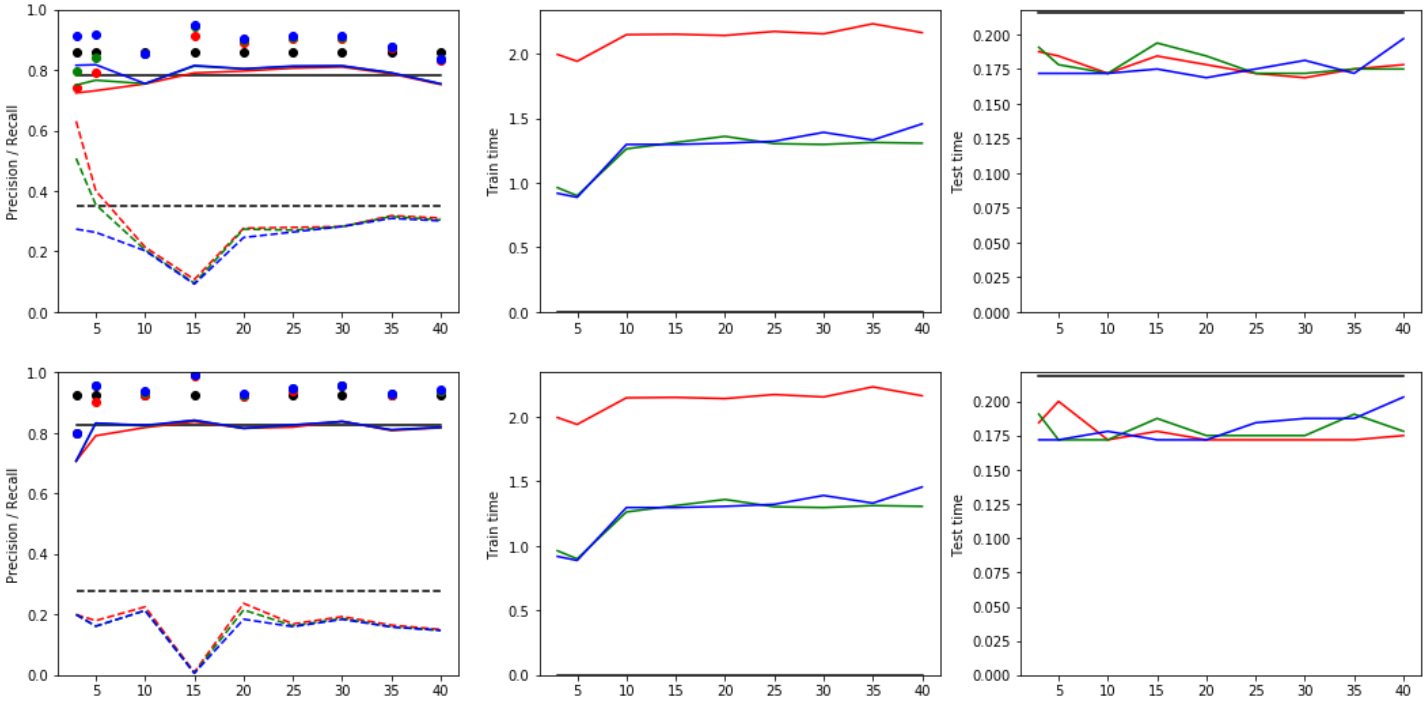
5. Resultados

Os resultados foram plotados em 6 gráficos. Em todos os gráficos o eixo x é a quantidade de protótipos gerada. Da coluna da esquerda para a direita o eixo y é: precisão / recall, tempo de treinamento e tempo de teste. Os gráficos da linha de cima são os testes com k = 1 e os de baixo com k = 3.

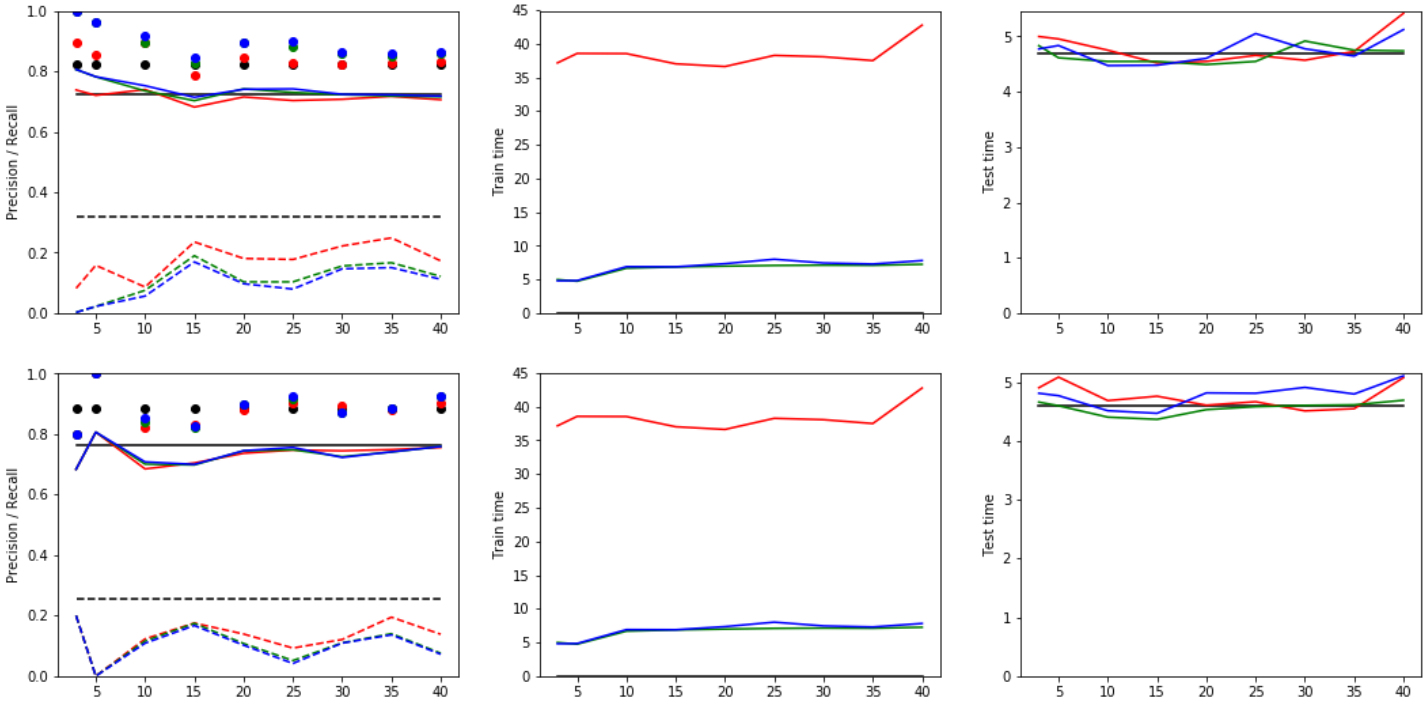
5.1. Legenda

- **Preto:** KNN sem seleção de protótipos.
- **Vermelho:** LVQ1
- **Verde:** LVQ2.1
- **Azul:** LVQ3
- **Tracejado (--):** Recall da classe minoritária (true)
- **Pontos (*):** Recall da classe majoritária (false)

5.2. KC1


Precision / Recall
Train time
Test time

5.3. JM1


Precision / Recall
Train time
Test time

5. Análise dos resultados

5.1. Tempo

O algoritmo KNN tende a ser rápido no treinamento, porém custoso na avaliação, já que o tempo de classificação é proporcional a quantidade de amostras usadas para treiná-lo. Sendo assim, a seleção de classificadores deveria tornar a avaliação mais rápida reduzindo o número de amostras no treinamento. Entretanto, essa diminuição foi pouco significativa no KC1 e inexistente no JM1. Possivelmente, o KNN implementado no scikit-learn possui alguma forma de otimizar a classificação das amostras para que ela não seja tão impactada pelo número de amostras no treinamento.

Dentre os LVQs, o LVQ1 é o mais lento, possivelmente porque ele sempre move os protótipos, enquanto os LVQs 2.1 e 3 nem sempre os movem devido à regra da janela. Como esses dois últimos são executados sobre os protótipos gerados pelo primeiro, isso implica no uso deles levar mais tempo que o primeiro, mas ainda assim não impactar muito o desempenho em tempo.

5.2. Acertos

Conforme a quantidade de protótipos aumenta, a precisão tende a se aproximar da precisão do KNN sem seleção de protótipos. O mesmo também ocorre com os recalls, que no caso da classe minoritária tendem a se manter abaixo do resultado sem a seleção de protótipos e se aproximam dele com um número maior deles. Usar k = 1 trouxe recalls maiores para a classe minoritária, tanto com quanto sem a seleção de protótipos.

5.3. Conclusão

A seleção de protótipos tem como principal benefício o tempo de execução, e ela foi pouco efetiva no mesmo. No KC1 os protótipos correspondiam a 0,71% (3 protótipos) a 9,48% (40 protótipos) da quantidade de amostras, mas o tempo de teste não diminuiu na mesma proporção. No JM1 a proporção é ainda menor (0,13% a 1,84%) e não houve diminuição significativa no tempo de classificação.

Como a técnica não demonstrou benefícios em seu principal propósito e ainda trouxe o ônus de aumentar enormemente o tempo de treinamento (que sem a seleção de protótipos é desprezível), ela demonstrou ser ineficiente para otimizar a classificação, pelo menos com o uso dos algoritmos testados.