Universidade Tecnológica Federal do Paraná
Campus Campo Mourão
Departamento de Computação - DACOM
Prof. Dr. Diego Bertolini
Disciplina: BCC35-G - Inteligência Artificial

Conteúdo: SVM
Data de Entrega: 14/05/2023

---

1) Baixe e instale a libSVM no seu diretório. Disponível nos links abaixo: *http://www.csie.ntu.edu.tw/~cjlin/libsvm/*

2) Compile o código para gerar os executáveis (Verifique se o gnuplot está instalado!)
apt-get install gnuplot
make all

3) Em caso de dúvidas, leia o documento disponível no link abaixo:
http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf

4) Baixe os conjuntos de treinamento e teste para este e outros experimentos (.SVM): download

5) Analise as bases de treinamento e teste e responda (ver os arquivos treino.SVM and teste.SVM):
a) Número de classes ; _____
b) Número de Instâncias no Treinamento ;_____
c) Número de Instâncias no Teste ;_____
6) Execute o script python que acompanha a libSVM, chamado easy_aula.py (copiar para o diretório tools). Esse script faz a busca pelos parâmetros do kernel Gaussiano (g) e da variável de custo (C). O script gera alguns arquivos. (>python2.7 easy_aula.py treino.svm teste.svm)
Liste quais são esses arquivos gerados:
_____
_____
_____
_____

6.1) Reporte a acurácia através do experimento acima._____
6.2) Reporte o número de vetores de suporte encontrados para cada classe e o total;
(NSV: _____)

6.3) Utilize o conjunto de treinamento para treinar um modelo e o mesmo para testar (>python2.7 easy.py train.svm train.svm). Descreva a taxa de acerto:_____

6.4) Inverta, utilize o conjunto de teste como treinamento e o de treinamento como teste (>python2.7 easyDiego.py test.svm train.svm). Descreva a taxa de acerto:_____

Segunda Parte:

Os dados dos arquivos “treino.svm” e “teste.svm”, são descritores de dígitos 0-9 da MNIST.
Utilizando o libSVM avalie:

Usando o scikit-learn(dados)

1) Taxa de acerto usando o easy.py (configuração padrão usando RBF) ;
2) Taxa de acerto usando modelos com outros kernels:
-t kernel_type : set type of kernel function (default 2)

0 -- linear: u'*v _____
1 -- polynomial: (gamma*u'*v + coef0)^degree _____
2 -- radial basis function: exp(-gamma*|u-v|^2) _____
3 -- sigmoid: tanh(gamma*u'*v + coef0) _____

Código

4) A taxa de acerto usando estes dados no k-NN com k = 3 ; _____

5) A taxa de acerto usando estes dados na Árvore de Decisão ; _____

Obs. Para utilizar outros parâmetros (t) vocês podem usar os parâmetros c e g já encontrados previamente com o easy_aula.py. Utilizem os arquivos já normalizados também (.scale). Segue um passo a passo.

1) Após treinar com RBF (easy_aula.py) use os arquivos gerados (.scale) e os parâmetros c e g encontrados.
2) Como os arquivos normalizados já foram gerados, não vamos utilizar o svm-scale (para normalizar os dados usando o z-score)
3) Vamos treinar um modelo usando um outro kernel (-t). Vamos avaliar o kernel linear por exemplo: svm-train -t 0 training_set_file.scale model_file_linear
4) Vamos usar o modelo gerado no nosso conjunto de teste:
svm-predict -b 1 test_file.scale model_file_linear output_file.predict

Terceira Parte:

Utilize o conjunto de dados extraído por vocês para o classificador SVM. Compare com os outros classificadores já vistos em aula.

Relatório

Escreva um relatório de no máximo 2 páginas relatando os experimentos/resultados. Não precisa matriz de confusão. Somente a acurácia para cada experimento.

*** Caso queira usar o scikit Learn ***

Código

%Matlab - Caso precise.

```
function [] = converteSVM(arquivo, nome)

dados = load(arquivo);
%dados = arquivo ;

nome = strcat(nome, '.SVM');

fid = fopen(nome,'w');
[l, c] = size(dados);
labels = dados(:, c);

for i = 1 : l
   fprintf(fid, '%d ', labels(i,1));
   for j = 1 : c -1
      fprintf(fid, ' %i:%f', j, dados(i,j));
   end
   fprintf(fid,'\n');

end


fclose(fid) ;
```