ORIE4741 Learning with Big Messy Data
Fall 2020

Midterm Report
Mental Health Study in Tech Industry

by
dh734 - Donghao Huong
lc977-Lihe Cao
ss2742 - Sukriti Sudhakar

1. Data Description

The dataset we use is separated into Answer, Question and Survey sheets. The Survey shows that this organization used five questionnaires from 2014 to 2019. The Questions collected all the questions asked in these five questionnaires and the Answer sheets record all the answers of these five questionnaires. We find that there are 4218 observations and 108 features in this dataset. Comparing the questions of these five questionnaires, we find that the questions differ in years but are the same from year 2017 to 2019. In the project, we only focus on the observation from these three questionnaires. Therefore, we finally used data with 1524 observations and 78 features. Among these 78 features, 57 of them are numerical data, 6 of them are text data, and the rest of the data are boolean data and nominal data. In this report, we do not include the feature text first. We plan to use the NLP tool to process these features in our following part of the project. In the next part, we will use encoding method to assign values to the nominal and ordinal data and fill up the missing values in them.

Data Visualization

Willingness to discuss mental health issues with employers

(All the figures are put in the next page)

We see in the graphs below(Fig1 and Fig2) that the percentage of people who have talked about their mental health issues with their coworkers and employers hasn't been very high in any of the years. In fact, this percentage has always been less than 1%. But, another important point to notice is that the percentage of people who have talked about their mental health issues with their employers has increased over the years. This shows some improvement in number of people coming out in open and talking more about their mental health and seeking help. In Fig3 we see that the percentage of people who have seeked help went up in year 2018 and went down again in year 2019, which follows an opposite trend as the percentage of people who discuss their mental health issues with coworkers, but this might be related to increased willingness to talk with employers.

In Figure 4 and 5 below, we see that the amount of importance given by employers to physical health of their employees is higher in comparison with the importance given to the mental health of an employer's employee. We see that the number of employers that pay 60% or more attention to physical health of employees (Fig4) is very as compared to the number of employers given that much attention for mental health is extremely low(acc to Fig5). A huge number of employers pay less than 30% attention to mental health of their employees. This must be another reason that employees might feel resistance to seek help.

Figure 6 is a simple box plot describe the age of the interviewees. We can see that most of the survey takers are between 20 and 55 years old.

We also made a correltaion matrix for all the numercial data and encoded data. Fig7 is the graph of the correltaion matrix. In the following part of the project, we will go futher into this graph and set up the significatn level and pick the important variable.

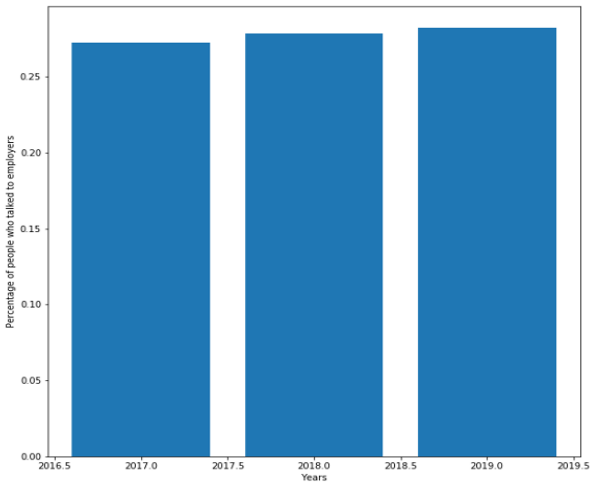


Figure 1

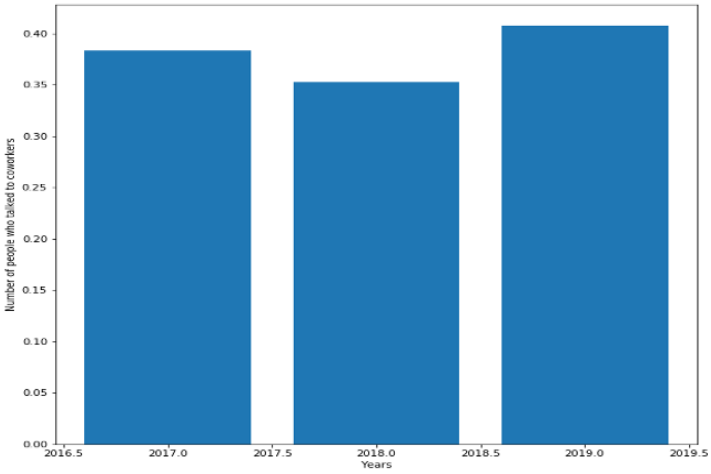


Figure 2

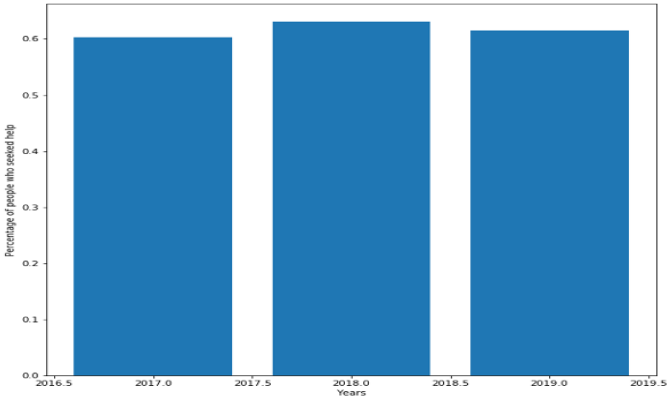
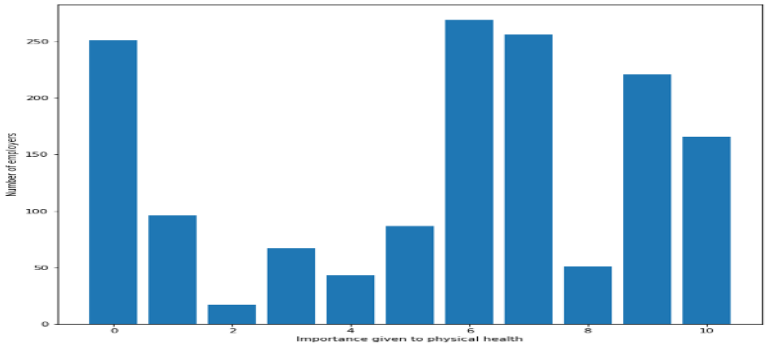


Figure 4



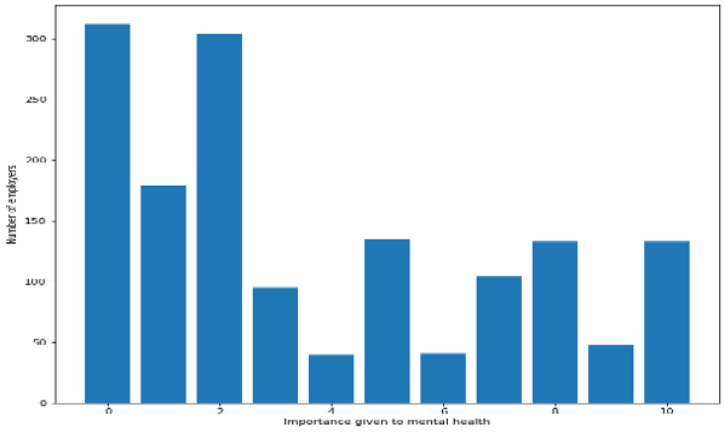


Figure 5

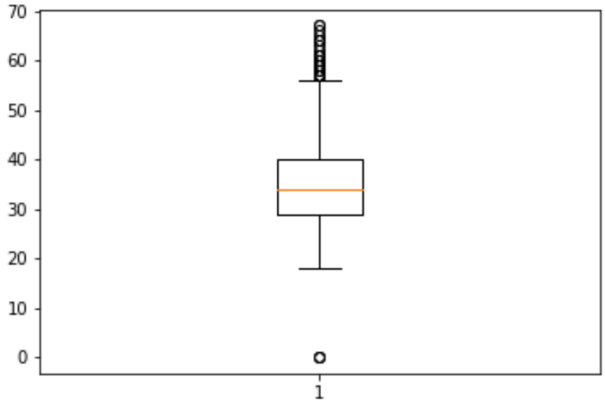


Figure 6

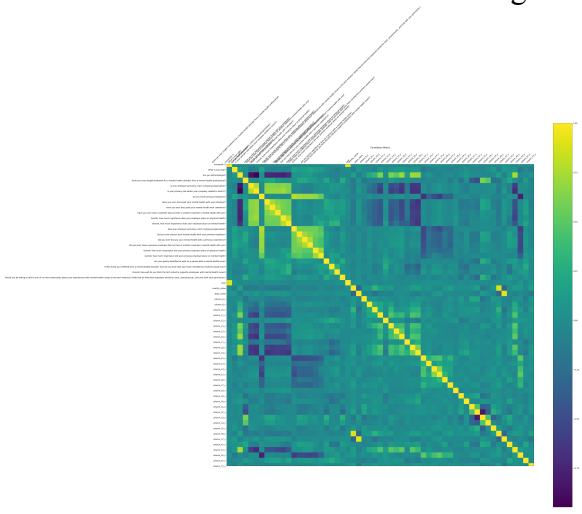


Figure 7

2. Data Preprocessing

(1) Nominal and Ordinal Data Encoding

For the boolean and the nominal data, we use the ordinal encoding for each response. For each column, we first count each label first, then sort them by the frequency descendingly. Then we assign the value to each label by using the order we got from the previous step. As a result, the data with a higher apparent frequency will have a higher value. There are two reasons we choose to use the ordinal encoding. The first one is that the ordinal encoding is the most straightforward. There are 32 features that need encoding. We can quickly assign value to each answer in the short time. The second one is that, unlike the one-hot encoding, the ordinal encoding will not increase the size of the whole dataset. Therefore, the data will not become super large and requires a large amount of computation power. In addition, large amount of features will also probably make the model become overfitting. However, the encoding method we use now may be too straight forward and makes our final model become too simple and consequently underfitting. We may consider other ways of encoding these features in the following part of the project.

(2) Fill up missing values

In our raw data, there are many mssing values. There are 19 questions which have more than 1100 missing values, where other questions have 223 missing answers at most. For now, we just throw them away these questions which have more than 1000 missing values and use the remaining features to fit our model. We will consider how to deal with these features later in the project. There are several ways to fill up the missing values of the remaining data.

(a) The simplest and widely used way is to take the mean of the remaining data assign it to the missing value. If the data of a feature is integer valued, we should round it first.

(b) For the bolean values "0/1", we can calculate the mean of the data, which is in [0,1], we then set the mean to be the probability for a missing data to have value 1 and use simulation to fill up the missing value. For example, if the mean is 0.6, we can generate a uniform random number in [0,1], if it falls in [0,0.4], we set the missing value to be 0, otherwise, we set the missing value to be 1.

(c) For each feature, we can take it out and let it be the output we want to predict and let other features to be the input examples. We can fit a model for each feature to predict the missing values.

In our project, we will fill up the missing values by calculating the mean of data of each feature before midterm and use this data to fit our final model to see the prediction result. If the result is reasonable, we will using other methods to fill up the missing values in the last part of our project to see whether we can refine our model.

(2) Separate Training and Test Sets

We get the training example and test example sets from the original dataset using "thumb rule". We first shuffle the data matrix and then take the first 80% of the data to be training examples and the rest 20% to be the test set.

3. Model Fitting

(1) Model and Parameter

Our project goal is to predict whether the interviewee has ever sought for a mental

treatment disorder from a mental health professional, which has bolean values 0 or 1. Therefore, we choose to use logistic regression and SVM as our models which are two most widely used models in solving classification problems. We then use "sklearn" packcage in Python to fit the model using our pre-processed data. Before the midterm, we have mainly used logistic regression to do the prediction. We have also tried to use the data to fit the SVM model. In the logistic regression model, we are going to solve the optimization problem

$$max\Sigma_{(x,y)\in S} log(1 + exp(-yw^T x))$$

We tried to add different regularization terms in the model. We fit the model by adding l_2 regularization term, l_1 regularization term and no regularization term. We also tried different solver such as 'liblinear', 'lbfgs' and 'saga' to see which slover gives better result. When running the algorithm, we set the maximum iteration to be 100000 times.
In the SVM model, we are going to solve the optimization problem below,

$$min\Sigma_{(x,y)\in S} l_h inge(x_i, y_i; w) + \lambda||w||^2$$

Before midterm, we didn't adjust the parameters.

(2) Prediction Result

(a) Logistic regression Here we will show some of the accuracy rate for the model using different regularization terms and solvers. We train the model using the training examples and get the accuracy rate using both the training and test set.
solver: 'lbfgs', regularization term: 'l_2 ', accuracy of training set: 84.49%, accuracy of test set: 80.66%
solver: 'liblinear', regularization term: 'l_1 ', accuracy of training set: 85.89%, accuracy of test set: 78.03%
solver: 'saga', regularization term: 'l_2 ', accuracy of training set: 75.88%, accuracy of test set: 74.09%
(b) SVM
We only fit the model using 'Radial basis function' as kernel function. The accuracy of training set: 99.02%, accuracy of test set: 79.67%. This is expectable since we didn't add penalty to the loss function, the error of the model applying on training set should be close to 0. Later in the project, we will try different kernels in the SVM model and adjust parameters such as the coefficient of the penalty and the kernel function and compare the results with the logistic regression model.

(3) Overfitting and Underfitting

In the previous part, we used our model to fit both the training set and the test set and calculate the accurate rate. We can observe that the accurate rates for predicting training examples are around 80%, which are quite high, so the model is not underfitting. We then compare the accurate rate of the test set with the training set, using different solvers and regularization terms respectively. From the results above, we can see they are very close. For example, when we use 'lbfgs' solver and 'l_2 ' regularization term, the accuracy of training set is 84.49% and accuracy of test set is 80.66%. The difference is: $\frac{84.49\% - 80.66\%}{84.49\%} = 4.53\%$. The difference is small enough for us to claim that the model is not overfitting.