

MÁY HỌC TRONG THỊ GIÁC MÁY TÍNH

HỆ THỐNG PHÂN BIỆT GIỚI TÍNH THÔNG QUA ẢNH CHÂN DUNG

14520529 – NGUYỄN CAO MINH, 14520178 – ĐOÀN TRÍ ĐỨC

*Lớp Khoa học Tài năng 2014, Khoa Khoa học máy tính, Trường Đại học Công nghệ Thông tin;
14520529@gm.uit.edu.vn, 14520178@gm.uit.edu.vn*

CS332.I11.KHTN

LÊ ĐÌNH DUY

MAI TIẾN DŨNG

TP. HỒ CHÍ MINH, 18/12/2017

MỤC LỤC

I. GIỚI THIỆU CHUNG:.....	2
II.PHƯƠNG PHÁP THỰC HIỆN:	2
III. CÁC KHÁI NIỆM CƠ BẢN:.....	6
IV. CÁC HÀM THỰC THI:.....	8
V.ĐÁNH GIÁ KẾT QUẢ:.....	14
VI. KẾT LUẬN:.....	15
VII.TÀI LIỆU THAM KHẢO:	15

I. GIỚI THIỆU CHUNG:

1. Đặt vấn đề:

- Với sự phát triển nhanh chóng và không ngừng của công nghệ, những rào cản của các thuật toán với đầu vào là khối lượng dữ liệu lớn đang ngày càng được tập trung giải quyết và áp dụng vào các công nghệ hiện tại.
- Trong lĩnh vực khoa học máy tính, “Deep learning” và “Machine Learning” đang được đầu tư và phát triển trong thời gian hiện tại, với các tiềm năng và ứng dụng mạnh mẽ, “Machine Learning” đang được dần tích hợp vào các lĩnh vực và các công nghệ hiện tại. Thị giác máy tính là một trong những lĩnh vực đang được tập trung hướng đến, với các kết quả khá tốt mà “Machine Learning” đã đạt được khi áp dụng vào các bài toán, các vấn đề liên quan đến thị giác máy tính đang mắc phải hứa hẹn sẽ được giải quyết trong tương lai gần với sự hỗ trợ đặc lực từ “Machine Learning”.

2. Mục tiêu:

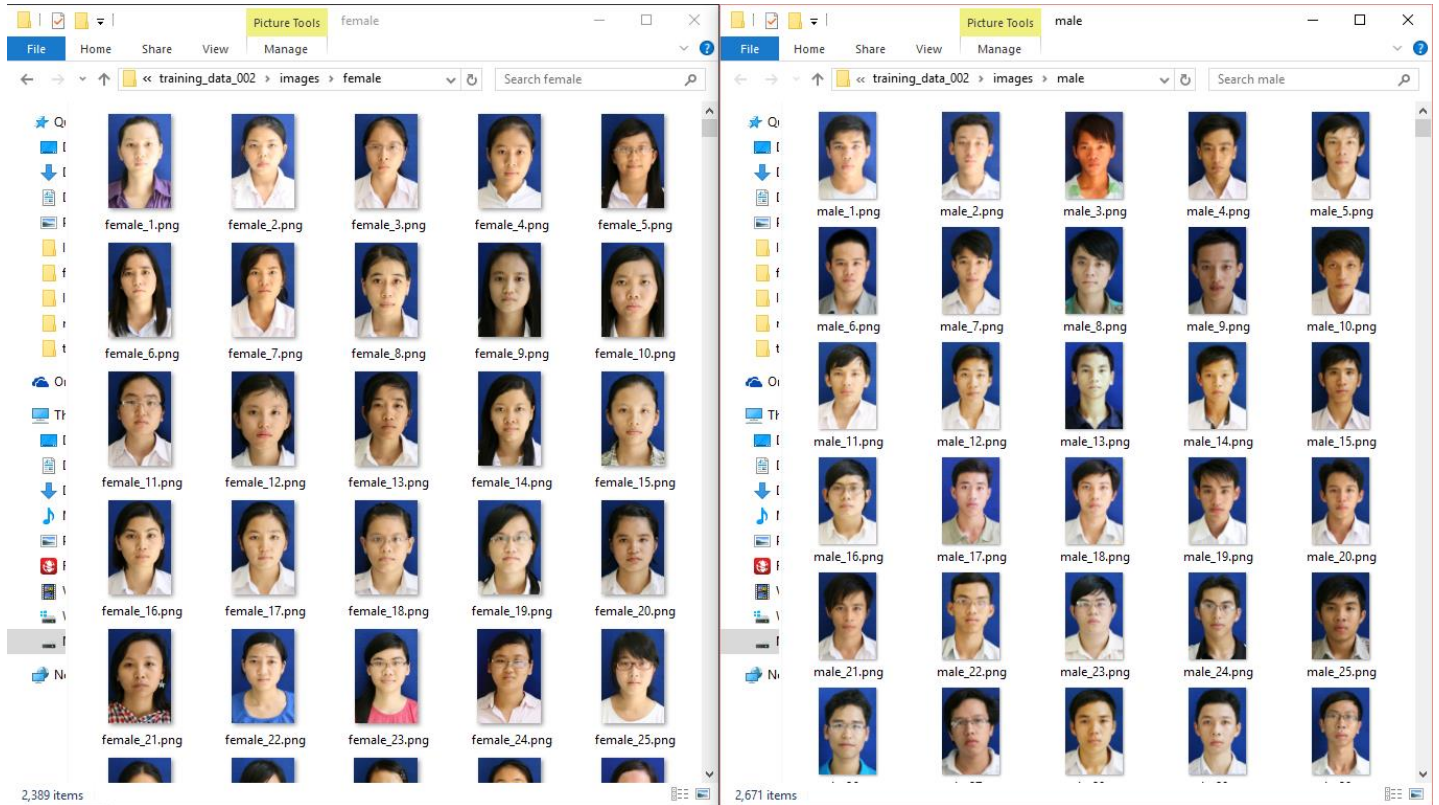
- Xây dựng một hệ thống ứng dụng liên quan đến thị giác máy tính áp dụng các kỹ thuật liên quan trong “Machine Learning”.
- Đề tài: xây dựng hệ thống dự đoán giới tính của một người thông qua gương mặt. Các kỹ thuật được áp dụng bao gồm:
 - Sử dụng mạng CNN của Oxford (VGG16 – OxfordNet) để rút trích đặc trưng từ các bức ảnh.
 - Áp dụng các thuật toán SVM để xây dựng model phân loại.

II. PHƯƠNG PHÁP THỰC HIỆN:

1. Khởi tạo bộ dataset:

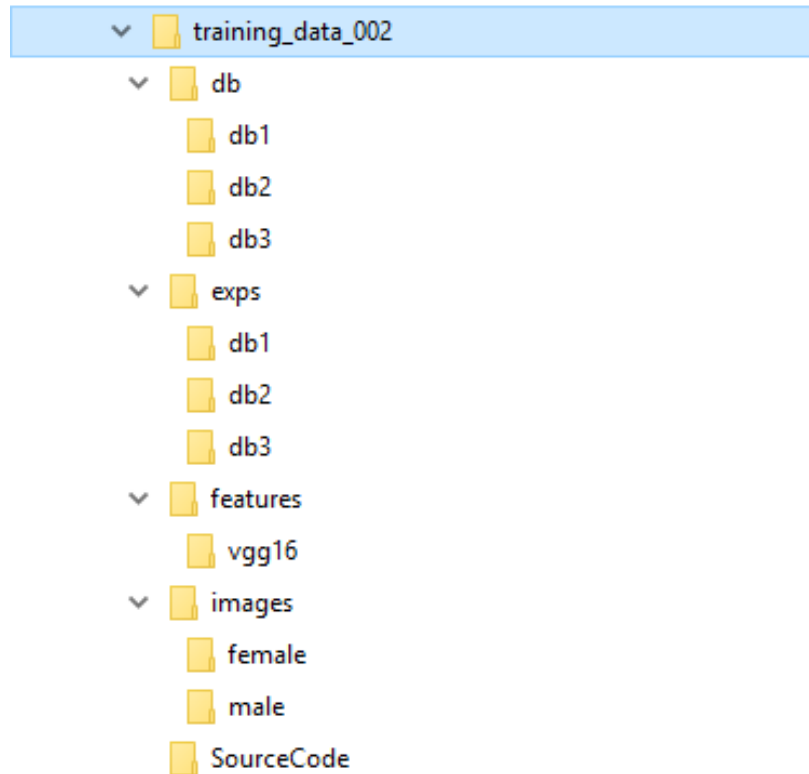
- Bộ dataset bao gồm 5000 tấm ảnh 3x4 của các bạn sinh viên khác nhau với kích thước (235x352), nguồn được lấy từ dữ liệu ký túc xá khu B Đại học Quốc gia Thành phố Hồ Chí Minh.

- Bộ dataset bao gồm:
 - o 2389 ảnh nữ.
 - o 2671 ảnh nam.

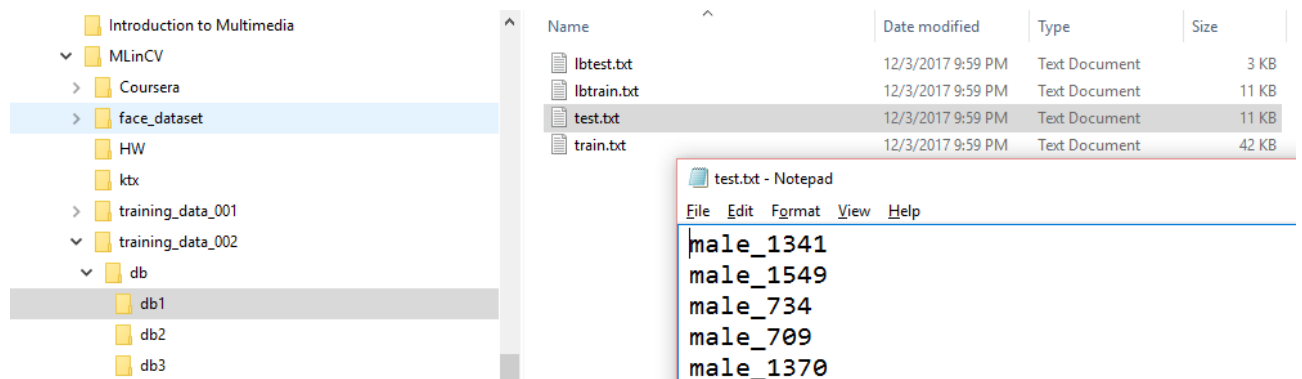


2. Xây dựng cấu trúc thư mục lưu trữ:

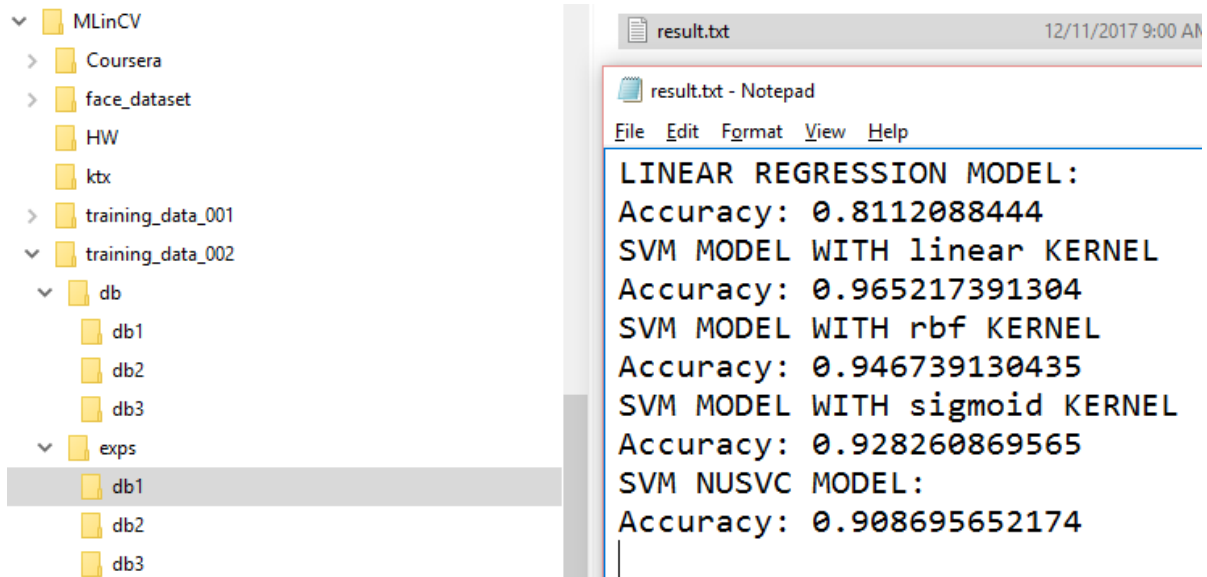
- Thư mục lưu trữ hệ thống có dạng như sau:



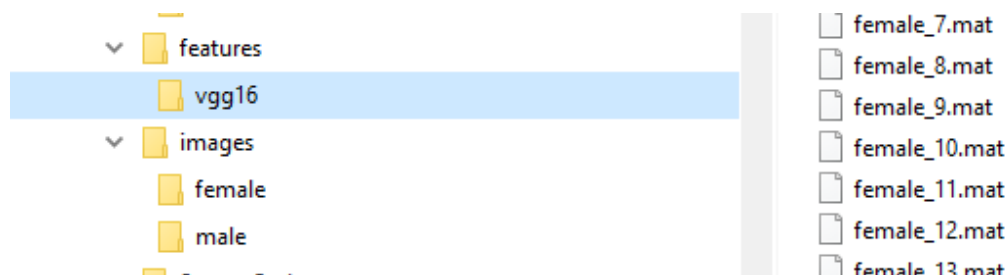
- training_data_002: Thư mục chính của project (thư mục duy nhất mà người dùng có thể thay đổi tên)
 - o db: Thư mục lưu trữ các bộ dữ liệu bao gồm các tập train và test dưới dạng file .txt



- exps: Thư mục lưu trữ kết quả so sánh độ chính xác khi áp dụng model ứng với các bộ dữ liệu trong thư mục db.



- Features: thư mục lưu trữ ma trận feature của các bức ảnh được rút trích thông qua mạng neural VGG16.



- Images: Thư mục chứa hình ảnh 3x4 (bộ dataset)
- SourceCode: Thư mục chứa source code thực thi.

3. Rút trích feature:

- Sử dụng thư viện của Keras để dùng mạng neural VGG16 rút trích đặc trưng từ các tấm ảnh trong bộ dataset và lưu lại thành ma trận feature. (file .mat)
- Kết quả rút trích feature được lấy ra tại output của layer “fc2” trong mô hình VGG16, vì ta chỉ cần lấy feature đặc trưng của tấm ảnh, nên không cần phải đi qua lớp softmax của mô hình VGG16 (sigmoid function).

- Nền của thư viện Keras được sử dụng là TensorFlow.
- 4. Xây dựng bộ dữ liệu train và test:
 - Bộ dữ liệu train và test được chạy bằng hàm random theo tỷ lệ 8:2, tức mỗi 10 ảnh thì có 8 ảnh dùng để train và 2 ảnh dùng để test.
 - Cụ thể đối với 1 bộ dữ liệu 5000 ảnh thì ta có 4000 ảnh dùng để train và 1000 ảnh dùng để test model được xây dựng từ bộ dữ liệu train.
 - Ứng với mỗi bộ dữ liệu train và test luôn có bộ label tương ứng dùng để đối chiếu trong việc training và evaluating trong hệ thống.
- 5. Xây dựng model:
 - Model được xây dựng dựa trên các bộ phân lớp sau:
 - o Linear Regression.
 - o Supported Vector Machine (SVM):
 - SVC:
 - Linear Kernel.
 - Rbf Kernel.
 - Sigmoid Kernel.
 - NuSVC.
- 6. Kiểm thử và so sánh:
 - Các mô hình đều được tính độ chính xác khi so sánh giữa label mà model dự đoán ra và label mà người dùng mong đợi. Các kết quả sẽ được lưu vào file result.txt của mỗi bộ dữ liệu trong folder exps.
 - Sau đó sẽ dùng phần mềm excel để tính toán độ lệch chuẩn và vẽ biểu đồ so sánh độ chính xác giữa các mô hình và đưa ra kết luận.

III. CÁC KHÁI NIỆM CƠ BẢN:

- VGG16^[1]:
 - o VGG16 hay còn được gọi là OxfordNet là một mạng di truyền dạng xoắn (convolutional neural network) được đặt tên và xây dựng bởi nhóm Visual Geometry Group từ Oxford.
 - o Mô hình này đã giành chiến thắng trong cuộc thi ILSVR (Image Net) năm 2014.
 - o Cho đến nay, nó vẫn được xem là một mô hình tuyệt vời khi áp dụng vào các bài toán cái đầu vào là hình ảnh, mặc dù rằng đã có nhiều mô hình cải tiến từ nó như ResNet hay Inception.

- Keras^[2]:
 - Keras là một API mạng di truyền cao cấp, là một thư viện neural network được viết cho ngôn ngữ python.
 - Được xây dựng dựa trên Tensorflow, CNTK hoặc Theano.
 - Được thiết kế để thực hiện các hàm tính toán thực thi với mạng di truyền sâu (deep neural network) một cách nhanh chóng với các cú pháp đơn giản.
- TensorFlow^[3]:
 - Là một thư viện mã nguồn mở dùng để tính toán dựa trên việc biến đổi và xử lý dữ liệu như một đồ thị.
 - Mỗi nodes trong đồ thị tương ứng với một phép toán, và các cạnh biểu thị cho số chiều của dữ liệu sử dụng cũng như thể hiện sự liên kết giữa các dữ liệu.
 - Kiến trúc của TensorFlow thích hợp cho cả CPU và GPU, có thể thực hiện cài đặt sử dụng trên các loại thiết bị và platform khác nhau như desktop, mobile, server, ... chỉ thông qua 1 API duy nhất.
 - TensorFlow được phát triển và nghiên cứu bởi tổ chức nghiên cứu Google's Machine Learning với mục đích giải quyết và thực hiện các bài toán liên quan đến máy học cũng như là các mạng di truyền sâu.
 - TensorFlow hiện tại đang được áp dụng trong rất nhiều lĩnh vực khác nhau để giải quyết các bài toán liên quan đến Machine Learning và Deep Neural Network.
- Linear Regression^[4]:
 - Trong thống kê, hồi qui tuyến tính là một cách tiếp cận tuyến tính để mô hình hóa mối quan hệ giữa một biến phụ thuộc vô hướng y và một hoặc nhiều biến giải thích (hoặc các biến độc lập) biểu thị X.
 - Trong hồi quy tuyến tính, các mối quan hệ được mô hình bằng các hàm dự báo tuyến tính mà các tham số mô hình không xác định là ước tính từ dữ liệu.
- Supported Vector Machine (SVM)^[5]:

- Là một tập các phương pháp học tập có giám sát (supervised), được sử dụng trong việc phân lớp (classification), hồi quy (regression), và phát hiện các điểm ngoại biên (outliers detection).
- SVC là một mô hình classification trong SVM^[6].
- NuSVC^[7]:
 - Nu-Support Vector Classification, tương tự như SVC nhưng sử dụng thêm tham số Nu để điều chỉnh độ lỗi có thể chấp nhận được của các support vectors.

IV. CÁC HÀM THỰC THI:

1. Môi trường, ngôn ngữ và các thư viện sử dụng:
 - Ngôn ngữ lập trình được sử dụng: Python 3.
 - Các thư viện phục vụ:
 - Tensorflow
 - Keras
 - Sklearn
2. Hàm khởi tạo ngẫu nhiên bộ dữ liệu train và test:
 - Hàm khởi tạo ngẫu nhiên 1 tập hơn 2000 số bao gồm các chữ số hoàn toàn ngẫu nhiên và khác nhau.
 - Sử dụng hàm chuyển đổi tập 2000 số thành tên của 2000 file ứng với mỗi folder Male và folder Female (tổng số 5000 ảnh), trong đó lấy 80% số ảnh của mỗi tập làm bộ ảnh trainer, 20% còn lại được sử dụng trong việc test.

- Mỗi bộ dữ liệu được thực hiện theo trình tự như trên, để đảm bảo tính khách quan của bộ dữ liệu, chương trình thực hiện khởi tạo ngẫu nhiên 3 bộ dữ liệu và tiến hành tính toán các giá trị trung bình khi áp dụng từng bộ dữ liệu vào các mô hình khác nhau.

```
27 #FUNCTION: RANDOM TRAINING DATA
28 def random_file(path_file):
29     import random
30     num_rand = random.sample(range(1,2301),2300) #2301,2300
31     #MALE
32     file_train_txt = open(path_file+"/train.txt","w")
33     file_train_lb = open(path_file+"/lbtrain.txt","w")
34     file_test_txt = open(path_file+"/test.txt","w")
35     file_test_lb = open(path_file+"/lbtest.txt","w")
36     for i in num_rand[0:1840]: #1840
37         file_train_txt.write("male_"+str(i)+"\n")
38         file_train_lb.write("1\n")
39     for i in num_rand[1840:2300]: #1840:2300
40         file_test_txt.write("male_"+str(i)+"\n")
41         file_test_lb.write("1\n")
42     file_train_txt.close()
43     file_train_lb.close()
44     file_test_txt.close()
45     file_test_lb.close()
46
47     #FEMALE
48     num_rand = random.sample(range(1,2301),2300)
49     file_train_txt = open(path_file+"/train.txt","a")
50     file_train_lb = open(path_file+"/lbtrain.txt","a")
51     file_test_txt = open(path_file+"/test.txt","a")
52     file_test_lb = open(path_file+"/lbtest.txt","a")
53     for i in num_rand[0:1840]:
54         file_train_txt.write("female_"+str(i)+"\n")
55         file_train_lb.write("0\n")
56     for i in num_rand[1840:2300]:
57         file_test_txt.write("female_"+str(i)+"\n")
58         file_test_lb.write("0\n")
59     file_train_txt.close()
60     file_train_lb.close()
61     file_test_txt.close()
62     file_test_lb.close()
63     print("DONE WRITE FILE")
```

3. Hàm rút trích feature:

- Sử dụng mạng VGG16 full-connected để rút trích các đặc trưng từ ảnh, kết quả được lấy từ layer “fc2” với output là ma trận chứa 4096 vector đặc trưng của bức ảnh (1x4096).
- Weight được sử dụng trong mạng VGG16 là bộ dataset của ImageNet.

```
65 #FUNCTION: EXTRACT FEATURE VGG16
66 def extract_fea_vgg16(path_fea,path_img):
67     from vgg16 import VGG16
68     from keras.preprocessing import image
69     from keras.models import Model
70     from imagenet_utils import preprocess_input
71     import numpy as np
72     import glob
73     import os
74     vgg16model = VGG16(weights='imagenet', include_top=True);
75     model = Model(inputs=vgg16model.input, outputs=vgg16model.get_layer('fc2').output)
76     i=1
77     for filename in glob.glob(os.path.join(path_img, '*.png')):
78         name_save=(filename.replace(path_img,"").replace(".png",""))
79         file_feature = open(path_fea+"/"+name_save+".mat","wb")
80         img_path=filename
81         img = image.load_img(img_path, target_size=(224, 224))
82         x = image.img_to_array(img)
83         x = np.expand_dims(x, axis=0)
84         x = preprocess_input(x)
85         features = model.predict(x)
86         np.savetxt(file_feature, features)
87         file_feature.close()
88         print("Step "+str(i)+" done")
89         i=i+1
90     print("DONE EXTRACT FEATURE")
```

4. Hàm xây dựng model và kết quả so sánh:

- Trước khi thực hiện việc đưa dữ liệu input vào các model để training, cần khởi tạo một ma trận chứa tất cả các feature đặc trưng của các ảnh sẽ dùng để training và test, cũng như ma trận chứa label tương ứng của các bộ ảnh đó.
- Hàm `get_training_input` sẽ trả về các ma trận:
 - o `arr_test/arr_train`: Ma trận chứa tất cả các feature của n ảnh dùng để test/train, ma trận có dạng (nx4096) – ma trận bao gồm n ảnh, mỗi ảnh có 4096 bộ feature đặc trưng.
 - o `arr_lbtest/arr_lbtrain`: Ma trận chứa các nhãn tương ứng với các ảnh của bộ test/train, ma trận có dạng (nx1) – ma trận bao gồm n ảnh, mỗi ảnh chứa 1 giá trị là 0: nếu là nữ và 1: nếu là nam

```
132 #FUNCTION: MERGE DATA TO 1 MATRIX
133 def get_training_input(path_training,path_features):
134     arr_test = initialize_input(path_training,path_features,"test.txt")
135     arr_lbtest = initialize_input(path_training,path_features,"lbtest.txt")
136     arr_train = initialize_input(path_training,path_features,"train.txt")
137     arr_lbtrain = initialize_input(path_training,path_features,"lbtrain.txt")
138     return [
139         arr_test,
140         arr_lbtest,
141         arr_train,
142         arr_lbtrain
143     ]
```

- Các model áp dụng:
 - Linear Regression Model:
 - Sử dụng thư viện sklearn, gọi model Linear Regression trong class linear model của sklearn.
 - Sau đó fit model với tập train và label train đã được khởi tạo từ trước thông qua hàm fit().
 - Sau khi model đã được training, ta sử dụng hàm predict() cho bộ dữ liệu test để model đưa ra các kết luận dựa trên dữ liệu đã được học tập.
 - Ta thực hiện phép so sánh giữa kết quả label mà model dự đoán ra được với ma trận label chuẩn mà ta đã khởi tạo từ trước.

```

145 #FUNCTION: LINEAR REGRESSION MODEL
146 def sklearn_linear_model(x_train, lbx_train, x_test, lbx_test, path_result):
147     from sklearn import linear_model
148     from sklearn.metrics import mean_squared_error
149     x_train=x_train.tolist()
150     x_test=x_test.tolist()
151     lbx_train=(lbx_train.ravel()).tolist()
152     lbx_test=(lbx_test.ravel()).tolist()
153
154     file_result_txt = open(path_result+"/result.txt", "w")
155     # Create linear regression object
156     regr = linear_model.LinearRegression()
157     # Train the model using the training sets
158     regr.fit(x_train, lbx_train)
159     # Make predictions using the testing set
160     x_pred = regr.predict(x_test)
161     print("Mean squared error: %.2f"
162           % mean_squared_error(lbx_test, x_pred))
163     result = 1 - mean_squared_error(lbx_test, x_pred)
164
165     file_result_txt.write("LINEAR REGRESSION MODEL:\n")
166     file_result_txt.write("Accuracy: "+str(result)+"\n")
167     file_result_txt.close()
168

```

- SVM.SVC:

- Cách thực hiện tương tự như model Linear Regression, tuy nhiên trong model SVC của thư viện SVM của sklearn. Ta có nhiều kernel để sử dụng: linear, rbf, sigmoid.
- Ngoài ra ta còn có thể áp dụng model NuSVC trong thư viện SVM, việc so sánh kết quả giữa model dự đoán từ dữ liệu học tập và kết quả chuẩn cũng tương tự như model Linear Regression.

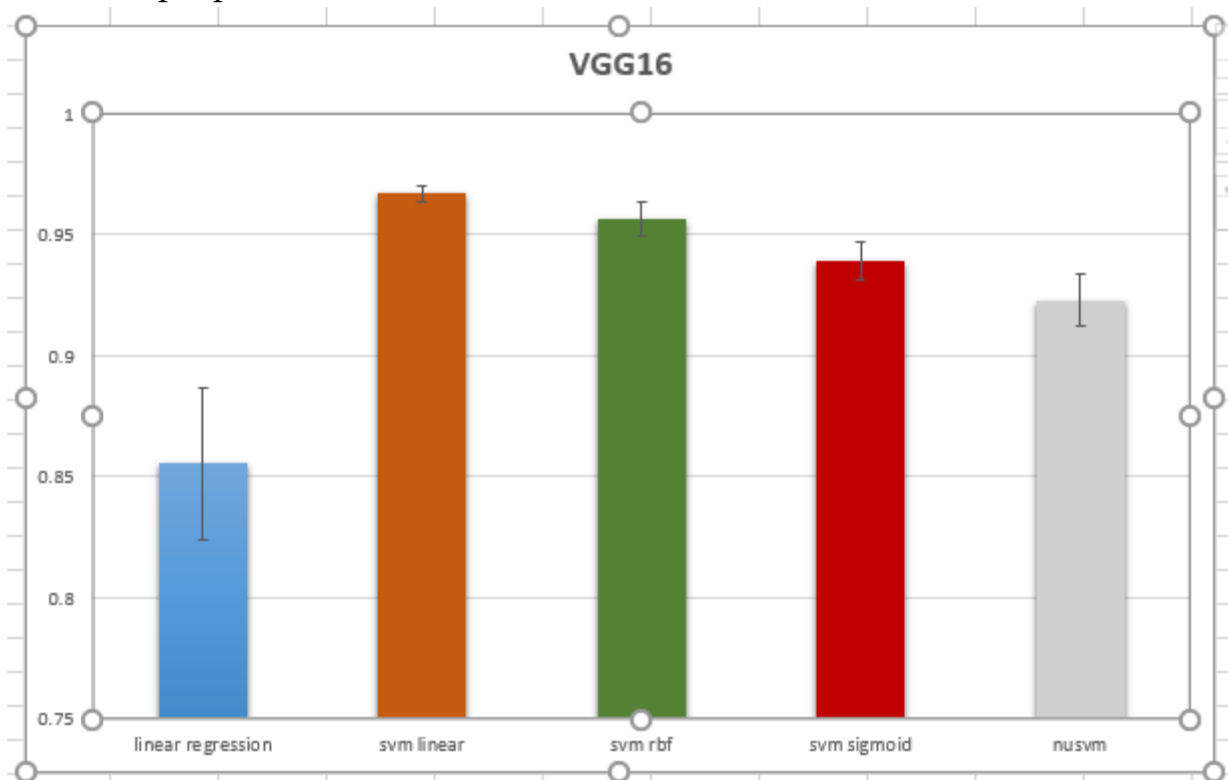
```
169
170 #FUNCTION: SVM MODEL
171 def sklearn_svm(x_train,lbx_train,x_test,lbx_test,kernel_type,kernel_input,path_result):
172     from sklearn import svm
173     from sklearn.metrics import mean_squared_error
174     x_train=x_train.tolist()
175     x_test=x_test.tolist()
176     lbx_train=(lbx_train.ravel()).tolist()
177     lbx_test=(lbx_test.ravel()).tolist()
178     file_result_txt = open(path_result+"/result.txt","a")
179
180     if kernel_type == "nonlinear":
181         clf = svm.NuSVC()
182         clf.fit(x_train, lbx_train)
183         x_pred = clf.predict(x_test)
184         # The mean squared error
185         print("NuSVC Kernel :")
186         print("Mean squared error: %.2f"
187             % mean_squared_error(lbx_test, x_pred))
188         result = 1 - mean_squared_error(lbx_test, x_pred)
189         # Explained variance score: 1 is perfect prediction
190         file_result_txt.write("SVM NUSVC MODEL:\n")
191         file_result_txt.write("Accuracy: "+str(result)+"\n")
192         file_result_txt.close()
193     else:
194         clf = svm.SVC(kernel=kernel_input)
195         clf.fit(x_train, lbx_train)
196         x_pred = clf.predict(x_test)
197         # The mean squared error
198         print("SVC Kernel " + kernel_input + " :")
199         print("Mean squared error: %.2f"
200             % mean_squared_error(lbx_test, x_pred))
201         result = 1 - mean_squared_error(lbx_test, x_pred)
202         file_result_txt.write("SVM MODEL "+ "WITH "+kernel_input+" KERNEL"+ "\n")
203         file_result_txt.write("Accuracy: "+str(result)+"\n")
204         file_result_txt.close()
205
```

V. ĐÁNH GIÁ KẾT QUẢ:

- Kết quả đạt được khi áp dụng các model trên 3 bộ dữ liệu được thống kê như sau:

2	Model	db1	db2	db3	Accuracy Score	Deviation
3	linear regression	0.811209	0.875004	0.879776	0.855329633	0.03125886
4	svm linear	0.965217	0.971739	0.96413	0.967028986	0.00336001
5	svm rbf	0.946739	0.963043	0.959783	0.956521739	0.00704428
6	svm sigmoid	0.928261	0.945652	0.943478	0.939130435	0.00773701
7	nusvm	0.908696	0.934783	0.925	0.922826087	0.01076032

- Trong đó:
 - o Accuracy Score: Độ chính xác trung bình của model (trung bình độ chính xác của 3 bộ dữ liệu)
 - o Deviation: Độ lệch chuẩn của model. Sử dụng hàm STDEV.P của Excel để tính toán.
- Ta có biểu đồ thống kê độ chính xác và độ lệch chuẩn giữa các phương pháp:



VI. KẾT LUẬN:

- Từ các số liệu và biểu đồ thống kê khi thực hiện trên các bộ dữ liệu khác nhau, ta thấy rằng model SVM với kernel linear cho ra kết quả tốt nhất (96,7%), nguyên do vì kết quả phân loại của chúng ta chỉ có 2 nhãn là 1 hoặc 0. Do đó những model có kernel là linear sẽ cho lại kết quả tốt hơn.
- Dù linear regression model vẫn sử dụng phương pháp áp dụng linear classify trong việc training và xây dựng model tuy nhiên, do input đầu vào của chúng là bộ feature của ảnh (1x4096), do đó linear regression không phát huy hết khả năng tối ưu của nó (vốn được áp dụng rất nhiều đối với dữ liệu 1 hoặc 2 chiều), trong khi đó input đầu vào của chúng ta là một ảnh có 4096 chiều vector đặc trưng.
- Material và Source Code:
https://github.com/bigredbug47/ML_Gender_Prediction_System

VII. TÀI LIỆU THAM KHẢO:

- [1] <https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html>
- [2] <https://en.wikipedia.org/wiki/Keras> - <https://keras.io/>
- [3] <https://www.tensorflow.org/>
- [4] <http://www.statisticssolutions.com/what-is-linear-regression/>
- [5][6] https://en.wikipedia.org/wiki/Support_vector_machine
- [7] <https://www.ncbi.nlm.nih.gov/pubmed/11516360>