Aprendizagem Automática

Trabalho Laboratorial – grupos de 3 alunos

Classificação de Críticas de Cinema no IMDb

1º Semestre de 2016/2017

Objectivos do trabalho: Este trabalho lida com a análise de dados de texto, em particular críticas de cinema do IMDb, e está dividido em duas principais tarefas:

1. **Classificação:**
   Aqui pretende-se determinar se uma crítica de cinema e positiva ou negativa, baseado no texto da crítica. Esta parte é composta das seguintes etapas:

   (a) Projectar um discriminante logístico avaliar o desempenho do mesmo para diferentes dimensões do vocabulário.

   (b) Projectar um classificador dos k-vizinhos mais próximo, e e testa-lo com duas métricas de distância: Euclidiana e de cosseno.

2. **Regressão:**
   Projecte um modelo de regressão linear para prever a pontuação de uma crítica num escala 1 a 10 valores baseado no texto da mesma. Adapte o modelo de regressão para classificar a crítica em positiva ou negativa e compare os resultados com os dos outros classificadores testados.

Dados: A IMDb, a Internet Movie Database, é uma base de dados que consiste em textos de críticas de cinema, recolhidas por Andrew Mass [1], e que se encontra disponível em `ai.stanford.edu/ amaas/data/sentiment/`. A ficheiros nesta base de dados encontram-se guardados em duas directorias de topo, `train/` com os dados de treino, e `test/`, com os dados de teste. Por sua vez, em cada uma destas directorias encontram-se duas sub-directorias `pos/` com os exemplos positivos e `neg/` com os negativos. A divisão das críticas em duas classes, positivas e negativas, é para a tarefa de classificação. Para a tarefa de regressão, a informação sobre o valor da pontuação (rating) encontra-se no nome do próprio ficheiro. Na tarefa de regressão é necessário extrair esta informação dos nomes dos ficheiros para criar a "matriz" $\mathbf{Y}$ de $1 \times N$, com as saídas desejadas (onde N é o número de documentos).
(Para mais informação sobre esta base de dados, ler o ficheiro `README` disponibilizado com a mesma, e para carregar a base de dados em ambiente Python, consultar os acetatos da disciplina sobre esta matéria).

Metodologias de Teste: O desempenho dos algoritmos projectados deve ser avaliado com base nos resultados obtidos no conjunto de teste. Para as tarefas de classificação deve reportar a

probabilidade de erro (ou acerto) e a matriz de confusão, mas também deve usar outras métricas de desempenho utilizadas em problemas de classificação binária. Para modelos de regressão, a métrica de desempenho é a potência do erro (pode também usar o coeficiente R^2 - método da classe `LinearRegression`). Deve também converter o modelo de regressão num classificador e comparar os resultados com os dos outros classificadores projectados.

Etapas do Trabalho: Para diferentes etapas do trabalho enumeradas no início do enunciado, tenha em conta o seguinte:

- Para poder trabalhar com dados de texto é primeiro necessário representar cada documento por um vector numérico. Esta é a representação "Bag of Words", em que os coeficientes devem ser obtidos através do método tf-idf. As classes do `scikit-learn`, `TfidfTransformer` e `TfidVectorizer` calculam esta representação e devolvem para um dado corpus, a matriz documento-termo. No entanto convém primeiro fazer uma limpeza dos documentos para remover símbolos de formatação (ex: mudanças de linha HTML), caracteres não alfabéticos, escolher termos com mais que um determinado número de caracteres e que apareçam em n ou mais documentos, fazer "stemming" ou outras técnicas que achar pertinentes. Todo o processo de limpeza deve ser descrito em detalhe.

1.(a) Nesta etapa, o objectivo é aferir o desempenho de um discriminante logístico para diferentes dimensões do vocabulário. Para controlar a dimensão do vocabulário pode usar o parâmetro `max_features` da função `TfidfVectorizer`. Pode também controlar o tamanho do vocabulário limitando o número mínimo de documentos em que um termo aparece (parâmetro `min_df`). Deve usar o discriminante logístico do `scikit-learn`: classe `LogisticRegression` que se encontra no módulo `linear_model`. Deve igualmente testar vocabulários com as seguintes dimensões: 5000, 10000, 15000, e 20000 (mas pode também testar outros valores). Adicionalmente, deve averiguar qual a dimensão mínima do vocabulário para a qual o desempenho dos classificadores seja ainda comparável aos resultados obtidos com vocabulários de maior dimensão.

1.(b) Nesta parte, pretende-se testar o classificador k-vizinhos mais próximos (kNN) usando duas métricas de distância, a Euclidiana e a de cosseno. Nas experiências relativas a esta parte, não teste diferentes dimensões do vocabulário. Escolha o vocabulário com a dimensão que achar apropriada. Teste os classificadores com os dados obtidos com a representação tf-idf, e teste igualmente com estes dados pré-processados de modo a terem média nula. Para ambas as métricas de distância, escolha o número de vizinhos que achar apropriado. Compare os resultados, com os obtidos na alínea anterior.

2. Nesta parte é projectar um modelo de regressão linear para prever a pontuação de uma crítica. As pontuações de cada documento estão no nome dos ficheiros `.txt`. Deve fazer um programa para extrair esta informação dos ficheiros de texto para um `NumPy` array.

Projectar o modelo de regressão equivale a estimar um vector $\mathbf{w}$ de dimensão $d+1$, onde d é a dimensão do vocabulário. O cálculo de $\mathbf{w}$ é obtido através da equação $\mathbf{w} = \mathbf{R}_{\mathbf{x}}^{-1}\mathbf{r}_{\mathbf{x}y}$. Note porém que na representação tf-idf, os dados podem ter uma dimensão muito elevada (d a dimensão do vocabulário é geralmente na ordem dos milhares), e isto inviabiliza o cálculo da matriz $\mathbf{R}_{\mathbf{x}}$ e da sua inversa. Por isso nesta tarefa, utilize um vocabulário com a dimensão menor possível mas que ainda obtenha desempenhos aceitáveis (abordado na alínea 1.(a)). A dimensão deste vocabulário pode ainda ser demasiado elevada, e assim sendo, deve pré-processar os dados com PCA, especificando todos os passos efectuados, como o número de componentes escolhidos, se foram testados vários números de componentes, se os dados foram branqueados, etc. Pode igualmente usar os modelos de regressão linear do `scikit-learn` que se encontram no sub-módulo `linear_model` como é o caso da classe `LinearRegression` - note que neste caso pode não ser necessário pré-processar os dados com PCA. Por fim converta os modelos de regressão num classificador para determinar se a crítica é positiva ou negativa, e compare os resultados com os obtidos com outros classificadores.

Elaboração do Relatório:

- O relatório terá no máximo 6 páginas. Não é necessário haver uma página de capa, nem um índice do relatório. No entanto este dever ser bem estruturado, ter uma introdução, uma conclusão, e uma descrição das experiências efectuadas e dos resultados obtidos. Os membros do grupo devem estar claramente identificados no início do relatório.
- A descrição das experiências feitas deve seguir a ordem dada neste enunciado. Para cada teste, deve brevemente descrever como foi obtida a matriz documento-termo, como foi limpo o vocabulário, o classificador usado, etc. Deve igualmente, reportar detalhadamente os resultados obtidos.
- Tendo em conta o espaço limitado do relatório, é preferível sempre que possível, descrever os resultados através de gráficos.
- Não inclua no relatório o código implementado.
- Deve comentar os resultados obtidos e as possíveis causas para os (bons/maus) desempenhos, e quando achar pertinente, complementar o seu raciocínio com gráficos ou imagens.
- Inclua na bibliografia todo o material consultado para elaborar o relatório.
- IMPORTANTE: Entregar unicamente o ficheiro do relatório. Terá que ser um ficheiro ".pdf" com o nome: `TP2_Axxxx_Axxxx_Axxxx.pdf`
  onde os "`Axxxx`" correspondem aos números de aluno dos 3 membros do grupo. Colocar os número em ordem crescente.

Referências

[1] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.