Pedro Morais nº 41019 Francisco do Carmo nº 41027 Pedro Bento nº 41082
7 de dezembro de 2016

APRENDIZAGEM AUTOMÁTICA
CLASSIFICAÇÃO DE FACES

Introdução

O objetivo deste trabalho consiste em classificar um conjunto de faces obtidas da base de dados '*labeled faces in the wild*' do *sklearn*. Neste relatório iremos explicar todos os passos realizados para atingir os objetivos propostos e mostrar quais foram os resultados obtidos para os diferentes classificadores usados (*KNeighborsClassifier* – k-NN – e o *DecisionTreeClassifier* – DTs).

Desenvolvimento

Inicialmente fazemos download da base de dados do "*labeled faces in the wild*". Esta base de dados é composta por 2370 imagens com 6510 componentes cada uma (93x70), num total de 34 classes. No entanto os dados estão bastante enviesados e de modo a prevenir que no treino de classificadores estes deem preferência às classes mais populosas, limitou-se o número de imagens de cada classe para no máximo 50, gerando então uma nova base de dados composta por 1410 imagens com os mesmos 6510 componentes por imagem e com 34 classes.

De modo a obter uma melhor estabilidade numérica multiplicámos cada imagem por um fator α de 255^{-1} , para os valores da escala de cinzento ficarem compreendidos entre [0, 1].

Conseguindo obter os dados de trabalho podemos então preparar o classificador que irá ser implementado. Para isso é necessário dividir os dados em *folds*, de modo a reduzir o risco de sobre-aprendizagem. No nosso caso dividimos os dados em 5 *folds*, que são gerados sempre da mesma maneira com a inicialização de uma *seed* (random_state = 0), de modo a obter sempre a mesma divisão dos dados e assim existir coerência na avaliação do desempenho. Em quantos mais *folds* os dados forem separados mais preciso será o nosso modelo. No entanto isto também pode ser uma desvantagem uma vez que quanto maior for o número de *folds*, mais elevado será o custo computacional, o que leva a que o processo seja mais lento.

Podemos então agora instanciar o classificador k-NN. De modo a que consigamos encontrar o valor ótimo de vizinhos para o qual se obtém o melhor desempenho no classificador, realizámos vários testes com diferentes valores de vizinhos e utilizando a validação cruzada conseguimos concluir que o valor ótimo de vizinhos é 1, tal como se pode observar na seguinte imagem.

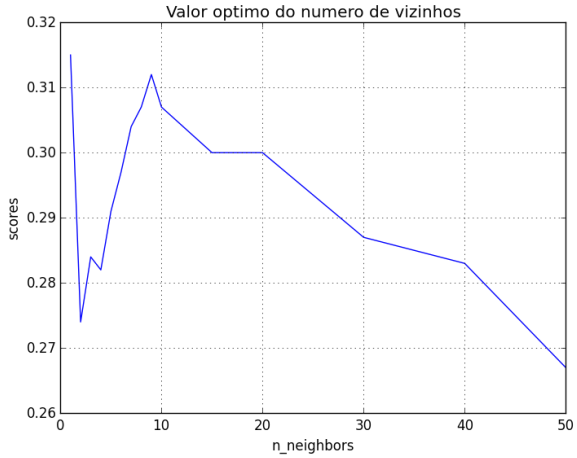

Figura 1 - Valor ótimo do número de vizinhos.

A validação cruzada é utilizada aqui como uma forma de estimar qual o desempenho do classificador. Para além disto, repetir tantas vezes quanto o número de *folds* a estimação das medidas de desempenho fornecer-nos-á mais e melhor informação sobre a variação do desempenho dos dados. Com este valor para os vizinhos, com o k-NN conseguimos obter uma eficácia média que ronda os 31,5%.

Para podermos ter uma melhor perceção de como atua o classificador na classificação das faces, obtivemos a seguinte matriz de confusão

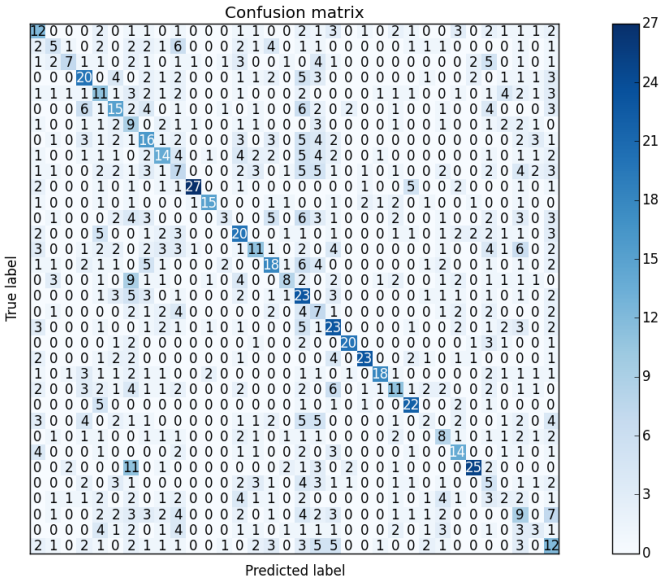

Figura 2 - Matriz de confusão para o classificador k-NN.

A partir da imagem podemos concluir que a diagonal da matriz de confusão não apresenta de todo valores satisfatórios, fazendo jus à probabilidade de 31,5% de acertos.

Agora aplicando uma transformação PCA aos dados e usando o mesmo classificador anterior vamos verificar se conseguimos obter resultados superiores ao obtidos no ponto anterior. Antes de realizarmos a transformação PCA é preciso retirar a média dos dados. Como estamos a

trabalhar com dados de grandes dimensões é importante recorrer a esta transformação de modo a projetar os dados na direção de maior variância de acordo com o número de componentes pretendidos. Neste ponto temos de testar vários valores de número de componentes para conseguirmos escolher qual apresenta a melhor probabilidade de acerto, neste teste o k-vizinhos é o valor estima anteriormente (k-vizinhos = 1). Como é possível verificar no gráfico seguinte o valor escolhido que obteve a melhor taxa de acerto (≈ 0,361) foi o de 200 componentes.

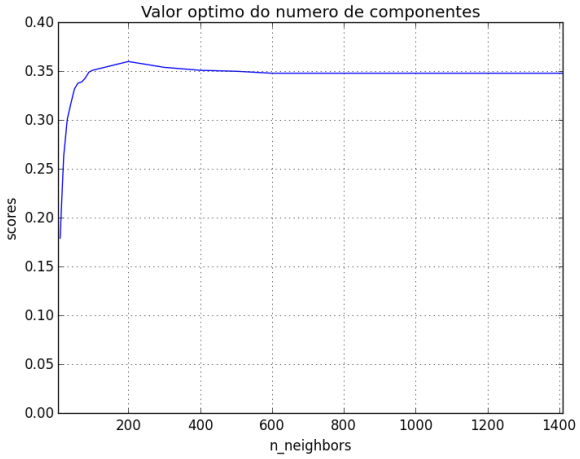

Figura 3 - Valor ótimo do número de componentes.

Iremos também apresentar a matriz de confusão para verificar como cada classe foi classificada. Através dos valores obtidos na matriz de confusão é possível verificar que ao aplicar a transformação PCA houve uma melhoria nos resultados.

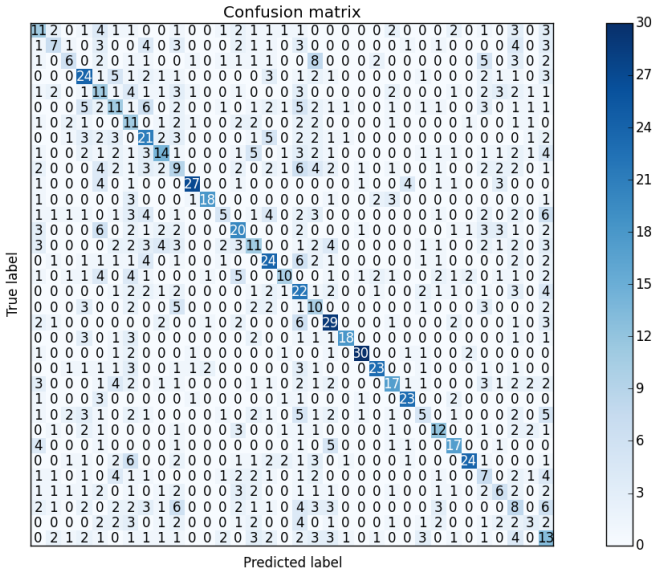

Figura 4 - Matriz de confusão do classificado k-NN com a transformação PCA.

No próximo teste vamos utilizar o número ótimo de componentes principais (200) calculado acima e normalizar a variância de cada dimensão de moda a que esta fique igual a 1. Assim sendo

os vetores das componentes irão ser multiplicados pela raiz quadrada do número de amostras e de seguida divididos pelo valor de decomposição para garantir que os resultados não estejam correlacionados com as variâncias dos componentes unitários. Este processo apresentou resultados de probabilidade de acertos (≈ 0,316) piores que a aplicação apenas da transformação PCA.

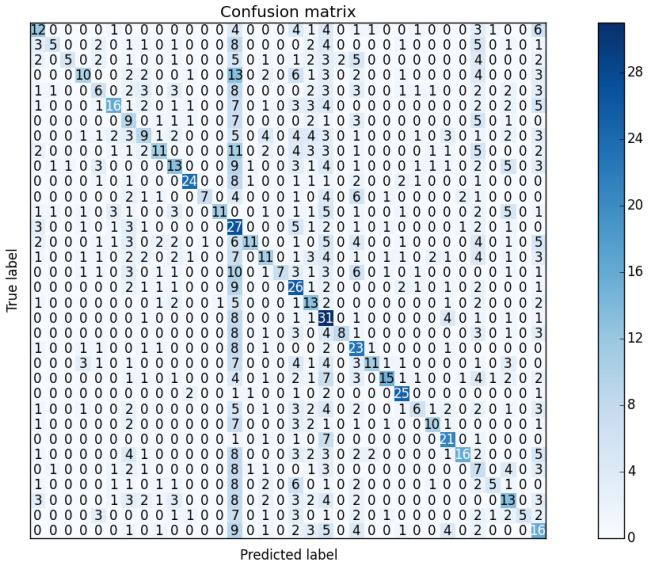

Figura 5 - Matriz de confusão com o classificador k-NN com a transformação PCA e variância = 1.

Depois de realizados todos os testes com o classificador *KNeighborsClassifier* vamos agora compará-lo com o classificador *DecisionTreeClassifier,* este classificador é um classificador que aplica um método de aprendizagem supervisionada podendo ser usado para classificação (quando as variáveis em análise podem assumir um valor finito) e regressão (quando as variáveis em análise podem tomar valores contínuos, como os números reais). O objetivo deste classificador é criar um modelo que preveja o valor da variável em análise através de uma aprendizagem baseada em regras de decisão simples inferidas através das características da amostra em estudo.

Este classificador nos dois testes realizados (com PCA e com PCA com variância = 1) apresentou resultados piores que o classificador de k-vizinhos mais próximos. A taxa de acertos com a PCA foi de aproximadamente 0,134 e com a variância = 1 o classificador teve um desempenho melhor, mas não o suficiente para classificar melhor que o k-NN, o valor de acertos obtido para este teste foi de 13,7% de acertos.

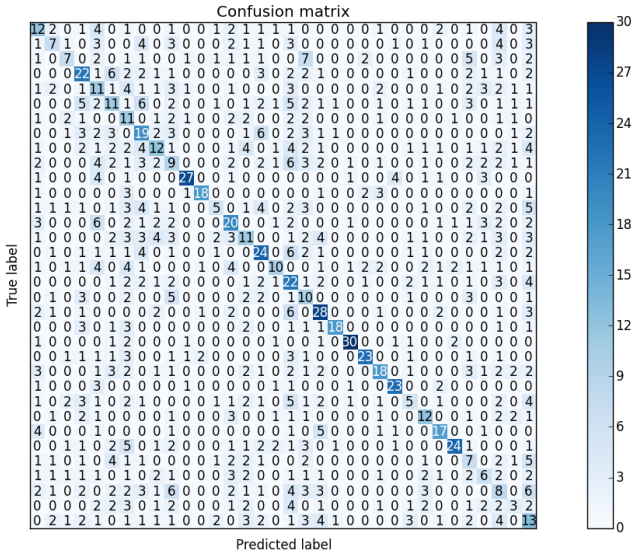

Figura 6 - Matriz de confusão com o classificador DTs com a transformação PCA.

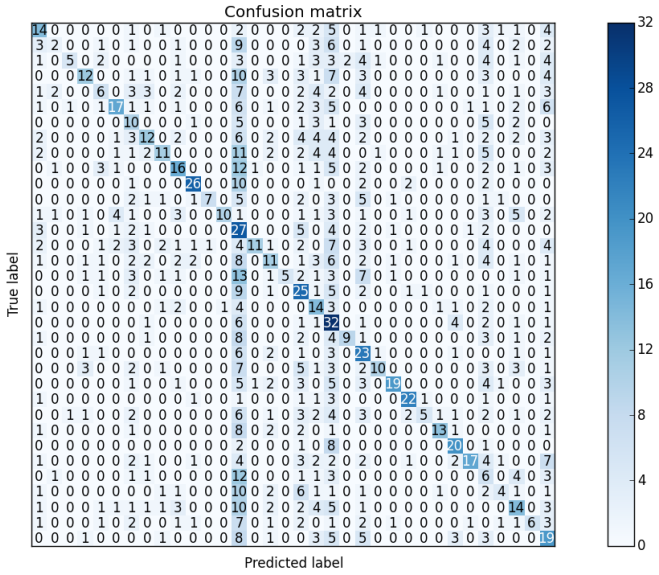

Figura 7 - Matriz de confusão com o classificador DTs com a transformação PCA e variância = 1.

Conclusão

Em suma, conseguimos verificar que ambos os classificadores usados apresentam taxas de acerto bastante más, tal como se pode verificar nos resultados obtidos. Podemos também concluir que o facto de não possuirmos um computador com elevada capacidade computacional, nos leva a utilizar também classificadores menos "poderosos", visto os outros serem muito mais pesados para as nossas máquinas.

Bibliografia

http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html (acedido a 1 de Dezembro de 2016)

http://scikit-learn.org/stable/modules/tree.html#tree (acedido a 4 de Dezembro de 2016)

https://books.google.pt/books?id=1-4lDQAAQBAJ&pg=PA195&lpg=PA195&dq=pca+whiten+%3D+true&source=bl&ots=26oNDKMNY_&sig=skDlxCW7FY7Q8NMAAE-bG8cDrs8&hl=pt-PT&sa=X&ved=0ahUKEwi-vfmo2eLQAhUJvRQKHWtVCvoQ6AEIVTAH#v=onepage&q=pca%20whiten%20%3D%20true&f=false (acedido a 1 de Dezembro de 2016)

https://en.wikipedia.org/wiki/Decision_tree_learning (acedido a 4 de Dezembro de 2016)