Raven Dataset – Masculino vs Feminino

Francisco Gonçalves, Nº Mec. 99788, MEI
Pedro Cavadas, Nº Mec. 85090, MEI

Mestrado Engenharia Informática,
Universidade de Aveiro

Abstract

This report presents the explanation for the machine learning project created in the scope of the second project of Data Exploration of the Master of Computer Science from the university of Aveiro.

Over the course of this report, we will explain the changes made to our data, with the purpose of removing outliers, and the strategy implemented for the analysis of our dataset.

We will also discuss, in further detail, our question, the reasons behind its choice, and what conclusions/answers we have reached by analysing the results obtained.

Introduction

Este relatório apresenta a explicação do projeto de machine learning, realizado no âmbito do segundo projeto da cadeira Exploração de dados do Mestrado em Engenharia Informática da Universidade de Aveiro.

Neste relatório iremos discutir/explicar a implementação do nosso segundo projeto da cadeira de Exploração de dados.

Como nosso tema nós decidimos tentar responder à pergunta: Há diferença nos resultados obtidos pelos dois géneros. Para isto, nós utilizámos a base de dados com os resultados obtidos no teste Raven aplicado a 21 estudantes de Desenho e Multimédia e a 24 estudantes de Engenharia Informática.

Iremos começar por discutir que tratamento foi feito aos dados disponibilizados; como foram implementados os diferentes classificadores testados, incluindo porções de código; discussão dos resultados obtidos após testes, com ênfase nos classificadores com melhor precisão e, por último, iremos referir que conclusões retirámos do testes, e tentar responder à pergunta original.

1. Dataset

Como referido anteriormente, o dataset escolhido foi o dataset correspondente aos teste de matrizes de Raven. Relativamente a ficheiros em concreto, foram usados os seguintes: ‘Overall_P100.xlsx',‘overall_energy_ratios.xlsx’, 'overall_immersion.xlsx', 'Overall_P300.xlsx', e ‘Informação_género’ (cujo conteúdo foi usado na criação do ficheiro ‘data.csv’).

Após estes ficheiros terem sido carregados, foram-lhes aplicado quatro operações de processamento de forma a ser feito o tratamento dos dados. De início, foram substituídos os valores correspondentes a NaN, carregados dos ficheiros 'Overall_P100.xlsx' e 'Overall_P300.xlsx', pela média da coluna do seu género. Isto teve o objetivo de minimizar o seu efeito nos resultados. Esta operação foi implementada da seguinte forma:

```
for target in classes:
    keep = overall['Gender'] == target
    target_df = overall[keep]
    overall[keep] = target_df.fillna(target_df.mean())
```

De seguida, foi aplicado o ‘Standard Scaling’. Esta operação foi implementada da seguinte forma:

```
# apply standard scaling
x = overall[columns].values
standard_scaler = StandardScaler()
x = standard_scaler.fit_transform(x)
topca = pd.DataFrame(x, columns = columns)
```

Logo após, foi aplicado o Kernel PCA aos dados, que consiste num procedimento matemático usado para converter um conjunto de observações num grupo de componentes principais, e é também usado de forma a reduzir as características (‘feature reduction’). Esta operação foi implementada da seguinte forma:

```
for i in range(1, 34):
    pca = KernelPCA(n_components = i, kernel = 'rbf', gamma = 15, fit_inverse_transform = True)
    x_pca = pca.fit_transform(topca)
```

Por último, foi feita a normalização dos dados, com o intuito de reduzir a redudância de dados e aumentar a integridade destes. Esta operação foi implementada da seguinte forma:

```
# normalization
    x = x_pca
    min_max_scaler = MinMaxScaler()
    x = min_max_scaler.fit_transform(x)
```

2. Implementação/Código

O código deste projeto está contido no ficheiro 'test.py'. De início, são definidas as classes masculino e feminino e é carregado o conteúdo do ficheiro criado denominado 'data.csv', que contém informação relativa a cada indivíduo. Mais concretamente, o número de respostas certas e erradas tanto para o treino como para o teste em si. De seguida, é calculado a proporção de respostas corretas. Na figura seguinte podemos observar o código que executa estas funções:

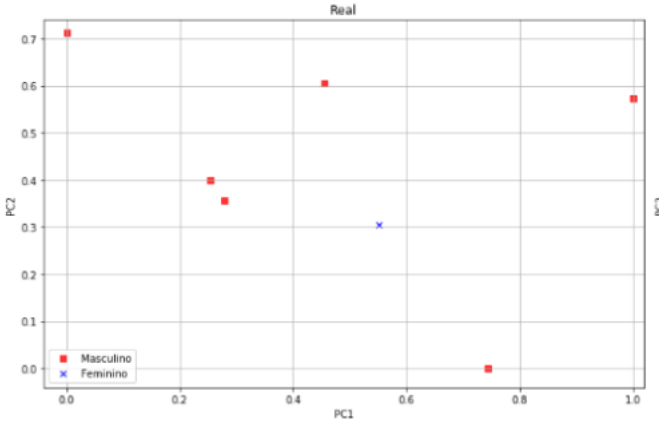
```
# set columns and classes
gicolumns = [ 'Indiv', 'Gender' ]
classes = [ 'Feminino', 'Masculino' ]

# load and filter gender info
gender_info = pd.read_csv(os.path.join('RAVEN', 'data.csv'), header = None, engine = 'python')
gender_info = gender_info[gender_info[1].isin(classes)]

# calculate ratios of correct answers
gender_info['training'] = gender_info[2] / (gender_info[2] + gender_info[3])
gender_info['testing'] = gender_info[4] / (gender_info[4] + gender_info[5])
```

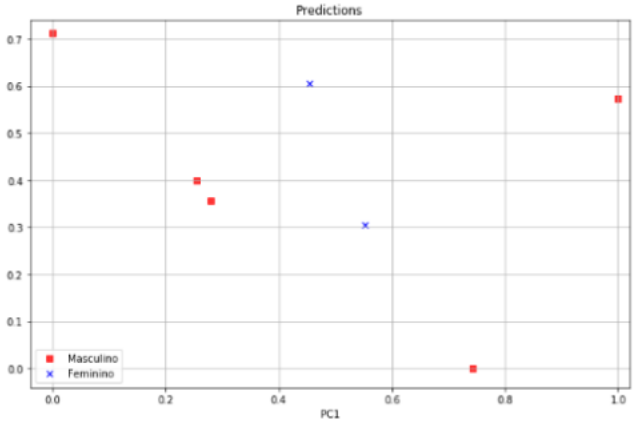
O loop seguinte corresponde ao treino dos classificadores. Como é possível observar, os classificadores testados foram os seguintes: Random Forest, MLP, SGD, Linear SVC, PPN e SVC.

```
amount = 2000
for i in range(amount):
    rf_classifier = RandomForestClassifier(max_depth = 3, min_samples_split = 5,n_estimators = 10, max_features = 'log2',
                                           oob_score = False)
    mlp_classifier = MLPClassifier(activation = 'tanh', hidden_layer_sizes = (10, 5), alpha = 0.01, max_iter = 5000)
    sgd_classifier = SGDClassifier(loss = 'log', max_iter = 100000)
    linear_svc_classifier = LinearSVC(C = 1.0, max_iter = 100000, tol = 1e-05, verbose = 0)
    ppn_classifier = Perceptron(penalty = None, alpha = 0.0001, fit_intercept = True, max_iter = 10000, tol = None,
                eta0 = 0.1, n_jobs = 1, random_state = 0, class_weight = None, warm_start = False)
    svc_classifier = SVC(C = 1.0, kernel = 'rbf', max_iter = 100000, tol = 1e-05, verbose = 0)

    training_size = math.floor(length * 0.8)
    testing_size = length - training_size

    training_choice = list(np.random.choice(X.shape[0], training_size, replace = False))
    testing_choice = [ i for i in range(X.shape[0]) if i not in training_choice ]

    X_training = X[training_choice, :]
    Y_training = [ y[value] for value in training_choice ]
X_testing = X[testing_choice, :]
Y_testing = [ y[value] for value in testing_choice ]

rf_classifier.fit(X_training, Y_training)
rf_Y_pred = rf_classifier.predict(X_testing)

mlp_classifier.fit(X_training, Y_training)
mlp_Y_pred = mlp_classifier.predict(X_testing)

sgd_classifier.fit(X_training, Y_training)
sgd_Y_pred = sgd_classifier.predict(X_testing)

linear_svc_classifier.fit(X_training, Y_training)
linear_svc_Y_pred = linear_svc_classifier.predict(X_testing)

ppn_classifier.fit(X_training, Y_training)
ppn_Y_pred = ppn_classifier.predict(X_testing)

svc_classifier.fit(X_training, Y_training)
svc_Y_pred = svc_classifier.predict(X_testing)

rf_accuracy += accuracy_score(Y_testing, rf_Y_pred)
mlp_accuracy += accuracy_score(Y_testing, mlp_Y_pred)
sgd_accuracy += accuracy_score(Y_testing, sgd_Y_pred)
linear_svc_accuracy += accuracy_score(Y_testing, linear_svc_Y_pred)
ppn_accuracy += accuracy_score(Y_testing, ppn_Y_pred)
svc_accuracy += accuracy_score(Y_testing, svc_Y_pred)
```

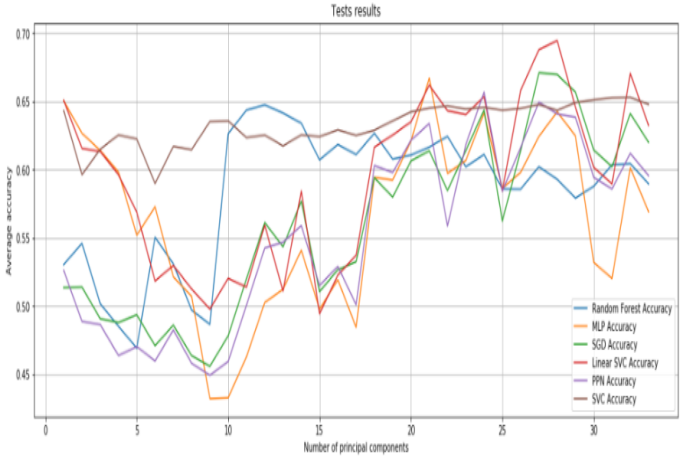
3. Resultados

Nas figuras abaixo podemos observar e comparar as predições feitas pelo classificador SVC Linear com os resultados reais (neste gráfico foram apenas usadas os dois primeiros componentes, devido ao alto número total destes). Desta forma, conseguimos perceber que de maneira semelhante, os resultados e as predições mostram as componentes do género masculino com um valor acima, em média, das do género feminino.





Conforme podemos observar na figura seguinte, o classificador SVC ao longo do tempo manteve uma boa precisão com diferente número de componentes. A precisão dos classificadores PPN e SGD teve um comportamente semelhante com o aumento do número de componentes (precisão melhorou com mais componentes). O classificador SVC linear atingiu a maior precisão de todos os classificadores, com uma precisão de aproximadamente 69,5%. O classificador PPN foi o classificador que, em média, teve os menores valores de precisão de entre todos.



4. Conclusão

Em suma, como é possível observar pelos gráficos acima disponibilizados e pelo ficheiro de texto com os resultados, o classificador com melhor precisão foi o Linear SVC com 28 features/componentes, que teve uma precisão de aproximadamente 0.695 ou 69%. É também possível concluir que, com base nos resultados disponíveis no dataset providenciado, o género masculino teve maior sucesso na resposta ao teste Raven.

5. Contribuição

Pedro Cavadas – Código

Francisco Gonçalves – Relatório

References

[1] Annushree Bablani, Damodar Reddy Edla, Diwakar Tripathi, and Ramalingaswamy Cheruku. Survey on brain-computer interface: An emerging computational intelligence paradigm. ACM Comput. Surv., 52(1):20:1–20:32, February 2019.

[2] Sayed Ahmed Alwedaie, Habib Al Khabbaz, Sayed Redha Hadi, and Riyadh Al Hakim. EEGBased Analysis for Learning through Virtual Reality Environment. Journal of Biosensors & Bioelectronics, 09(01):1–6, feb 2018.

[3] https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60