Importing libraries and datafiles

In [2]:

```
import pandas as pd
import numpy as np
x =  pd.read_csv("C:\\Users\\hp\\Downloads\\micro_high_res_every_frame.csv")
y = pd.read_csv("C:\\Users\\hp\\Downloads\\updated_fer.csv")
```

Conv Fer to range from 1- 100

In [3]:

```
y[["angry","disgust","fear","happy","sad","surprise","neutral"]] *= 100
```

1) Compute model averaged sentiment scores for each individual sentiment type from different models (FER, DeepFace etc)

In [4]:

```
# fer_mean = y.mean(axis=0)
deep_mean = x[["df_angry","df_disgust","df_fear","df_happy","df_sad","df_surprise","df_
# fer_mean = pd.DataFrame(fer_mean)
deep_mean = pd.DataFrame(deep_mean)
```

In [5]:

```
# fer_mean=fer_mean.iloc[1:]
# fer_mean=fer_mean.T
# print(fer_mean)
```

In [6]:

```
deep_mean = deep_mean.T
print(deep_mean)
```

```
   df_angry  df_disgust    df_fear    df_happy     df_sad  df_surprise  \
0   7.47512    0.134416  15.583923  12.856685  18.135053     3.090492

   df_neutral
0   42.724311
```

Compute Storyline negativity scores for StoryTIME and screenPLAY.

In [7]:

```
DF_Overall_Negativitys_core = deep_mean[["df_angry","df_disgust","df_fear","df_sad"]].m
DF_Overall_Negativitys_core = pd.DataFrame(DF_Overall_Negativitys_core)
DF_Overall_Negativitys_core.columns = ["DF_Overall_Negativitys_core"]
DF_Overall_Negativitys_core.head()
```

Out[7]:

	DF_Overall_Negativitys_core
0	10.332128

In [8]:

```
DF_Overall_Aggression_score = deep_mean[["df_angry","df_disgust"]].mean(axis=1)
DF_Overall_Aggression_score = pd.DataFrame(DF_Overall_Aggression_score)
```

```
DF_Overall_Aggression_score.columns = ["DF_Overall_Aggression_score"]
print(DF_Overall_Aggression_score)
```

```
   DF_Overall_Aggression_score
0                     3.804768
```

In [9]:
```
DF_Depression_score = deep_mean[["df_sad","df_fear"]].mean(axis=1)
DF_Depression_score = pd.DataFrame(DF_Depression_score)
DF_Depression_score.columns = ["DF_Depression_score"]
print(DF_Depression_score)
```

```
   DF_Depression_score
0            16.859488
```

Compute Storyline.face.sentiment score for each frame (row) by dividing the happiness score by the sum of the happiness + negative sentiment scores. This score shows the percentage of net face sentiment that is positive (0% to 100%). Put into StoryTIME.

In [10]:
```
Storyline_face_sentiment = (x[["df_happy"]] /(x[["df_happy"]] + DF_Overall_Negativitys_
Storyline_face_sentiment.columns = ["Storyline_face_sentiment"]
Storyline_face_sentiment.head()
```

Out[10]:

	Storyline_face_sentiment
0	0.000036
1	34.708970
2	0.000008
3	0.000011
4	0.000007

Add Core Facial expression scores to storyTIME and screenPLAY

In [11]:
```
Core_Expression_Scores_DeepFace_Model_y = pd.concat([deep_mean,DF_Overall_Negativitys_c
Core_Expression_Scores_DeepFace_Model_y.head()
```

Out[11]:

	df_angry	df_disgust	df_fear	df_happy	df_sad	df_surprise	df_neutral	DF_Overall_Negativitys_
0	7.47512	0.134416	15.583923	12.856685	18.135053	3.090492	42.724311	10.33
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

Save DataFrame in CSV

```
In [12]: # Core_Expression_Scores_DeepFace_Model_y.to_csv (r"D:\Master's\Intern\Core_Expression_
```

*** Ignore this part as it was used for earlier version of the doc**

```
In [13]: # avg_modal = pd.DataFrame(np.vstack([fer_mean, deep_mean]), columns = list(fer_mean.co
         # avg_modal.head()
```

```
In [14]: # avg_modal = avg_modal.mean(axis=0)
         # print(avg_modal)
```

2) Define positivity score for each frame (row) in each video at 24 fps from the average of smiling and happiness scores from different models. (rows)

```
In [15]: # fer = y[['angry',   'disgust',      'fear',     'happy',       'sad',  'surprise',
         # deep =  x[["df_angry","df_disgust","df_fear","df_happy","df_sad","df_surprise","df_ne
         # fer.head()
```

```
In [16]: # deep.head()
```

```
In [17]: # deep = deep.rename({'df_happy': 'happy','df_angry':'angry','df_disgust':'disgust','df_
         # deep.head()
```

```
In [18]: # positive_score=pd.concat([deep[["happy"]],fer[["happy"]]],axis=1)
         # positive_score.head()
```

```
In [19]: # positive_score['positive_score'] = positive_score.mean(axis=1)
         # positive_score.head()
```

3)Compute negativity score from the average of sadness, anger, disgust, contempt and fear scores from different models for each frame (row) in each video at 24 fps. (row)

```
In [20]: # negative_score=pd.concat([deep[["angry","disgust","sad","fear"]],fer[["angry","disgus
         # negative_score['negative_score'] = negative_score.mean(axis=1)
         # negative_score.head()
```

4)Compute face.sentiment score for each frame (row) by dividing the positivity score by the sum of the positive + negative sentiment scores. This score shows the percentage of net face sentiment that is positive (0% to 100%).

In [21]:

```
# face_sentiment_score = positive_score['positive_score'] / negative_score['negative_sc
```

In [22]:

```
# face_sentiment_score["face_sentiment_score"]= pd.DataFrame(face_sentiment_score)
```

*** Ignore till here as it was used for earlier version of the doc**

Face Alphabet - Facial Sentiment Analysis Algorithm

Compute the question (Q) level facial sentiment score from all frames for a given question

In [23]:

```
facial_sentimental_report = x[["subject_id","assessment_id","question_id","question_typ
facial_sentimental_report.head()
```

Out[23]:

	subject_id	assessment_id	question_id	question_type	main_question	start_answer_datetime	end_
0	63edbeb0-b664-4b51-a9c8-cfe648e5f3ac	4d86c48f-8a2e-4366-84dc-bf9a2e58223f	958a18c2-67d5-48e9-8464-9f2615a3f84f	video	Click begin, stare into the camera, then look ...	2022-02-07T19:26:16.298000Z	07T
1	63edbeb0-b664-4b51-a9c8-cfe648e5f3ac	4d86c48f-8a2e-4366-84dc-bf9a2e58223f	958a18c2-67d5-48e9-8464-9f2615a3f84f	video	Click begin, stare into the camera, then look ...	2022-02-07T19:26:16.298000Z	07T
2	63edbeb0-b664-4b51-a9c8-cfe648e5f3ac	4d86c48f-8a2e-4366-84dc-bf9a2e58223f	958a18c2-67d5-48e9-8464-9f2615a3f84f	video	Click begin, stare into the camera, then look ...	2022-02-07T19:26:16.298000Z	07T
3	63edbeb0-b664-4b51-a9c8-cfe648e5f3ac	4d86c48f-8a2e-4366-84dc-bf9a2e58223f	958a18c2-67d5-48e9-8464-9f2615a3f84f	video	Click begin, stare into the camera, then look ...	2022-02-07T19:26:16.298000Z	07T

	subject_id	assessment_id	question_id	question_type	main_question	start_answer_datetime	end_
4	63edbeb0-b664-4b51-a9c8-cfe648e5f3ac	4d86c48f-8a2e-4366-84dc-bf9a2e58223f	958a18c2-67d5-48e9-8464-9f2615a3f84f	video	Click begin, stare into the camera, then look ...	2022-02-07T19:26:16.298000Z	07T

In [24]:

```
facial_sentimental_report_Q = facial_sentimental_report.groupby(['question_id']).sum()
Q_Face_Sentiment_Score_QuestionID = facial_sentimental_report_Q[["df_happy"]]/(facial_s

Q_Face_Sentiment_Score_QuestionID.columns = ["Q_Face_Sentiment_Score_QuestionID"]
print(Q_Face_Sentiment_Score_QuestionID)
```

```
                                      Q_Face_Sentiment_Score_QuestionID
question_id
1924574b-dc9a-4822-8f25-28943d0531a9                           0.998039
237e6469-5bfc-434f-81c8-afba91956066                           0.999840
533ad77a-8cd0-4201-8254-d552a679f3c3                           0.999968
546d29a9-c6eb-43ae-8df5-f106c781a8b4                           0.999901
7272d619-e9ef-4b44-98dd-9b8eb4530151                           0.999820
74490236-9e9e-4f0e-ab37-4c72ca21e15e                           0.999812
8b4362d5-8af2-4875-bcd8-6ffe06c30228                           0.999806
958a18c2-67d5-48e9-8464-9f2615a3f84f                           0.998910
cb52904c-1edb-4922-bf84-8a7b50ce2b3e                           0.999809
ef51d3ee-6709-49d7-9819-07ba799205ea                           0.999623
```

Compute the assessment (A) level facial sentiment score from all frames for a given assessment

In [25]:

```
facial_sentimental_report_A = facial_sentimental_report.groupby(['assessment_id']).sum(
A_Face_Sentiment_Score_AssessmentID = facial_sentimental_report_A[["df_happy"]]/(facial_
A_Face_Sentiment_Score_AssessmentID.columns = ["A_Face_Sentiment_Score_AssessmentID"]
print(A_Face_Sentiment_Score_AssessmentID)
```

```
                                      A_Face_Sentiment_Score_AssessmentID
assessment_id
0df79265-944e-43f9-a0bd-39d8e5757a6d                             0.999919
23222d65-20dc-4247-8495-8b98f52c2c6f                             0.999726
2d03cee3-2af3-4b99-b97f-9280f81ae964                             0.999799
4143b281-9e12-4f3c-9da9-6ee7d3f8c060                             0.998748
4d86c48f-8a2e-4366-84dc-bf9a2e58223f                             0.999849
5878ccf2-052a-400b-8682-ed7b9db29f50                             0.999814
5e703137-4029-4f52-b5de-b200751e87cf                             0.997853
5e7c6635-e024-4268-83ad-590016cd1046                             0.998920
664554ac-ff5a-4d3c-af27-c6c09ee9f816                             0.998007
6b6ee892-6d2e-45c2-a44f-9cd50ae436a8                             0.999721
b8409cdc-bf94-4f3a-a2d3-ffb029f2e965                             0.999150
c3e3616b-61ca-48c2-9f2c-8325825f0782                             0.998158
cac929f6-fb4c-4415-845b-ed42dec63a86                             0.999906
d5968d42-783a-4dfe-9af7-2bb284230a9d                             0.999783
d6889be5-3d64-4293-8049-53466ca2329a                             0.984145
dcc4a978-da9b-48d6-9142-3a7f5e90306a                             0.999471
f7bab369-0fbd-4bfc-85d6-2452c945be0f                             0.999802
f7f724bb-9348-4648-bf4c-b21c66604931                             0.999744
f94450cf-c403-4142-a8be-85ffa46ec49c                             0.999715
fa3846ac-f7a5-4991-bc6a-3a89236641e2                             0.999492
```

Compute the total emotion score by summing the scores for each micro-expression for every frame by question and assessment.

In [26]:

```
total_Facial_Emotion_Score_questionID = facial_sentimental_report.groupby(['question_id
total_Facial_Emotion_Score_questionID_x = total_Facial_Emotion_Score_questionID.sum(axi

total_Facial_Emotion_Score_questionID_x = pd.DataFrame(total_Facial_Emotion_Score_quest
total_Facial_Emotion_Score_questionID_x = total_Facial_Emotion_Score_questionID_x.renam
total_Facial_Emotion_Score_questionID_x = total_Facial_Emotion_Score_questionID_x.assig

total_Facial_Emotion_Score_questionID_x
```

Out[26]:

	question_id	total_Facial_Emotion_Score_questionID
Index		
0	1924574b-dc9a-4822-8f25-28943d0531a9	557000.003706
1	237e6469-5bfc-434f-81c8-afba91956066	478700.003308
2	533ad77a-8cd0-4201-8254-d552a679f3c3	674900.003909
3	546d29a9-c6eb-43ae-8df5-f106c781a8b4	658800.004111
4	7272d619-e9ef-4b44-98dd-9b8eb4530151	649200.003992
5	74490236-9e9e-4f0e-ab37-4c72ca21e15e	651700.003827
6	8b4362d5-8af2-4875-bcd8-6ffe06c30228	651300.003510
7	958a18c2-67d5-48e9-8464-9f2615a3f84f	335100.002351
8	cb52904c-1edb-4922-bf84-8a7b50ce2b3e	699800.003251
9	ef51d3ee-6709-49d7-9819-07ba799205ea	480600.003004

In [27]:

```
total_Facial_Emotion_Score_assessmentID = facial_sentimental_report.groupby(['assessmen
total_Facial_Emotion_Score_assessmentID_x = total_Facial_Emotion_Score_assessmentID.sum
total_Facial_Emotion_Score_assessmentID_x = pd.DataFrame(total_Facial_Emotion_Score_ass
total_Facial_Emotion_Score_assessmentID_x = total_Facial_Emotion_Score_assessmentID_x.r
total_Facial_Emotion_Score_assessmentID_x = total_Facial_Emotion_Score_assessmentID_x.a

total_Facial_Emotion_Score_assessmentID_x
```

Out[27]:

	assessment_id	total_Facial_Emotion_Score_assessmentID
Index		
0	0df79265-944e-43f9-a0bd-39d8e5757a6d	398300.002575
1	23222d65-20dc-4247-8495-8b98f52c2c6f	512000.002504
2	2d03cee3-2af3-4b99-b97f-9280f81ae964	322700.002331
3	4143b281-9e12-4f3c-9da9-6ee7d3f8c060	105700.000612
4	4d86c48f-8a2e-4366-84dc-bf9a2e58223f	515900.002427
5	5878ccf2-052a-400b-8682-ed7b9db29f50	309900.002254

	assessment_id	total_Facial_Emotion_Score_assessmentID
Index		
6	5e703137-4029-4f52-b5de-b200751e87cf	30100.000218
7	5e7c6635-e024-4268-83ad-590016cd1046	238100.001048
8	664554ac-ff5a-4d3c-af27-c6c09ee9f816	63200.000575
9	6b6ee892-6d2e-45c2-a44f-9cd50ae436a8	341600.002166
10	b8409cdc-bf94-4f3a-a2d3-ffb029f2e965	76400.000333
11	c3e3616b-61ca-48c2-9f2c-8325825f0782	86600.000139
12	cac929f6-fb4c-4415-845b-ed42dec63a86	522800.003276
13	d5968d42-783a-4dfe-9af7-2bb284230a9d	346000.001779
14	d6889be5-3d64-4293-8049-53466ca2329a	291800.001568
15	dcc4a978-da9b-48d6-9142-3a7f5e90306a	195500.001580
16	f7bab369-0fbd-4bfc-85d6-2452c945be0f	398600.002627
17	f7f724bb-9348-4648-bf4c-b21c66604931	436300.002982
18	f94450cf-c403-4142-a8be-85ffa46ec49c	320100.001900
19	fa3846ac-f7a5-4991-bc6a-3a89236641e2	325500.002076

Compute the following Storyline Net Emotion scores for each question by summing the scores for each micro-expression across all frames by question

In [28]:

```
Storyline_Net_Emotion_scores = facial_sentimental_report.groupby(['question_id']).sum()

Storyline_angry_Sum_questionID = Storyline_Net_Emotion_scores[['df_angry']].reset_index
Storyline_angry_Sum_questionID.columns = ["Storyline_angry_Sum_questionID"]
Storyline_angry_Sum_questionID = Storyline_angry_Sum_questionID.assign(Index=range(len(

Storyline_disgust_Sum_questionID = Storyline_Net_Emotion_scores[['df_disgust']].reset_i
Storyline_disgust_Sum_questionID.columns = ["Storyline_disgust_Sum_questionID"]
Storyline_disgust_Sum_questionID = Storyline_disgust_Sum_questionID.assign(Index=range(

Storyline_fear_Sum_questionID = Storyline_Net_Emotion_scores[['df_fear']].reset_index(d
Storyline_fear_Sum_questionID.columns = ["Storyline_fear_Sum_questionID"]
Storyline_fear_Sum_questionID = Storyline_fear_Sum_questionID.assign(Index=range(len(St

Storyline_happy_Sum_questionID = Storyline_Net_Emotion_scores[['df_happy']].reset_index
Storyline_happy_Sum_questionID.columns = ["Storyline_happy_Sum_questionID"]
Storyline_happy_Sum_questionID = Storyline_happy_Sum_questionID.assign(Index=range(len(

Storyline_sad_Sum_questionID = Storyline_Net_Emotion_scores[['df_sad']].reset_index(dro
Storyline_sad_Sum_questionID.columns = ["Storyline_sad_Sum_questionID"]
Storyline_sad_Sum_questionID = Storyline_sad_Sum_questionID.assign(Index=range(len(Stor
Storyline_surprise_Sum_questionID = Storyline_Net_Emotion_scores[['df_surprise']].reset_
Storyline_surprise_Sum_questionID.columns = ["Storyline_surprise_Sum_questionID"]
Storyline_surprise_Sum_questionID = Storyline_surprise_Sum_questionID.assign(Index=rang
Storyline_Neutral_Sum_questionID = Storyline_Net_Emotion_scores[['df_neutral']].reset_i
Storyline_Neutral_Sum_questionID.columns = ["Storyline_Neutral_Sum_questionID"]
Storyline_Neutral_Sum_questionID = Storyline_Neutral_Sum_questionID.assign(Index=range(

Storyline_Neutral_Sum_questionID
```

Out[28]:

	Storyline_Neutral_Sum_questionID
Index	
0	424780.119243
1	249933.024604
2	168241.791554
3	270663.968314
4	219138.555276
5	290114.336943
6	237673.710369
7	176693.703525
8	244428.535007
9	212192.985795

Compute the following Storyline Net Emotion scores for each assessment by summing the scores for each micro-expression across all frames in an assessment

In [29]:

```
Storyline_Net_Emotion_scores_a = facial_sentimental_report.groupby(['assessment_id']).s

Storyline_angry_Sum_assessmentID = Storyline_Net_Emotion_scores_a[['df_angry']].reset_i
Storyline_angry_Sum_assessmentID.columns = ["Storyline_angry_Sum_assessmentID"]
Storyline_angry_Sum_assessmentID = Storyline_angry_Sum_assessmentID.assign(Index=range(

Storyline_disgust_Sum_assessmentID = Storyline_Net_Emotion_scores_a[['df_disgust']].res
Storyline_disgust_Sum_assessmentID.columns = ["Storyline_disgust_Sum_assessmentID"]
Storyline_disgust_Sum_assessmentID = Storyline_disgust_Sum_assessmentID.assign(Index=ra

Storyline_fear_Sum_assessmentID = Storyline_Net_Emotion_scores_a[['df_fear']].reset_ind
Storyline_fear_Sum_assessmentID.columns = ["Storyline_fear_Sum_assessmentID"]
Storyline_fear_Sum_assessmentID = Storyline_fear_Sum_assessmentID.assign(Index=range(le

Storyline_happy_Sum_assessmentID = Storyline_Net_Emotion_scores_a[['df_happy']].reset_i
Storyline_happy_Sum_assessmentID.columns = ["Storyline_happy_Sum_assessmentID"]
Storyline_happy_Sum_assessmentID = Storyline_happy_Sum_assessmentID.assign(Index=range(

Storyline_sad_Sum_assessmentID = Storyline_Net_Emotion_scores_a[['df_sad']].reset_index
Storyline_sad_Sum_assessmentID.columns = ["Storyline_sad_Sum_assessmentID"]
```

```
Storyline_sad_Sum_assessmentID = Storyline_sad_Sum_assessmentID.assign(Index=range(len(

Storyline_surprise_Sum_assessmentID = Storyline_Net_Emotion_scores_a[['df_surprise']].r
Storyline_surprise_Sum_assessmentID.columns = ["Storyline_surprise_Sum_assessmentID"]
Storyline_surprise_Sum_assessmentID = Storyline_surprise_Sum_assessmentID.assign(Index=

Storyline_Neutral_Sum_assessmentID = Storyline_Net_Emotion_scores_a[['df_neutral']].res
Storyline_Neutral_Sum_assessmentID.columns = ["Storyline_Neutral_Sum_assessmentID"]
Storyline_Neutral_Sum_assessmentID = Storyline_Neutral_Sum_assessmentID.assign(Index=ra
Storyline_Neutral_Sum_assessmentID
```

```
Out[29]:        Storyline_Neutral_Sum_assessmentID
         Index
             0                       145427.190817
             1                       197653.266110
             2                       130670.854916
             3                        63183.022632
             4                       100755.007347
             5                       236457.581563
             6                        10056.311740
             7                        54605.692779
             8                        40172.058480
             9                       145091.879633
            10                        30434.995146
            11                        22027.760118
            12                       266705.291335
            13                        42141.542148
            14                        46384.677904
            15                       127614.444725
            16                       239253.134537
            17                       184387.122854
            18                       230855.584963
            19                       179983.310888
```

Compute the following proportional facial emotion scores for each question by summing the scores for each micro-expression and normalizing to the Total Emotion Score (above)

In [74]:

```
Proportion_Neutral_questionID = Storyline_Neutral_Sum_questionID['Storyline_Neutral_Sum
Proportion_Neutral_questionID = pd.DataFrame(Proportion_Neutral_questionID)
Proportion_Neutral_questionID.columns = ["Proportion_Neutral_questionID"]
Proportion_Neutral_questionID
```

Out[74]:

Index	Proportion_Neutral_questionID
0	0.762621
1	0.522108
2	0.249284
3	0.410844
4	0.337552
5	0.445165
6	0.364922
7	0.527286
8	0.349283
9	0.441517

In [76]:

```
Proportion_Surprise_questionID = Storyline_surprise_Sum_questionID['Storyline_surprise_
Proportion_Surprise_questionID = pd.DataFrame(Proportion_Surprise_questionID)
Proportion_Surprise_questionID.columns = ["Proportion_Surprise_questionID"]
Proportion_Surprise_questionID
```

Out[76]:

Index	Proportion_Surprise_questionID
0	0.015371
1	0.016537
2	0.022285
3	0.007531
4	0.118856
5	0.060106
6	0.003503
7	0.007829
8	0.016091
9	0.023758

In [77]:

```
Proportion_Sad_questionID = Storyline_sad_Sum_questionID['Storyline_sad_Sum_questionID'
Proportion_Sad_questionID = pd.DataFrame(Proportion_Sad_questionID)
Proportion_Sad_questionID.columns = ["Proportion_Sad_questionID"]
Proportion_Sad_questionID
```

Out[77]:

	Proportion_Sad_questionID
Index	
0	0.111420
1	0.137083
2	0.098200
3	0.221228
4	0.152019
5	0.144684
6	0.191333
7	0.208781
8	0.296571
9	0.257511

In [78]:

```
Proportion_Happy_questionID = Storyline_happy_Sum_questionID['Storyline_happy_Sum_quest
Proportion_Happy_questionID = pd.DataFrame(Proportion_Happy_questionID)
Proportion_Happy_questionID.columns = ["Proportion_Happy_questionID"]
Proportion_Happy_questionID
```

Out[78]:

	Proportion_Happy_questionID
Index	
0	0.009441
1	0.134665
2	0.474750
3	0.157791
4	0.088538
5	0.084323
6	0.081585
7	0.028255
8	0.077088
9	0.056991

In [79]:

```
Proportion_Fear_questionID = Storyline_fear_Sum_questionID['Storyline_fear_Sum_question
Proportion_Fear_questionID = pd.DataFrame(Proportion_Fear_questionID)
Proportion_Fear_questionID.columns = ["Proportion_Fear_questionID"]
Proportion_Fear_questionID
```

Out[79]:

	Proportion_Fear_questionID
Index	
0	0.080686

	Proportion_Fear_questionID
Index	
1	0.153088
2	0.120376
3	0.145398
4	0.249769
5	0.190103
6	0.135115
7	0.168566
8	0.163414
9	0.144632

In [80]:

```
Proportion_Angry_questionID = Storyline_angry_Sum_questionID['Storyline_angry_Sum_quest
Proportion_Angry_questionID = pd.DataFrame(Proportion_Angry_questionID)
Proportion_Angry_questionID.columns = ["Proportion_Angry_questionID"]
Proportion_Angry_questionID
```

Out[80]:

	Proportion_Angry_questionID
Index	
0	0.020289
1	0.035756
2	0.033285
3	0.055823
4	0.052838
5	0.075281
6	0.221096
7	0.059135
8	0.093391
9	0.075194

In [81]:

```
Proportion_Disgust_questionID = Storyline_disgust_Sum_questionID['Storyline_disgust_Sum
Proportion_Disgust_questionID = pd.DataFrame(Proportion_Disgust_questionID)
Proportion_Disgust_questionID.columns = ["Proportion_Disgust_questionID"]
Proportion_Disgust_questionID
```

Out[81]:

	Proportion_Disgust_questionID
Index	
0	0.000172
1	0.000761

	Proportion_Disgust_questionID
Index	
2	0.001820
3	0.001386
4	0.000429
5	0.000338
6	0.002446
7	0.000147
8	0.004162
9	0.000398

Compute the following proportional facial emotion scores for each assessment by summing the scores for each micro-expression and normalizing to the Total Emotion Score (above)

In [96]:

```
Proportion_Happy_assessmentID = Storyline_happy_Sum_assessmentID['Storyline_happy_Sum_a
Proportion_Happy_assessmentID = pd.DataFrame(Proportion_Happy_assessmentID)
Proportion_Happy_assessmentID.columns = ["Proportion_Happy_assessmentID"]
Proportion_Happy_assessmentID
```

Out[96]:

	Proportion_Happy_assessmentID
Index	
0	0.321984
1	0.073501
2	0.159495
3	0.077970
4	0.132810
5	0.179680
6	0.159526
7	0.040127
8	0.081853
9	0.108464
10	0.159007
11	0.064661
12	0.209489

	Proportion_Happy_assessmentID
Index	
13	0.137394
14	0.002198
15	0.099816
16	0.131112
17	0.092372
18	0.113172
19	0.062477

In [97]:

```
Proportion_Angry_assessmentID = Storyline_angry_Sum_assessmentID['Storyline_angry_Sum_a
Proportion_Angry_assessmentID = pd.DataFrame(Proportion_Angry_assessmentID)
Proportion_Angry_assessmentID.columns = ["Proportion_Angry_assessmentID"]
Proportion_Angry_assessmentID
```

Out[97]:

	Proportion_Angry_assessmentID
Index	
0	0.034010
1	0.132176
2	0.086818
3	0.036580
4	0.006986
5	0.028435
6	0.108315
7	0.307761
8	0.033429
9	0.099365
10	0.039329
11	0.063711
12	0.025459
13	0.068836
14	0.248795
15	0.013916
16	0.070664
17	0.070117
18	0.005066
19	0.051824

In [98]:

```
Proportion_Disgust_assessmentID = Storyline_disgust_Sum_assessmentID['Storyline_disgust
Proportion_Disgust_assessmentID = pd.DataFrame(Proportion_Disgust_assessmentID)
Proportion_Disgust_assessmentID.columns = ["Proportion_Disgust_assessmentID"]
Proportion_Disgust_assessmentID
```

Out[98]:

	Proportion_Disgust_assessmentID
Index	
0	0.000148
1	0.005486
2	0.002724
3	0.000006
4	0.000398
5	0.000006
6	0.021323
7	0.000733
8	0.001355
9	0.000185
10	0.000264
11	0.000095
12	0.002976
13	0.001251
14	0.000269
15	0.000127
16	0.000039
17	0.001587
18	0.000006
19	0.000297

In [99]:

```
Proportion_Fear_assessmentID = Storyline_fear_Sum_assessmentID['Storyline_fear_Sum_asse
Proportion_Fear_assessmentID = pd.DataFrame(Proportion_Fear_assessmentID)
Proportion_Fear_assessmentID.columns = ["Proportion_Fear_assessmentID"]
Proportion_Fear_assessmentID
```

Out[99]:

	Proportion_Fear_assessmentID
Index	
0	0.074243
1	0.160632
2	0.078923

	Proportion_Fear_assessmentID
Index	
3	0.073459
4	0.422237
5	0.004643
6	0.134101
7	0.134302
8	0.117109
9	0.085162
10	0.184026
11	0.519985
12	0.092295
13	0.244961
14	0.254932
15	0.152222
16	0.064016
17	0.253677
18	0.074230
19	0.051062

In [100…

```
Proportion_Sad_assessmentID = Storyline_sad_Sum_assessmentID['Storyline_sad_Sum_assessm
Proportion_Sad_assessmentID = pd.DataFrame(Proportion_Sad_assessmentID)
Proportion_Sad_assessmentID.columns = ["Proportion_Sad_assessmentID"]
Proportion_Sad_assessmentID
```

Out[100…

	Proportion_Sad_assessmentID
Index	
0	0.158357
1	0.201337
2	0.254685
3	0.213973
4	0.215125
5	0.009108
6	0.222079
7	0.280910
8	0.091042
9	0.249445

	Proportion_Sad_assessmentID
Index	
10	0.069862
11	0.080244
12	0.092562
13	0.401558
14	0.334532
15	0.075022
16	0.118010
17	0.134359
18	0.035303
19	0.247005

In [83]:

```
Proportion_Surprise_assessmentID = Storyline_surprise_Sum_assessmentID['Storyline_surpr
Proportion_Surprise_assessmentID = pd.DataFrame(Proportion_Surprise_assessmentID)
Proportion_Surprise_assessmentID.columns = ["Proportion_Surprise_assessmentID"]
Proportion_Surprise_assessmentID
```

Out[83]:

	Proportion_Surprise_assessmentID
Index	
0	0.046139
1	0.040827
2	0.012426
3	0.000254
4	0.027145
5	0.015116
6	0.020559
7	0.006828
8	0.039577
9	0.032637
10	0.149148
11	0.016943
12	0.067071
13	0.024204
14	0.000314
15	0.006138
16	0.015925

	Proportion_Surprise_assessmentID
Index	
17	0.025272
18	0.051025
19	0.034391

In [82]:

```
Proportion_Neutral_assessmentID = Storyline_Neutral_Sum_assessmentID['Storyline_Neutral
Proportion_Neutral_assessmentID = pd.DataFrame(Proportion_Neutral_assessmentID)
Proportion_Neutral_assessmentID.columns = ["Proportion_Neutral_assessmentID"]
Proportion_Neutral_assessmentID
```

Out[82]:

	Proportion_Neutral_assessmentID
Index	
0	0.365120
1	0.386042
2	0.404930
3	0.597758
4	0.195299
5	0.763013
6	0.334097
7	0.229339
8	0.635634
9	0.424742
10	0.398364
11	0.254362
12	0.510148
13	0.121796
14	0.158961
15	0.652759
16	0.600234
17	0.422615
18	0.721198
19	0.552944

Creating two csv files for assessments and questions

In [58]:

```
subject_id = facial_sentimental_report.groupby(['subject_id']).sum()
subject_id =pd.DataFrame(subject_id.index)
assessment_id = facial_sentimental_report.groupby(['assessment_id']).sum()
assessment_id =pd.DataFrame(assessment_id.index)
question_id = facial_sentimental_report.groupby(['question_id']).sum()
question_id = pd.DataFrame(question_id.index)
question_type = facial_sentimental_report.groupby(['question_type']).sum()
question_type = pd.DataFrame(question_type.index)
main_question = facial_sentimental_report.groupby(['main_question']).sum()
main_question = pd.DataFrame(main_question.index)
question_id
```

Out[58]:

```
                               question_id
0   1924574b-dc9a-4822-8f25-28943d0531a9
1    237e6469-5bfc-434f-81c8-afba91956066
2   533ad77a-8cd0-4201-8254-d552a679f3c3
3    546d29a9-c6eb-43ae-8df5-f106c781a8b4
4   7272d619-e9ef-4b44-98dd-9b8eb4530151
5    74490236-9e9e-4f0e-ab37-4c72ca21e15e
6    8b4362d5-8af2-4875-bcd8-6ffe06c30228
7    958a18c2-67d5-48e9-8464-9f2615a3f84f
8   cb52904c-1edb-4922-bf84-8a7b50ce2b3e
9   ef51d3ee-6709-49d7-9819-07ba799205ea
```

In [104…

```
face_alpha_report = pd.merge(question_id, total_Facial_Emotion_Score_questionID_x, on="
face_alpha_report = pd.merge(face_alpha_report, Q_Face_Sentiment_Score_QuestionID, on="
face_alpha_report = face_alpha_report.join(Storyline_angry_Sum_questionID)
face_alpha_report = face_alpha_report.join(Storyline_disgust_Sum_questionID)
face_alpha_report = face_alpha_report.join(Storyline_fear_Sum_questionID)
face_alpha_report = face_alpha_report.join(Storyline_happy_Sum_questionID)
face_alpha_report = face_alpha_report.join(Storyline_sad_Sum_questionID)
face_alpha_report = face_alpha_report.join(Storyline_surprise_Sum_questionID)
face_alpha_report = face_alpha_report.join(Storyline_Neutral_Sum_questionID)
face_alpha_report = face_alpha_report.join(Proportion_Neutral_questionID)
face_alpha_report = face_alpha_report.join(Proportion_Surprise_questionID)
face_alpha_report = face_alpha_report.join(Proportion_Sad_questionID)
face_alpha_report = face_alpha_report.join(Proportion_Happy_questionID)
face_alpha_report = face_alpha_report.join(Proportion_Fear_questionID)
face_alpha_report = face_alpha_report.join(Proportion_Angry_questionID)
face_alpha_report_QID = face_alpha_report.join(Proportion_Disgust_questionID)
face_alpha_report_QID.to_csv (r"D:\Master's\Intern\face_alpha_report_QID.csv", index =


face_alpha_report_x = pd.merge(assessment_id, A_Face_Sentiment_Score_AssessmentID, on="
face_alpha_report_x = pd.merge(face_alpha_report_x, total_Facial_Emotion_Score_assessme
face_alpha_report_x = face_alpha_report_x.join(Storyline_angry_Sum_assessmentID)
face_alpha_report_x = face_alpha_report_x.join(Storyline_disgust_Sum_assessmentID)
face_alpha_report_x = face_alpha_report_x.join(Storyline_fear_Sum_assessmentID)
face_alpha_report_x = face_alpha_report_x.join(Storyline_happy_Sum_assessmentID)
face_alpha_report_x = face_alpha_report_x.join(Storyline_sad_Sum_assessmentID)
face_alpha_report_x = face_alpha_report_x.join(Storyline_surprise_Sum_assessmentID)
face_alpha_report_x = face_alpha_report_x.join(Storyline_Neutral_Sum_assessmentID)
face_alpha_report_x = face_alpha_report_x.join(Proportion_Happy_assessmentID)
face_alpha_report_x = face_alpha_report_x.join(Proportion_Angry_assessmentID)
face_alpha_report_x = face_alpha_report_x.join(Proportion_Disgust_assessmentID)
face_alpha_report_x = face_alpha_report_x.join(Proportion_Fear_assessmentID)
face_alpha_report_x = face_alpha_report_x.join(Proportion_Sad_assessmentID)
face_alpha_report_x = face_alpha_report_x.join(Proportion_Surprise_assessmentID)
face_alpha_report_AID = face_alpha_report_x.join(Proportion_Neutral_assessmentID)
face_alpha_report_AID.to_csv (r"D:\Master's\Intern\face_alpha_report_AID.csv", index =
```

In [93]:

```
face_alpha_report_QID
```

Out[93]:

	question_id	total_Facial_Emotion_Score_questionID	Q_Face_Sentiment_Score_QuestionID	Storyline_an
0	1924574b-dc9a-4822-8f25-28943d0531a9	557000.003706	0.998039	
1	237e6469-5bfc-434f-81c8-afba91956066	478700.003308	0.999840	
2	533ad77a-8cd0-4201-8254-d552a679f3c3	674900.003909	0.999968	
3	546d29a9-c6eb-43ae-8df5-f106c781a8b4	658800.004111	0.999901	
4	7272d619-e9ef-4b44-98dd-9b8eb4530151	649200.003992	0.999820	
5	74490236-9e9e-4f0e-ab37-4c72ca21e15e	651700.003827	0.999812	
6	8b4362d5-8af2-4875-bcd8-6ffe06c30228	651300.003510	0.999806	
7	958a18c2-67d5-48e9-8464-9f2615a3f84f	335100.002351	0.998910	
8	cb52904c-1edb-4922-bf84-8a7b50ce2b3e	699800.003251	0.999809	

	question_id	total_Facial_Emotion_Score_questionID	Q_Face_Sentiment_Score_QuestionID	Storyline_an
9	ef51d3ee-6709-49d7-9819-07ba799205ea	480600.003004	0.999623	

In [102…

```
face_alpha_report_AID
```

Out[102…

	assessment_id	A_Face_Sentiment_Score_AssessmentID	total_Facial_Emotion_Score_assessmentID	Storyl
0	0df79265-944e-43f9-a0bd-39d8e5757a6d	0.999919	398300.002575	
1	23222d65-20dc-4247-8495-8b98f52c2c6f	0.999726	512000.002504	
2	2d03cee3-2af3-4b99-b97f-9280f81ae964	0.999799	322700.002331	
3	4143b281-9e12-4f3c-9da9-6ee7d3f8c060	0.998748	105700.000612	
4	4d86c48f-8a2e-4366-84dc-bf9a2e58223f	0.999849	515900.002427	
5	5878ccf2-052a-400b-8682-ed7b9db29f50	0.999814	309900.002254	
6	5e703137-4029-4f52-b5de-b200751e87cf	0.997853	30100.000218	
7	5e7c6635-e024-4268-83ad-590016cd1046	0.998920	238100.001048	
8	664554ac-ff5a-4d3c-af27-c6c09ee9f816	0.998007	63200.000575	
9	6b6ee892-6d2e-45c2-a44f-9cd50ae436a8	0.999721	341600.002166	

	assessment_id	A_Face_Sentiment_Score_AssessmentID	total_Facial_Emotion_Score_assessmentID	Story
10	b8409cdc-bf94-4f3a-a2d3-ffb029f2e965	0.999150	76400.000333	
11	c3e3616b-61ca-48c2-9f2c-8325825f0782	0.998158	86600.000139	
12	cac929f6-fb4c-4415-845b-ed42dec63a86	0.999906	522800.003276	
13	d5968d42-783a-4dfe-9af7-2bb284230a9d	0.999783	346000.001779	
14	d6889be5-3d64-4293-8049-53466ca2329a	0.984145	291800.001568	
15	dcc4a978-da9b-48d6-9142-3a7f5e90306a	0.999471	195500.001580	
16	f7bab369-0fbd-4bfc-85d6-2452c945be0f	0.999802	398600.002627	
17	f7f724bb-9348-4648-bf4c-b21c66604931	0.999744	436300.002982	
18	f94450cf-c403-4142-a8be-85ffa46ec49c	0.999715	320100.001900	
19	fa3846ac-f7a5-4991-bc6a-3a89236641e2	0.999492	325500.002076	

```
In [ ]:
```