AMATH 582: HOMEWORK 4

VINSENSIUS

Applied Mathematics Department, University of Washington, Seattle, WA
vxvinsen@uw.edu

Abstract. A data set of 1984 voting results of US Congress is used to train a semi-supervised learning (SSL) model and a spectral clustering using Laplacian embedding [6]. Both models are trying to classify the party affiliations of the Congress members based on their voting pattern for 19 bills. The labels used for party labelling are ± 1, where 1 for Democrats and -1 for Republicans. Fiddler vector is used to determined the accuracy of the clustering model, which aids in finding the optimal value of the variance of the Laplacian graph that has highest accuracy. Optimal value of variance, σ^* is found to be 2.14 with highest accuracy about 88%. Then SSL model is applied with varying number of eigenvectors (M), which determines the geometry of the clusters and varying the number of labelled sets (J). With different M and J, the accuracies are found and then examined. The highest accuracy value is found to be for combination of M = 4 and J = 10. However, further study into the effect of varying M and J to the SSL model accuracy needs to be done since there is no clear pattern. Furthermore, the SSL model accuracy is sensitive to value of σ, which affects the clustering part of the SSL model.

1. Introduction and Overview

Semi-supervised learning (SSL) is a type of machine learning that combines the label identification from supervised learning and the structure identification from unsupervised learning. The idea is that the model will be given labelled data-sets and unlabelled data-sets, which the number of unlabelled data-sets are much larger than the labelled ones. The model is then try to label the unlabelled data-sets based on labelled data-sets. This method of semi-supervised learning is called label propagation. The main idea is that the model will predict the clustering first, then using the information about the clusters to predict the labels based on the labelled data sets. SSL is used widely because only a fraction of the datasets are labelled, which saves a lot of time and money. This is because labelling datasets can be time consuming and expensive especially if the data requires human expertise such as audio of different languages, where people are required to transcribe or translate it for model to learn.

The data given is a voting results for 19 bills from Congress [6]. The first task is to find optimal variance parameter σ^*, which gives the highest accuracy for classifying the party affiliations of the Congress member. The next task is to apply SSL model based on the number of eigenvectors(M) and number of labelled data set (J) for classification the party affiliations of the congress member. We will apply the SSL model based on combinations of M $=[2, 3, 4, 5, 6]$ and J $=[5, 10, 20, 40]$.

2. Theoretical Background

Laplacian graph is a type of weighted undirected graph called proximity graph. The idea is that to created a weighted graph which is based on the distance between each pair of input points. The simplest model if the distance between each pair of point is bigger than threshold, it will assign a weight 0 and 1 otherwise. This is because with pair of points that are lower than the threshold

Date: March 12, 2022.

will form its own cluster and thus disconnected cluster of points are created. The weight graph is typically defined as:

$$W_{ij} = \exp\Big(- \frac{||\underline{x_i} - \underline{x_j}||_2^2}{2\sigma^2}\Big) \tag{1}$$

The underline represent a vector, $x_k \in X$ where X is the input matrix, and σ represents the variance parameter that dictate how big or small the cluster is. Thus, the steps of creation of unnormalized Laplacian graph as follows. First, create a degree vector, $d_j = \sum_{i=0}^{N-1} W_{ji}$. Then, create a diagonal degree matrix,D, which is a diagonal matrix with diagonal entries of the degree vector. Lastly, create the unnormalized laplacian graph, $\tilde{L} = D - W$, which is a matrix.

Next, we move on to spectral clustering. The idea of the spectral clustering is that creating a feature map such that it can transform the input data and makes clustering easier. By using Laplacian graph, we can exploit the geometry of the graph to determine the clusters based on the eigenvectors. We can compute the eigendecomposition of $\tilde{L}$ and look at its eigenvector matrix $\tilde{Q}$ and arrange eigenvalues in ascending order with its corresponding eigenvector.

Then by plotting the first few eigenvectors, we can then see which eigenvector that provides the most direct information about the cluster classification. Note that using choosing one eigenvector for clustering only works for binary classification. If we want to classify more than 2 objects, we need to use more eigenvectors to give us more information about the relation between pairs of clusters.

For binary classification, usually the first eigenvector is constant since the eigenvalue is close to zero. Thus, usually the second eigenvector (q_1) provides most of the information about the clusters for binary classification. This second eigenvector is known to be the Fiedler vector. The q_1 eigenvector is important because it can be used as approximation of the label by taking the sign of the vector if the labels used are ± 1.

Next, we will cover semi-supervised learning. The idea is that given datasets which a fraction of it is labelled, and get the model to label the rest of the unlabelled dataset by incorporating clustering methods into it. We can use the graph Laplacians as our feature maps and then apply regression based on the eigenvectors and the labelled datasets. After that, we can do label propagation to the rest of the unlabelled data sets by multiplying the the coefficient of the regression with the unlabelled dataset. For current problem, we will apply linear regression or Ridge regression with near zero regularization terms, since the M, number of eigenvectors is much smaller than J, number of labelled sets [2].

The SSL and clustering accuracies are defined as

$$\text{accuracy} = 1 - \frac{1}{435} \cdot \text{number of misclassified members} \tag{2}$$

3. Algorithm Implementation and Development

Python packages that are used:numpy [1], matplotlib [3], scikit-learn [5], pandas[4] and scipy [7].

Pre-processing The preprocessing part is separating the data into the inputs (X) and the labels (Y) used for the comparison of the model. Pandas package is used for sorting the data. The labels used for classifying the parties are (+1) for Democrats and (-1) for Republicans. Create another X and Y matrices under different names that are arranged based on their parties so that the clustering model works better in separating the parties.

Clustering First, the weight function W. Then create euclidean distance matrix for values of each pair of input points using scipy package. For initial σ choose a value somewhere between $(0, 4]$. The input used should be the inputs that have been reordered based on party affiliations. Unnormalized Laplacian matrices are then computed based on the distance matrix and the weight function. Compute the eigen-decomposition of the the Laplacian matrices and sort the eigenvalues in the ascending order and with their respective eigenvectors. Then, using the sign of the first non-zero (second) eigenvector (q_1), the accuracy can be calculated. After doing the initial calculation, divide the $(0, 4]$ segment into many points and find the optimal σ based on the highest accuracy. Note that since it is an unsupervised learning, the labels sometimes switches sign. To do that use a conditional statement that flips the sign of the q_1 if the accuracy is less than 0.5. The value is chosen because the accuracy values have symmetry about 0.5 value when plotted. Furthermore, note that after a certain σ the accuracy values are fluctuating around some value. Thus, the optimal sigma that is chosen is the value above the critical σ where the accuracy values transition where the accuracy value is at the highest point of the fluctuation.

Semi-Supervised Learning For this portion, use the original order of input and use the optimal σ found in the clustering part. Then, use the optimal σ to generate matrix of eigenvectors, Q using the clustering code. Based on J and M that are chosen, create another matrix, A that is has first J columns and M rows of the Q matrix. Now, apply linear regression model on matrix A and the first J rows of column vector Y. Use Ridge regression from scikit-learn package to do the regression with very small regularization term since the assumption is that M is much smaller than J. After applying the Ridge regression, extract the coefficients of the regression and multiply it by the first M columns of Q matrix to get the predicted labels of the entire data. Take the sign of the predicted labels and compare it to vector Y. Finally, calculate the accuracy of the model. Note that during the matrix multiplication the coefficients might need to get transpose due to the results being a row vector. Another precaution is that the resulted accuracy might be reversed since the labelling is random from the clustering algorithm.

4. Computational Results

The section will discuss the results from analyzing the data. Figure 1 shows how the variance parameter is varied from 0.2 to 4 across 50 data points. The optimal σ^* is chosen to be 2.14 and shown in the graph. The value is chosen as the optimal σ^* since it is the first σ with the highest clustering accuracy, around 88% .

Notice that there are two points with σ near 0.5 with very high accuracy. These two points are anomaly because the way the algorithm is set up. The algorithm is set up such that if the accuracy is below 0.5 it will change the sign of q_1 and redo the accuracy calculation. A more robust algorithm is needed so that it will reverse the sign of q_1 such that the accuracy is in increasing manner rather than just purely using symmetry at 0.5 to flip all the accuracy below 0.5. However, this method provides with needed information since the optimal σ beyond the anomaly σ values and based on the geometry of the accuracy curve.

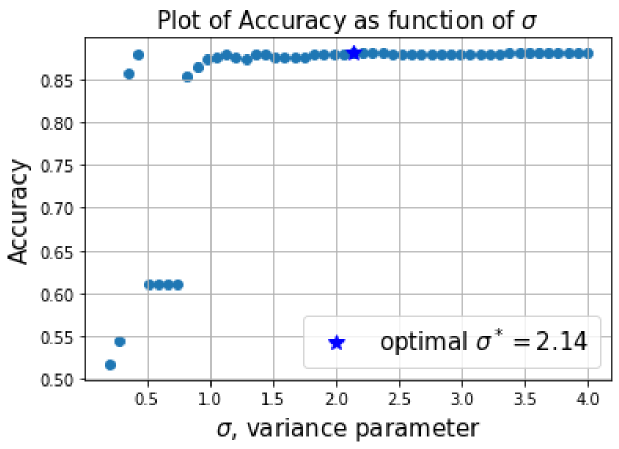

Figure 1. The accuracy with varying σ from 0.2 to 4 using 50 data points.

Table 1 provides information of the accuracies of the SSL classifier model when M and J are varied. Looking at the table, it can be seen that M = 2 is the most consistent value of M because the accuracy values does not fluctuate much. This makes sense that when M = 2, it is far less than all the J values used in the SSL model. For M = 3 it is also similar to M = 2 but more fluctuation and higher average. However, if we look at M = 4 and M = 5, particularly when J = 5, they have very low accuracy. This is because it is breaking the assumption when we are applying the regression that M needs to be far less than J. Thus, when M is considerable to J but still smaller than J, it might be that the minimizer function have problem solving the problem and might need regularization term. At the same time, if M is considerable to J but bigger than J, it might be because the eigenvector or the geometry will dictate more on the classifier than the label set. As a result, any error that the labelled set brings such as uneven distribution might be reduced.

If we examine further about the accuracy values, there is no clear pattern whether more labelled set might be useful for the model. This might be because the labelled set might bring some bias to the model since the data is already uneven in the first place. So, using more labelled data might not be useful if it is uneven distribution. It is better to use smaller labelled data sets with more even distribution so that the classifier able to use the information in conjunction with the geometry from the eigenvectors. We can see having smaller labelled data sets is useful because the highest accuracy is combination of M = 4 with J = 10. However, it can not be ruled out that at J = 10 the labelled data sets are coincidentally even coupled with the correct number eigenvectors to produce the highest accuracy of this SSL model.

J \ M	2	3	4	5	6
5	87.4%	89.9%	39.8%	41.4%	86.7 %
10	87.4%	86.7%	92%	78.4%	81.4 %
20	87.8%	87.8%	89.9%	85.1%	89.4 %
40	87.6%	87.6%	89.2%	86.2%	86.4 %

Table 1. The table shows how M, number of chosen eigenvectors from the Laplacian embedding and J, number of the seen labels vary with accuracy with $\sigma^*= 2.14$

5. Summary and Conclusions

From the politician dataset, we learnt how to apply clustering algorithm using Laplacian graph and semi-supervised learning through Laplacian graph embedding. Through clustering, we found optimal σ values that produces the highest accuracy. Using the optimal σ we apply semi-supervised learning through Laplacian graph embedding with varying number of labelled dataset and feature maps (eigenvectors). The results is that if the feature maps are considerably much smaller than the labelled dataset, the accuracy of the learning is very high and does not fluctuate much.

Therefore, to further study the data, we can apply labelling in different way such as one-hot labelling such that the model can work better. To study more on the variation M and J on the data accuracy, we can employ more combinations to see if there is a clear relationship between M and J such that one can write approximate the ratio to achieve certain accuracy if needed. Furthermore, a regularization term can be added into the consideration when M is roughly on the same magnitude as J when using Ridge regression.

Acknowledgements

The author is thankful to Prof. Hosseni in teaching the material and providing the demo that is similar to the problem. I would like thank Raymond Mead for helping my clustering algorithm.

I would like to thank 482/582 discord community for providing helpful ideas for report writing. The discord community also helps me figure out other programming issue and guidance on the problem.

References

[1] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020.

[2] B. Hosseini. Lectures 19-22. University of Washington-Seattle (LOW 216), Feb-March 2022. AMATH 482/582.

[3] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[4] W. McKinney. Data structures for statistical computing in python. In S. van der Walt and J. Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 51 – 56, 2010.

[5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[6] J. C. Schlimmer. Incremental adjustment of representations for learning. In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 79–90. Elsevier, 1987.

[7] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.