Atividade 6 - Redução de Dimensionalidade para Classificação via PCA

Prof. Dr. Juliano Henrique Foleis

Descrição da Atividade

Nesta atividade você vai investigar o efeito da redução de dimensionalidade em um problema de classificação via PCA em um dataset esparso. Sua implementação deve ser feita em Python em um caderno no Jupyter.

Nesta atividade vamos trabalhar com um subconjunto da base de dados "Dota 2 Games Results". Este subconjunto da base de dados consiste em dados de 9264 partidas de Dota 2. Você deve fazer um classificador para predizer qual dos dois times vencerá a partida. São 116 características por partida. Na ordem, estas características são:

- ID do Cluster (relacionado à localização geográfica do servidor)
- Modo de jogo
- Tipo de jogo (ranqueado, etc)
- O restante das colunas indicam quais heróis participaram da partida. O valor 1 indica que o herói estava na equipe 1, e -1 indica que o herói estava na outra equipe. Como cada partida é 5x5, cada partida tem 5 heróis marcados com 1 e 5 heróis marcados com -1. Este link contém os heróis correspondentes na sequência das colunas do dataset.
- Equipe vencedora (1 ou -1) - **variável de saída**.

Note que a grande maioria dos dados serão iguais a zero, uma vez que das 116 colunas apenas 13 serão diferentes de zero! Neste caso dizemos que o dataset é esparso. Nestes casos a redução de dimensionalidade tende a funcionar bem, até melhorando o desempenho de alguns classificadores em alguns casos.

Documente cada um dos passos indicados a seguir no Jupyter:

1. Visualize o espaço formado de características usando todos os atributos. Use PCA para reduzir a dimensionalidade.
2. Avalie o desempenho do classificador KNN usando validação cruzada em dois níveis, conforme discutimos nas aulas de otimização de hiperparâmetros. A validação cruzada no primeiro deve ser em 10 vias, enquanto no segundo nível deve ser em 5 vias. A validação cruzada no segundo nível deve selecionar a melhor combinação de hiperparâmetros. Os parâmetros a serem otimizados são os mesmos que estudamos em sala de aula. Utilize a métrica *macro f1-score* para avaliar o desempenho do classificador. Imprima os resultados usando a função `classification_report`. Imprima a soma das matrizes de confusão dos folds com a melhor combinação de parâmetros. Use a função `do_cv` que está em utils.py para facilitar!
3. Avalie o desempenho do classificador `SVM` usando validação cruzada em apenas um nível. Neste caso, use apenas um conjunto de validação para encontrar a melhor combinação de C e *gamma* (γ) de acordo com o que vimos na aula sobre SVM. Use o kernel `rbf`.
4. Avalie o desempenho do classificador `Random Forest` usando validação cruzada em dois níveis, da mesma forma que no item *2*. Os parâmetros a serem otimizados são os mesmos que estudamos em sala de aula (Tópico 8).

5. Utilize PCA para reduzir a dimensionalidade dos vetores de características. Avalie novamente os classificadores da mesma forma que nos itens 2, 3 e 4 acima, mas desta vez ao invés de usar as colunas originais do dataset, você vai usar apenas as projeções obtidas com PCA. Avalie os resultados obtidos com projeções para 2, 8, 16, 32 e 64 dimensões. A função `do_cv` que está em utils.py implementa esse procedimento de redução de dimensionalidade pelo parâmetro `dim_red`. Veja o caderno no github para ver como usar esta funcionalidade.
6. Imprima os resultados de todos os testes em sequência para observar a diferença no desempenho dos sistemas que usaram redução de dimensionalidade em relação aos sistemas que não utilizaram.
7. Faça o teste da hipótese nula (pelo Teste-T) entre todos os resultados obtidos e os resultados obtidos sem redução de dimensionalidade. Para isso, use a função `print_t_tests` que está em utils.py. Neste passo a saída será uma tabela, onde as linhas representam os resultados de todas as combinações avaliadas e as colunas representam os resultados sem a redução de dimensionalidade com SVM, KNN, e RandomForest.
8. Usando os dados exibidos nos itens **6** e **7** acima, indique e justifique quais foram os melhores resultados obtidos. Identifique e justifique também os piores resultados obtidos. Você notou algum padrão interessante? Quais? Responda estas perguntas em uma célula no final do notebook.

Instruções e Entrega

- A maioria dos passos acima estão prontos nos cadernos das Semanas 4–9 disponibilizados no GitHub.
- Capriche no seu *notebook*: coloque textos explicativos, faça gráficos que julgar necessário, etc. Aproveite para aprender como usar as ferramentas!
- A atividade deve ser feita em um Jupyter Notebook. Você pode usar o *Google Colab* se quiser, mas é necessário entregar o arquivo *.ipynb*. Caso coloque código em arquivos *.py* por favor entregar junto com o *.ipynb* em um arquivo *.zip*.
- A entrega deverá ser realizada via Moodle, na *Atividade 6*.
- O trabalho é individual.
- Não é permitido alterar o arquivo que contém a base de dados (**dota2DatasetTrain_header_01.csv**)!

BONS ESTUDOS!