Understanding Yelp User's Influence

Individual Report 2

Contents

1 Introduction

This report is an exploration of the meaningfulness of the **Elite Users** for the **Yelp** platform, by means of statistical tools.

1.1 Resources

The **Yelp Dataset** is a resource available online, released for the **Yelp Challenge**. The dataset, is divided into 5 files:

- **yelp_academic_dataset_business.json**
- **yelp_academic_dataset_user.json**
- **yelp_academic_dataset_review.json**
- yelp_academic_dataset_checkin.json
- yelp_academic_dataset_tip.json

For the purpose of this project, only the first three files are needed.

1.2 Goals

This project has been carried out for two main reasons:

- **Understanding user influence**
- **Modelling user influence**

1.3 Hypotheses

1. The data is **clean enough** to avoid performing data cleaning. Enough means that the final result won't be deviated so much, w.r.t. performing data cleaning, to invalidate the research.

2. The influence analysis between normal user and elite user is not going to be critically affected by the **overlapping of effects**. Since the influence is going to be computed as a *before-after effect* on the average star-rating of a business, having two elite users reviewing the same restaurants would result in an overlapping of influences, hardening the influence analysis. For elite users this is not going to be such an issue since, despite knowing that, usually, human activities follow Gaussian alike distributions, the number of elite users: 50773 is less than the number of businesses: 85901, thus such an event can be considered quite rare. This issue, however would definitely show up considering normal users, due to their number: 635783. However, *random sampling* as many normal users as many elite users (needed also for computational purposes) deals with such issue since, on average, together with the number of normal users, the rarity of the overlapping of events decreases by more than 10 times.

2 Methodology

2.1 Information Retrieving

For the creation of the sub-dataset that is used, only a few features are retrieved: from the **Business File**: business_id, categories; from the **User File**: user_id, votes, yelping_since, fans, friends, compliments, review_count, years_elite; from the **Review File**: date, business_id, stars, user_id.
Nonetheless, not only the pre-existing features are used: *2 more features* are computed in the pre-processing step:

- **Average Star Impact**: Given the point in time a review is released from a user, the difference between the average star rating (for that business), before and after that point in time, is computed. This is also referred to as the **before-after effect**. For a single user, the *average_star_impact* is the *mean* between all before-after effects regarding that user.
- **Average Influence**: Real number between $[-1, 1]$ which describes whether the a user has *negative* or *positive* influence. **Negative** influence shows up when, despite a user reviews *positively* a business, the average star rating of that business, after the user's review, *drops* below its previous average; the vice-versa is still negative influence. **Positive** influence, on the other hand, manifests when, considering one *positive* review in time, the average star-rating of the business *increases* w.r.t. its previous average star rating. The vice-versa is still Positive Influence. Given these concepts, the average_influence is computed with this formula:

  $$AverageInfluence(u) = \frac{1}{R(u)} \sum_{i=1}^{R(u)} RelativeInfluence(r_i)$$

  .

  $R(u) = Number\ of\ reviews\ of\ user\ u$

  $$RelativeInfluence(r_i) = \begin{cases} +1, & \text{if the i-th review shows } \textbf{Positive} \text{ influence} \\ -1, & \text{if the i-th review shows } \textbf{Negative} \text{ influence} \end{cases}$$

2.2 Feature and Metric Engineering

Modelling the user influence is the most critical component of the project. Of all the features, only a few are considered for the Influence Metric:

- **F1**: **interactions**: Sum of all the attributes of the **votes** data structure from the user json.
- **F2**: **fans**
- **F3**: **compliments**: Sum of all the attributes of the **compliments** data structure.
- **F4**: **review_count**
- **F5**: **number_of_years_elite**: Number of years a user has been elite.

Given these parameters, the influence metric is defined as a weighted average over the first five features, normalized following the *z-score*, everything multiplied by the Average Influence.

$$Influence(u) = \left(\sum_{i=1}^{5} F_i(u) \times w_i \right) \times AverageInfluence(u)$$

2.3 Parameter Modelling

Parameter modelling can be either **supervised** or **unsupervised**. Since it is a crucial component of the project, both approaches are used and their results are compared to spot a possible coherence or incoherence of the results (assuming both approaches are feasible).

2.3.1 Supervised

The issue that arises for supervised learning is the lack of true targets, necessary to create a supervised learner. The only actual **targets** y_i that can be used are given by the **average_star_impact** feature. The **data points** x_i, instead, are 5-dimensional: the 5 features F_1 to F_5 characterize each user. Since the data is rather complex, a linear approach cannot be used, moreover, the size of the dataset (only 50000 data points) denies the use of a Neural Network. Such a task, however is suitable for **Support Vector Machines**, thus 3 Support Vector Regressors are trained with different kernels $K(w, x_i, b)$.

- **Radial Basis Function**
- **Polynomial**
- **Logit**

SVRs are trained by optimizing the objective function:

$$\min \tfrac{1}{2}||w||^2$$
$$s.t. \begin{cases} y_i - K(w, x_i, b) \leq \epsilon \\ K(w, x_i, b) - y_i \leq \epsilon \end{cases}, \epsilon = PenalizationThreshold, b = Margin$$

2.3.2 Unsupervised

If the data is hard to learn and the supervised learning doesn't work, the weights cannot be inferred manually by a human since that would lead to overfitting. This is why, to model the weights, an unsupervised data-driven approach is necessary in such cases. **Principal Component Analysis** is the approach chosen for this task.

PCA performs an orthogonal transformation $\mathbf{R} = \mathbf{VDV^T}$ on the (Pearson) correlation matrix $\mathbf{R}$ to get, from a set of usually correlated features, a set of linearly independent eigenvectors $\mathbf{V}$, named principal components. Such components have different importance, given by the value of the associated eigenvalues $\mathbf{D}$, and are sorted following the direction of maximum variance they express.

PCA is performed on the feature (F_1 to F_5) matrix for both normal and elite users.
Computed the PCA, the elements of the first PC, are used as weights for the Influence Metric.

2.4 Influence Analysis

After modelling the influence metric, the chosen features are plugged in together with their weights, computed with the aforementioned approaches. The analysis is composed of two main steps:

1. Understand the Probability Density Functions of both elite and normal users influences, comparing it to a Gaussian, by means of a **Kolmogorov–Smirnov** test.
2. Compare the elite and normal user influence distribution, by means of the **Kullback–Leibler** divergence.

3 Results

3.1 Support Vector Regression

These table sum up the performances (on both Test and Train sets) of the 3 aforementioned SVR , plus the Mean: a simple horizontal line that has the value of the mean of the train set. This result is shown only for Elite users due to the meaninglessness of carrying on such approach.

Test				
	RBF	POLY	LOGIT	MEAN
MSE	0.1536	0.1533	109084.27	0.1533
R^2	-0.0025	-0.0005	-711574.05	-0.0003

Train				
	RBF	POLY	LOGIT	MEAN
MSE	0.1588	0.1602	103070.18	0.1600
R^2	0.0073	-0.0015	-644114.64	0.0

Table 1: Performance of the algorithms on Test and Train sets for Elite users

3.2 Principal Component Analysis

The first principal component for both elite and normal users, has all *negative* values. Since the direction of the eigenvector is not important, a simple inversion of sign is applied. The results are shown in the following table.

PCA Weights					
	Interactions	Fans	Compliments	Review Count	Years Elite
Elite	0.4966	0.4797	0.5291	0.4116	0.2717
Normal	0.4021	0.5188	0.5380	0.5287	nan

Table 2: Principal Component Analysis Weights

3.3 Influence Behavior

Figure 1 is a four panel image that shows the behavior of the influence metric with PCA weights and the contribution of each feature to the influence metric: having $weight = 1$ for that feature and zero for all the others.

3.4 Probability Density Function

Table 3 shows the main properties of the PDF for elite and normal users' influence.

First Four Moments				
	Mean	Standard Deviation	Skewness	Kurtosis
Elite	0.0315	0.7468	54.7941	5766.17
Normal	0.0118	0.5803	4.8306	1843.35

Table 3: First four moments of the Influence PDF and KS Test w.r.t. a Standard Gaussian

The **Kolmogorov–Smirnov** test w.r.t. a **normal null-hypothesis**:

- **Elite User Influence Distribution**: $p\text{-}value = 0.0$
- **Normal User Influence Distribution**: $p\text{-}value = 0.0$

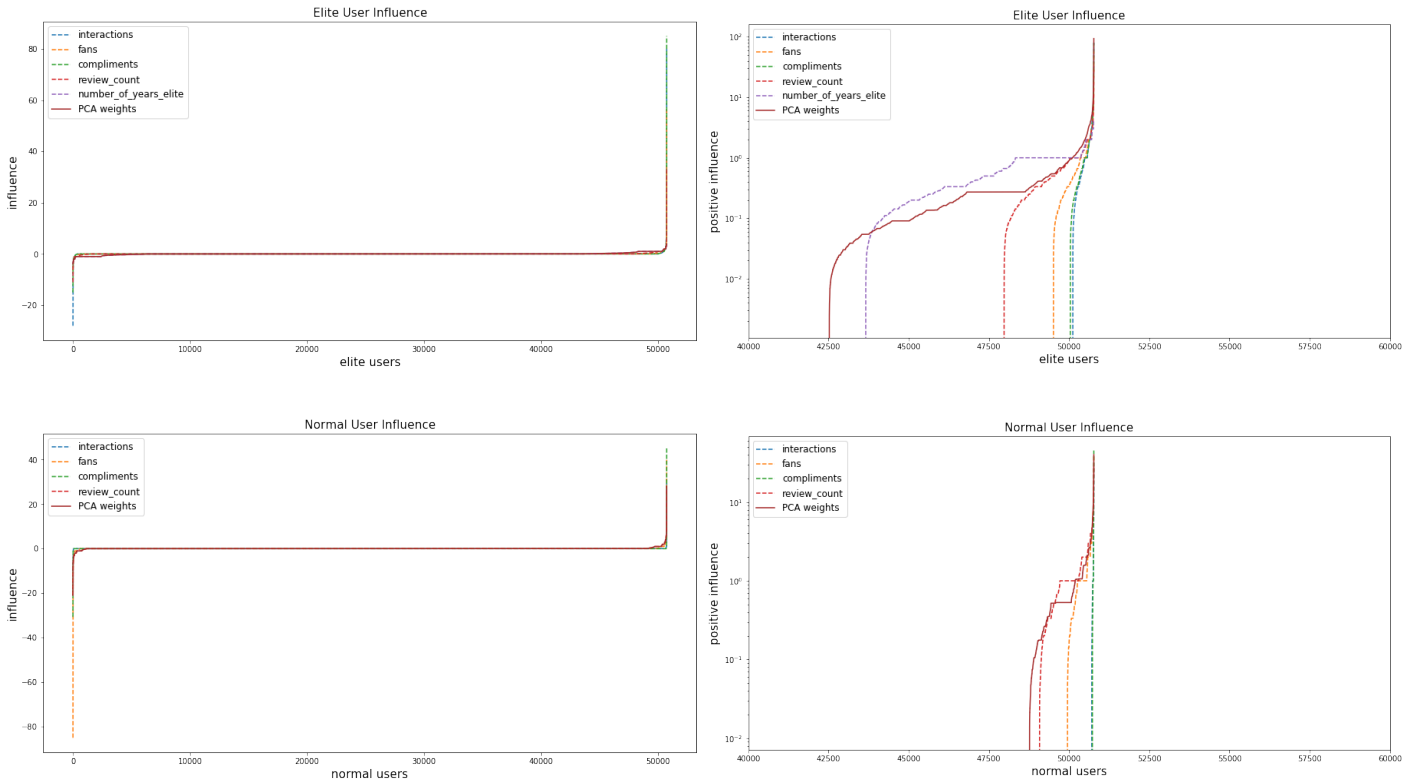

Figure 1: [top left] Elite user influence, linear scale. [tip right] Elite user positive influence, semi-logarithmic scale. [bottom left] Normal user influence, linear scale. [bottom right] Normal user positive influence, semi-logarithmic scale.

3.5 Kullback–Leibler Divergence

Figure 2 is the result of the Kullback-Liebler divergence, computed between the elite user PDF and normal user PDF in both senses.

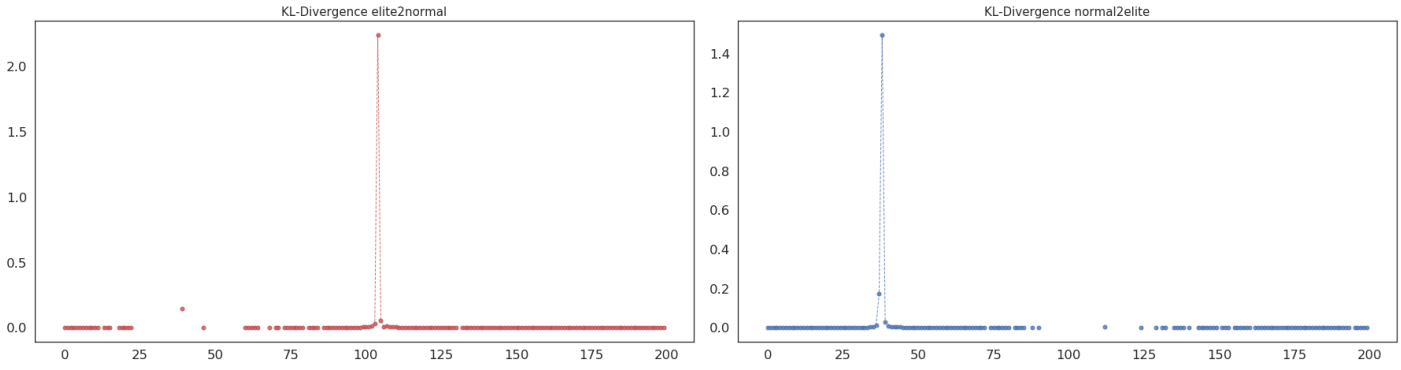

Figure 2: KL Divergence, both senses

4 Comments

4.1 Support Vector Regression

The SVRs trained on 80% of the data, clearly show that is not possible to learn from this data. On the Train, for both RBF and Polynomial kernels, the Mean Squared Error is almost the same as a simple Mean model, while a Logit kernel is not suitable at all for this task. On the Test (20%) the situation is basically the same. This is confirmed by the coefficient of determination R^2 which is negative, showing how the models perform worse than a simple horizontal line!

Given such poor results, there is probably no supervised algorithm that can achieve reasonably good results. This is due to the little percentage of users that actually have non-null influence: the 19.02% of the elite users have some influence, instead, only 6.73% of the normal users, seem to have some influence, and this could be due to the *overlapping of effects* discussed in hypothesis 2.

One possible solution would be to retrieve somehow *more data*, that is meaningful, thus, showing non-null influence.

4.2 Principal Component Analysis

At this point, an unsupervised approach is the only one feasible. As for all unsupervised approaches, however, it is hard to precisely infer something from the result. This said, knowing how PCA works, the weights are expressed by the direction of maximum variance, thus, if actually used as weights, the influence metric should show more variation than just uniform weights.

4.3 Influence Behavior

For elite user influence, the dashed lines represent the impact that one single feature has on the influence (e.g. giving weight *1* to that feature and *0* to all the others); the solid line, instead, shows the behavior of the PCA weighting. As we can see, the plot shows how the number_of_years_elite is the most meaningful feature, since it shows *less null-influence* than the others. Moreover, it shows how the PCA weighting is actually meaningful, since, such weighted influence shows *even less null-influence* than the number_of_years_elite feature!

This impact is much weaker for the normal user influence, that, in general, has *more null-influential* users and seem to have also more negative than positive influence (for the elite users is the opposite).

4.4 Probability Density Functions

Still, the high non-informativeness of the data is a fact. The influence mean is almost zero and the standard deviation is also low, considering that the maximum absolute value is 93.47 for elite users and 44.10 for normal users, using PCA weighting.

Seen this behavior, hypothesizing that the influence PDF behaves as a Gaussian is reasonable, but the **KS Test** actually shows the opposite, rejecting the null-hypothesis with $p\text{-}value = 0.0$ for both elite and normal user distributions.

Even though, between elite and normal users the difference is slightly visible on the plots in Figure 1, it is hard to infer anything from the the **KL-Divergence** in Figure 2: the distributions are equal for most

of the body. Having most of the support zero-valued, means zero information loss, except for one single spike that resembles an outlier. That spike actually underlines the difference between the percentage of users having *non-null influence* between elite and normal users: *non-null influence* elite users are more than *non-null influence* normal users.

5 Conclusions

This study shows how only a small amount of elite users have actual influence, thus modelling a proper metric and setting correctly the hyperparameters is *extremely hard*. Given more data, performing a crude **filtering**/data-cleaning using an **influence-threshold** would lead to better and more meaningful results.

Nonetheless, since, the difference between elite users and normal users is still visible, saying that the elite status is not meaningful for the Yelp Platform would be a *big mistake*.

6 References and Word Count

6.1 References

- **matplotlib**: Python library used to *plot* distributions.
- **pandas**: Python library used to *retrieve* information.
- **numpy**, **scipy**: Python libraries used to perform *computations*.
- **sklearn**: Python libraries used to perform *training*.

6.2 Word Count

1906 words