Universidade Federal de Pernambuco – UFPE

Centro de Informática – CIn

Aprendizagem de Máquina

Lista de Exercícios 1 - Relatório

Recife – 2019.1

Aprendizagem de Máquina

Lista de Exercícios 1 - 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: [Data de entrega]

Sumário

1. Introdução

Este trabalho trata-se de um relatório dos resultados das atividades definidas na lista de exercícios da cadeira de Aprendizagem de Máquina. Ela consiste em avaliar o funcionamento de 3 modelos de KNN para 2 datasets diferentes.

Os modelos utilizados são: KNN básico (sem pesos), KNN com peso e KNN adaptativo. Cada modelo foi testado com diferentes valores para k: 1, 2, 3, 5, 7, 9, 11, 13 e 15. Suas taxas de acerto, tempos de treinamento e tempos de teste foram coletadas para avaliar qual seria o mais adequado para cada dataset.

2. Metodologia

Para avaliar a eficiência das variações do KNN 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 KNN foi então executada uma vez para cada valor de k citado no tópico anterior 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.

Nenhum tratamento foi realizado nas bases de dados além de manter as proporções de cada classe em cada grupo do k-folds.

3. Datasets

Os datasets utilizados são ambos do repositório promise, datasets CM1 e KC1.

O dataset CM1 contém 2 classes: true e false, com 449 amostras false (90,16%) e 49 amostras true (9,84%), totalizando 498 amostras.

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

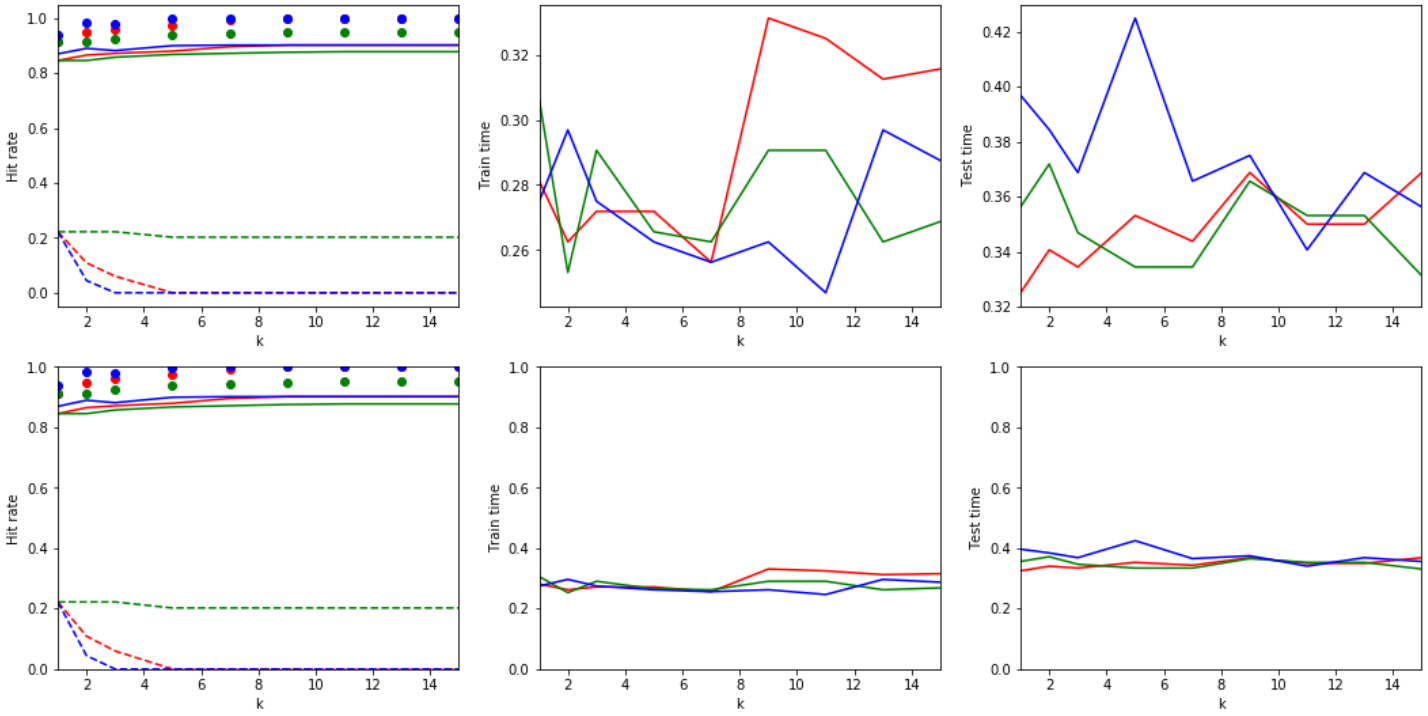
4. Resultados

Os resultados foram plotados em 6 gráficos, 2 para cada estatística. Os gráficos de cima são iguais aos de baixo porém com escalas diferentes, os de baixo tem escalas começando em zero. Da esquerda para a direita são os gráficos de: taxa de acerto, tempo de treinamento e tempo de teste.

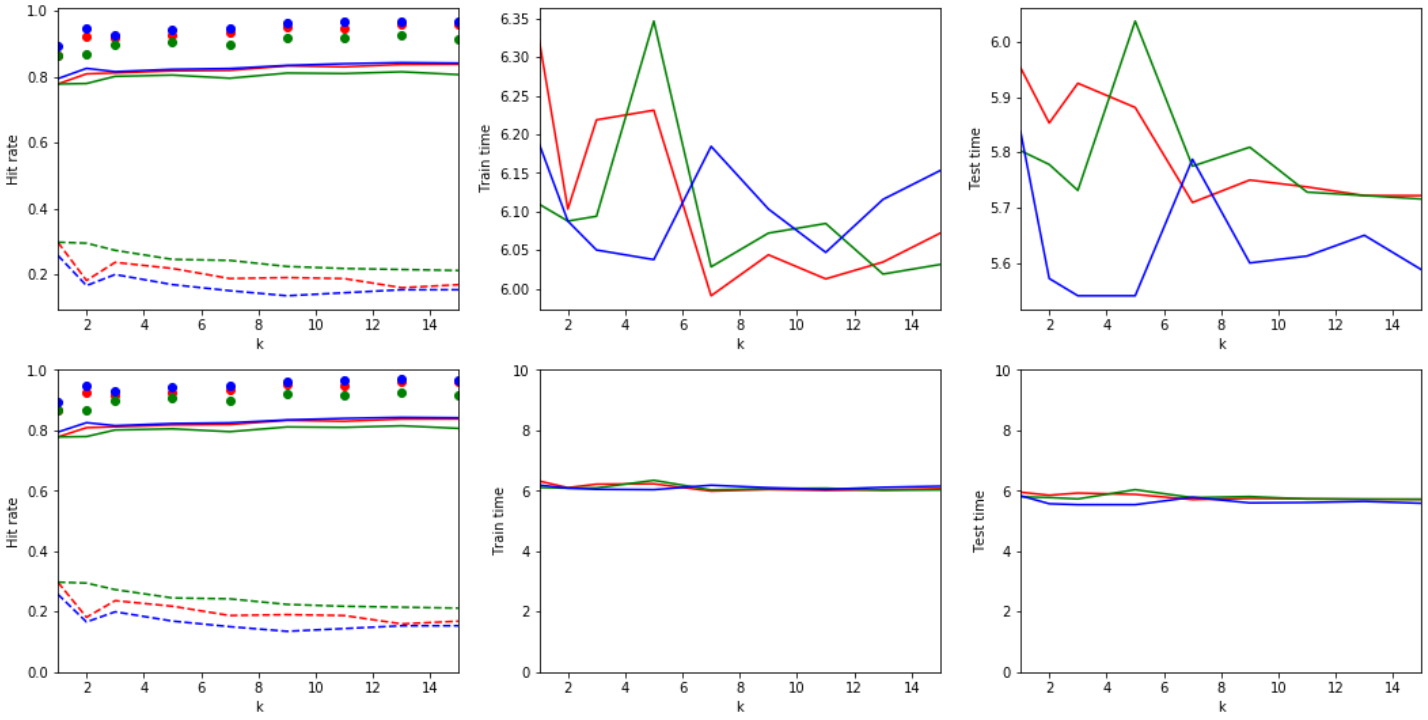
Cada cor representa um modelo de KNN: **sem peso**, **com peso** e **adaptativo**. No gráfico da esquerda (taxa de acerto), os segmentos são as médias das taxas de acertos enquanto os pontos são os acertos na

classe majoritária (false em ambos os datasets) e os segmentos tracejados são os acertos na classe minoritária.

4.1. CM1



4.2. KC1



5. Análise dos resultados

Os tempos de treinamento e teste foram bastante similares entre as 3 variações, com diferenças desprezíveis entre elas. Apesar disso, é possível observar que estão próximos. Em um algoritmo como o KNN espera-se que o treinamento seja mais rápido que o teste, pois o treinamento consiste apenas em armazenar amostras, enquanto as operações no geral ocorrem na classificação. Entretanto, na implementação utilizada, o cálculo dos raios para as distâncias adaptativas é feito no treinamento. Como esse cálculo só é usado no modelo adaptativo, o treinamento dos outros 2 poderia ser mais rápido se ele fosse removido. Além disso, o treinamento é realizado com 4 vezes a quantidade de amostras do teste, então o tempo de execução por amostra do treinamento é aproximadamente ¼ do teste, confirmando a hipótese de que o treinamento é mais rápido.

O valor de k não demonstrou efeito significativo no teste. Isso foi esperado devido a implementação utilizada para a seleção das k amostras mais próximas: uma ordenação das mesmas por ordem crescente de distância seguida de uma seleção das k primeiras amostras da lista ordenada gerada. O tempo de execução dessa implementação não é afetado pelo valor de k.

Em relação a taxa de acerto, os KNNs sem peso e adaptativo tiveram as maiores taxas de acerto, tendendo a crescer e convergir para altos valores de k. Mas é importante lembrar que uma taxa de acerto geral maior não necessariamente implica em um classificador melhor. Como os datasets estavam desbalanceados e com poucas classes, era fácil conseguir altas taxas de acerto simplesmente classificando todas as amostras com a classe majoritária.

Esse fenômeno foi observado em um teste do primeiro dataset com 10 grupos no k-fold. Como a classe minoritária possuía apenas 49 amostras, cada grupo ficou com 4 ou 5 amostras, que eram sempre minoria nas classificações com k = 11, 13, ou 15. Isso fazia com que todas as amostras fossem classificadas como o grupo majoritário.

Por isso, os gráficos de taxa de acerto tiveram os recalls de ambas as classes incluídas neles. Com isso, foi possível confirmar que com o aumento do k, a classe minoritária passava a ser menos classificada, consequentemente, o recall da mesma caia. No KC1, os recalls da classe minoritária em todos os modelos tendiam a cair, embora o modelo com peso tenha sempre se mantido com um recall maior que o sem peso que sempre se manteve com um recall acima do adaptativo. No CM1, a partir de k = 5, o recall do KNN com peso se manteve em 20% e os outros 2 zeraram, tornando os modelos inúteis.

Minha conclusão final é que o melhor modelo em ambos os casos, é o KNN com peso. Apesar de sua taxa de acerto mais baixa, ele teve os maiores recalls para a classe minoritária. Considerando os contextos dos datasets (presença de defeitos em softwares), é importante que os defeitos existentes sejam encontrados. Também por isso, considero que mesmo esse modelo é ineficiente para os datasets. Os recalls da classe minoritária nunca estiveram acima de 30%, o que significa que 70% dos defeitos não seriam encontrados, independente do que fosse feito.

Quanto ao k, como ele influenciou negativamente o recall, k = 1 pode ser o valor mais interessante. k = 3 produziu taxas de acerto ligeiramente maiores, mas acredito que isso não compensa nem a redução do recall em KC1, nem as possibilidades de otimização na classificação, principalmente a possibilidade de substituir a ordenação por um min na seleção da classe, que tem complexidade menor.