[Dacon] 블럭 장난감 제조 공정 최적화 경진대회

_ (팀명)

2020년 월 일 (제출날짜)

1. 본 코드는 대회 참가를 돕고자 단순 예시를 작성한 것으로 참고용으로 사용바랍니다.
2. 본 코드는 자유롭게 수정하여 사용 할 수 있습니다.
3. 추가 모듈 보러가기: https://bit.ly/36MNs76 (https://bit.ly/36MNs76)

1. 라이브러리 및 데이터

Library & Data

In [2]:

```
import pandas as pd
import numpy as np
import multiprocessing
import warnings
from copy import deepcopy
from module.genome import Genome, genome_score
import datetime
warnings.filterwarnings(action='ignore')
np.random.seed(777)
```

In [3]:

```
!python --version
print('Pandas : %s'%(pd.__version__))
print('Numpy : %s'%(np.__version__))
```

```
Python 3.7.7
Pandas : 1.0.3
Numpy : 1.18.1
```

2. 데이터 전처리

Data Cleansing & Pre-Processing

In [4]:

```
# 입력하세요.
```

3. 탐색적 자료분석

Exploratory Data Analysis

In [5]:

```
# 입력하세요.
```

4. 변수 선택 및 모델 구축

Feature Engineering & Initial Modeling

In [6]:

```
CPU_CORE = multiprocessing.cpu_count() # 멀티프로세싱 CPU 사용 수
N_POPULATION = 300                      # 세대당 생성수
N_BEST = 20                             # 베스트 수
N_CHILDREN = 10                         # 자손 유전자 수
PROB_MUTATION = 0.4                    # 돌연변이
REVERSE = True                         # 배열 순서 (False: ascending order, True: descending order)

score_ini = 10                         # 초기 점수
input_length = 125                     # 입력 데이터 길이
output_length_1 = 5 * 2                # Event (CHECK_1~4, PROCESS)
output_length_2 = 12 * 2               # MOL(0~5.5, step:0.5)
h1 = 50                                # 히든레이어1 노드 수
h2 = 50                                # 히든레이어2 노드 수
h3 = 50                                # 히든레이어3 노드 수
EPOCHS = 300                           # 반복 횟수

genomes = []
for _ in range(N_POPULATION):
    genome = Genome(score_ini, input_length, output_length_1, output_length_2, h1, h2, h3)
    genomes.append(genome)
try:
    for i in range(N_BEST):
        genomes[i] = best_genomes[i]
except:
    best_genomes = []
    for _ in range(N_BEST):
        genome = Genome(score_ini, input_length, output_length_1, output_length_2, h1, h2, h3)
        best_genomes.append(genome)
```

In [7]:

```
best_genomes[0].forward(np.zeros((1, 125)))
```

Out[7]:

```
('CHECK_1', 'CHECK_1', 0.0, 0.0)
```

5. 모델 학습 및 검증

Model Tuning & Evaluation

1. PRT는 고정값 사용
2. Event A, Event B (MOL_A, MOL_B) 를 같은 값으로 제한
3. Event는 CHECK와 PROCESS 만 사용함
4. 목적 함수로 수요 부족분만 고려함
5. Event와 MOL에 대해 인공신경망 모델을 만들어 유전 알고리즘으로 학습

[illegible]

In [8]:

```
n_gen = 1
score_history = []
high_score_history = []
mean_score_history = []
best_gen = None
best_score_ever = 0
while n_gen <= EPOCHS:
    print('EPOCH', n_gen, datetime.datetime.now())
    genomes = np.array(genomes)
    while len(genomes)%CPU_CORE != 0:
        genomes = np.append(genomes, Genome(score_ini, input_length, output_length_1, output_length_
    genomes = genomes.reshape((len(genomes)//CPU_CORE, CPU_CORE))

    for idx, _genomes in enumerate(genomes):
        if __name__ == '__main__':
            pool = multiprocessing.Pool(processes=CPU_CORE)
            genomes[idx] = pool.map(genome_score, _genomes)
            pool.close()
            pool.join()
    genomes = list(genomes.reshape(genomes.shape[0]*genomes.shape[1]))

     # score에 따라 정렬
    genomes.sort(key=lambda x: x.score, reverse=REVERSE)

    # 평균 점수
    s = 0
    for i in range(N_BEST):
        s += genomes[i].score
    s /= N_BEST

    # Best Score
    bs = genomes[0].score

    # Best Model 추가
    if best_genomes is not None:
        genomes.extend(best_genomes)

    # score에 따라 정렬
    genomes.sort(key=lambda x: x.score, reverse=REVERSE)

    score_history.append([n_gen, genomes[0].score])
    high_score_history.append([n_gen, bs])
    mean_score_history.append([n_gen, s])

    if genomes[0].score > best_score_ever:
        best_score_ever = genomes[0].score
        best_gen = genomes[0]

    # 결과 출력
    print('EPOCH #%s₩tHistory Best Score: %s₩tBest Score: %s₩tMean Score: %s' % (n_gen, genomes[0].s

    # 모델 업데이트
    best_genomes = deepcopy(genomes[:N_BEST])

    # CHILDREN 생성
    for i in range(N_CHILDREN):
        new_genome = deepcopy(best_genomes[0])
        a_genome = np.random.choice(best_genomes)
        b_genome = np.random.choice(best_genomes)
        for j in range(input_length):
            cut = np.random.randint(new_genome.w1.shape[1])
            new_genome.w1[j, :cut] = a_genome.w1[j, :cut]
            new_genome.w1[j, cut:] = b_genome.w1[j, cut:]

        for j in range(h1):
            cut = np.random.randint(new_genome.w2.shape[1])
            new_genome.w2[j, :cut] = a_genome.w2[j, :cut]
            new_genome.w2[j, cut:] = b_genome.w2[j, cut:]

        for j in range(h2):
            cut = np.random.randint(new_genome.w3.shape[1])
            new_genome.w3[j, :cut] = a_genome.w3[j, :cut]
            new_genome.w3[j, cut:] = b_genome.w3[j, cut:]

        for j in range(h3):
            cut = np.random.randint(new_genome.w4.shape[1])
            new_genome.w4[j, :cut] = a_genome.w4[j, :cut]
            new_genome.w4[j, cut:] = b_genome.w4[j, cut:]

        for j in range(input_length):
            cut = np.random.randint(new_genome.w5.shape[1])
            new_genome.w5[j, :cut] = a_genome.w5[j, :cut]
            new_genome.w5[j, cut:] = b_genome.w5[j, cut:]

        for j in range(h1):
            cut = np.random.randint(new_genome.w6.shape[1])
            new_genome.w6[j, :cut] = a_genome.w6[j, :cut]
            new_genome.w6[j, cut:] = b_genome.w6[j, cut:]

        for j in range(h2):
            cut = np.random.randint(new_genome.w7.shape[1])
            new_genome.w7[j, :cut] = a_genome.w7[j, :cut]
            new_genome.w7[j, cut:] = b_genome.w7[j, cut:]

        for j in range(h3):
            cut = np.random.randint(new_genome.w8.shape[1])
            new_genome.w8[j, :cut] = a_genome.w8[j, :cut]
            new_genome.w8[j, cut:] = b_genome.w8[j, cut:]

        best_genomes.append(new_genome)

    # 모델 초기화
    genomes = []
    for i in range(int(N_POPULATION / len(best_genomes))):
        for bg in best_genomes:
            new_genome = deepcopy(bg)
            mean = 0
            stddev = 0.2
            # 50% 확률로 모델 변형
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w1 += new_genome.w1 * np.random.normal(mean, stddev, size=(input_length,
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w2 += new_genome.w2 * np.random.normal(mean, stddev, size=(h1, h2)) * np
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w3 += new_genome.w3 * np.random.normal(mean, stddev, size=(h2, h3)) * np
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w4 += new_genome.w4 * np.random.normal(mean, stddev, size=(h3, output_le
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w5 += new_genome.w5 * np.random.normal(mean, stddev, size=(input_length,
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w6 += new_genome.w6 * np.random.normal(mean, stddev, size=(h1, h2)) * np
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w7 += new_genome.w7 * np.random.normal(mean, stddev, size=(h2, h3)) * np
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w8 += new_genome.w8 * np.random.normal(mean, stddev, size=(h3, output_len
            genomes.append(new_genome)

    if REVERSE:
        if bs < score_ini:
            genomes[len(genomes)//2:] = [Genome(score_ini, input_length, output_length_1, output_len
    else:
        if bs > score_ini:
            genomes[len(genomes)//2:] = [Genome(score_ini, input_length, output_length_1, output_len

    n_gen += 1
```

```
EPOCH 1 2020-06-28 12:05:51.216003
EPOCH #1        History Best Score: 82.6869582127399    Best Score: 82.68695821273
99      Mean Score: 79.76876010534907
EPOCH 2 2020-06-28 12:10:06.630384
EPOCH #2        History Best Score: 83.83997335859024   Best Score: 83.83997335859
024     Mean Score: 82.53936537102126
EPOCH 3 2020-06-28 12:14:44.404786
EPOCH #3        History Best Score: 86.00186999839943   Best Score: 86.00186999839
943     Mean Score: 83.57946752817057
EPOCH 4 2020-06-28 12:19:43.802378
EPOCH #4        History Best Score: 86.00186999839943   Best Score: 85.78838143297
729     Mean Score: 84.58583819521654
EPOCH 5 2020-06-28 12:24:52.194858
EPOCH #5        History Best Score: 86.00186999839943   Best Score: 86.00186999839
943     Mean Score: 85.0754801183257
EPOCH 6 2020-06-28 12:30:53.014518
EPOCH #6        History Best Score: 86.00186999839943   Best Score: 85.62539173348
269     Mean Score: 83.0644396164399
EPOCH 7 2020-06-28 12:36:35.425397
EPOCH #7        History Best Score: 86.00186999839943   Best Score: 86.00186999839
943     Mean Score: 83.85238588386225
EPOCH 8 2020-06-28 12:41:22.206497
EPOCH #8        History Best Score: 86.00186999839943   Best Score: 86.00186999839
943     Mean Score: 83.70574988410434
EPOCH 9 2020-06-28 12:45:48.513585
EPOCH #9        History Best Score: 86.00186999839943   Best Score: 86.00186999839
943     Mean Score: 84.67572926711853
EPOCH 10 2020-06-28 12:50:24.679715
EPOCH #10       History Best Score: 86.00790855409831   Best Score: 86.00790855409
831     Mean Score: 84.91629480722919
EPOCH 11 2020-06-28 12:54:54.801382
EPOCH #11       History Best Score: 86.00790855409831   Best Score: 85.68524673564
391     Mean Score: 85.09180775113876
EPOCH 12 2020-06-28 12:59:30.960789
EPOCH #12       History Best Score: 86.12835740286806   Best Score: 86.12835740286
806     Mean Score: 85.47453538051106
EPOCH 13 2020-06-28 13:04:13.369139
EPOCH #13       History Best Score: 86.12835740286806   Best Score: 86.00971557407
291     Mean Score: 83.21133964940373
EPOCH 14 2020-06-28 13:08:55.401278
EPOCH #14       History Best Score: 86.25345928199066   Best Score: 86.25345928199
066     Mean Score: 83.91787585754341
EPOCH 15 2020-06-28 13:13:16.180506
EPOCH #15       History Best Score: 86.25345928199066   Best Score: 85.72148820782
758     Mean Score: 82.66563588393906
EPOCH 16 2020-06-28 13:17:18.392418
EPOCH #16       History Best Score: 86.50938117399771   Best Score: 86.50938117399
771     Mean Score: 83.04059348102643
EPOCH 17 2020-06-28 13:21:19.634424
EPOCH #17       History Best Score: 86.88939083888184   Best Score: 86.88939083888
184     Mean Score: 83.04901042461897
EPOCH 18 2020-06-28 13:25:21.274521
EPOCH #18       History Best Score: 87.27176081224349   Best Score: 87.27176081224
349     Mean Score: 84.15547990634954
EPOCH 19 2020-06-28 13:29:22.491870
EPOCH #19       History Best Score: 87.27176081224349   Best Score: 87.11650209841
04      Mean Score: 85.59355882783942
EPOCH 20 2020-06-28 13:33:24.115290
EPOCH #20       History Best Score: 87.57459548650526   Best Score: 87.57459548650
526     Mean Score: 86.07775649653173
EPOCH 21 2020-06-28 13:37:26.452395
EPOCH #21       History Best Score: 87.57459548650526   Best Score: 86.93290954485
647     Mean Score: 84.15379670810238
EPOCH 22 2020-06-28 13:41:29.419829
EPOCH #22       History Best Score: 87.80434354515691   Best Score: 87.80434354515
691     Mean Score: 84.9330629050012
EPOCH 23 2020-06-28 13:45:29.988073
EPOCH #23       History Best Score: 87.80434354515691   Best Score: 87.80434354515
691     Mean Score: 84.66166916481085
EPOCH 24 2020-06-28 13:49:32.185294
EPOCH #24       History Best Score: 87.89620264675278   Best Score: 87.89620264675
278     Mean Score: 85.99744120196013
EPOCH 25 2020-06-28 13:53:32.842477
EPOCH #25       History Best Score: 88.08642913252903   Best Score: 88.08642913252
903     Mean Score: 85.29718314402429
EPOCH 26 2020-06-28 13:57:34.696960
EPOCH #26       History Best Score: 88.48808801495998   Best Score: 88.48808801495
998     Mean Score: 85.58805838437915
EPOCH 27 2020-06-28 14:01:44.714118
EPOCH #27       History Best Score: 88.48808801495998   Best Score: 87.95396338587
986     Mean Score: 84.51000154220338
EPOCH 28 2020-06-28 14:05:51.065762
EPOCH #28       History Best Score: 88.48808801495998   Best Score: 88.31427370669
196     Mean Score: 83.4300515839381
EPOCH 29 2020-06-28 14:09:57.472962
EPOCH #29       History Best Score: 88.48808801495998   Best Score: 86.26068490419
445     Mean Score: 84.09396053802622
EPOCH 30 2020-06-28 14:14:02.428981
EPOCH #30       History Best Score: 88.48808801495998   Best Score: 88.29707260920
843     Mean Score: 82.87598518993218
EPOCH 31 2020-06-28 14:18:07.980000
EPOCH #31       History Best Score: 88.48808801495998   Best Score: 87.28556721902
329     Mean Score: 83.43625036596609
EPOCH 32 2020-06-28 14:22:13.396555
EPOCH #32       History Best Score: 88.48808801495998   Best Score: 86.04687683847
078     Mean Score: 82.82977246663141
EPOCH 33 2020-06-28 14:26:19.129013
EPOCH #33       History Best Score: 88.48808801495998   Best Score: 87.91279638645
159     Mean Score: 84.17759037173553
EPOCH 34 2020-06-28 14:30:25.150382
EPOCH #34       History Best Score: 88.48808801495998   Best Score: 87.20208732004
683     Mean Score: 82.65179533056452
EPOCH 35 2020-06-28 14:34:31.090190
EPOCH #35       History Best Score: 88.48808801495998   Best Score: 88.48808801495
```

```
998     Mean Score: 84.49332531612863
EPOCH 36 2020-06-28 14:38:37.764235
EPOCH #36       History Best Score: 88.48808801495998   Best Score: 88.48808801495
998     Mean Score: 85.16880541909377
EPOCH 37 2020-06-28 14:42:46.120017
EPOCH #37       History Best Score: 88.48808801495998   Best Score: 87.90158008259
698     Mean Score: 85.69513926372997
EPOCH 38 2020-06-28 14:46:57.651219
EPOCH #38       History Best Score: 88.48808801495998   Best Score: 88.19108545004
843     Mean Score: 85.86920748257418
EPOCH 39 2020-06-28 14:51:04.845301
EPOCH #39       History Best Score: 88.48808801495998   Best Score: 88.03500217966
791     Mean Score: 85.44241160670865
EPOCH 40 2020-06-28 14:55:11.101013
EPOCH #40       History Best Score: 88.48808801495998   Best Score: 88.48808801495
998     Mean Score: 86.23155682973368
EPOCH 41 2020-06-28 14:59:18.027502
EPOCH #41       History Best Score: 88.53444873755907   Best Score: 88.53444873755
907     Mean Score: 85.5982314890281
EPOCH 42 2020-06-28 15:03:25.194588
EPOCH #42       History Best Score: 88.53444873755907   Best Score: 88.48808801495
998     Mean Score: 86.97519242980901
EPOCH 43 2020-06-28 15:07:31.544116
EPOCH #43       History Best Score: 88.733825527644     Best Score: 88.73382552764
4       Mean Score: 87.59936405615463
EPOCH 44 2020-06-28 15:11:36.892880
EPOCH #44       History Best Score: 88.733825527644     Best Score: 88.48808801495
998     Mean Score: 86.52789761428511
EPOCH 45 2020-06-28 15:15:41.606579
EPOCH #45       History Best Score: 88.733825527644     Best Score: 86.80240780342
62      Mean Score: 85.3459088293601
EPOCH 46 2020-06-28 15:19:46.931973
EPOCH #46       History Best Score: 88.7643955622637    Best Score: 88.76439556226
37      Mean Score: 85.06132327501327
EPOCH 47 2020-06-28 15:23:52.155377
EPOCH #47       History Best Score: 88.7643955622637    Best Score: 87.83885725801
77      Mean Score: 85.61836790336835
EPOCH 48 2020-06-28 15:27:57.849435
EPOCH #48       History Best Score: 88.7643955622637    Best Score: 88.23336732120
195     Mean Score: 84.78868412879181
EPOCH 49 2020-06-28 15:32:03.787467
EPOCH #49       History Best Score: 88.7643955622637    Best Score: 88.68662734560
071     Mean Score: 84.92360372992026
EPOCH 50 2020-06-28 15:36:08.823038
EPOCH #50       History Best Score: 88.7643955622637    Best Score: 88.76439556226
37      Mean Score: 85.03941033147234
EPOCH 51 2020-06-28 15:40:13.364482
EPOCH #51       History Best Score: 88.7643955622637    Best Score: 88.73083543491
825     Mean Score: 85.1223510429688
EPOCH 52 2020-06-28 15:44:17.884383
EPOCH #52       History Best Score: 88.7643955622637    Best Score: 88.20854706316
246     Mean Score: 84.75070912170288
EPOCH 53 2020-06-28 15:48:24.316822
EPOCH #53       History Best Score: 88.7643955622637    Best Score: 88.03507130232
931     Mean Score: 84.17255770436577
EPOCH 54 2020-06-28 15:52:29.449371
EPOCH #54       History Best Score: 88.7643955622637    Best Score: 88.76439556226
37      Mean Score: 85.53179817534605
EPOCH 55 2020-06-28 15:56:34.072299
EPOCH #55       History Best Score: 88.7643955622637    Best Score: 88.33041338730
84      Mean Score: 85.3367471274926
```

```
EPOCH 56 2020-06-28 16:00:39.104144
EPOCH #56       History Best Score: 88.7643955622637    Best Score: 88.63545537855
717     Mean Score: 86.39956093285485
EPOCH 57 2020-06-28 16:04:44.009651
EPOCH #57       History Best Score: 88.7643955622637    Best Score: 88.76439556226
37      Mean Score: 86.4990950768383
EPOCH 58 2020-06-28 16:08:49.115399
EPOCH #58       History Best Score: 88.7643955622637    Best Score: 88.60776176828
671     Mean Score: 85.9056959069861

EPOCH 59 2020-06-28 16:12:53.994333
EPOCH #59       History Best Score: 88.7643955622637    Best Score: 88.58542623951
524     Mean Score: 86.28655366084368
EPOCH 60 2020-06-28 16:16:59.290230
EPOCH #60       History Best Score: 88.82879699227331   Best Score: 88.82879699227
331     Mean Score: 85.8848273454164
EPOCH 61 2020-06-28 16:21:04.381654
EPOCH #61       History Best Score: 88.82879699227331   Best Score: 88.76439556226
37      Mean Score: 87.15793181003373
EPOCH 62 2020-06-28 16:25:08.804796
EPOCH #62       History Best Score: 88.82879699227331   Best Score: 88.76439556226
37      Mean Score: 87.90899858774722
EPOCH 63 2020-06-28 16:29:14.181438
EPOCH #63       History Best Score: 88.82879699227331   Best Score: 87.73427467133
71      Mean Score: 85.14375350797508
EPOCH 64 2020-06-28 16:33:19.318772
EPOCH #64       History Best Score: 88.86558530151152   Best Score: 88.86558530151
152     Mean Score: 86.24597237612683
EPOCH 65 2020-06-28 16:37:25.087996
EPOCH #65       History Best Score: 88.86558530151152   Best Score: 87.81603787061
651     Mean Score: 86.03333186895695
EPOCH 66 2020-06-28 16:41:31.158693
EPOCH #66       History Best Score: 88.86558530151152   Best Score: 88.23478694706
094     Mean Score: 86.38851359550586
EPOCH 67 2020-06-28 16:45:36.883918
EPOCH #67       History Best Score: 88.86558530151152   Best Score: 87.97975750541
538     Mean Score: 85.6990920200806
EPOCH 68 2020-06-28 16:49:43.520287
EPOCH #68       History Best Score: 88.87357035135508   Best Score: 88.87357035135
508     Mean Score: 86.08994915029817
EPOCH 69 2020-06-28 16:53:49.229907
EPOCH #69       History Best Score: 88.87357035135508   Best Score: 88.71935707937
402     Mean Score: 85.63251330304021
EPOCH 70 2020-06-28 16:57:54.063383
EPOCH #70       History Best Score: 88.87357035135508   Best Score: 88.87357035135
508     Mean Score: 87.0267125139359
EPOCH 71 2020-06-28 17:01:59.969467
EPOCH #71       History Best Score: 88.87357035135508   Best Score: 88.64417405024
695     Mean Score: 86.89458622099653
EPOCH 72 2020-06-28 17:06:07.683909
EPOCH #72       History Best Score: 88.87357035135508   Best Score: 88.87006598602
862     Mean Score: 86.5506811500659
EPOCH 73 2020-06-28 17:10:15.399061
EPOCH #73       History Best Score: 89.11860619222306   Best Score: 89.11860619222
306     Mean Score: 86.82578436555995
EPOCH 74 2020-06-28 17:14:22.025022
EPOCH #74       History Best Score: 89.11860619222306   Best Score: 88.87357035135
508     Mean Score: 87.45685764387574
EPOCH 75 2020-06-28 17:18:29.273152
EPOCH #75       History Best Score: 89.11860619222306   Best Score: 88.92582431845
824     Mean Score: 87.45241632855445
```

```
EPOCH 76 2020-06-28 17:22:36.495790
EPOCH #76       History Best Score: 89.11860619222306   Best Score: 88.53022150280
039     Mean Score: 86.89523287117409
EPOCH 77 2020-06-28 17:26:44.038318
EPOCH #77       History Best Score: 89.11860619222306   Best Score: 87.61434808267
562     Mean Score: 86.11103824387888
EPOCH 78 2020-06-28 17:30:51.568751
EPOCH #78       History Best Score: 89.11860619222306   Best Score: 88.01775131386
819     Mean Score: 86.51465759859272
EPOCH 79 2020-06-28 17:34:59.489474
EPOCH #79       History Best Score: 89.11860619222306   Best Score: 89.05753463122
139     Mean Score: 85.79588752421981
EPOCH 80 2020-06-28 17:39:07.571052
EPOCH #80       History Best Score: 89.11860619222306   Best Score: 88.06523090194
628     Mean Score: 86.30098478287653
EPOCH 81 2020-06-28 17:43:14.489581
EPOCH #81       History Best Score: 89.11860619222306   Best Score: 88.22174334725
105     Mean Score: 86.2503433399394
EPOCH 82 2020-06-28 17:47:27.356423
EPOCH #82       History Best Score: 89.11860619222306   Best Score: 88.42115362342
527     Mean Score: 86.75285376747692
EPOCH 83 2020-06-28 17:51:55.549779
EPOCH #83       History Best Score: 89.11860619222306   Best Score: 87.61244091164
495     Mean Score: 85.79592660156439
EPOCH 84 2020-06-28 17:56:25.331640
EPOCH #84       History Best Score: 89.11860619222306   Best Score: 88.29564898959
566     Mean Score: 86.36823472704074
EPOCH 85 2020-06-28 18:00:55.660039
EPOCH #85       History Best Score: 89.11860619222306   Best Score: 88.57472236499
038     Mean Score: 87.11536535320298
EPOCH 86 2020-06-28 18:05:23.919105
EPOCH #86       History Best Score: 89.11860619222306   Best Score: 88.79869391963
277     Mean Score: 86.96390032143574
EPOCH 87 2020-06-28 18:09:45.930127
EPOCH #87       History Best Score: 89.11860619222306   Best Score: 89.11860619222
306     Mean Score: 86.8723812418915
EPOCH 88 2020-06-28 18:14:10.457606
EPOCH #88       History Best Score: 89.11860619222306   Best Score: 88.99674109692
441     Mean Score: 87.43275615399054
EPOCH 89 2020-06-28 18:18:46.925493
EPOCH #89       History Best Score: 89.11860619222306   Best Score: 88.10688298885
16      Mean Score: 86.33229375410704
EPOCH 90 2020-06-28 18:23:07.954422
EPOCH #90       History Best Score: 89.11860619222306   Best Score: 89.11860619222
306     Mean Score: 86.82676509324033
EPOCH 91 2020-06-28 18:27:28.554465
EPOCH #91       History Best Score: 89.11860619222306   Best Score: 88.54955802970
309     Mean Score: 86.97312302021177
EPOCH 92 2020-06-28 18:31:48.423841
EPOCH #92       History Best Score: 89.11860619222306   Best Score: 89.11860619222
306     Mean Score: 87.13046556723745
EPOCH 93 2020-06-28 18:36:13.218469
EPOCH #93       History Best Score: 89.11860619222306   Best Score: 89.04233102504
605     Mean Score: 87.18467420031993
EPOCH 94 2020-06-28 18:40:38.253872
EPOCH #94       History Best Score: 89.11860619222306   Best Score: 88.82575805286
686     Mean Score: 87.3912922650513
EPOCH 95 2020-06-28 18:46:08.798597
EPOCH #95       History Best Score: 89.16698683259384   Best Score: 89.16698683259
384     Mean Score: 87.39033473186399
EPOCH 96 2020-06-28 18:51:15.278981
EPOCH #96       History Best Score: 89.16698683259384   Best Score: 89.11860619222
306     Mean Score: 87.75374116420379
EPOCH 97 2020-06-28 18:56:07.811912
EPOCH #97       History Best Score: 89.16698683259384   Best Score: 89.16698683259
384     Mean Score: 88.08210024137634
EPOCH 98 2020-06-28 19:01:09.151009
EPOCH #98       History Best Score: 89.16698683259384   Best Score: 88.89719832028
861     Mean Score: 86.63314466942931
EPOCH 99 2020-06-28 19:06:09.225621
EPOCH #99       History Best Score: 89.16698683259384   Best Score: 89.16698683259
384     Mean Score: 87.11852306070229
EPOCH 100 2020-06-28 19:11:21.864490
EPOCH #100      History Best Score: 89.16698683259384   Best Score: 89.16698683259
384     Mean Score: 87.64049844504733
EPOCH 101 2020-06-28 19:16:34.751445
EPOCH #101      History Best Score: 89.16698683259384   Best Score: 89.16698683259
384     Mean Score: 88.3037057273194
EPOCH 102 2020-06-28 19:21:27.475411
EPOCH #102      History Best Score: 89.16698683259384   Best Score: 89.16698683259
384     Mean Score: 87.93173720915874
EPOCH 103 2020-06-28 19:26:09.627751
EPOCH #103      History Best Score: 89.17310925712187   Best Score: 89.17310925712
187     Mean Score: 88.55158691806017
EPOCH 104 2020-06-28 19:30:52.681106
EPOCH #104      History Best Score: 89.44781991734158   Best Score: 89.44781991734
158     Mean Score: 88.75805959479375
EPOCH 105 2020-06-28 19:35:27.584793
EPOCH #105      History Best Score: 89.44781991734158   Best Score: 89.21800611535
866     Mean Score: 88.82420121084472
EPOCH 106 2020-06-28 19:39:46.757104
EPOCH #106      History Best Score: 89.44781991734158   Best Score: 89.16698683259
384     Mean Score: 88.34000483244208
EPOCH 107 2020-06-28 19:44:04.725604
EPOCH #107      History Best Score: 89.44781991734158   Best Score: 88.98914190513
729     Mean Score: 87.96730809936336
EPOCH 108 2020-06-28 19:48:26.302106
EPOCH #108      History Best Score: 89.44781991734158   Best Score: 89.25296467094
695     Mean Score: 87.86645572478491
EPOCH 109 2020-06-28 19:52:51.696032
EPOCH #109      History Best Score: 89.44781991734158   Best Score: 89.27105115291
991     Mean Score: 88.33737703462555
EPOCH 110 2020-06-28 19:57:16.218659
EPOCH #110      History Best Score: 89.44781991734158   Best Score: 89.38360465769
998     Mean Score: 88.55210363299491
EPOCH 111 2020-06-28 20:01:39.664912
EPOCH #111      History Best Score: 89.45135776875736   Best Score: 89.45135776875
736     Mean Score: 88.76697286981621
EPOCH 112 2020-06-28 20:06:04.196439
EPOCH #112      History Best Score: 89.45135776875736   Best Score: 89.21957135962
433     Mean Score: 88.38487665291483
EPOCH 113 2020-06-28 20:10:29.653079
EPOCH #113      History Best Score: 89.45135776875736   Best Score: 89.38859807875
866     Mean Score: 88.70074735421494
EPOCH 114 2020-06-28 20:14:53.763725
EPOCH #114      History Best Score: 89.45135776875736   Best Score: 88.75901566872
489     Mean Score: 88.04425212813314
EPOCH 115 2020-06-28 20:19:25.357362
EPOCH #115      History Best Score: 89.45135776875736   Best Score: 89.21247107984
651     Mean Score: 88.36283922714105

EPOCH 116 2020-06-28 20:24:13.728862
EPOCH #116      History Best Score: 89.45135776875736   Best Score: 89.0484850923924
```

```
1       Mean Score: 88.41447955397759
EPOCH 117 2020-06-28 20:29:01.411540
EPOCH #117      History Best Score: 89.45135776875736   Best Score: 89.1745141368131
Mean Score: 88.1386157127176
EPOCH 118 2020-06-28 20:34:06.849945
EPOCH #118      History Best Score: 89.53326873692622   Best Score: 89.5332687369262
2       Mean Score: 88.32229591380619
EPOCH 119 2020-06-28 20:38:53.708901
EPOCH #119      History Best Score: 89.53326873692622   Best Score: 89.0225462760857
5       Mean Score: 88.26293444744728
EPOCH 120 2020-06-28 20:43:45.597125
EPOCH #120      History Best Score: 89.53326873692622   Best Score: 89.0656775879447
6       Mean Score: 88.07829923230837
EPOCH 121 2020-06-28 20:48:48.839010
EPOCH #121      History Best Score: 89.53326873692622   Best Score: 89.5332687369262
2       Mean Score: 88.24399964609196
EPOCH 122 2020-06-28 20:53:50.719258
EPOCH #122      History Best Score: 89.53326873692622   Best Score: 89.1302375392578
3       Mean Score: 88.11850788443796
EPOCH 123 2020-06-28 20:59:24.084112
EPOCH #123      History Best Score: 89.53326873692622   Best Score: 88.8881638348392
3       Mean Score: 88.24645855420583
EPOCH 124 2020-06-28 21:04:43.567352
EPOCH #124      History Best Score: 89.53326873692622   Best Score: 89.2524270502472
6       Mean Score: 88.11007066486476
EPOCH 125 2020-06-28 21:10:05.259227
EPOCH #125      History Best Score: 89.53326873692622   Best Score: 89.2221488668645
3       Mean Score: 88.25402886808111
EPOCH 126 2020-06-28 21:15:18.474585
EPOCH #126      History Best Score: 89.53326873692622   Best Score: 89.2129558601117
Mean Score: 88.15607893869374
EPOCH 127 2020-06-28 21:20:24.899806
EPOCH #127      History Best Score: 89.53326873692622   Best Score: 89.5332687369262
2       Mean Score: 88.33005295103075
EPOCH 128 2020-06-28 21:26:19.062571
EPOCH #128      History Best Score: 89.53326873692622   Best Score: 89.5332687369262
2       Mean Score: 88.30003454597012
EPOCH 129 2020-06-28 21:32:10.398857
EPOCH #129      History Best Score: 89.53326873692622   Best Score: 89.3621256355060
6       Mean Score: 88.37055400737862
EPOCH 130 2020-06-28 21:37:59.095151
EPOCH #130      History Best Score: 89.53326873692622   Best Score: 89.2589184361812
Mean Score: 88.44037933902146
EPOCH 131 2020-06-28 21:43:45.278967
EPOCH #131      History Best Score: 89.53326873692622   Best Score: 89.5332687369262
2       Mean Score: 88.29619793105122
EPOCH 132 2020-06-28 21:49:29.825651
EPOCH #132      History Best Score: 89.53326873692622   Best Score: 89.5332687369262
2       Mean Score: 88.29224792424647
EPOCH 133 2020-06-28 21:55:22.645491
EPOCH #133      History Best Score: 89.53326873692622   Best Score: 89.5332687369262
2       Mean Score: 88.50618517254398
EPOCH 134 2020-06-28 21:59:56.157159
EPOCH #134      History Best Score: 89.53326873692622   Best Score: 89.1904740985099
4       Mean Score: 88.45031092909153
EPOCH 135 2020-06-28 22:04:26.688200
EPOCH #135      History Best Score: 89.53326873692622   Best Score: 89.5332687369262
2       Mean Score: 88.620524668649
EPOCH 136 2020-06-28 22:08:54.313146
EPOCH #136      History Best Score: 89.53326873692622   Best Score: 89.5332687369262
2       Mean Score: 88.71456174850228
```

```
EPOCH 137 2020-06-28 22:13:20.986082
EPOCH #137      History Best Score: 89.53326873692622   Best Score: 89.5332687369262
2       Mean Score: 88.91156909055776
EPOCH 138 2020-06-28 22:17:44.569072
EPOCH #138      History Best Score: 89.68895478241262   Best Score: 89.6889547824126
2       Mean Score: 89.24727874834528
EPOCH 139 2020-06-28 22:22:14.507237
EPOCH #139      History Best Score: 89.68895478241262   Best Score: 89.5867702912644
6       Mean Score: 89.11316880399056
EPOCH 140 2020-06-28 22:26:39.491162
EPOCH #140      History Best Score: 89.68895478241262   Best Score: 89.5772316712046
9       Mean Score: 89.10221846877637
EPOCH 141 2020-06-28 22:31:05.278515
EPOCH #141      History Best Score: 89.72509794681119   Best Score: 89.7250979468111
9       Mean Score: 89.30923736814084
EPOCH 142 2020-06-28 22:35:32.097352
EPOCH #142      History Best Score: 89.72509794681119   Best Score: 89.6889547824126
2       Mean Score: 89.28392411191972
EPOCH 143 2020-06-28 22:40:45.770600
EPOCH #143      History Best Score: 89.72509794681119   Best Score: 89.5752707381040
7       Mean Score: 89.19323989587977
EPOCH 144 2020-06-28 22:46:13.223989
EPOCH #144      History Best Score: 89.84716119373914   Best Score: 89.8471611937391
4       Mean Score: 89.25069317740899
EPOCH 145 2020-06-28 22:51:35.293303
EPOCH #145      History Best Score: 89.84716119373914   Best Score: 89.7824442571817
7       Mean Score: 89.24843192938552
EPOCH 146 2020-06-28 22:56:47.571284
EPOCH #146      History Best Score: 89.84716119373914   Best Score: 89.6839456935506
7       Mean Score: 89.23042728864361
EPOCH 147 2020-06-28 23:01:55.243854
EPOCH #147      History Best Score: 89.84716119373914   Best Score: 89.7250979468111
9       Mean Score: 89.3081821108709
EPOCH 148 2020-06-28 23:07:06.872162
EPOCH #148      History Best Score: 89.8833940639916    Best Score: 89.8833940639916
Mean Score: 89.1029466836944
EPOCH 149 2020-06-28 23:11:35.462994
EPOCH #149      History Best Score: 89.8833940639916    Best Score: 89.8397871882222
3       Mean Score: 89.30150590630103
EPOCH 150 2020-06-28 23:16:12.918289
EPOCH #150      History Best Score: 89.8833940639916    Best Score: 89.6384039976860
8       Mean Score: 89.07833715760857
EPOCH 151 2020-06-28 23:21:01.502926
EPOCH #151      History Best Score: 89.97046035385273   Best Score: 89.9704603538527
3       Mean Score: 89.15221896031531
EPOCH 152 2020-06-28 23:26:20.048265
EPOCH #152      History Best Score: 89.97046035385273   Best Score: 89.5576395440607
8       Mean Score: 89.16621556193805
EPOCH 153 2020-06-28 23:31:53.320032
EPOCH #153      History Best Score: 89.97046035385273   Best Score: 89.6154110356018
5       Mean Score: 89.06760941272465
EPOCH 154 2020-06-28 23:36:44.405198
EPOCH #154      History Best Score: 89.97046035385273   Best Score: 89.8426301264821
8       Mean Score: 89.20183327122533
EPOCH 155 2020-06-28 23:42:03.463045
EPOCH #155      History Best Score: 89.97046035385273   Best Score: 89.5847718783207
7       Mean Score: 89.00166802198284
EPOCH 156 2020-06-28 23:46:54.786327


---------------------------------------------------------------------------
```

```
----
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-8-31ece2cd64e8> in <module>
     15     if __name__ == '__main__':
     16         pool = multiprocessing.Pool(processes=CPU_CORE)
---> 17         genomes[idx] = pool.map(genome_score, _genomes)
     18         pool.close()
     19         pool.join()

~₩anaconda3₩lib₩multiprocessing₩pool.py in map(self, func, iterable, chu
nksize)
    266         in a list that is returned.
    267         '''
--> 268         return self._map_async(func, iterable, mapstar, chunksize).get
()
    269
    270     def starmap(self, func, iterable, chunksize=None):

~₩anaconda3₩lib₩multiprocessing₩pool.py in get(self, timeout)
    649
    650     def get(self, timeout=None):
--> 651         self.wait(timeout)
    652         if not self.ready():
    653             raise TimeoutError

~₩anaconda3₩lib₩multiprocessing₩pool.py in wait(self, timeout)
    646
    647     def wait(self, timeout=None):
--> 648         self._event.wait(timeout)
    649
    650     def get(self, timeout=None):

~₩anaconda3₩lib₩threading.py in wait(self, timeout)
    550             signaled = self._flag
    551             if not signaled:
--> 552                 signaled = self._cond.wait(timeout)
    553             return signaled
    554

~₩anaconda3₩lib₩threading.py in wait(self, timeout)
    294         try:    # restore state no matter what (e.g., KeyboardInte
rrupt)
    295             if timeout is None:
--> 296                 waiter.acquire()
    297                 gotit = True
    298             else:

KeyboardInterrupt:
```

In [*]:

```
score_history = []
high_score_history = []
mean_score_history = []

while n_gen <= EPOCHS:
    print('EPOCH', n_gen, datetime.datetime.now())
    genomes = np.array(genomes)
    while len(genomes)%CPU_CORE != 0:
        genomes = np.append(genomes, Genome(score_ini, input_length, output_length_1, output_length_
    genomes = genomes.reshape((len(genomes)//CPU_CORE, CPU_CORE))

    for idx, _genomes in enumerate(genomes):
        if __name__ == '__main__':
            pool = multiprocessing.Pool(processes=CPU_CORE)
            genomes[idx] = pool.map(genome_score, _genomes)
            pool.close()
            pool.join()
    genomes = list(genomes.reshape(genomes.shape[0]*genomes.shape[1]))

    # score에 따라 정렬
    genomes.sort(key=lambda x: x.score, reverse=REVERSE)

    # 평균 점수
    s = 0
    for i in range(N_BEST):
        s += genomes[i].score
    s /= N_BEST

    # Best Score
    bs = genomes[0].score

    # Best Model 추가
    if best_genomes is not None:
        genomes.extend(best_genomes)

    # score에 따라 정렬
    genomes.sort(key=lambda x: x.score, reverse=REVERSE)

    score_history.append([n_gen, genomes[0].score])
    high_score_history.append([n_gen, bs])
    mean_score_history.append([n_gen, s])

    if genomes[0].score > best_score_ever:
        best_score_ever = genomes[0].score
        best_gen = genomes[0]

    # 결과 출력
    print('EPOCH #%s₩tHistory Best Score: %s₩tBest Score: %s₩tMean Score: %s' % (n_gen, genomes[0].s

    # 모델 업데이트
    best_genomes = deepcopy(genomes[:N_BEST])

    # CHILDREN 생성
    for i in range(N_CHILDREN):
        new_genome = deepcopy(best_genomes[0])
        a_genome = np.random.choice(best_genomes)
        b_genome = np.random.choice(best_genomes)

        for j in range(input_length):
            cut = np.random.randint(new_genome.w1.shape[1])
            new_genome.w1[j, :cut] = a_genome.w1[j, :cut]
            new_genome.w1[j, cut:] = b_genome.w1[j, cut:]

        for j in range(h1):
            cut = np.random.randint(new_genome.w2.shape[1])
            new_genome.w2[j, :cut] = a_genome.w2[j, :cut]
            new_genome.w2[j, cut:] = b_genome.w2[j, cut:]

        for j in range(h2):
            cut = np.random.randint(new_genome.w3.shape[1])
            new_genome.w3[j, :cut] = a_genome.w3[j, :cut]
            new_genome.w3[j, cut:] = b_genome.w3[j, cut:]

        for j in range(h3):
            cut = np.random.randint(new_genome.w4.shape[1])
            new_genome.w4[j, :cut] = a_genome.w4[j, :cut]
            new_genome.w4[j, cut:] = b_genome.w4[j, cut:]

        for j in range(input_length):
            cut = np.random.randint(new_genome.w5.shape[1])
            new_genome.w5[j, :cut] = a_genome.w5[j, :cut]
            new_genome.w5[j, cut:] = b_genome.w5[j, cut:]

        for j in range(h1):
            cut = np.random.randint(new_genome.w6.shape[1])
            new_genome.w6[j, :cut] = a_genome.w6[j, :cut]
            new_genome.w6[j, cut:] = b_genome.w6[j, cut:]

        for j in range(h2):
            cut = np.random.randint(new_genome.w7.shape[1])
            new_genome.w7[j, :cut] = a_genome.w7[j, :cut]
            new_genome.w7[j, cut:] = b_genome.w7[j, cut:]

        for j in range(h3):
            cut = np.random.randint(new_genome.w8.shape[1])
            new_genome.w8[j, :cut] = a_genome.w8[j, :cut]
            new_genome.w8[j, cut:] = b_genome.w8[j, cut:]

        best_genomes.append(new_genome)

    # 모델 초기화
    genomes = []
    for i in range(int(N_POPULATION / len(best_genomes))):
        for bg in best_genomes:
            new_genome = deepcopy(bg)
            mean = 0
            stddev = 0.2
            # 50% 확률로 모델 변형
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w1 += new_genome.w1 * np.random.normal(mean, stddev, size=(input_length,
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w2 += new_genome.w2 * np.random.normal(mean, stddev, size=(h1, h2)) * np
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w3 += new_genome.w3 * np.random.normal(mean, stddev, size=(h2, h3)) * np
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w4 += new_genome.w4 * np.random.normal(mean, stddev, size=(h3, output_le
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w5 += new_genome.w5 * np.random.normal(mean, stddev, size=(input_length,
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w6 += new_genome.w6 * np.random.normal(mean, stddev, size=(h1, h2)) * np
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w7 += new_genome.w7 * np.random.normal(mean, stddev, size=(h2, h3)) * np
            if np.random.uniform(0, 1) < PROB_MUTATION:
                new_genome.w8 += new_genome.w8 * np.random.normal(mean, stddev, size=(h3, output_le
            genomes.append(new_genome)

    if REVERSE:
        if bs < score_ini:
            genomes[len(genomes)//2:] = [Genome(score_ini, input_length, output_length_1, output_le
    else:
        if bs > score_ini:
            genomes[len(genomes)//2:] = [Genome(score_ini, input_length, output_length_1, output_le

    n_gen += 1
```

```
EPOCH 156 2020-06-29 01:34:12.917482
EPOCH #156      History Best Score: 89.97046035385273   Best Score: 89.9704603538527
3       Mean Score: 89.51574597329774
EPOCH 157 2020-06-29 01:39:41.770176
EPOCH #157      History Best Score: 89.97046035385273   Best Score: 89.9704603538527
3       Mean Score: 89.47766181384623
EPOCH 158 2020-06-29 01:45:19.673681
```

6. 결과 및 결언

Conclusion & Discussion

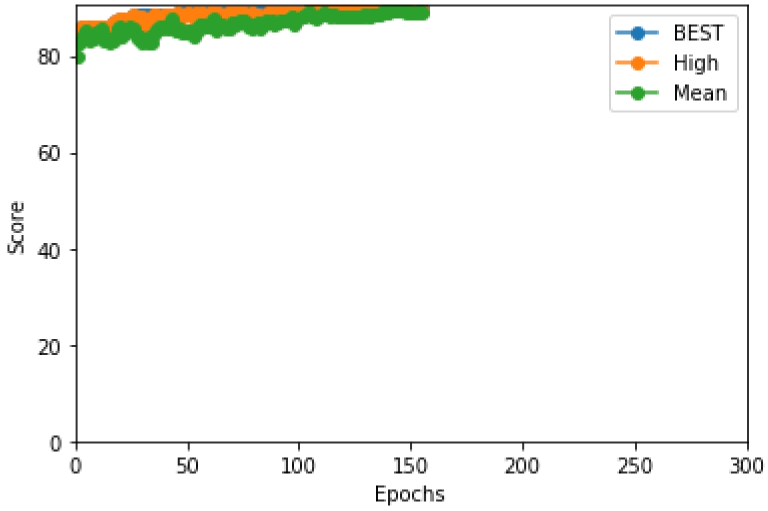
결과 그래프

In [9]:

```
import matplotlib.pyplot as plt

# Score Graph
score_history = np.array(score_history)
high_score_history = np.array(high_score_history)
mean_score_history = np.array(mean_score_history)

plt.plot(score_history[:,0], score_history[:,1], '-o', label='BEST')
plt.plot(high_score_history[:,0], high_score_history[:,1], '-o', label='High')
plt.plot(mean_score_history[:,0], mean_score_history[:,1], '-o', label='Mean')
plt.legend()
plt.xlim(0, EPOCHS)
plt.ylim(bottom=0)
plt.xlabel('Epochs')
plt.ylabel('Score')
plt.show()
```



Submission 파일 만들기

In [10]:

```
# 재고 계산
from module.simulator import Simulator
simulator = Simulator()
order = pd.read_csv('module/order.csv')
submission = best_gen.predict(order)
_, df_stock = simulator.get_score(submission)

# PRT 개수 계산
PRTs = df_stock[['PRT_1', 'PRT_2', 'PRT_3', 'PRT_4']].values
PRTs = (PRTs[:-1] - PRTs[1:])[24*23:]
PRTs = np.ceil(PRTs * 1.1)
PAD = np.zeros((24*23+1, 4))
PRTs = np.append(PRTs, PAD, axis=0).astype(int)

# Submission 파일에 PRT 입력
submission.loc[:, 'PRT_1':'PRT_4'] = PRTs
submission.to_csv('Dacon_baseline2.csv', index=False)
```

점수 향상 팁

해당 코드는 단순한 모델로 다음 방법으로 점수 향상을 꾀할 수 있습니다.

1. 성형 공정 2개 라인을 따로 모델링
2. CHANGE, STOP 이벤트 활용
3. 수요 초과분 외 다양한 양상을 반영하는 목적함수
4. 유전 알고리즘 외 효율적인 학습 기법