FACE RECOGNITION

Group 1 - Annie, Constance, Cooper

TABLE OF CONTENTS

Aim and Business Values

Data & EDA

Modeling

Conclusions and Future Improvements

Next Steps

Q & A

AIM

DETECT FACES
PREDICT GENDER, AGE & EMOTION

BUSINESS VALUES

SHOPPING MALL & SHOPS - MARKETING CAMPAIGN

SMART MARKETING - ADVERTISEMENT

GAMBLING INDUSTRY SECURITY - E.G CASINO

OVERALL ARCHITECTURE

Input image

identify faces

Make predictions

age prediction

gender prediction

emotion prediction

Render predictions to screen

DATA

DATA STRUCTURE

EDA

DATA

1/ UTKFACE

Gender, age, race

20k+ cropped face images

male and female

age 1-116

5 ethnicities

2/ EMOTION DATASET

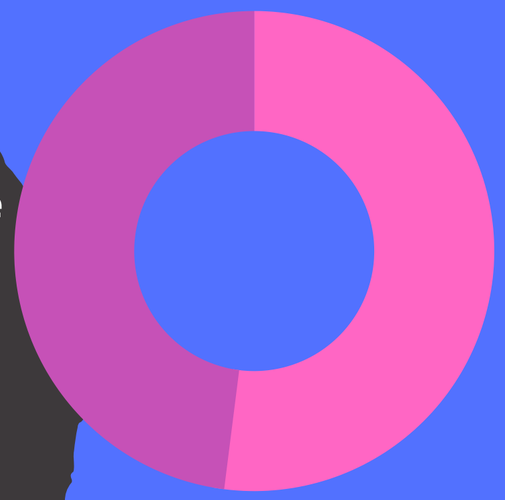
35k+ cropped face images

Contains 7 emotions (happiness, neutral, sadness, anger, surprise, disgust, fear)

48x48 pixel

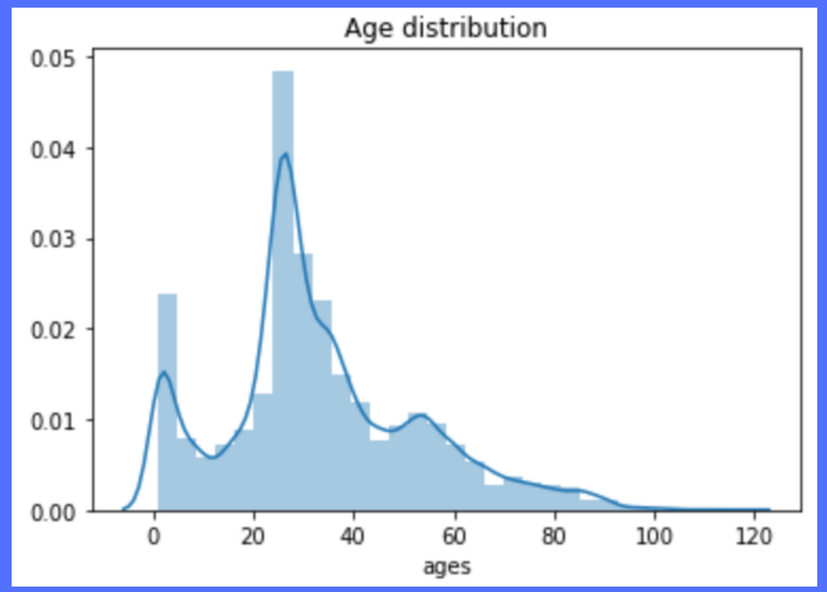
EDA

Gender



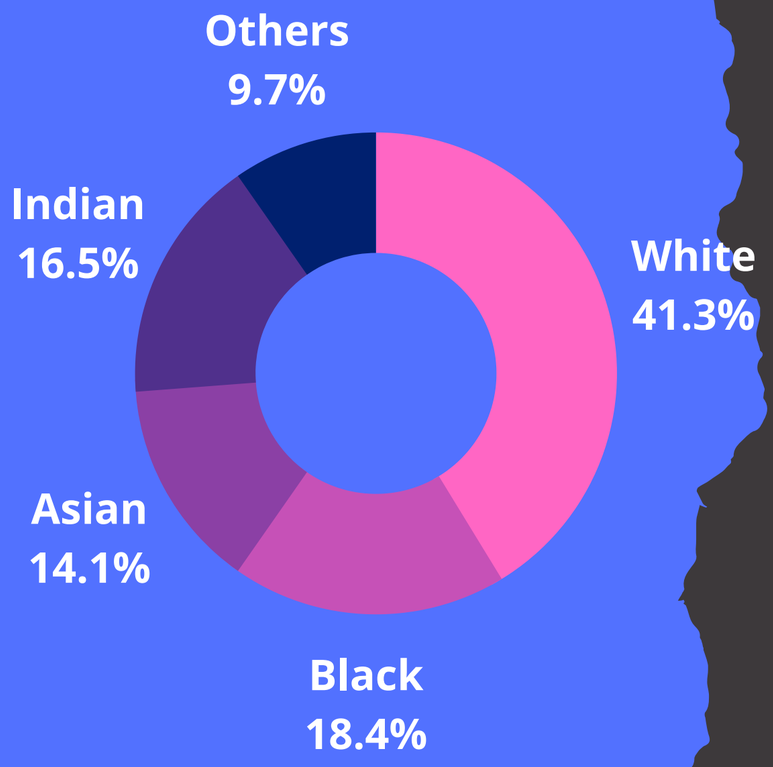
0 - Male - 12,391
1 - Female - 11,314

Age



Age distribution:
Majority age group: around 30
Skewed dataset

Race



0 - White - 10,078
1 - Black - 4,526
2 - Asian - 3,975
3 - Indian - 3,434
4 - Others - 1,692

MODELLING

GENDER, AGE

FACE DETECTION - MTCNN

EMOTION

GENDER PREDICTION

random sample 5000 images

save training time

2 classes

'Male' and 'Female'

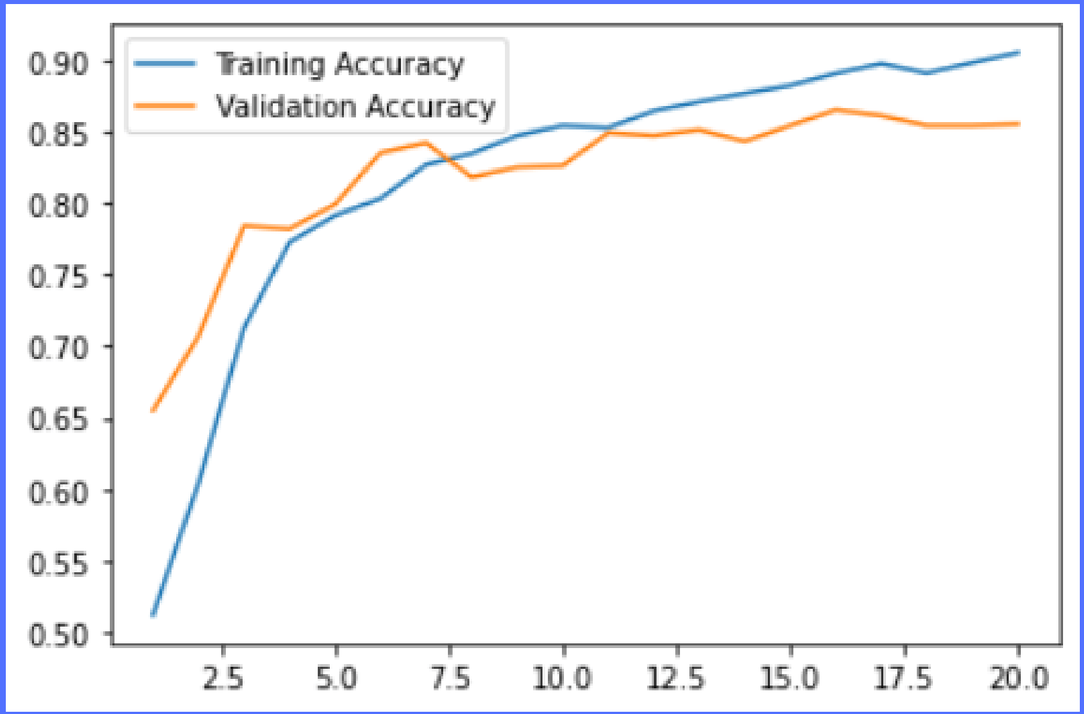
```
gender_model=Sequential()
gender_model.add(Conv2D(32,(3, 3),activation='relu', input_shape=(64,64,3)))
gender_model.add(Conv2D(32,(3, 3),activation='relu'))
gender_model.add(MaxPooling2D(2, 2))
gender_model.add(Conv2D(32,(3, 3),activation='relu'))
gender_model.add(Conv2D(32,(3, 3),activation='relu'))
gender_model.add(MaxPooling2D(2, 2))
gender_model.add(Conv2D(64,(3, 3),activation='relu'))
gender_model.add(Conv2D(64,(3, 3),activation='relu'))
gender_model.add(MaxPooling2D(2, 2))
gender_model.add(Dropout(0.25))
gender_model.add(Flatten())

gender_model.add(Dense(128, activation='relu'))
gender_model.add(Dense(64, activation='relu'))
gender_model.add(Dense(32, activation='relu'))
gender_model.add(Dropout(0.5))
gender_model.add(Dense(1, activation='sigmoid'))
gender_model.compile(optimizer='adam', loss ='binary_crossentropy', metrics=['accuracy'])
```

```
from tensorflow.keras import callbacks
early_stop=callbacks.EarlyStopping(monitor='val_loss', patience=3)
hist=gender_model.fit(X_train, y_train, epochs=20, batch_size=20, validation_data=(X_test, y_test),callbacks=[early_stop])
```

GENDER PREDICTION

Test accuracy = ~0.85

slightly overfit



AGE PREDICTION

random sample 5000 images

save training time

10 classes

'1-2', '3-11', '12-17', '18-24', '25-34', '35-44', '45-54', '55-64', '65-80', '81-116'

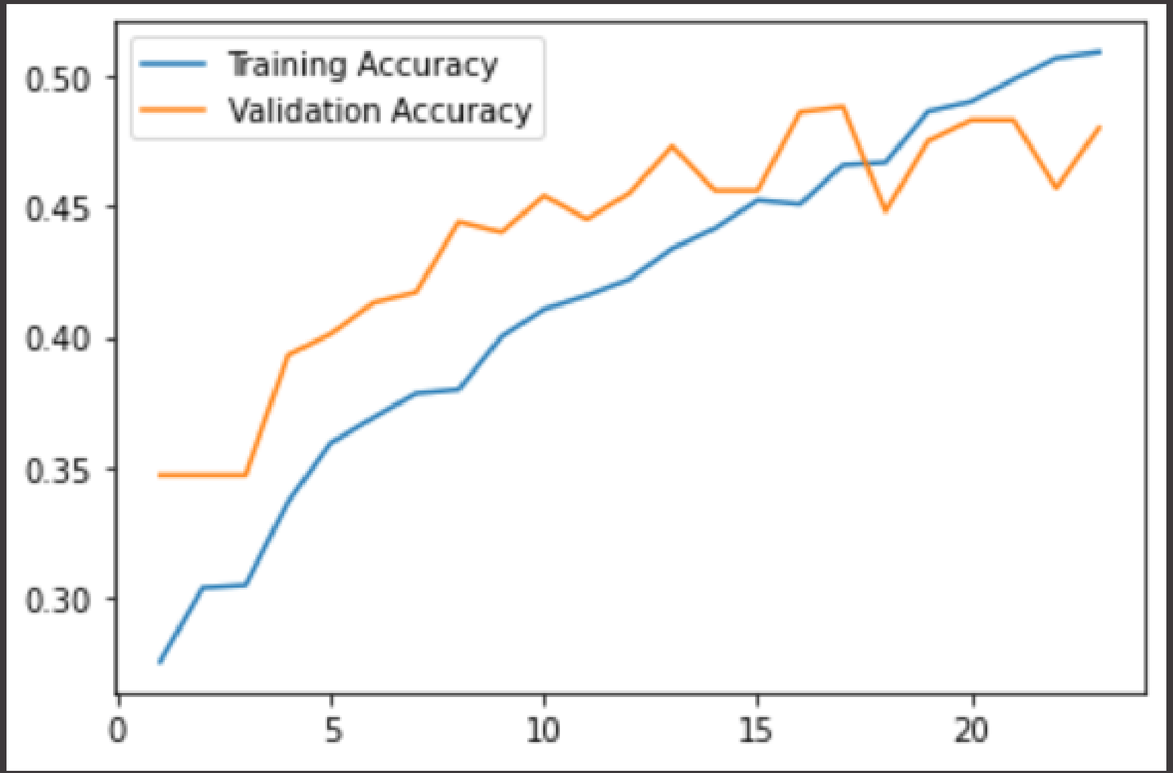
```
age_model=Sequential()
age_model.add(Conv2D(32,(3, 3),activation='relu', input_shape=(64,64,3)))
age_model.add(MaxPooling2D(2, 2))
age_model.add(Conv2D(64,(3, 3),activation='relu'))
age_model.add(MaxPooling2D(2, 2))
age_model.add(Conv2D(128,(3, 3),activation='relu'))
age_model.add(MaxPooling2D(2, 2))
age_model.add(Conv2D(256,(3, 3),activation='relu'))
age_model.add(MaxPooling2D(2, 2))
age_model.add(Dropout(0.25))
age_model.add(Flatten())

age_model.add(Dense(128, activation='relu'))
age_model.add(Dense(64, activation='relu'))
age_model.add(Dense(32, activation='relu'))
age_model.add(Dropout(0.5))
age_model.add(Dense(10, activation='softmax'))
age_model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001), loss ='categorical_crossentropy', metrics=['accuracy']
```

```
from tensorflow.keras import callbacks
early_stop=callbacks.EarlyStopping(monitor='val_loss', patience=3)
hist=age_model.fit(X_train, y_train, epochs=50, batch_size=20, validation_data=(X_test, y_test),callbacks=[early_stop])
```

AGE PREDICTION

Test accuracy = ~0.46

slightly overfit

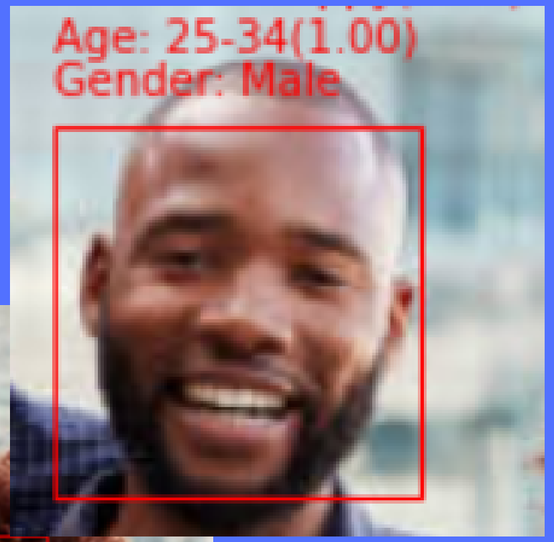


MTCNN MODEL

```
def face_detection(path, faces):
  data=pyplot.imread(path)
  pyplot.imshow(data)
  fig=pyplot.gcf()
  fig.set_size_inches(15,10)
  ax=pyplot.gca()
  for face in faces:
    x,y,width,height=face['box']
    # center alignment
    center=[x+(width/2),y+(height/2)]
    max_border=max(width,height)
    left=max(int(center[0]-(max_border/2)),0)
    right=max(int(center[0]+(max_border/2)),0)
    top=max(int(center[1]-(max_border/2)),0)
    bottom=max(int(center[1]+(max_border/2)),0)
    # crop the face
    cropped_image_i=data[top:top+max_border,left:left+max_border,:]
    # resize image to fit the age and gender model
    cropped_image=np.array(Image.fromarray(cropped_image_i).resize([64,64]))
    # resize image to fit emo model
    cropped_image_emo=np.array(Image.fromarray(cropped_image_i).resize([48,48]))
    cropped_image_emo=cv2.cvtColor(cropped_image_emo,cv2.COLOR_BGR2GRAY)
    cropped_image_emo=np.expand_dims(cropped_image_emo,0)
    # create predictions
    gender_pred=gender_model.predict(cropped_image.reshape(1,64,64,3))
    age_pred=age_model.predict(cropped_image.reshape(1,64,64,3))
    emo_pred=emo_model.predict(cropped_image_emo.reshape(1,48,48,1))
    # create the box around the face
    rect=Rectangle((left,top),max_border,max_border,fill=False,color='red')
    # add the box
    ax.add_patch(rect)
    # add gender prediction
    gender_text='Female' if gender_pred > 0.5 else 'Male'
    ax.text(left,top-(image.shape[0]*0.014),'Gender: {}'.format(gender_text),fontsize=10,color='red')
    # # add age prediction
    age_index=int(np.argmax(age_pred))
    age_confident=age_pred[0][age_index]
    age_text=age_labels[age_index]
    ax.text(left,top-(image.shape[0]*0.033),'Age: {}({:.2f})'.format(age_text,age_confident),fontsize=10,color='red')
    # add emotion prediction
    emo_index=np.argmax(emo_pred)
    emo_confident=np.max(emo_pred)
    emo_text=emo_dict[emo_index]
    ax.text(left, top-(image.shape[0]*0.055),'Emotion: {}({:.2f})'.format(emo_text,emo_confident),fontsize=10,color='red')
  # show the resulting image
  pyplot.show()
```

FACE DETECTION

MTCNN



EMOTION PREDICTION

```python
train_dir = "train" #passing the path with training images
test_dir = "test"   #passing the path with testing images
img_size = 48 #original size of the image
```

```python
train_datagen = ImageDataGenerator(rotation_range = 22,
                                   width_shift_range = 0.1,
                                   height_shift_range = 0.1,
                                   horizontal_flip = True,
                                   rescale = 1./255,
                                   #zoom_range = 0.2,
                                   validation_split = 0.2
                                  )
validation_datagen = ImageDataGenerator(rescale = 1./255,
                                   validation_split = 0.2)
```

```python
train_generator = train_datagen.flow_from_directory(directory = train_dir,
                                                    target_size = (img_size,img_size),
                                                    batch_size = 64,
                                                    color_mode = "grayscale",
                                                    class_mode = "categorical",
                                                    subset = "training"
                                                   )
validation_generator = validation_datagen.flow_from_directory( directory = test_dir,
                                                               target_size = (img_size,img_size),
                                                               batch_size = 64,
                                                               color_mode = "grayscale",
                                                               class_mode = "categorical",
                                                               subset = "validation"
                                                              )
```

```
Found 10093 images belonging to 3 classes.
Found 622 images belonging to 3 classes.
```

3 Classes:

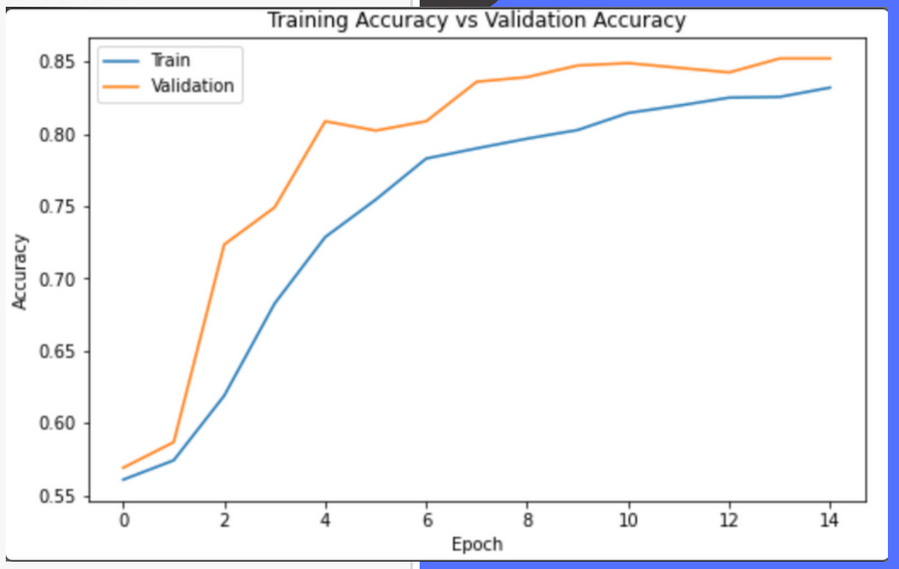
- Disgust
- Neutral
- Happy

EMOTION PREDICTION

```python
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(48,48,1)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, kernel_size=(3,3), activation='relu' ))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())

model.add(Dense(256, activation='relu'))
model.add(Dropout(0.25))

model.add(Dense(512, activation='relu'))
model.add(Dropout(0.25))

model.add(Dense(3, activation='softmax'))

model.compile(optimizer = Adam(lr=0.0001),loss='categorical_crossentropy', metrics=['accuracy'])
```

```python
epochs = 15
batch_size = 64
```

```python
history = model.fit(x = train_generator,epochs = epochs,validation_data = validation_generator)
```



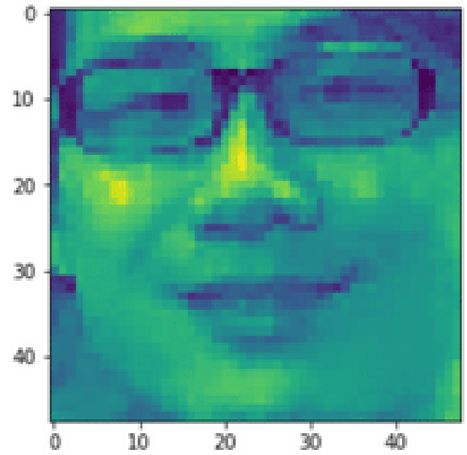
Training accuracy = ~0.83

Validation accuracy = ~0.85

EMOTION PREDICTION

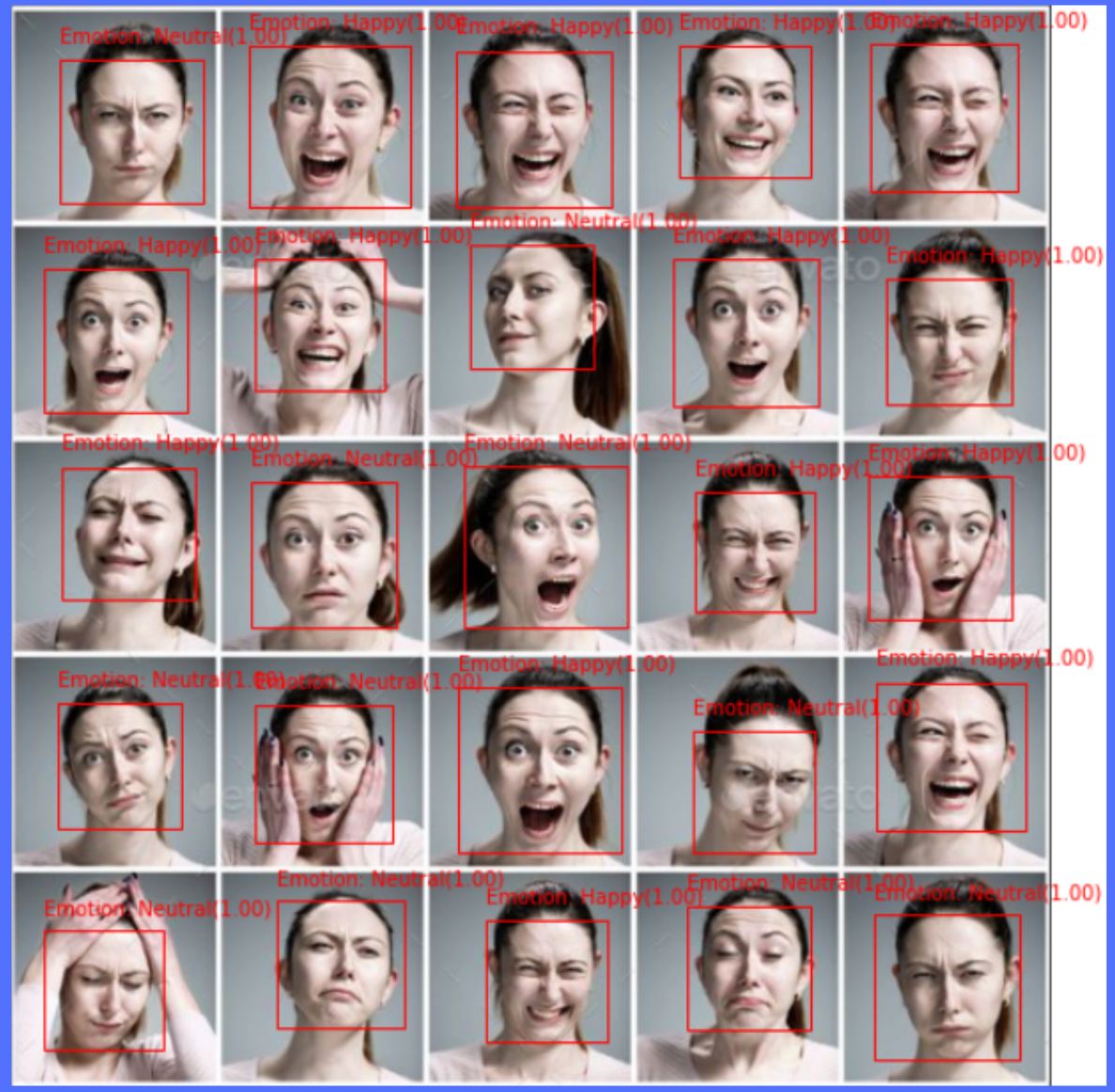
```python
label_dict = {0:'Disgust',1:'Happy',2:'Neutral'}
# img = image.load_img("pic(614, 2636, 800, 2450).jpg",target_size = (48,48),color_mode = "grayscale")
img = image.load_img("pic(684, 1758, 813, 1629).jpg",target_size = (48,48),color_mode = "grayscale")
img = np.array(img)
plt.imshow(img)
img = np.expand_dims(img,axis = 0) #makes image shape (1,48,48)
img = img.reshape(1,48,48,1)
result = emo_model.predict(img)
result = list(result[0])
print(result)
img_index = result.index(max(result))
print("Emotion: ",label_dict[img_index])
```

```
[0.0, 1.0, 0.0]
Emotion:  Happy
```



Final result

Final result - Emotion



age prediction on elderly

Emotion prediction on ”disgust”

CONCLUSIONS & FUTURE IMPROVEMENTS

CONCLUSIONS

GENDER(85%), AGE(46%), EMOTION(83%)

ambiguous for human

DATASET SELECTION

Skewed data on Age and Emotion

CATEGORICAL CLASSIFICATION - AGE & EMOTION

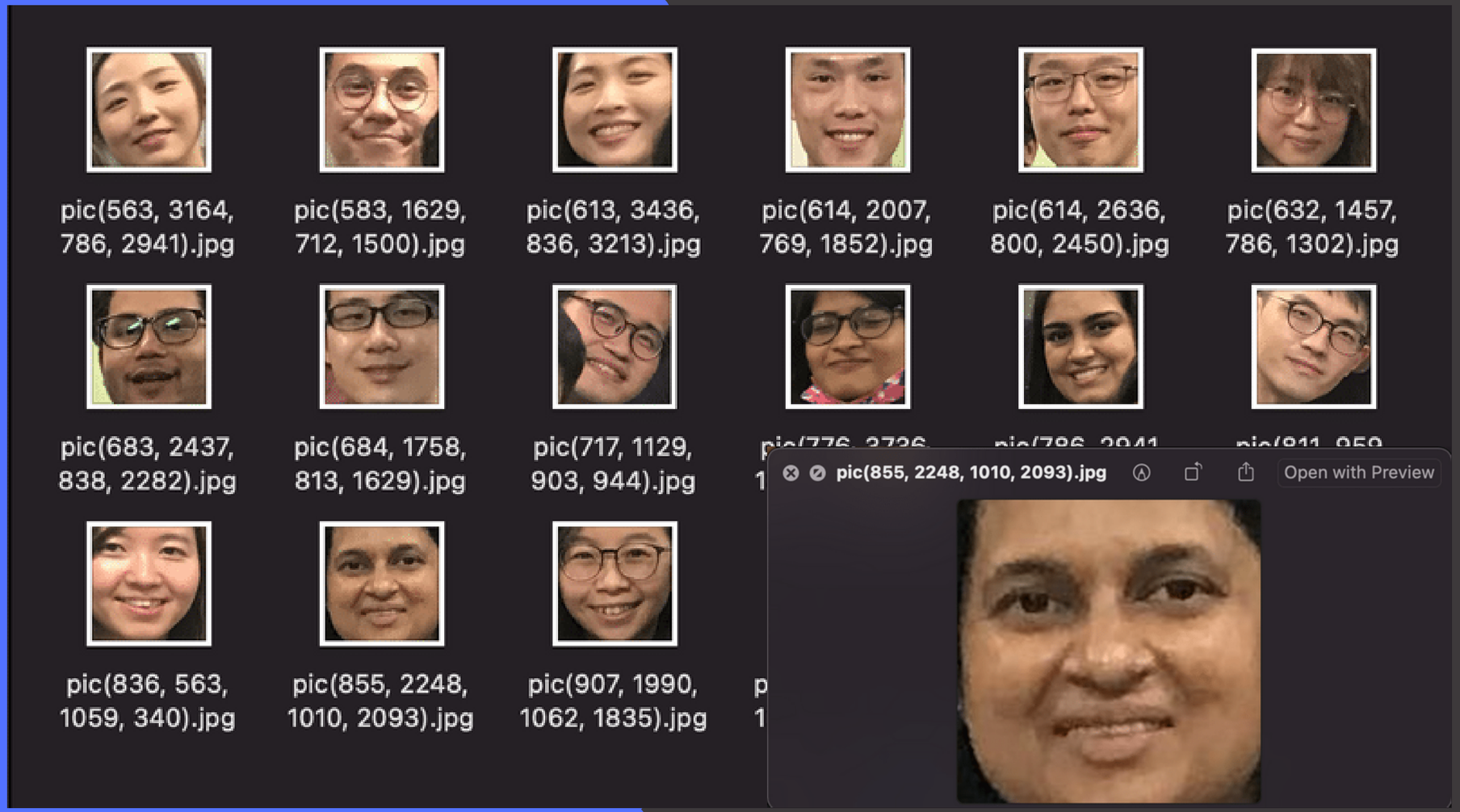
The fewer the classes, the more accurate the model can be

TECHNICAL ISSUES

Libraries installation, training time, deployment of Streamlit

NEXT STEP

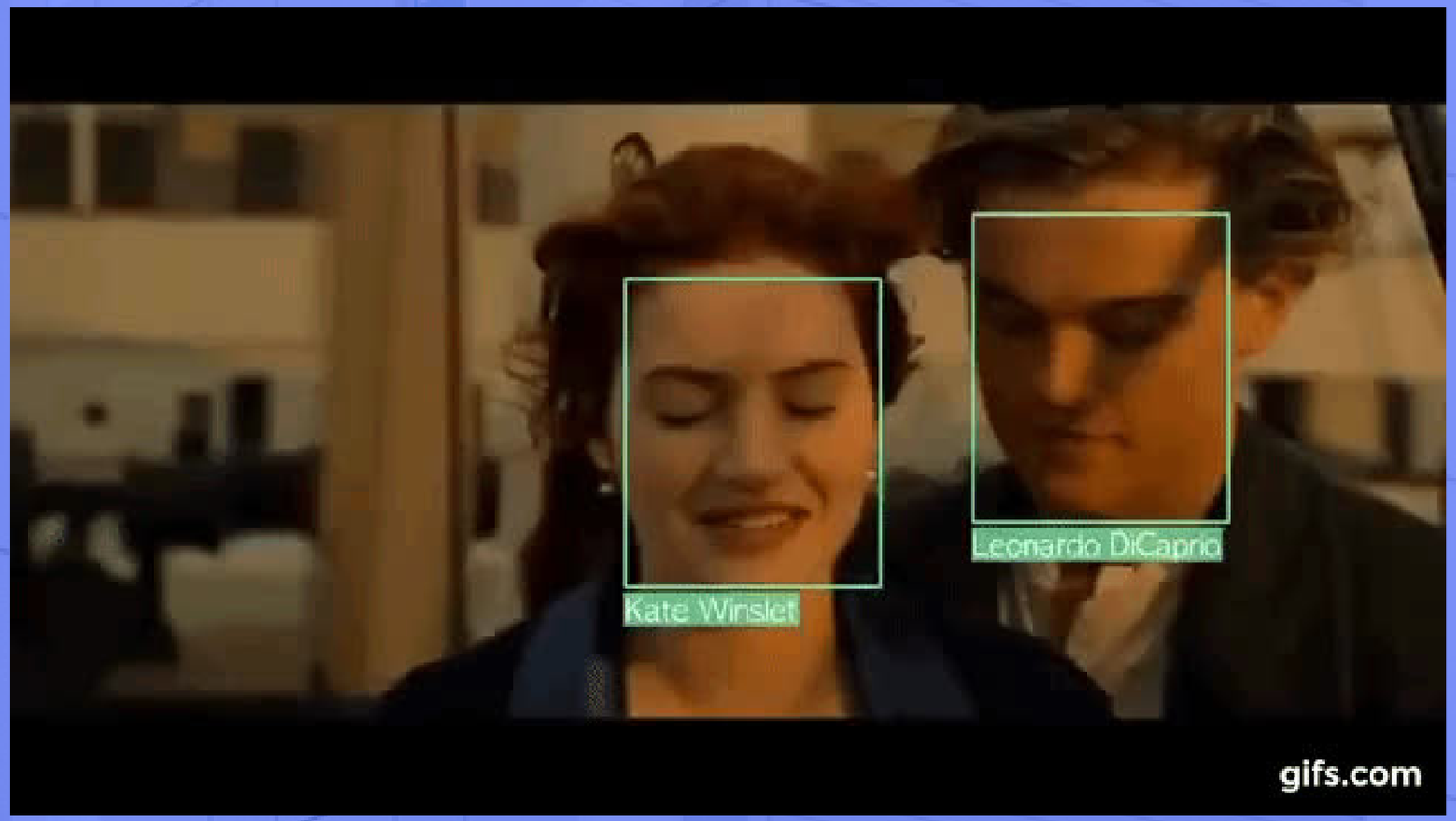
Crop and save labelled data



Label data include: Gender, Age and Emotion

NEXT STEP

apply on real time video



source: https://www.youtube.com/watch?v=lofQMB0LWJ0

NEXT STEPS

Apply Transfer Learning to enhance accuracy e.g. VGGFace

Race prediction model

THANK YOU

Q & A