RESEARCH

Introduction of Profile Areas of Data Science :Project 10

Suresh Kumar Choudhary, Frenny Macwan and Aman Jain

Full list of author information is available at the end of the article

Abstract

Goal of the Project: To employ cluster analysis and dimensionality reduction (using SVD) to extract patterns from large data of Facebook users and their likes and use the extracted patterns to build models aimed at predicting psychological outcomes.

Main Result of the Project: Analyzed large Facebook dataset using SVD dimension reduction technique and predicted real-life outcomes using Logistic Regression.

Personal Key Leanings: Big Data, SVD, Behavioural Traits

Estimate working hours: 24

Project Evaluation: 2

Number of Words: 1243

Keywords: SVD; Big Data; Personality Traits

Scientific Background

Recently due to profile proliferation digitally, large samples of data are being collected containing traces of human behaviours, communication, and social interactions. These data aids us to understand the human nature very well, but there comes a challenge during the analysis of this ample amount of data. Through singular value decomposition (SVD) we extract patterns and reduce the dimensionality of large data sets and efficiently store them eventually using the data to build predictive models. The SVD is also extremely useful in all areas of science, engineering, and statistics, such as signal processing, least squares fitting of data, and process control.

Goal

To employ cluster analysis and dimensionality reduction (using SVD) to extract patterns from large data of Facebook users and their likes and use the extracted patterns to build models aimed at predicting psychological outcomes.

Data

The sample data set [1] we used for analysis contains psychodemographic profiles of number 110,728 Facebook users and their Facebook Likes. For simplicity and manageability, the sample is limited to U.S. users.

1 **users.csv :** This file contains psychodemographic user profiles. It has 110,728 rows (excluding the row holding column names) and 9 columns: anonymous

user ID, gender ("0" for male and "1" for female), age, political views ("0" for Democrat and "1" for Republican), and five scores on a 100-item-long International Personality Item Pool questionnaire measuring the five-factor (i.e., Openness, Conscientiousness, Extroversion, Agreeableness, and Neuroticism) model of personality (Goldberg et al., 2006).

2 **likes.csv :** This file contains anonymous IDs and names of 1,580,284 Facebook Likes. It has two columns: ID and name.

3 **users-likes.csv :** This file contains the associations between users and their Likes, stored as user–Like pairs. It has 10,612,326 rows and 2 columns: user ID and Like ID. An existence of a user–Like pair implies that a given user had the corresponding Like on their profile.

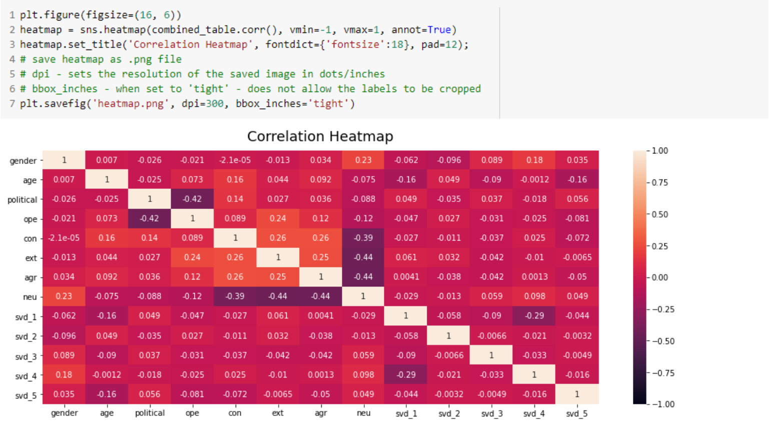
Interpreting Clusters and Dimensions

- **Background and Goal**: One of the main considerations regarding data dimensionality reduction is selecting the right number (in our case 5) of dimensions or clusters to extract . If the goal is to get an insight of the data, a small number of clusters and dimensions would be easier to showcase, whereas for training the models a larger number of dimensions or clusters will retain more information from the original matrix, thus enabling more accurate predictions.
- **Method**: We have used the cor-relation scores (Heat Maps) to study about the dimensions and the features. (As shown in Figure : 1)
- **Result**: After executing the TruncatedSVD module the 7062 features were combined into 5 dimensions (svd_1, svd_2, svd_3, svd_4, svd_5).As per the co-relation heat map, we can clearly conclude that the Politically inclined people are the least open and are least willing to listen to others. Dimension svd_1 are positively co-related to the agreeableness and extra-version personality traits. Also after combining we saw 20% information gain, which can been seen in Figure : 2
- **Discussion**: More number of dimension would have been difficult to manage but would have lost less Information which might lead to better accuracy and prediction of models. Since we have taken 5 dimensions, we have lost 80% of the information.

Reducing the Dimensionality of the User–Like Matrix Using SVD

- **Background and Goal**: Singular Value Decomposition (SVD) is a popular dimensionality reduction technique widely employed in various contexts, spanning computational social sciences, machine learning, signal processing, natural language processing, and computer vision. SVD represents a given matrix (of size m rows n columns) as a product of three matrices: a Matrix U (of size m x k) containing left singular vectors; a non-negative square diagonal Matrix $\sum$ (of size k) containing singular values; and a Matrix V (of size n x k) containing right singular vectors, where k is the number of dimensions that the researcher chose to extract. The goal is to achieve dimensionality reduction using SVD.
- **Method**: To achieve this, we used SVD to extract patterns from the user–Like Matrix M constructed.

Figure 1 Co-relation Matrix

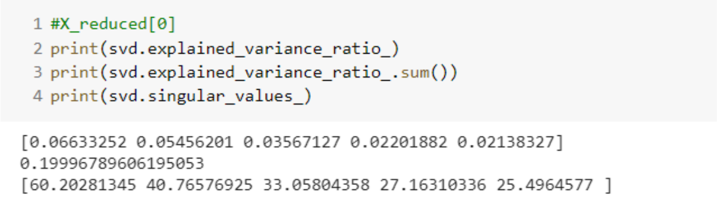


```
plt.figure(figsize=(16, 6))
heatmap = sns.heatmap(combined_table.corr(), vmin=-1, vmax=1, annot=True)
heatmap.set_title('Correlation Heatmap', fontdict={'fontsize':18}, pad=12);
# save heatmap as .png file
# dpi - sets the resolution of the saved image in dots/inches
# bbox_inches - when set to 'tight' - does not allow the labels to be cropped
plt.savefig('heatmap.png', dpi=300, bbox_inches='tight')
```

Correlation Heatmap

	gender	age	political	ope	con	ext	agr	neu	svd_1	svd_2	svd_3	svd_4	svd_5
gender	1	0.007	-0.026	-0.021	-2.1e-05	-0.013	0.034	0.23	-0.062	-0.096	0.089	0.18	0.035
age	0.007	1	-0.025	0.073	0.16	0.044	0.092	-0.075	-0.16	0.049	-0.09	-0.0012	-0.16
political	-0.026	-0.025	1	-0.42	0.14	0.027	0.036	-0.088	0.049	-0.035	0.037	-0.018	0.056
ope	-0.021	0.073	-0.42	1	0.089	0.24	0.12	-0.12	-0.047	0.027	-0.031	-0.025	-0.081
con	-2.1e-05	0.16	0.14	0.089	1	0.26	0.26	-0.39	-0.027	-0.011	-0.037	0.025	-0.072
ext	-0.013	0.044	0.027	0.24	0.26	1	0.25	-0.44	0.061	0.032	-0.042	-0.01	-0.0065
agr	0.034	0.092	0.036	0.12	0.26	0.25	1	-0.44	0.0041	-0.038	-0.042	0.0013	-0.05
neu	0.23	-0.075	-0.088	-0.12	-0.39	-0.44	-0.44	1	-0.029	-0.013	0.059	0.098	0.049
svd_1	-0.062	-0.16	0.049	-0.047	-0.027	0.061	0.0041	-0.029	1	-0.058	-0.09	-0.29	-0.044
svd_2	-0.096	0.049	-0.035	0.027	-0.011	0.032	-0.038	-0.013	-0.058	1	-0.0066	-0.021	-0.0032
svd_3	0.089	-0.09	0.037	-0.031	-0.037	-0.042	-0.042	0.059	-0.09	-0.0066	1	-0.033	-0.0049
svd_4	0.18	-0.0012	-0.018	-0.025	0.025	-0.01	0.0013	0.098	-0.29	-0.021	-0.033	1	-0.016
svd_5	0.035	-0.16	0.056	-0.081	-0.072	-0.0065	-0.05	0.049	-0.044	-0.0032	-0.0049	-0.016	1

Figure 2 Information Gain and Loss

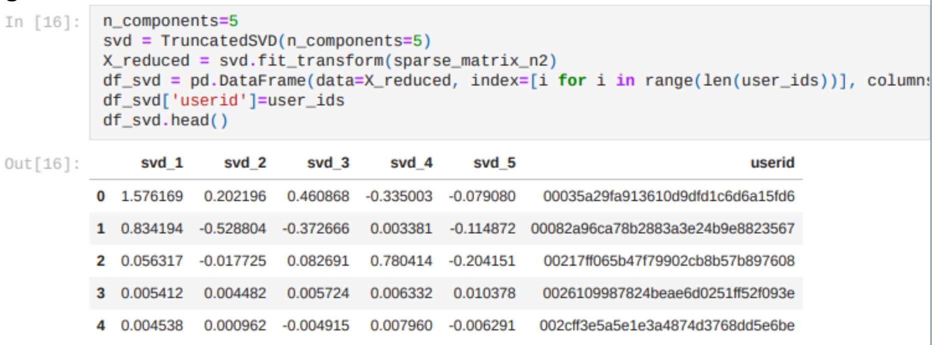


```
#X_reduced[0]
print(svd.explained_variance_ratio_)
print(svd.explained_variance_ratio_.sum())
print(svd.singular_values_)
```

```
[0.06633252 0.05456201 0.03567127 0.02201882 0.02138327]
0.19996789606195053
[60.20281345 40.76576925 33.05804358 27.16310336 25.4964577 ]
```

- **Result**: We used TruncatedSVD of sklearn package to compute SVD and can be shown in Fig 3. For the sake of simplicity, we select a relatively small value of n_components as 5.

Figure 3 Users' score on k=5



```
In [16]:  n_components=5
          svd = TruncatedSVD(n_components=5)
          X_reduced = svd.fit_transform(sparse_matrix_n2)
          df_svd = pd.DataFrame(data=X_reduced, index=[i for i in range(len(user_ids))], columns
          df_svd['userid']=user_ids
          df_svd.head()
```

Out[16]:

	svd_1	svd_2	svd_3	svd_4	svd_5	userid
0	1.576169	0.202196	0.460868	-0.335003	-0.079080	00035a29fa913610d9dfd1c6d6a15fd6
1	0.834194	-0.528804	-0.372666	0.003381	-0.114872	00082a96ca78b2883a3e24b9e8823567
2	0.056317	-0.017725	0.082691	0.780414	-0.204151	00217ff065b47f79902cb8b57b897608
3	0.005412	0.004482	0.005724	0.006332	0.010378	0026109987824beae6d0251ff52f093e
4	0.004538	0.000962	-0.004915	0.007960	-0.006291	002cff3e5a5e1e3a4874d3768dd5e6be

Predicting Real-Life Outcomes With Facebook Likes

To accomplish this hands-on, we followed the research paper titled as "Mining Big Data to Extract Patterns and Predict Real-Life Outcomes" [2].

- **Goal:** In this hands-on section, we need to build and demonstrate prediction models for Real-Life Outcomes based on the SVD dimensions extracted from the sparse matrix (user–Like Matrix).
- **Method:** The following steps followd to complete this task:

  1 Generated the sparse matrix with the help of pandas groupby functionalities which as follows:

  $$sparse\_matrix = df.groupby([′userid′,′ likeid′]).size().unstack(fill\_value = 0)$$

  2 Trimmed the dataset(matrix): Matrix rows with the users trimmed as who had less than 2 Facebook likes and columns with likeid trimmed as which was liked by less than two users which as follows:

  $$sparse\_matrix\_n1 = sparse_{m}atrix_{n}[:, np.sum(sparse\_matrix, axis = 0) > 1]$$

  $$sparse\_matrix\_n2 = sparse_{m}atrix_{n}1[np.sum(sparse\_matrix, axis = 1) > 1]$$

  3 Extracted the dimention using SVD with the help of python scikit-learn library

  $$svd = TruncatedSVD(n_{c}omponents = 5)$$

  $$X\_reduced = svd.fit_{t}ransform(sparse\_matrix\_n2)$$

  4 Applied the Linear regression and logistic regression method of scikit learn library on reduced dimension dataset. We splitted the dataset as train set(66%) and test set (33%) to train these models.
- **Results:** Due to memory constraint, we used subset of data (50957 user-like pairs) as sparse matrix consumes too much memory(RAM). Using this dataset we got a sparse matrix of dimension 25257 X 7062. After trimming we had our matrix with dimension 9992 X 2342. We applied SVD on this matrix to get reduced dimension. So for this purpose we used the number of components from 1 to 50. With the help of linear regression(for continous response variable) and logistic regression(for categorical respinse varaible) we calculated the accuracy. For Gender and Political variable we calculated the Area under curve(AUC) with the help of logistic regression model. For remaining variables, we calculated the Pearson's correlation coefficient with the help of actual values and predicted values. The accuracies can be seen in table 1 and code screenshot for the same in Figure 5 for 50 components of SVD.

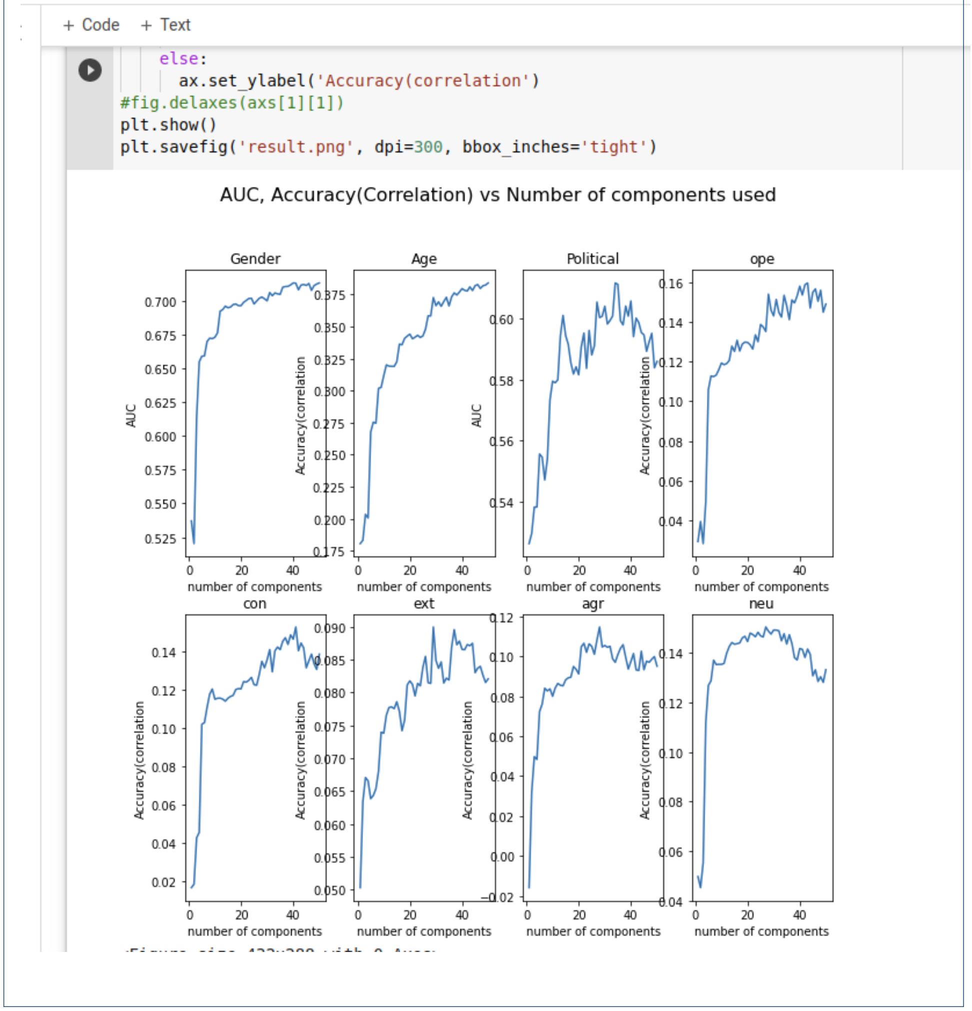
  We got AUC 0.71 and 0.59 for Gender variable and Political variable respectively and highest accuracy(correlation) for Age variable as 0.38 if we use the 50 dimensions for likes.
- **Discussion:** If we increase the number of components, the accuracy also increases till some dimension(components) and later it decreaes, which can be seen in figure 4. From figure 4, we can conclude, For variable Gender and

Age, if we increase then it might possible we may get better results. And for remaining variables if we increase the number of components(dimension) then accuracy decreases so there is no need to further increase the number of dimension, only we need to find the point where accuracies are maximum. As we used subset of dataset(only around 5% data) so actual result is not reflected which is given in paper.

Table 1 Approximate Prediction Accuracy Based on k = 50 Singular Value Decomposition (SVD) Dimensions

Real-Life Outcomes(Response Variable)	Accuracy	Algorithm Used
Gender(AUC)	0.7131	SVD+Logistic regression
Political views(AUC)	0.5859	SVD+Logistic regression
Age	0.3841	SVD+Linear regression
Openness	0.1489	SVD+Linear regression
Conscientiousness	0.1384	SVD+Linear regression
Extroversion	0.0821	SVD+Linear regression
Agreeableness	0.0951	SVD+Linear regression
Neuroticism	0.1333	SVD+Linear regression

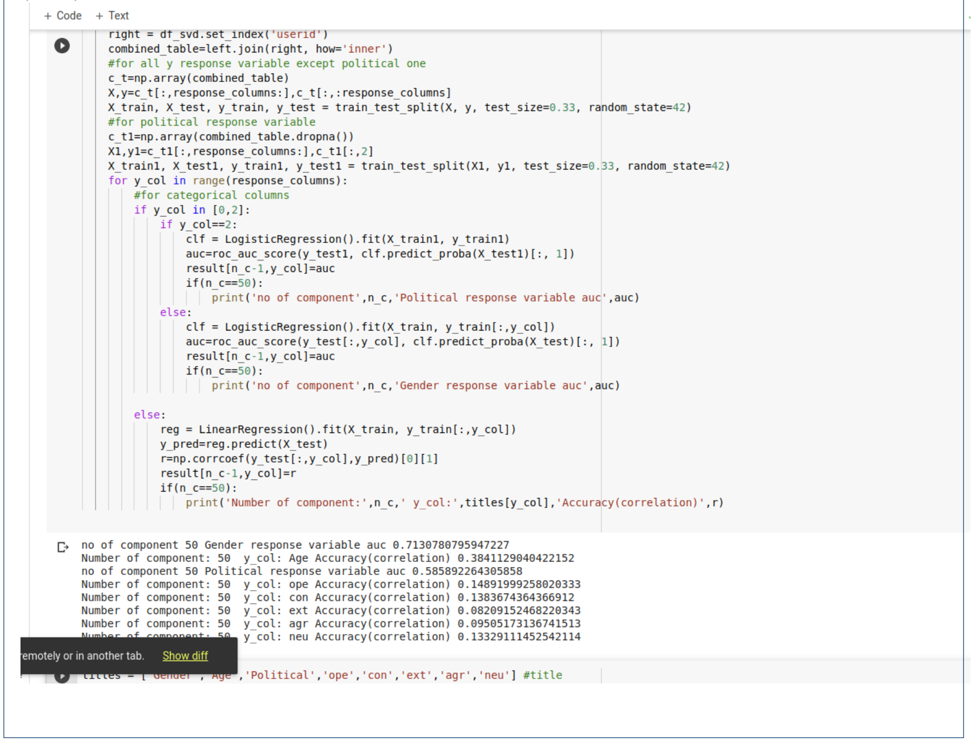
Figure 4 Relationship between the accuracy and the number of the singular value decomposition dimen- sions used.



```
    else:
        ax.set_ylabel('Accuracy(correlation')
#fig.delaxes(axs[1][1])
plt.show()
plt.savefig('result.png', dpi=300, bbox_inches='tight')
```

```
<Figure size 432x288 with 0 Axes>
```

Figure 5 Approximate Prediction Accuracy Based on k = 50 Singular Value Decomposition (SVD) Dimensions



```
right = df_svd.set_index('userid')
combined_table=left.join(right, how='inner')
#for all y response variable except political one
c_t=np.array(combined_table)
X,y=c_t[:,response_columns:],c_t[:,:response_columns]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
#for political response variable
c_t1=np.array(combined_table.dropna())
X1,y1=c_t1[:,response_columns:],c_t1[:,2]
X_train1, X_test1, y_train1, y_test1 = train_test_split(X1, y1, test_size=0.33, random_state=42)
for y_col in range(response_columns):
    #for categorical columns
    if y_col in [0,2]:
        if y_col==2:
            clf = LogisticRegression().fit(X_train1, y_train1)
            auc=roc_auc_score(y_test1, clf.predict_proba(X_test1)[:, 1])
            result[n_c-1,y_col]=auc
            if(n_c==50):
                print('no of component',n_c,'Political response variable auc',auc)
        else:
            clf = LogisticRegression().fit(X_train, y_train[:,y_col])
            auc=roc_auc_score(y_test[:,y_col], clf.predict_proba(X_test)[:, 1])
            result[n_c-1,y_col]=auc
            if(n_c==50):
                print('no of component',n_c,'Gender response variable auc',auc)

    else:
        reg = LinearRegression().fit(X_train, y_train[:,y_col])
        y_pred=reg.predict(X_test)
        r=np.corrcoef(y_test[:,y_col],y_pred)[0][1]
        result[n_c-1,y_col]=r
        if(n_c==50):
            print('Number of component:',n_c,' y_col:',titles[y_col],'Accuracy(correlation)',r)
```

```
no of component 50 Gender response variable auc 0.7130780795947227
Number of component: 50  y_col: Age Accuracy(correlation) 0.3841129040422152
no of component 50 Political response variable auc 0.585892264305858
Number of component: 50  y_col: ope Accuracy(correlation) 0.14891999258020333
Number of component: 50  y_col: con Accuracy(correlation) 0.1383674364366912
Number of component: 50  y_col: ext Accuracy(correlation) 0.08209152468220343
Number of component: 50  y_col: agr Accuracy(correlation) 0.09505173136741513
Number of component: 50  y_col: neu Accuracy(correlation) 0.13329111452542114
```

```
titles = ['Gender','Age','Political','ope','con','ext','agr','neu'] #title
```

Appendix

Code -

- Aman - Correlation
- Suresh - Sparse Matrix Creation, SVD, Linear regression and logistic regression
- Frenny - Result plot

Report -

- Aman - Scientific background, Reducing the Dimensionality of the User–Like Matrix Using SVD
- Suresh - Predicting Real-Life Outcomes With Facebook Likes
- Frenny - Abstract, Goal, dataset and Interpreting Clusters and Dimensions

Author details

References

1. Dataset. https://drive.google.com/file/d/15v2umVbv1OarPfNrePH9V-KPk7qqA6Cg
2. Michal, H.J. Yilun: Mining big data to extract patterns and predict real-life outcomes. Psychological methods **21**(4), 493 (2016)