Análise da posição de uma pessoa baseado em imagens utilizando SVM

Introdução

Este trabalho foi desenvolvido por Igor Carvalho de Brito Batista, Rony de Sena Lourenço e Thatiana Jéssica da Silva Ribeiro.

A finalidade deste trabalho é relatar a experiência com o algoritmo de machine learning Support Vector Machine (SVM) destinado à classificação de pessoas sentadas ou levantadas.

Metodologia

O modelo de aprendizado de máquina utilizado neste trabalho foi o Support Vector Machine, algoritmo responsável pelo reconhecimento de padrões. Quando os dados são linearmente separáveis, o SVM faz aplicação da regressão linear para separar uma nuvem de pontos em conjuntos estabelecidos por padrões.

Este modelo tem como entrada uma base de dados e o objetivo final é classificá-los em dois grupos. Para que esse modelo funcione, assim como qualquer outro modelo em rede, é necessário passar pela etapa do treinamento. Essa etapa tem por objetivo fazer com que o modelo aprenda os padrões a partir de uma determinada amostra, dessa forma, quando um dado desconhecido for fornecido à rede, ela será apta à designar a qual classe a amostra pertence. Depois, foi possível fazer a validação do resultado deste algoritmo com os dados de teste, durante esta fase, foi possível determinar o funcionamento da rede com alguns dados que não foram utilizados na etapa de treinamento.

Para o caso de um problema linearmente separável, o SVM, primeiramente, pega um conjunto de dados para treinar, mapeando-os no espaço multidimensional e utiliza a regressão para um hiperplano que seja capaz de separar em dois grupos

no espaço. Logo após esse período de treinamento, o algoritmo terá a capacidade de avaliar novos dados e definir a qual grupo esse esses novos dados pertencem.

Códigos

Para o desenvolvimento da atividade, foi utilizado o código cedido pelo professor em aula. O código faz uso das bibliotecas numpy, matplotlib, scikit-learn, keras e pandas.

A base de dados foi obtida a partir de um vídeo de uma pessoa se levantando e se sentando em uma cadeira. Através de um software, foi possível separar os frames e, com o auxílio de um código em Python, extraímos os dados das coordenadas x e y de determinados membros do corpo da pessoa em todos os frames. Esses dados foram colocados em um arquivo .csv.

A biblioteca pandas foi utilizada para importar o dataset em csv contendo todos os dados. Em seguida, as características que vão ser utilizadas para classificar, ou seja, coordenadas referentes às partes do corpo (que estão localizadas nas diferentes colunas do csv) são selecionadas. Depois, houve a separação do conjunto de teste. Todos os dados da tabela passaram por uma normalização para que os mesmos se apresentassem em ordem de grandeza próxima e não causasse problemas decorrentes de valores extremos. Essas passos podem ser observados na imagem a seguir.

```
#X = dataset.iloc[:,0:10].values ##considerando somente partes do rosto
#X = dataset.iloc[:,10:22].values ##considerando somente membros superiores
#X = dataset.iloc[:,22:34].values ##considerando somente membros inferiores
#X = dataset.iloc[:,[0,1,6,7,10,11,18,19,22,23,26,27]].values ##considerando 2 partes de cada tipo de membro bem mal distribuido
#X = dataset.iloc[:,[0,1,6,7,12,13,18,19,24,25,26,27,32,33]].values ##considerando 2 partes de cada tipo de membro e o tornozelo
X = dataset.iloc[:,[0,1]].values ##considerando somente nariz

#X = dataset.iloc[:,0:34].values ##considerando tudo para mostrar overfitting
y = dataset.iloc[:,34].values

print(X[0,:])
print("#################")
print(X[0:34,0])
print("#################")
print(y)
```

```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)
```

```
# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

Figura 01 - Escolha das colunas da tabela, separação do conjunto de teste e normalização dos dados.

Na figura abaixo é possível visualizar a escolha dos parâmetros do algoritmo SVM.

```
[21] # Fitting SVM to the Training set
     from sklearn.svm import SVC
     from sklearn.pipeline import Pipeline

     #classifier = SVC(kernel = 'linear', random_state = 0)
     classifier = Pipeline((
     ("scaler", StandardScaler()),
     ("svm_clf", SVC(kernel="rbf", gamma=0.05, C=0.10))
     ))



     classifier.fit(X_train, y_train)
     print(X_train.shape)
```

Figura 02 - Parâmetros do algoritmo de aprendizado de máquina SVM.

Por fim, o conjunto de teste é utilizado para fazer uma predição e com base no resultado é obtida a matriz de confusão e a taxa de acerto é calculada. A figura abaixo exemplifica o uso do conjunto de teste.

```
[22] # Predicting the Test set results
     y_pred = classifier.predict(X_test)

     print(y_test[0:35])
     print(y_pred[0:35])
```

```
['sentado' 'levantado' 'sentado' 'sentado' 'levantado' 'levantado'
 'sentado' 'levantado' 'sentado' 'levantado' 'sentado' 'levantado'
 'levantado' 'sentado' 'sentado' 'sentado' 'sentado' 'sentado' 'sentado'
 'sentado' 'sentado' 'levantado' 'sentado' 'levantado' 'sentado']
['sentado' 'sentado' 'sentado' 'sentado' 'sentado' 'sentado' 'sentado'
 'sentado' 'sentado' 'levantado' 'sentado' 'sentado' 'sentado' 'sentado'
 'sentado' 'sentado' 'sentado' 'sentado' 'sentado' 'sentado' 'sentado'
 'sentado' 'sentado' 'levantado' 'sentado']
```

```
[23] # Making the Confusion Matrix
     from sklearn.metrics import confusion_matrix
     cm = confusion_matrix(y_test, y_pred)

     print(cm)
     print((cm[0,0]+cm[1,1])/len(y_test))
```

```
[[ 2  7]
 [ 0 16]]
0.72
```

Figura 3 - Uso do conjunto de treinamento para cálculo da matriz de confusão e taxa de acerto.

Análise de sensibilidade - escolha de pontos representativos

Para esta análise, os parâmetros do algoritmo do SVM foram mantidos constantes e iguais a: 25% dos dados no conjunto de testes, kernel rbf, gama = 0,05 e C = 0,10. Foram alterados somente as escolhas de membros representativos para efetuar o treinamento da rede, conforme descrito nos tópicos abaixo.

- Situação 1

Inicialmente, escolheu-se como coordenadas representativas para o processo de classificação somente as partes que compõem o rosto do indivíduo (colunas 0 até 9 do arquivo csv).

- Situação 2

Em seguida, escolheu-se como coordenadas representativas para o processo de classificação somente os membros superiores do indivíduo (colunas 10 até 21 do arquivo csv).

- Situação 3

Em seguida, escolheu-se como coordenadas representativas para o processo de classificação somente os membros inferiores do indivíduo (colunas 22 até 33 do arquivo csv).

- Situação 4

Em seguida, escolheu-se como coordenadas representativas para o processo de classificação duas partes do rosto e duas partes de cada membro do indivíduo (nariz,orelha esquerda,ombro esquerdo,punho esquerdo, quadril esquerdo, joelho esquerdo) situadas nas colunas 0,1,6,7,10,11,18,19,22,23,26,27 do arquivo csv.

- Situação 5

Como na situação anterior a escolha de membros representativos não foi bem distribuída (só foi utilizados membros do lado esquerdo do corpo), escolheu-se agora como coordenadas representativas para o processo de classificação partes do corpo mais diversificadas (nariz,orelha esquerda,ombro direito,punho esquerdo, quadril direito, joelho esquerdo, tornozelo direito) situadas nas colunas 0,1,6,7,12,13,18,19,24,25,26,27,32,33 do arquivo csv.

- Situação 6

No último cenário analisado, foi escolhido pontos representativos somente do nariz (colunas 0 e 1).

Os resultados obtidos são sintetizados na Tabela 1.

Tabela 1 - Cenários analisados

	Situação 1	Situação 2	Situação 3	Situação 4	Situação 5	Situação 6
Matriz de confusão	9 0 0 16	9 0 0 16	8 1 0 16	9 0 0 16	9 0 0 16	2 7 0 16
Taxa de acerto	100 %	100 %	96 %	100 %	100 %	72 %

Como foram selecionadas apenas imagens na qual o indivíduo encontra-se ou sentado ou em pé para compor a base de dados, a taxa de acerto foi elevada em todas as situações na qual mais de uma característica é considerada. Além disso, as imagens foram feitas no mesmo local e do mesmo integrante do grupo, assim, quase não há variação entre elas. Logo, esse efeito de alta taxa de acerto é justificável visto que as amostras de treino estão muito similar às amostras de teste. A última situação foi incluída nesta análise justamente para demonstrar que foi necessário reduzir o número de membros representativos a um para que a taxa de acerto fosse reduzida.

Análise de sensibilidade à mudanças dos parâmetros da rede

Nessa análise, três parâmetros do algoritmo serão variados: gama, C e o tamanho do conjunto de teste. O parâmetro gama influencia em como será definida a *decision boundary*.

O parâmetro C controla o *tradeoff* entre o quão suave a *decision boundary* será ou o quão corretamente ela vai separar as classes. De forma simplificada, ao variar o valor do parâmetro C, o algoritmo tentará englobar mais pontos corretamente se o valor de C for alto ou irá generalizar melhor se o valor C for baixo. Porém a alta generalização causada pelo parâmetro C pode ocasionar *Underfitting* e valores muito altos de C podem acarretar *Overfitting*.

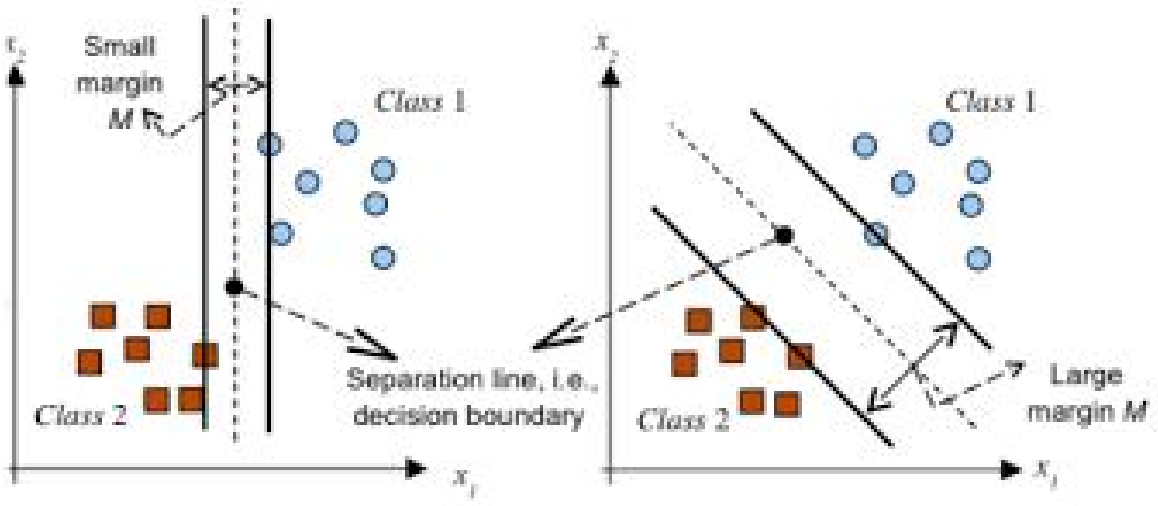

Figura 4 - Efeito do dimensionamento do parâmetro C.

O parâmetro gama (γ) é responsável por dar maior flexibilidade à fronteira de decisão. Entretanto, se a fronteira de decisão se tornar muito maleável, há o risco da ocorrência de *Overfitting*.

Kernel RBF é exponencial, um maior valor de gama faz com que amostras que distantes tenham maior influência, enquanto valores de gama menores faz com que distâncias menores influenciam menos.

$$K = e^{-\gamma|x_i - x_j|^2}$$

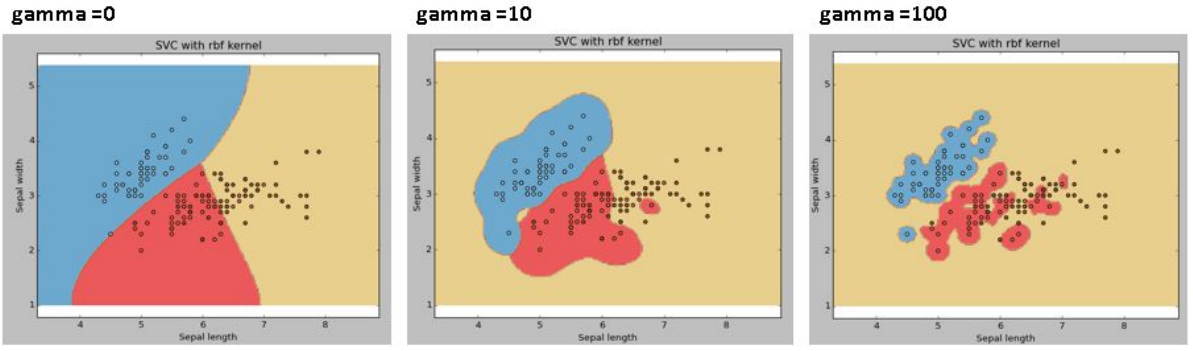

Figura 5 - Efeito do dimensionamento do parâmetro gama.

Para esta análise foi escolhido seguir com a escolha de características da situação 6 da análise mostrada anteriormente (somente coordenadas x e y do nariz são tomadas). Os parâmetros C e gama eram de 0,10 e 0,05, respectivamente. A matriz de confusão obtida nesse caso é mostrada abaixo.

```
[[ 2  7]
 [ 0 16]]
```

E a taxa de acerto foi de 72 %. Na Tabela 2 são propostos cenários com variações dos parâmetros gama e de C.

Tabela 2 - Cenários analisados

	Situação 7	Situação 8	Situação 9	Situação 10	Situação 11	Situação 12	Situação 13	Situação 14	Situação 15
Gama	0,1	1	10	0,1	1	10	0,1	1	10
C	0,01	0,01	0,01	1	1	1	100	100	100
% amostras para teste	25 %	25 %	25 %	25 %	25 %	25%	25 %	25 %	25 %
Matriz de confusão	0 9 0 16	0 9 0 16	0 9 0 16	9 0 0 16	9 0 0 16	9 0 0 16	9 0 0 16	9 0 0 16	9 0 0 16
Taxa de acerto	64 %	64 %	64 %	100 %	100 %	100 %	100 %	100 %	100 %

As linhas da matriz de confusão correspondem ao que foi predito pelo algoritmo de machine learning, o SVM, e as colunas correspondem aos dados que são verdadeiros. Nesse caso, existe duas possibilidades de saída: indivíduo sentado e levantado. Utilizando o comando unique, counts = np.unique(y_test, return_counts=True), foi possível notar que com o conjunto de teste de 25 %, temos o seguinte output:

```
['levantado' 'sentado'] [ 9 16]
```

Assim, é possível definir que o output da matriz de confusão equivale a: primeira linha - levantado; segunda linha - sentado.

```
[[ 9  0]
```

```
 [ 0 16]]
```

Agora, é possível analisar os resultados obtidos na Tabela de forma quantitativa e buscar uma interpretação apropriada.

- Situações 7, 8 e 9: manteve-se constante o valor de C e a percebemos que ao alterar gama não houve alterações no processo de classificação. O gama não influenciou no resultado do processo devido à proximidade dos pontos, nuvens de pontos muito densas se dão em razão da baixa diversidade do conjunto de dados.
- Situações 10, 11 e 12: manteve-se constante o valor de C e, novamente, ao alterar o valor do parâmetro gama não foram obtidas diferenças nos resultados da matriz de confusão. Entretanto, por conta do aumento do parâmetro C em relação às situações anteriores não foram obtidos falsos positivos e nem falsos negativos na matriz de confusão. A taxa de acerto do algoritmo foi de 100%.
- Situações 13, 14, e 15: o valor do parâmetro C foi fixado em 100 e o valor do parâmetro gama foi alterado variado entre os casos. Como resultado, não foram obtidas diferenças na matriz de confusão e na taxa de acerto do algoritmo em relação aos casos 10, 11 e 12.

O mapa de contorno só pôde ser obtido para esta segunda análise porque somente nela tem-se apenas dois atributos (coordenadas x e y do nariz), portanto, é possível visualizar num espaço 2D a distribuição espacial do resultado do processo de classificação. Na Figura 6, é mostrado o mapa de contorno para a seguinte configuração dos parâmetros do kernel rbf: gama = 0.05 e C = 0.1, na qual a taxa de acerto obtida pela matriz de confusão foi de 72 %. O rótulo de valor 0 foi dado a situação sentado e o rótulo de valor 1 foi dado a situação levantado.

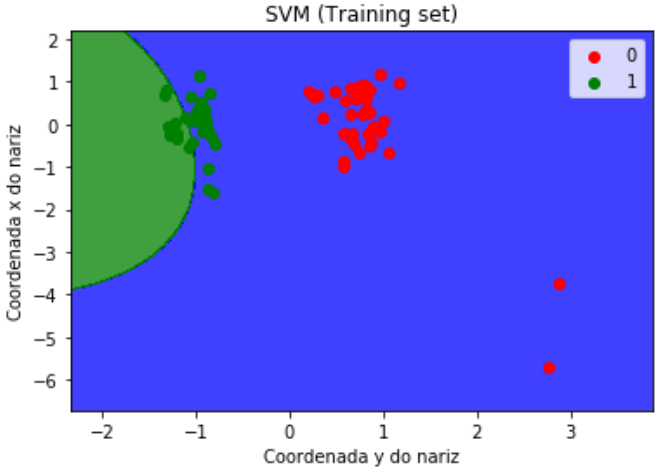

Figura 6 - Mapa de contorno para gama = 0.05 e C = 0.1

Já para a Figura 7, os parâmetros do kernel rbf foram alterados para: gama = 0.1 e C = 100.

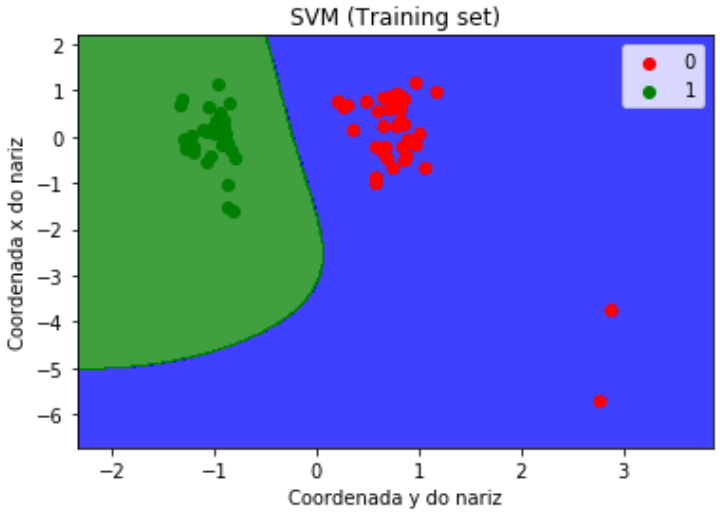

Figura 7 - Mapa de contorno para gama = 0.1 e C = 100.

Com o aumento dos parâmetros C e gama foi verificado que, como esperado, a fronteira de decisão foi melhorada e portanto amostras de indivíduo levantado que no caso anterior tinham sido classificadas incorretamente, foram então classificadas de maneira correta.