Friedman1

May 18, 2022

[1]:

```
import warnings
warnings.filterwarnings('ignore')
```

[2]:

```
import scrapbook as sb
import pandas as pd
import numpy as np
import seaborn as sns
import numpy as np
from statistics import mean, median
import matplotlib.pyplot as plt
```

1 Baseline

[3]:

```
books = sb.read_notebooks("./BaseLine_Model_Output")
baseLine_data = []
for nb in books.notebooks:
    nbList=[nb.scraps['Stats Model MAE'].data,nb.scraps['Catboost MAE'].data]
    baseLine_data.append(nbList)
df = pd.DataFrame(baseLine_data, columns = ["Stats Model","Catboost"])
baseline_data = np.array(baseLine_data)
stats = median(baseline_data[:,0])
catboost = median(baseline_data[:,1])
display(df)
print(df.median(axis=0))
```

```
   Stats Model  Catboost
0     0.446618  0.256896
1     0.355889  0.362786
2     0.539765  0.383997
3     0.516100  0.378521
4     0.465442  0.329466
5     0.472790  0.418192
6     0.372389  0.349403
7     0.407086  0.349494
8     0.410168  0.233684
9     0.272517  0.196731
```

```
Stats Model    0.428393
```

```
Catboost       0.349449
dtype: float64
```

2 GAN Model

[4]:

```
book = sb.read_notebooks("./GAN_Output")
gan_data = []
gan_mse = []
for nb in book.notebooks:
    metrics = nb.scraps['GAN_1 Metrics'].data
    for i in range(1000):
        gan_mse.append(metrics[0][i])
    nbList = [nb.scraps['GAN Model MSE'].data,
              nb.scraps['GAN Model MAE'].data,
              nb.scraps['GAN Model Euclidean distance'].data,
              nb.scraps['GAN Model Manhattan Distance'].data]
    gan_data.append(nbList)

df = pd.DataFrame(gan_data, columns = ['MSE','MAE','Euclidean␣
 ↪Distance','Manhattan Distance'])
display(df.style)
print("MEDIAN:")
print(df.median(axis = 0))
gan_data = np.array(gan_data)
gan_median = median(gan_data[:,1])
```

```
<pandas.io.formats.style.Styler at 0x7fae921459d0>
```

```
MEDIAN:
MSE                   0.085055
MAE                   0.208172
Euclidean Distance    1.292234
Manhattan Distance    4.163449
dtype: float64
```

3 ABC_GAN Analysis

3.1 ABC Pre-generator - Catboost

[5]:

```
book = sb.read_notebooks("./ABC_GAN_Catboost")
paramVal = [1,0.1,0.01]
abc_mae = [[] for i in range(3)]
abc_mae_skip = [[] for i in range(3)]
abc_mae_mean = [[] for i in range(3)]
abc_mae_skip_mean = [[] for i in range(3)]
abc_weights = [[] for i in range(3)]
prior_model = [[] for i in range(3)]
```

```
abc_pre_generator = [[] for i in range(3)]

for nb in book.notebooks:
    metrics1 = np.array(nb.scraps['ABC_GAN_1 Metrics'].data)
    metrics3 = np.array(nb.scraps['ABC_GAN_3 Metrics'].data)
    paramVar = float(nb.papermill_dataframe.iloc[0]['value'])

    #Divide data according to parameters
    for i in range(3):
        if paramVar == paramVal[i]:
            for j in range(100):
                abc_mae[i].append(metrics1[1,j])
                abc_mae_skip[i].append(metrics3[1,j])
            abc_weights[i].append(nb.scraps['Skip Connection Weight'].data)
            prior_model[i].append(nb.scraps['Prior Model MAE'].data)
            abc_pre_generator[i].append(nb.scraps['ABC Pre-generator MAE'].data)
            abc_mae_skip_mean[i].append(mean(metrics3[1,:]))
            abc_mae_mean[i].append(mean(metrics1[1,:]))
```

[6]:

```
data = [[] for i in range(3)]
data_median_catboost = [[] for i in range(3)]
for i in range(3):
    for j in range(len(abc_weights[i])):
        data[i].append([prior_model[i][j],paramVal[i],
                                  ␣
 ↪abc_pre_generator[i][j],abc_weights[i][j],abc_mae_mean[i][j],abc_mae_skip_mean[i][j]])

    df = pd.DataFrame(data[i], columns = ['Baseline','Variance','Prior␣
 ↪Model','Weight','ABC_GAN','Skip_GAN'])
    data_median_catboost[i] = [ df['Baseline'].median(),df['Variance'].
 ↪median(), df['Prior Model'].median(),
                              df['ABC_GAN'].median(), df['Skip_GAN'].
 ↪median(),df['Weight'].median()]
print(data_median_catboost)
```

```
[[0.3284895998228049, 1.0, 0.9342310346052503, 0.20056221077172087,
0.16983164138183932, 0.36616359651088715], [0.3422213657088792, 0.1,
0.3556469392849807, 0.3052056774329394, 0.24653722626157104,
0.22563840448856354], [0.28883367551810446, 0.01, 0.28719960297101377,
0.27654515282996, 0.2850077768463807, 0.008500191383063793]]
```

3.2 ABC Pre-generator - Stats

[7]:

```
book = sb.read_notebooks("./ABC_GAN_Stats")
paramVal = [1,0.1,0.01]
abc_mae = [[] for i in range(3)]
abc_mae_skip = [[] for i in range(3)]
```

```
abc_mae_mean = [[] for i in range(3)]
abc_mae_skip_mean = [[] for i in range(3)]
abc_weights = [[] for i in range(3)]
prior_model = [[] for i in range(3)]
abc_pre_generator = [[] for i in range(3)]

for nb in book.notebooks:
    metrics1 = np.array(nb.scraps['ABC_GAN_1 Metrics'].data)
    metrics3 = np.array(nb.scraps['ABC_GAN_3 Metrics'].data)
    paramVar = float(nb.papermill_dataframe.iloc[0]['value'])

    #Divide data according to parameters
    for i in range(3):
        if paramVar == paramVal[i]:
            for j in range(100):
                abc_mae[i].append(metrics1[1,j])
                abc_mae_skip[i].append(metrics3[1,j])
            abc_weights[i].append(nb.scraps['Skip Connection Weight'].data)
            prior_model[i].append(nb.scraps['Prior Model MAE'].data)
            abc_pre_generator[i].append(nb.scraps['ABC Pre-generator MAE'].data)
            abc_mae_skip_mean[i].append(mean(metrics3[1,:]))
            abc_mae_mean[i].append(mean(metrics1[1,:]))
```

[8]:

```
data = [[] for i in range(3)]
data_median_stats = [[] for i in range(3)]
for i in range(3):
    for j in range(len(abc_weights[i])):
        data[i].append([paramVal[i],prior_model[i][j],
                         ␣
 ↪abc_pre_generator[i][j],abc_weights[i][j],abc_mae_mean[i][j],abc_mae_skip_mean[i][j]])

    df = pd.DataFrame(data[i], columns = ['Variance','Baseline','Prior␣
 ↪Model','Weight','ABC_GAN','Skip_GAN'])
    data_median_stats[i] = [df['Baseline'].median(),df['Variance'].median(),␣
 ↪df['Prior Model'].median(), df['ABC_GAN'].median(), df['Skip_GAN'].median(),
                            df['Weight'].median()]
print(data_median_stats)
```

```
[[0.3299035718500814, 1.0, 0.8106776979131838, 0.27565625952207484,
0.18991099416287616, 0.9643011093139648], [0.45901400777813817, 0.1,
0.4538427825330652, 0.31216624486965594, 0.15889515103814192,
0.17494748532772064], [0.39844206415763644, 0.01, 0.3990571479950601,
0.3003488202616572, 0.2715642233293504, 0.1280488520860672]]
```

[9]:

```
#Output Table
output_data_stats = [[stats, gan_median] for i in range(3)]
for i in range(3):
    for j in range(6):
        output_data_stats[i].append(data_median_stats[i][j])

df_stats = pd.DataFrame(output_data_stats,␣
 ↪columns=['Baseline','GAN','Baseline','Variance','Prior␣
 ↪Model','ABC_GAN','Skip_GAN','Weight'])

output_data_catboost = [[catboost, gan_median] for i in range(3)]
for i in range(3):
    for j in range(6):
        output_data_catboost[i].append(data_median_catboost[i][j])
df_catboost = pd.
 ↪DataFrame(output_data_catboost,columns=['Baseline','GAN','Baseline','Variance','Prior␣
 ↪Model','ABC_GAN','Skip_GAN','Weight'])

display(df_stats)
display(df_catboost)
```

```
   Baseline       GAN  Baseline  Variance  Prior Model   ABC_GAN  Skip_GAN  \
0  0.428393  0.208172  0.329904      1.00     0.810678  0.275656  0.189911
1  0.428393  0.208172  0.459014      0.10     0.453843  0.312166  0.158895
2  0.428393  0.208172  0.398442      0.01     0.399057  0.300349  0.271564

     Weight
0  0.964301
1  0.174947
2  0.128049
```

```
   Baseline       GAN  Baseline  Variance  Prior Model   ABC_GAN  Skip_GAN  \
0  0.349449  0.208172  0.328490      1.00     0.934231  0.200562  0.169832
1  0.349449  0.208172  0.342221      0.10     0.355647  0.305206  0.246537
2  0.349449  0.208172  0.288834      0.01     0.287200  0.276545  0.285008

     Weight
0  0.366164
1  0.225638
2  0.008500
```