Machine Learning (2021 Fall semester)

Sangeun Chae
2018314760

1. Dataset preprocessing

1) Yelp dataset 의 json 파일에 포함된 'user id', 'business id', 'stars' 항목을 사용하여 User-item Interaction Matrix 을 구성합니다. 100 개의 가장 많이 사용된 user id 와 가장 많이 사용된 business id 을 가지는 data 만을 사용하세요.

```
df = pd.DataFrame(json_data).loc[:, ['user_id', 'business_id', 'stars']]

user_id = df['user_id'].tolist()
user_id_100 = list()
temp_user = Counter(user_id).most_common(100)
for key in temp_user:
  user_id_100.append(key[0])


business_id = df['business_id'].tolist()
business_id_50 = list()
temp_business = Counter(business_id).most_common(50)
for key in temp_business:
  business_id_50.append(key[0])
```

Figure 1: Data processing (Most frequent data processing)

User id, Business id, Stars 을 Column 로 가지는 dataframe 을 생성하고, json 파일에서 값을 읽어 해당 dataframe 에 저장한다. 이후, list 에 data frame 의 user id 와 business 값을 각각 저장한다. 해당 list 에 counter 를 사용하여 가장 많이 사용된 100 개의 user id 와 가장 많이 사용된 50 개의 business id 를 저장한다.

```
interac_matrix = np.zeros((100,50))

for x, a in tqdm(enumerate(user_id_100)):
  temp = df['user_id'] == a
  for y, b in enumerate(business_id_50):
    temp_ = df['business_id'] == b
    data = df[temp & temp_]
    if len(data) > 0:
      data = data.iloc[[0], [2]].values[0]
      interac_matrix[x][y] = data
```

Figure 2: Data processing using & operation

이후, 저장된 100 개의 user id 와 50 개의 business id 를 이용하여 for 문을 순회하며, dataframe 에 저장된 json 값들 중, user id 와 business id 를 모두 가지는 row 를 따로 추출하여, 0 으로 초기화 된 matrix 에 순차적으로 값을 저장한다. 반복문이 종료되고 나면, matrix 에는 가장 많이 사용된 100 개의 user id 와 가장 많이 사용된 50 개의 business 가 matrix 의 축을 이루게 되고, 해당 축에는 stars 값들이 저장되게 된다.

2) Dataset 을 random 하게 training:test= 80:20 으로 나누세요. Training dataset 에는 각각 user_id 와 각 business_id 에 최소한 한개의 data 가 존재해야 합니다.

```
[10] non_zero = np.count_nonzero(interac_matrix != 0)

     test_count = int(non_zero * 0.2)
     train_count = int(non_zero - test_count)
```

```
print(non_zero)
print(test_count)
print(train_count)
```

```
789
157
632
```

Figure 3: Separate dataset (80:20)

1)번 과정에서 생성된 matrix 를 바탕으로 dataset 를 80:20 비율로 분리한다. 우선, matrix 에서 유의미한 값들이 저장된 수를 구하고(0 이 아닌 값), 유의미한 값들의 수를 80:20 으로 나눈다. **[Figure 3]** 에서 확인할 수 있듯이, 유의미한 값들은 총 789 개 있음을 알 수 있고, 해당 값들을 632:157 (80:20) 로 나눴음을 알 수 있다.

또한, Training dataset 에는 각 user_id 와 business_id 에 최소한 한개의 데이터가 존재해야 한다는 조건을 만족시키기 위해, 각각의 value 가 속한 좌표에서 모든 행의 합이 해당 value 의 값과 같거나, 모든 열의 합이 해당 value 의 값이 같으면 데이터를 분리하지 않았다. 위의 조건이 의미하는 바는, 해당 value 는 user id 와 business id 가 유일하게 하나밖에 존재하지 않는다는 의미이다. 따라서 해당 데이터를 test data 로 분리하게 된다면, 해당 데이터는 유의미한 학습 데이터가 아니게 된다.

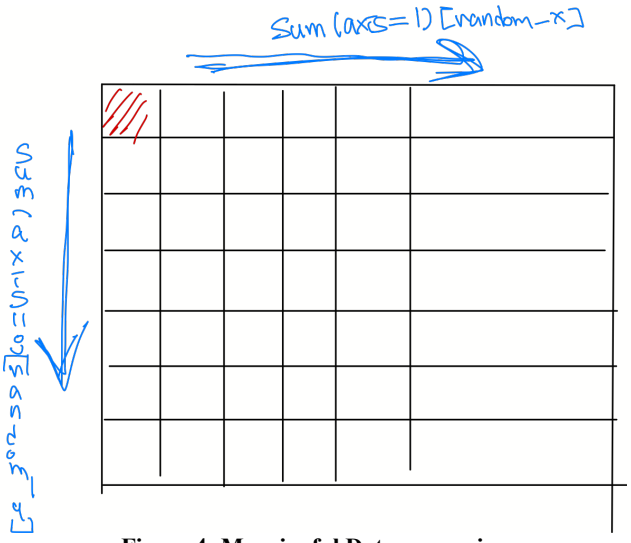

Figure 4: Meaningful Data processing

2. Train and Test

1) 학습 데이터를 활용하여 User-item Interaction Matrix(Rating Matrix)를 User Matrix 와 Item Matrix 로 Decomposition 한 후, 생성된 User Matrix 를 통해 예측한 값을 테스트 데이터와 비교해보세요.

2) 적절한 Loss function 을 구성하고 Stochastic Gradient Descent(SGD)을 사용하여 optimization problem 을 해결하세요.

3) 테스트 data 대한 RMSE(Root Mean Squared Error)값이 어떻게 변화하는지 비교하세요.

4) Latent vector (rank) 크기를 각각 5, 10, 15, 20 으로 변경하면서 1)~3)을 반복하세요.

Loss Function 은 다른 명칭으로 Cost Function 이라고 한다. Loss function 은 데이터를 토대로 산출한 모델의 예측 값과 실제 값과의 차이를 표현하는 지표이다. Loss function 에서는 bias 값을 이용하는데, bias 는 하나의 뉴런으로 입력된 모든 값을 다 더한 다음, 이 값이 더 해주는 상수이다. 이 값은 하나의 뉴런에서 활성화 함수를 거쳐 최종적으로 출력되는 값을 조절하는 역할을 한다. 이번 과제에서 사용하는 loss function 은 RMSE(Root Mean Squared Error) 이며, 희귀를 사용하는 loss function 이다. RMSE 는 예측 값과 정답 값의 차이에 절대값을 취하여, 그 값들을 전부 더하고, 개수로 나누어 평균을 낸 값을 루트를 씌운 값이다. 아래의 그래프는 Rank 를 변화시키면서 측정된 RMSE 를 나타낸 것이다.

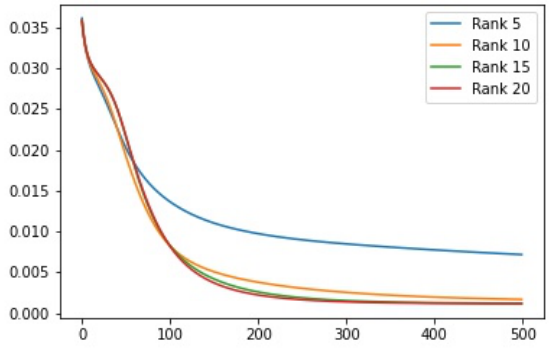

Figure 5: Training Result

[Figure 5] 를 통해 확인할 수 있듯이, 전반적으로 학습이 진행될수록 loss 값이 감소하는 것을 확인할 수 있다. 또한 Rank 값이 커질수록, loss 값이 감소하는 것을 확인할 수 있다. 다음은 각 Rank 별 Test 상황에서의 RMSE 값을 나타낸 것이다. Test 도 Train 과 마찬가지로, Rank 값이 증가할 수록 loss 값이 감소하는 것을 알 수 있다.

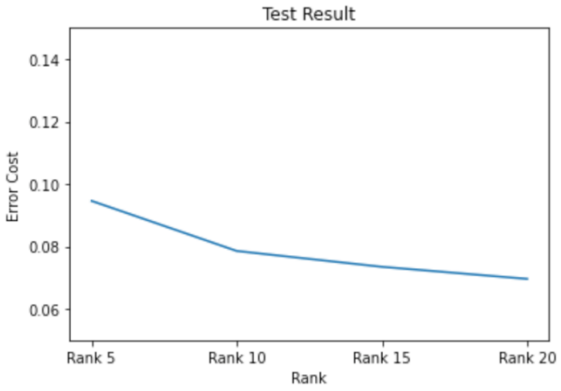

Figure 6: Test Result

Rank	Cost
Rank5	0.1044
Rank10	0.0771
Rank15	0.0763
Rank20	0.0730

Table 1: Test Result