RESEARCH

Introduction to the profile areas of data sciences: project 10

Rohan Chitte Natalja Amiridze

Abstract

Goal of the project: The goal of this project was to perform the analyses described in the tutorial paper for each of the SVD hands-on parts in Python

Main results of the project: Using statistical approach with SVD it was possible to reduce dimensionality of the data, extract needed features, build models (linear and logistic regression) and make predictions of psychological outcomes.

Personal key learnings: We learnt about different types of dimension reduction techniques, working with sparse matrix and singular value decomposition (SVD)

Estimated working hours: 14

Project evaluation: 2

Number of words: 1366

1 Scientific Background

Nowadays the big data is constantly increasing. This makes the task of optimal and active processing this amount of data of great importance. To address the problem there are many analytic approaches. Considering the paper by Kosinski et al.[1], there are two complementary approaches described. One of them is based on SVD (singular value decomposition) and another one Both use dimensionality reduction to extract features from large datasets. Then the extracted features are used to build models that can predict psychological outcomes. The benefit of these approaches is that predictive models based on digital footprints can be used to develop psychometric measures that are useful for psychological research and practice.

2 Goal

The goal was to perform the statistical analysis of the big data with SVD, which was described in the tutorial paper[1]

3 Data

The sample dataset used in this project contains psychodemographic profiles of 110,728 Facebook users and their Facebook likes. For simplicity the samples are limited to US users. The sample dataset comprises of three .csv-files: users.csv, likes.csv and users-likes.csv

4 Sample, Constructing Matrix and Trimming Matrix

To load the data, Pandas Framework was used to import the CSV files into pandas dataframes users, likes and ul (user-likes). For convenience, sparse matrix was used to represent the user-Like data(digital footprint) in matrix form. Sparse matrices to store data that contains a large number of zero-valued elements can both save a significant amount of memory and speed up the processing of that data. First, we matched the "userid" from users and user id from ul and stored its index value in user_row. Then, we did the same thing for "likeid" in likes and ul dataframe and stored the index value of the all the matches in like_row. We then appended user_row and like_row as columns in ul dataframe to resemble the output generated by R's match() function. Further, we used scipy's sparse library to construct a sparse matrix where the i parameter indicates the location row and columns according to user_row and like_row respectively. We named this matrix "mmatrix". A function called sp_matr_update was used to remove the least frequent data points from the user-Like matrix. A threshold of minimum 50 likes per user and 150 users per like was used to trim the Matrix. This process is repeated in a loop with range=9 to reduce the matrix significantly. The users that were deleted from this matrix were also removed from the dataframe "user" and saved as another dataframe called newusers.

5 Reducing the Dimensionality of the User–Like Matrix Using SVD

5.1 Background and goal

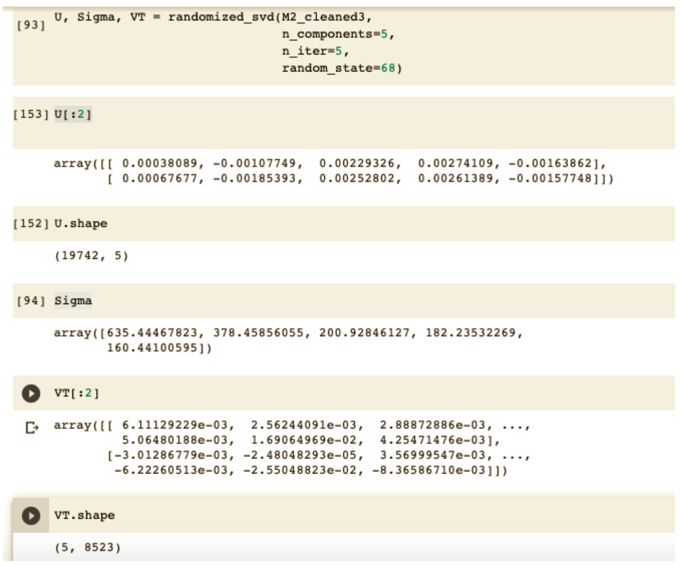
SVD is a matrix decomposition method. It is used to reduce the dimensionality of the matrix and extract patterns from the user–Like Matrix constructed in the previous section. Moreover, to simplify the SVD's Dimensions, Varimax rotation was implemented.

5.2 Method

Singular value decomposition (SVD) with parameter n_component (which corresponds to k) = 5 is implemented using sklearn's randomized_svd to reduce the dimensionality of the Matrix. The output is 3 matrices U, Sigma and V where U contains left singular vectors and V contains right singualr vectors. Furtherahead, factor rotation techniques is used to simplify SVD dimensions and increase their interpretability by mapping the original space into a new orthagonal rotates space using Varimax. We used the Varimax function to rotate SVD's dimensions and the code which was adapted through https://cutt.ly/fj34qt4 Wikipedia. SVD's V dimension was rotated using varimax function and the output of varimax was stored in variable v_vot. In order to rotate the other dimension u, we calculated the dot product of Matrix M and v_rot using numpy and stored it as u_rot.

5.3 Results

SVD represents a given matrix (of sizemrows*ncolumns) asa product of three matrices: a Matrix U (of size m*k) containing left singular vectors; a non-negative square diagonal Matrix*(ofsizek) containing singular values; and a MatrixV(of sizen*k) containing right singular vectors, where k is the number of dimensions that the researcher chose to extract. A popular approach to selecting the right number k of SVD dimensions to extract is plotting the singular values against k.The optimum k lies at the “knee” of the resulting scree plot which according to the figure



```
[93] U, Sigma, VT = randomized_svd(M2_cleaned3,
                                   n_components=5,
                                   n_iter=5,
                                   random_state=68)

[153] U[:2]

     array([[ 0.00038089, -0.00107749,  0.00229326,  0.00274109, -0.00163862],
            [ 0.00067677, -0.00185393,  0.00252802,  0.00261389, -0.00157748]])

[152] U.shape

     (19742, 5)

[94] Sigma

     array([635.44467823, 378.45856055, 200.92846127, 182.23532269,
            160.44100595])

     VT[:2]

     array([[ 6.11129229e-03,  2.56244091e-03,  2.88872886e-03, ...,
              5.06480188e-03,  1.69064969e-02,  4.25471476e-03],
            [-3.01286779e-03, -2.48048293e-05,  3.56999547e-03, ...,
             -6.22260513e-03, -2.55048823e-02, -8.36586710e-03]])

     VT.shape

     (5, 8523)
```

Figure 1 SVD and its dimensions

```python
import matplotlib.pyplot as plt
plt.plot(Sigma)
plt.ylabel('Msvd$d')
plt.show()
```

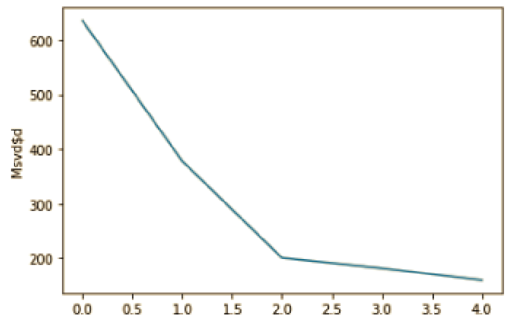
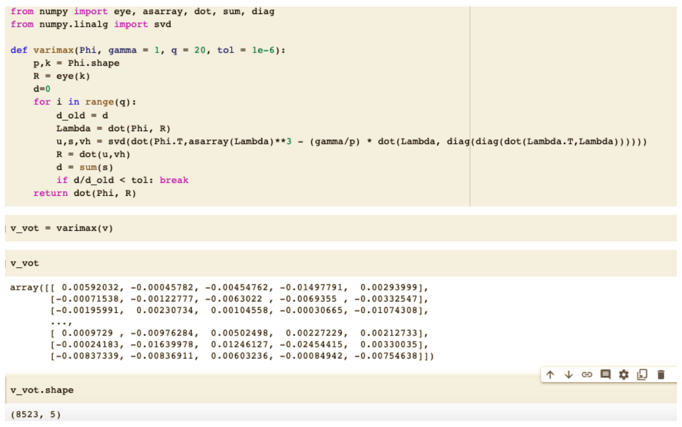


Figure 2 Sigma(Singular values) vs K

below lies at value 2 which means factor 3. Varimax is one of the most popular orthogonal rotations.It minimizes both the number of dimensions related to each variable and the number of variables related to each dimension, thus improving the interpretability of the data.

5.4 Discussion

SVD's play a virtal role in Machine learning as often a set of data can be represented in a 2-d matrix where the columns are features and the rows are data points. the Matrix computations using all of the values in a matrix are sometimes redundant and infact computationally expensive. To address such a problem, SVD decomposes the matrix and resolves this issue.



```
from numpy import eye, asarray, dot, sum, diag
from numpy.linalg import svd

def varimax(Phi, gamma = 1, q = 20, tol = 1e-6):
    p,k = Phi.shape
    R = eye(k)
    d=0
    for i in range(q):
        d_old = d
        Lambda = dot(Phi, R)
        u,s,vh = svd(dot(Phi.T,asarray(Lambda)**3 - (gamma/p) * dot(Lambda, diag(diag(dot(Lambda.T,Lambda))))))
        R = dot(u,vh)
        d = sum(s)
        if d/d_old < tol: break
    return dot(Phi, R)
```

```
v_vot = varimax(v)
```

```
v_vot
```

```
array([[ 0.00592032, -0.00045782, -0.00454762, -0.01497791,  0.00293999],
       [-0.00071538, -0.00122777, -0.0063022 , -0.0069355 , -0.00332547],
       [-0.00195991,  0.00230734,  0.00104558, -0.00030665, -0.01074308],
       ...,
       [ 0.0009729 , -0.00976284,  0.00502498,  0.00227229,  0.00212733],
       [-0.00024183, -0.01639978,  0.01246127, -0.02454415,  0.00330035],
       [-0.00837339, -0.00836911,  0.00603236, -0.00084942, -0.00754638]])
```

```
v_vot.shape
```

```
(8523, 5)
```

Figure 3 varimax

6 Interpreting Clusters and Dimensions

6.1 Background and goal

A correlation matrix is used to summarize data, as an input into a more advanced analysis, and as a diagnostic for advanced analyses. The goal is to find correlations between user scores on the varimax-rotated SVD dimensions(Urot) and psychodemographic user traits.

6.2 Method

To draw pairwise correlation between multidimensional matrices, we used one vectorized solution using broadcasting:
https://stackoverflow.com/questions/33650188/efficient-pairwise-correlation-for-two-matrices-of-features/3365144233651442.

6.3 Results

The correlation matrix between user scores on the varimax-rotated SVD dimensions (Urot) and psychodemographic user traits is more easily interpreted using a heatmap. The results show that openness correlates strongly with SDA3 SDA2, con correlates with SDA1 and SDA5, ext with SDA2, neu with SDA 1, 4 and 5. Overall, it can be seen that SDA correlate with personality traits.

6.4 Discussion

Correlation is a powerful tool which can be used to summarize a large dataset and to identify and visualize patterns in the given data.

7 Predicting Real-Life Outcomes With Facebook Likes

7.1 Method

To prepare our regression model to make predictions, the data was first divided into 10 folds/subsections. For our test data we selected one fold and the rest as our train data. We imported sklearn logistic regression and linear regression package for binomial and continuous data respectively to predict gender and once to predict openness (ope). Fit() function was used to fit the model. Finally, to make predictions predict() function was used. Sklearn's roc_curve and roc_auc_score was used

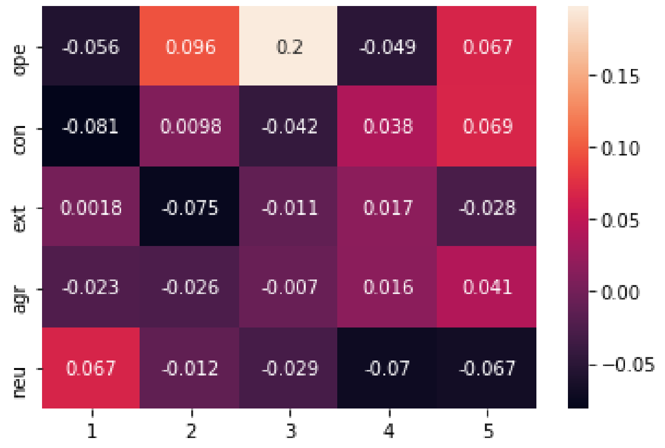


Figure 4 The heat map presenting correlations between k=5 varimaxrotated singular value decomposition dimensions and users' psychodemographic traits

to display the roc curve and roc auc score for gender. Metric's package was used to find the error rates for openness. Last, we investigated the correlation between cross-validated prediction accuracy and the number k of SVD dimensions. For this we stored the SVD dimensions for different k values. Also accuracies were estimated for different folds to predict openness and political.

7.2 Results

For logistic regression with gender as target variable we get accuracy of 0.854 and the roc curve (see Figure 5).

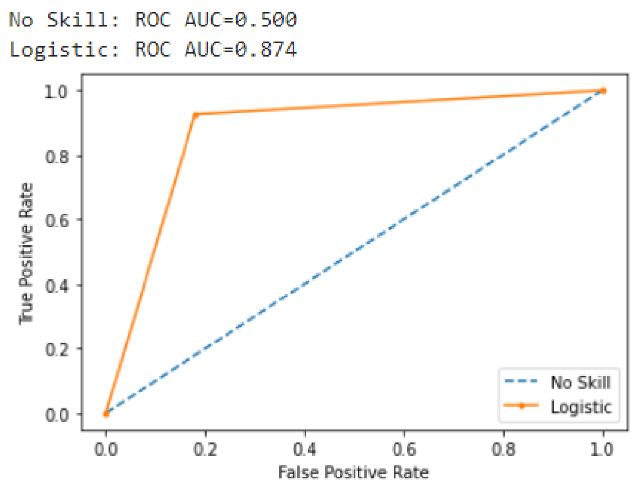
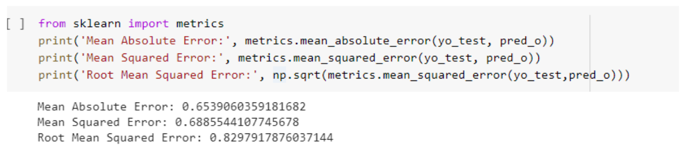


Figure 5 The roc curve for logistic regression with target variable gender

For linear regression with openness as target variable we get mse metrics (see Figure 6). As values of openness range from -5.22 to 1.9 the error is relatively high but not extremely high.



```
[ ]  from sklearn import metrics
     print('Mean Absolute Error:', metrics.mean_absolute_error(yo_test, pred_o))
     print('Mean Squared Error:', metrics.mean_squared_error(yo_test, pred_o))
     print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(yo_test,pred_o)))

     Mean Absolute Error: 0.6539060359181682
     Mean Squared Error: 0.6885544107745678
     Root Mean Squared Error: 0.8297917876037144
```

Figure 6 The mse metrics for linear regression with target variable openness

By investigating the correlation values of real openness values and predicted values for different column number of rotated matrix v and plotted this dependency (see Figure 7) With growing value of k the correlation value increases and 0.48 by k

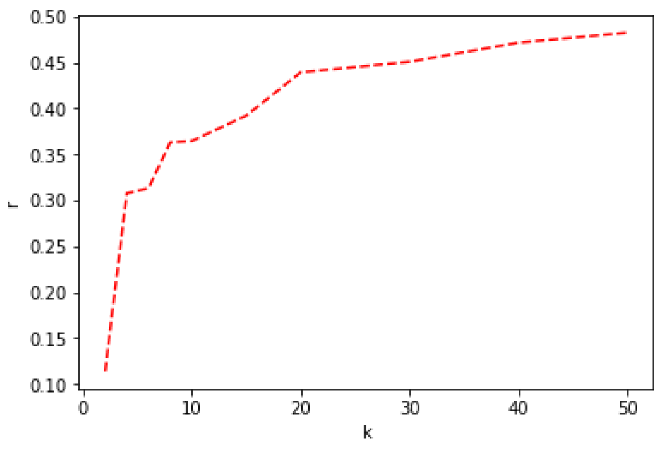
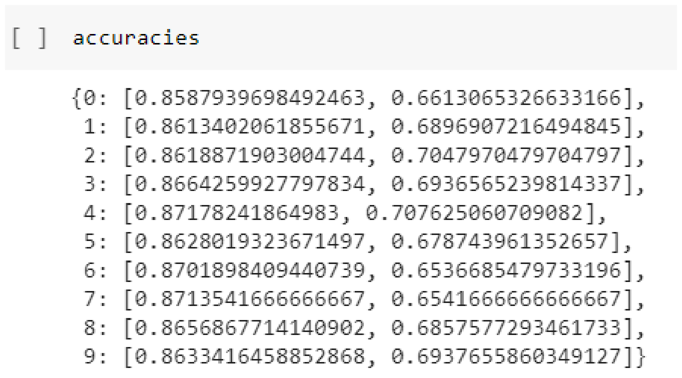


Figure 7 The correlation r in dependence from k values by linear regression with target variable openness

= 50. For different folds we could estimate accuracies for logistic regression with target variables gender and political. For all folds accuracy values of the model for gender is nearly 0.86 and for political is nearly 0.68 (see Figure 8)



```
[ ]  accuracies

     {0: [0.8587939698492463, 0.6613065326633166],
      1: [0.8613402061855671, 0.6896907216494845],
      2: [0.8618871903004744, 0.7047970479704797],
      3: [0.8664259927797834, 0.6936565239814337],
      4: [0.87178241864983, 0.707625060709082],
      5: [0.8628019323671497, 0.678743961352657],
      6: [0.8701898409440739, 0.6536685479733196],
      7: [0.8713541666666667, 0.6541666666666667],
      8: [0.8656867714140902, 0.6857577293461733],
      9: [0.8633416458852868, 0.6937655860349127]}
```

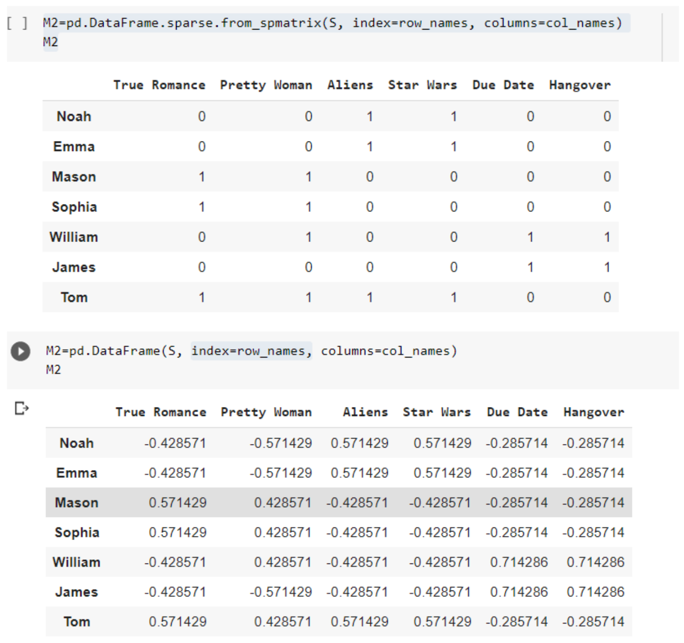
Figure 8 The accuracies for logistic regression with target variables gender and political for 10 folds

7.3 Discussion

For different target variables it was possible to use SVD approach and predict demographic and also psychological features using components of SVD.

8 Comparison of SVD by centered and uncentered sparse matrix

We investigated the differences of SVD components matrices by using centered and not centered sparse matrices using a reduced sparse matrix. The matrice are shown



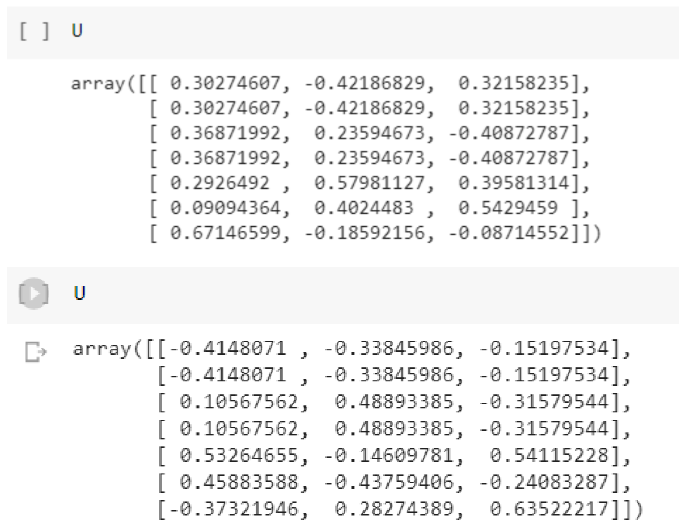
```
[ ]  M2=pd.DataFrame.sparse.from_spmatrix(S, index=row_names, columns=col_names)
     M2
```

	True Romance	Pretty Woman	Aliens	Star Wars	Due Date	Hangover
Noah	0	0	1	1	0	0
Emma	0	0	1	1	0	0
Mason	1	1	0	0	0	0
Sophia	1	1	0	0	0	0
William	0	1	0	0	1	1
James	0	0	0	0	1	1
Tom	1	1	1	1	0	0

```
M2=pd.DataFrame(S, index=row_names, columns=col_names)
M2
```

	True Romance	Pretty Woman	Aliens	Star Wars	Due Date	Hangover
Noah	-0.428571	-0.571429	0.571429	0.571429	-0.285714	-0.285714
Emma	-0.428571	-0.571429	0.571429	0.571429	-0.285714	-0.285714
Mason	0.571429	0.428571	-0.428571	-0.428571	-0.285714	-0.285714
Sophia	0.571429	0.428571	-0.428571	-0.428571	-0.285714	-0.285714
William	-0.428571	0.428571	-0.428571	-0.428571	0.714286	0.714286
James	-0.428571	-0.571429	-0.428571	-0.428571	0.714286	0.714286
Tom	0.571429	0.428571	0.571429	0.571429	-0.285714	-0.285714

Figure 9 Not centered and centered sparse matrices



```
[ ]  U

array([[ 0.30274607, -0.42186829,  0.32158235],
       [ 0.30274607, -0.42186829,  0.32158235],
       [ 0.36871992,  0.23594673, -0.40872787],
       [ 0.36871992,  0.23594673, -0.40872787],
       [ 0.2926492 ,  0.57981127,  0.39581314],
       [ 0.09094364,  0.4024483 ,  0.5429459 ],
       [ 0.67146599, -0.18592156, -0.08714552]])
```

```
U

array([[-0.4148071 , -0.33845986, -0.15197534],
       [-0.4148071 , -0.33845986, -0.15197534],
       [ 0.10567562,  0.48893385, -0.31579544],
       [ 0.10567562,  0.48893385, -0.31579544],
       [ 0.53264655, -0.14609781,  0.54115228],
       [ 0.45883588, -0.43759406, -0.24083287],
       [-0.37321946,  0.28274389,  0.63522217]])
```

Figure 10 Not centered and centered U matrices

in Figure 9 and also their U components in Figure 10.

9 Appendix

Name	Work Description
Rohan	Results, Discussions, Code
Natalja	Scientific Background, Goal, Data, Results, Discussion, Code

Table 1 Task responsibilities

References

1. Kosinski, M., Wang, Y., Lakkaraju, H., Leskovec, J.: Mining big data to extract patterns and predict real-life outcomes. Psychological methods **21 4**, 493–506 (2016)