Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет Компьютерных сетей и систем

Кафедра Информатики

ЛАБОРАТОРНАЯ РАБОТА №9
«Рекомендательные системы»

БГУИР 1-40 81 04

Магистрант:
гр. 858642
Кукареко А.В.

Проверил:
Стержанов М. В.

Минск, 2019

ХОД РАБОТЫ

Задание.

Набор данных ex9_movies.mat представляет собой файл формата *.mat (т.е. сохраненного из Matlab). Набор содержит две матрицы Y и R - рейтинг 1682 фильмов среди 943 пользователей. Значение Rij может быть равно 0 или 1 в зависимости от того оценил ли пользователь j фильм i. Матрица Y содержит числа от 1 до 5 - оценки в баллах пользователей, выставленные фильмам.

1. Загрузите данные ex9_movies.mat из файла.
2. Выберите число признаков фильмов (n) для реализации алгоритма коллаборативной фильтрации.
3. Реализуйте функцию стоимости для алгоритма.
4. Реализуйте функцию вычисления градиентов.
5. При реализации используйте векторизацию для ускорения процесса обучения.
6. Добавьте L2-регуляризацию в модель.
7. Обучите модель с помощью градиентного спуска или других методов оптимизации.
8. Добавьте несколько оценок фильмов от себя. Файл movie_ids.txt содержит индексы каждого из фильмов.
9. Сделайте рекомендации для себя. Совпали ли они с реальностью?
10. Также обучите модель с помощью сингулярного разложения матриц. Отличаются ли полученные результаты?
11. Ответы на вопросы представьте в виде отчета.

Результат выполнения:

1. Загрузите данные ex9_movies.mat из файла.

```
movie_data = scipy.io.loadmat('ex9_movies.mat')
Y = movie_data['Y']
R = movie_data['R']

Nm, Nu = Y.shape

print(f'Y.shape = {Y.shape}')
print(f'R.shape = {R.shape}')

print(f'Users: {Nu}, Movies: {Nm}')
```

```
Y.shape = (1682, 943)
R.shape = (1682, 943)
Users: 943, Movies: 1682
```

2. Выберите число признаков фильмов (n) для реализации алгоритма коллаборативной фильтрации.

После нескольких экспериментов, было решено выбрать 100 фич. При таком количестве фич, система показывала что-то что я теоретически мог-бы посмотреть.

3. Реализуйте функцию стоимости для алгоритма.

Функция стоимости:

```
def h(theta, X):
    return np.dot(X, theta.T)
```

```
def cost(theta, X, y, r):
    y_pred = h(theta, X)
    y_pred = y_pred * r

    return np.sum(np.power((y_pred - y), 2))
```

```
def J_combined(theta, X, y, r, lmb = 0.):
    reg = 0
    error = cost(theta, X, y, r)

    if lmb != 0:
        reg += lmb * np.sum(np.square(theta))
        reg += lmb * np.sum(np.square(X))

    return (error + reg) / 2
```

4. Реализуйте функцию вычисления градиентов.

Функция вычисления градиентов:

```
def gd_step(data, Y, R, Nm, Nu, Nf, lmb = 0.):
    theta, X = roll_up(data, Nm, Nu, Nf)

    error = (h(theta, X) * R) - Y

    X_gd = np.dot(error, theta)
    theta_gd = np.dot(error.T, X)

    if lmb != 0:
        X_gd += lmb * X
        theta_gd += lmb * theta

    return unroll(theta_gd, X_gd)
```

5. При реализации используйте векторизацию для ускорения процесса обучения.

Векторизация была добавлена во время выполнения задания № 4.

6. Добавьте L2-регуляризацию в модель.

Регуляризация была добавлена во время выполнения задания № 4.

7. Обучите модель с помощью градиентного спуска или других методов оптимизации.

Функция обучения модели:

```
def build_model(init_theta, init_X, Y, R, Nm, Nu, Nf, lmb = 0.):
    data = optimize.fmin_cg(
        J_gd,
        x0=unroll(init_theta, init_X),
        fprime=gd_step,
        args=(Y, R, Nm, Nu, Nf, lmb),
        maxiter=400,
        disp=True
    )

    return roll_up(data, Nm, Nu, Nf)
```

8. Добавьте несколько оценок фильмов от себя. Файл movie_ids.txt содержит индексы каждого из фильмов.

Я добавил оценки 20 фильмам:

```
Toy Story (1995)            3
GoldenEye (1995)            5
Star Wars (1977)            3
Lion King, The (1994)       2
Mask, The (1994)            3
Terminator 2: Judgment Day (1991)       5
Back to the Future (1985)       3
Indiana Jones and the Last Crusade (1989)       5
Star Trek: First Contact (1996) 3
Die Hard 2 (1990)       4
Star Trek VI: The Undiscovered Country (1991)   3
Star Trek: The Wrath of Khan (1982)     3
Star Trek III: The Search for Spock (1984)      3
Star Trek IV: The Voyage Home (1986)    3
Fifth Element, The (1997)       5
Naked Gun 33 1/3: The Final Insult (1994)       3
Mission: Impossible (1996)      5
Spy Hard (1996) 3
Jackie Chan's First Strike (1996)       5
Die Hard: With a Vengeance (1995)       5
```

9. Сделайте рекомендации для себя. Совпали ли они с реальностью?

Рекомендации, которые выдала мне система:

```
recommend_movies(y_pred_gd, my_R, 943, 10)
```

```
Top 10:
5.37 - Three Colors: White (1994)
5.33 - Man Who Knew Too Little, The (1997)
5.28 - Fly Away Home (1996)
5.27 - Tin Drum, The (Blechtrommel, Die) (1979)
5.07 - Private Benjamin (1980)
5.07 - Richard III (1995)
5.07 - Pink Floyd - The Wall (1982)
5.03 - Face/Off (1997)
5.02 - Home Alone (1990)
5.01 - Men in Black (1997)
```

Некоторые фильмы из этого списка я бы посмотрел или уже смотрел и они мне понравились. Из 10 фильмов мне интересными показались только 3.

Для получения такого результата, алгоритм совершил 400 итераций и достиг функции стоимости в «2303» единиц.

MSE для предсказанных результатов получилось «10.016».

10. Также обучите модель с помощью сингулярного разложения матриц. Отличаются ли полученные результаты?

Функция обучения использующая сингулярное разложение:

```
def predict_using_svd(Y, feat_count):
    U, Sigma, Vt = np.linalg.svd(Y)

    U_f = U[:, : feat_count]
    S_f = np.diag(Sigma[: feat_count])
    Vt_f = Vt[: feat_count]

    predictions = np.dot(np.dot(U_f, S_f), Vt_f)

    return predictions
```

Сингулярное разложение справилось значительно быстрее чем градиентный спуск. Ошибка MSE для предсказанных результатов тоже значительно меньше: «0.246».

Рекомендации которые мне выдал алгоритм:

```
recommend_movies(y_pred_svd, my_R, 943, 10)
```

```
Top 10:
1.47 - Grand Day Out, A (1992)
1.47 - Brazil (1985)
1.25 - Manon of the Spring (Manon des sources) (1986)
1.21 - Age of Innocence, The (1993)
1.09 - Monty Python and the Holy Grail (1974)
1.07 - Highlander (1986)
1.07 - Haunted World of Edward D. Wood Jr., The (1995)
1.01 - D3: The Mighty Ducks (1996)
1.01 - Black Sheep (1996)
1.00 - Kolya (1996)
```

Из рекомендуемых 10 фильмов я бы тоже посмотрел 3, как и в предыдущем задании, однако качество выборки мне показалось хуже, так как в своем рейтинге я поставил 5 балов всем «action» фильмам, а мне предложили только 1 фильм – это «Highlander».

Для повышения точности нужно оценить больше фильмов, и возможно провести эксперименты на более свежих данных.

Вывод.

В ходе выполнения лабораторной работы я ознакомился с задачей рекомендательных систем и узнал несколько подходов решения этой задачи. Так же в ходе лабораторной работы я реализовал алгоритм «коллаборативной фильтрации», создал две рекомендательные системы: на основе «коллаборативной фильтрации», на основе «сингулярного разложения матриц» и сравнил результаты их работы.