Mineração de dados

Trabalho prático: Classificação

1- Descrição do problema:

O IMDb é a maior base de filmes, seriados, programas de televisão e semelhantes existente na internet. Sua reputação vem do fato de ela possuir não apenas dados técnicos sobre os filmes, mas também de os usuários poderem postar comentários e dar notas a cada filme da base, auxiliando assim outras pessoas a decidirem se vale ou não a pena assistir o filme em questão.

Sua base contém milhares de *reviews*, e é atualizada constantemente. Entretanto, como os *reviews* deixados pelas pessoas são textos, ela precisa saber se eles são positivos ou negativos em relação ao filme. Isso é um grande problema, dada a enorme quantidade de dados, e logo não pode ser feito manualmente. Mas classificar os *reviews* é extremamente importante para sistemas de recomendação de filmes, e não há informação mais valiosa para eles que extrair diretamente o que os próprios usuários estão comentando sobre os filmes que assistiram.

Para resolver esse problema, o IMDb forneceu a você uma amostra da base de dados deles contendo mais de mil *reviews* diferentes, que á foram classificados como positivos ou negativos, para servir como treino. Ele entregou ainda outra amostra, que é de *reviews* ainda não classificados. Cada *review* é formado por um texto (conjunto de palavras) que expressam o sentimento do usuário em relação a um filme em questão.

Para isso, sua tarefa é classificar os novos *reviews* que chegam no sistema do IMDb diariamente.

2- Avaliação

Você deve implementar o algoritmo KNN (K-Nearest Neighbors) de forma a classificar as entradas do arquivo de teste.

Você deve ainda resolver uma forma de escolher o melhor k para a base. Cuidado com *overfitting*!

Compare ainda os resultados gerados com e sem a utilização de *stopwords*.

Analise a qualidade das soluções geradas usando precisão e revocação, e qualquer outra métrica de acurácia que você julgar importante/necessária.

3- Formato

O arquivo entregue contém uma pasta com os arquivos de treino, teste e uma lista de stopwords.

3.1- Formato de entrada

O programa receberá dois arquivos de entrada: um para treino e outro para teste. O formato de ambos será o mesmo: cada linha consistem em um instância na forma:

<classe> <uma ou mais palavras separadas por espaço>

Exemplo de treino:

1 filme muito bom 1 excelente atuação dos atores

```
0 nunca vi filme pior em toda minha vida
0 lamentável ,  muito ruim mesmo
1 brilhante !
```

E de teste:

```
0 filme muito ruim
1 o melhor filem da minha vida !
```

Note que apesar de o arquivo de teste também conter as classes reais de cada instância, essa informação não deve ser levada em conta na hora da classificação, mas apenas para avaliação da mesma.

O programa deve também receber como entrada o número k de vizinhos mais próximos a serem analisados a fim de gerar a comparação. O formado de execução deve ser o seguinte:

```
$> comando –i <arquivo de treino> -t <arquivo teste> -k <número de vizinhos>
```

Você pode incluir outros parâmetros caso sinta necessidade. Entretanto, eles devem ser opcionais, logo, escolha um valor default para que, apenas com o comando acima seu programa seja executado normalmente. Não esqueça de especificar os novos parâmetros em um arquivo README.txt

3-2 Formato de Saída

Para cada instancia do teste, imprimir a qual classe foi atribuída pelo classificador, uma por linha. Para os exemplos de entrada acima, usando k=3, a saída deveria ser:

```
0
1
```

4- O que entregar:

- Código com a implementação (C++ ou Python). Neste trabalho não é permitir utilizar bibliotecas de mineração de dados.
- Arquivo README.txt: contendo, resumidamente, os comandos necessários para a execução do seu programa.
- Um arquivo em formato pdf, abordando, pelo menos, os seguintes pontos:
    - Uma breve descrição do algoritmo implementada;
    - Uma análise de como o algoritmo se comporta quando variamos os parâmetros e tamanho da entrada;
    - Discussão sobre a forma de escolha do k;
    - Análise da base de dados fornecida;
    - Discussão sobre a qualidade da solução com e sem o uso de stopwords.