More viz about feature

May 12, 2020

```python
[1]: import pandas as pd
     import matplotlib.pyplot as plt
```

```python
[2]: baseline=pd.read_csv('baseline_clean_df.csv')
     log=pd.read_csv('logistic_regression_trained_on_outside.csv')
```

```python
[3]: def get_first(x):
         return x.split("'")[1]
     def encoder(x):
         if x.lower()=='joy':
             return 0
         elif x.lower()=='sad':
             return 1
         elif x.lower()=='surprise':
             return 2
         elif x.lower()=='fear':
             return 3
         elif x.lower=='disgust':
             return 4
         else:
             return 5
     baseline['emotion']=baseline['Emotion'].apply(get_first)
     log['emotion_cluster']=log['predict'].apply(encoder)
     numls=['scene_avg_p', 'scene_avg_a', 'scene_avg_d',
            'scene_avg_blur', 'scene_avg_optical_flow']
```

```python
[7]: rols=['Unnamed: 0','scene_avg_p', 'scene_avg_a', 'scene_avg_d',
           'scene_avg_blur', 'scene_avg_optical_flow']
```
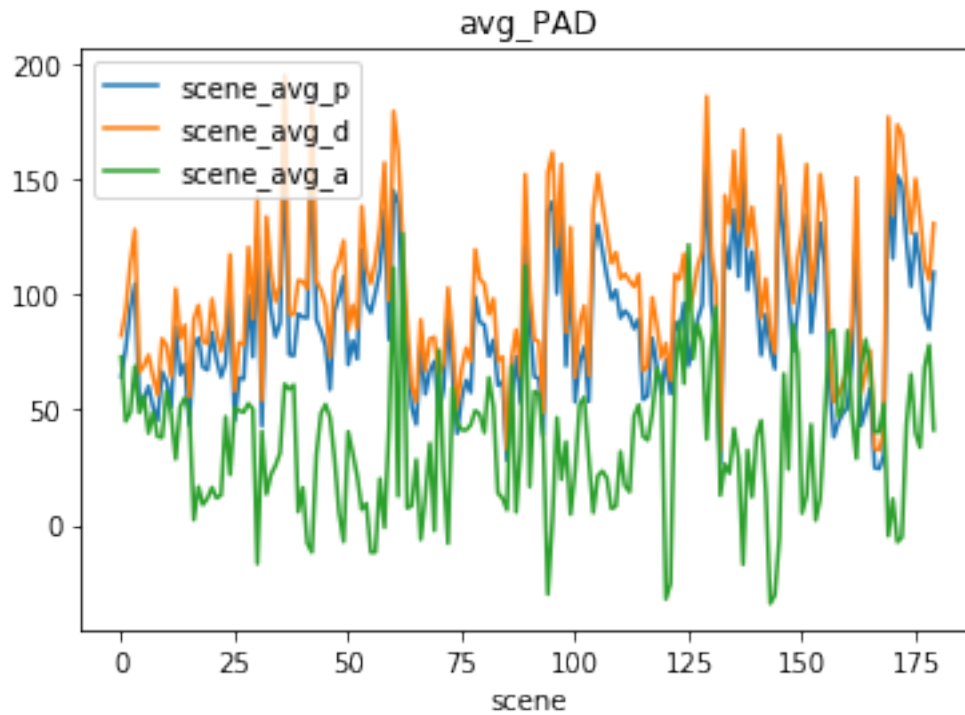
```python
[11]: baseline[['scene_avg_blur']].plot()
      plt.xlabel('scene')
      plt.title('avg_blur')
```

```python
[11]: Text(0.5, 1.0, 'avg_blur')
```

avg_blur
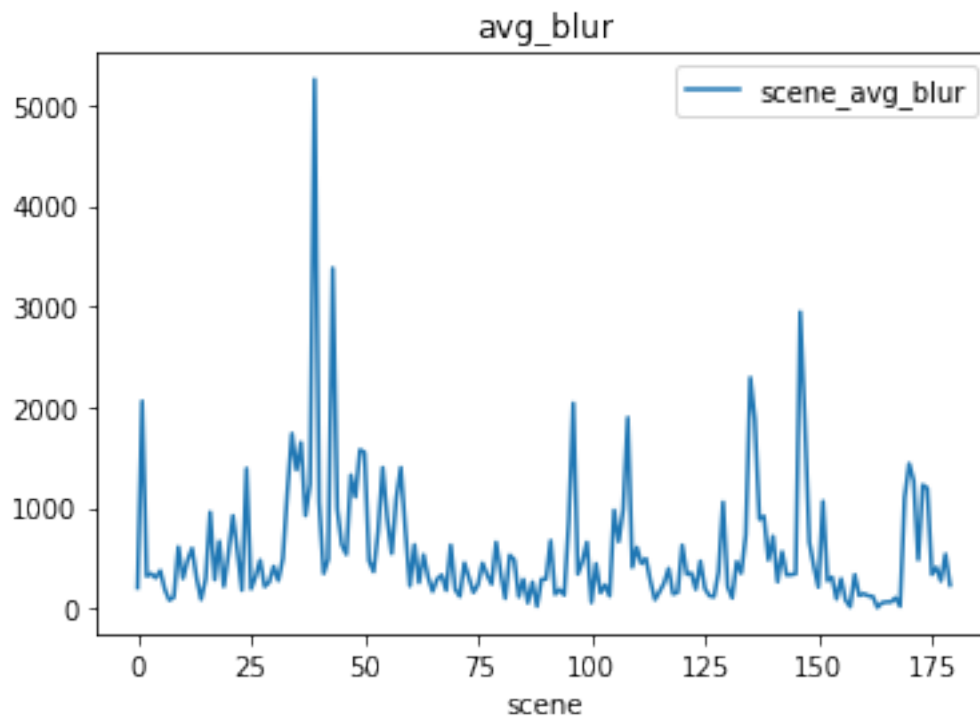
```
[12]: baseline[['scene_avg_p', 'scene_avg_d', 'scene_avg_a']].plot()
      plt.xlabel('scene')
      plt.title('avg_PAD')
```
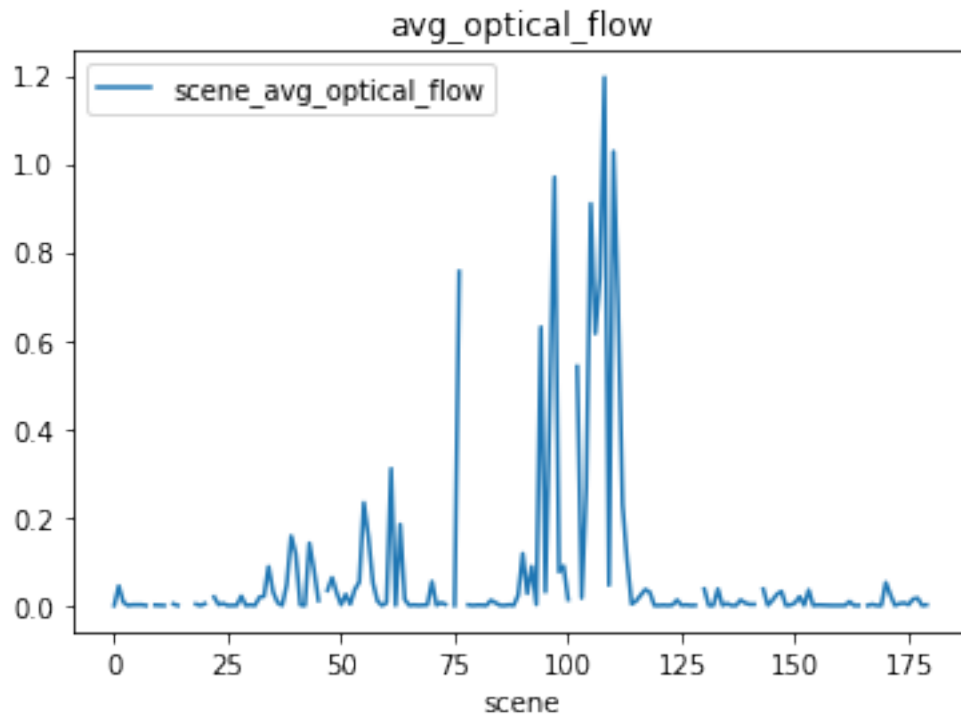
```
[12]: Text(0.5, 1.0, 'avg_PAD')
```

avg_PAD

avg_p is average pleasure, avg_d is average dominance, avg_a is averge arousal

```
[15]: baseline[['scene_avg_optical_flow']].plot()
      plt.xlabel('scene')
      plt.title('avg_optical_flow')
```
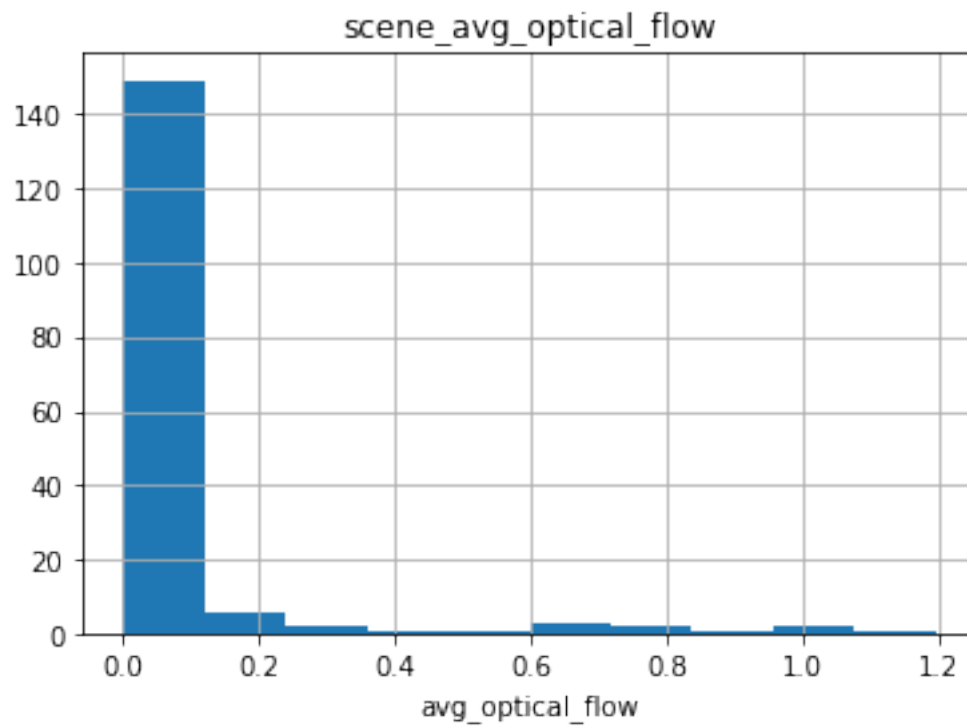
```
[15]: Text(0.5, 1.0, 'avg_optical_flow')
```
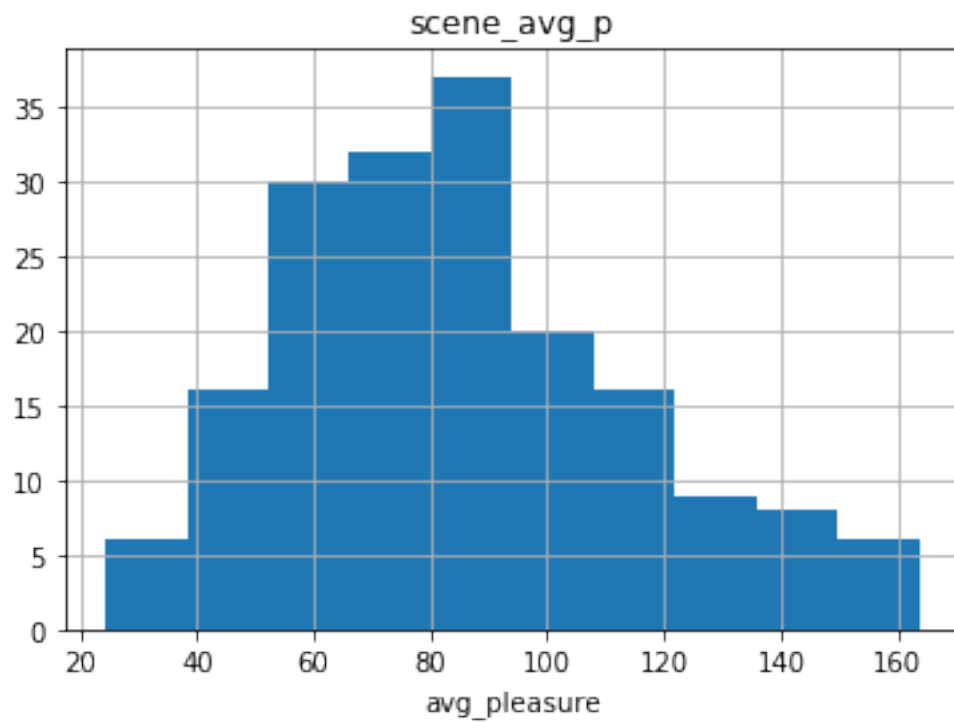
avg_optical_flow

Histgoram of each features

```
[30]: baseline[['scene_avg_optical_flow']].hist()
      plt.xlabel('avg_optical_flow')
```

```
[30]: Text(0.5, 0, 'avg_optical_flow')
```

scene_avg_optical_flow

avg_optical_flow

[29]: `baseline[['scene_avg_p']].hist()`
`plt.xlabel('avg_pleasure')`

[29]: `Text(0.5, 0, 'avg_pleasure')`

scene_avg_p
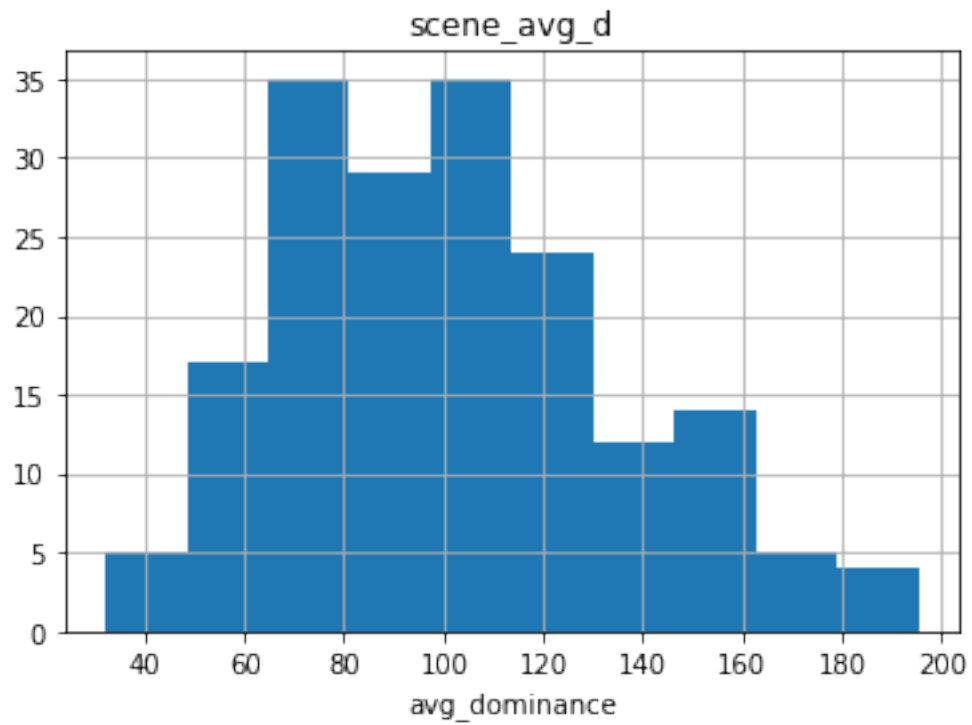
avg_pleasure

```
[28]: baseline[['scene_avg_blur']].hist()
      plt.xlabel('avg_blur')
```

[28]: Text(0.5, 0, 'avg_blur')
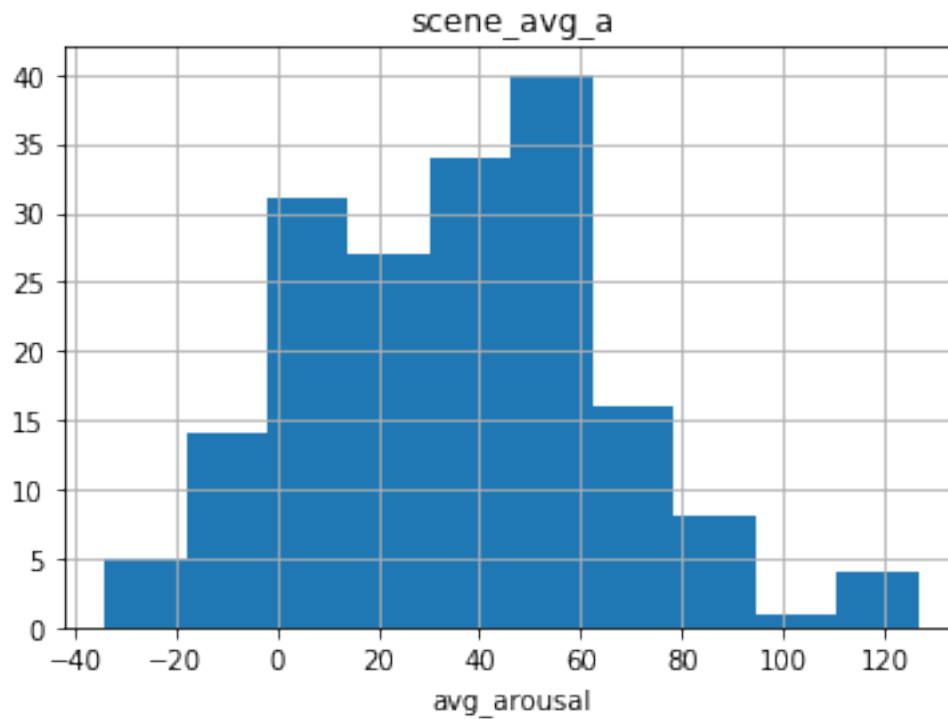
scene_avg_blur
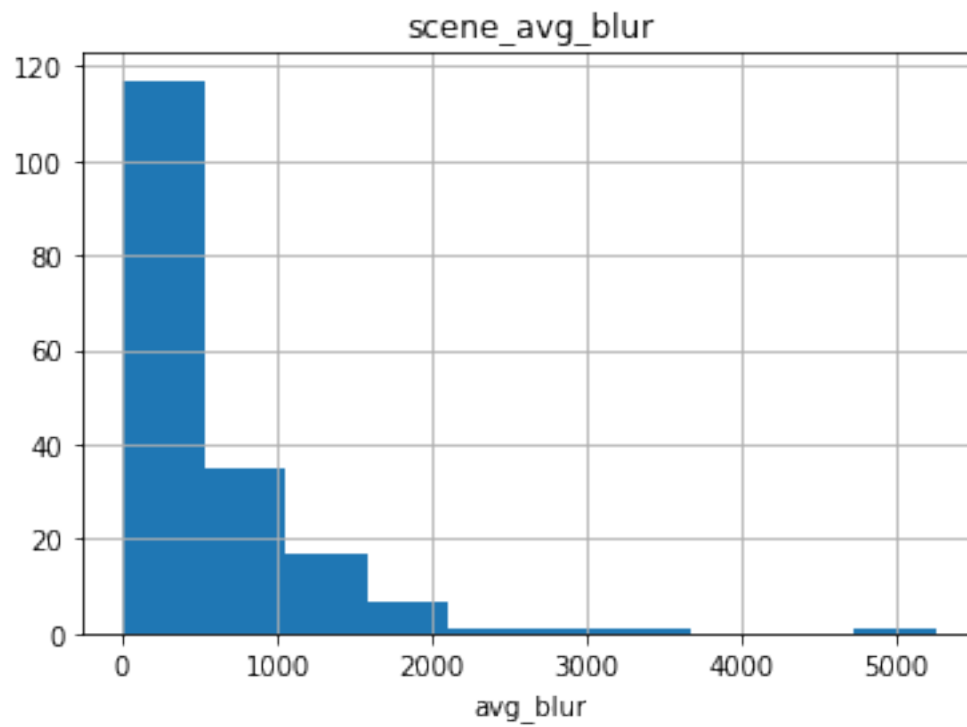
```
[31]: baseline[['scene_avg_d']].hist()
      plt.xlabel('avg_dominance')
```

[31]: Text(0.5, 0, 'avg_dominance')

scene_avg_d

```
[32]: baseline[['scene_avg_a']].hist()
      plt.xlabel('avg_arousal')
```

```
[32]: Text(0.5, 0, 'avg_arousal')
```

Some stats groupby movies:

```
[17]: numdf2=log.drop(['predict', 'emotion_cluster'], axis=1)
```

```
[20]: numdf2.groupby('scene_movie').mean()
```

```
[20]:                scene_avg_p  scene_avg_a  scene_avg_d  scene_avg_blur  \
      scene_movie
      big_hero_6       69.257722    39.951352    84.183492      460.885642
      cars_3           95.184971    21.719709   112.009275     1173.280825
      incredible_2     72.636551    38.717727    87.936714      305.694960
      toy_story_4      88.053817    30.353011   104.758193      491.155222
      up               96.371533    38.839275   115.278433      686.101881
      wall_e_          88.987489    42.153594   107.144932      399.754346

                    scene_avg_optical_flow
      scene_movie
      big_hero_6                   0.004825
      cars_3                       0.048813
      incredible_2                 0.049380
      toy_story_4                  0.309847
      up                           0.008049
      wall_e_                      0.006766
```

```
[19]: numdf2.groupby('scene_movie').max()
```

```
[19]:                scene_name  scene_avg_p  scene_avg_a  scene_avg_d  \
      scene_movie
      big_hero_6      big_hero_69  104.689522    72.981351   128.197153
      cars_3             cars_39   163.632920    61.172539   195.204988
      incredible_2   incredible_29  145.156260   126.587839  179.546329
      toy_story_4     toy_story_49  140.287836    68.247325   161.740315
      up                   up9      157.959511   121.647637  186.010694
      wall_e_            wall_e_9   154.245262    84.614697   177.070653

                     scene_avg_blur  scene_avg_optical_flow
      scene_movie
      big_hero_6        2058.214860                0.044003
      cars_3            5255.479130                0.232639
      incredible_2       656.606464                0.755713
      toy_story_4       2037.561794                1.195580
      up                2944.690295                0.037222
      wall_e_           1437.663826                0.051978
```

above tables show mean values and max values about every status for each movie
Emotion frequency in baseline model for each movie:

```
[24]: basenum=baseline[['emotion', 'scene_movie']]
      basenum.groupby(['scene_movie', 'emotion']).size()
```

```
[24]: scene_movie     emotion
      big_hero_6      Joy          17
                      Surprise     13
      cars_3          Joy          28
                      Surprise      2
      incredible_2    Joy          16
                      Surprise     14
      toy_story_4     Joy          22
                      Surprise      8
      up              Joy          28
                      Surprise      2
      wall_e_         Joy          19
                      Surprise     11
      dtype: int64
```

```
[ ]:
```