Locally aware ensemble of Linear Classifiers

Archit Rathore & Yash Gangrade 15th December 2017

1 Introduction

We explore ways to combine multiple linear classifiers based on spatial information in this project. The core idea is to create an ensemble of multiple linear classifiers, each of which is good at classifying points in its local vicinity and the final prediction for any point is a combination of prediction from each of the local classifiers weighted by the distance of the point from classifier. This has the effect of "anchoring" each classifier to a small region of the instace space. Our method first trains a gaussian mixture model (GMM) on the training data and specified number of centers k. We then use the likelihood of a point with respect to a each me center as a weighing scheme to train as many linear classifier as the number of centers. This has the effect of adapting each classifier to robustly classify point around that particular center but would perform poorly on the complete data. We then provide a method to combine the predictions of the linear classifier. Our experiments show that our method achieves better accuracy for binary classification against a single global classifier.

2 Literature Review

There has been some work on combination of linear classifiers. In (Dietterich, 2000)[1], different ensemble methods for constructing a set of classifiers and using voted prediction to classify a new point. Paper by (Duin, 2002)[2] presents a discussion on the use of combination of linear classifiers and how they should be trained combined to produce an optimal result in the case where a single classifier gives a suboptimal result. Paper by (Erdogen, 2010) explores the number of methods of combining the outputs of a set of linear classifiers. One of the effective method of combination is using a linearly weighted combination rule [3]. (Ponti Jr., 2011)[5] discusses different approaches to make an ensemble of classifiers, effectiveness of each method and how it can be improved. Paper by (Tumer, 1996)[6] provides an analytical approach for quantifying the improvements achieved in the classification results by using a combination of linear classifiers.

3 Method Overview

3.1 Data Generation

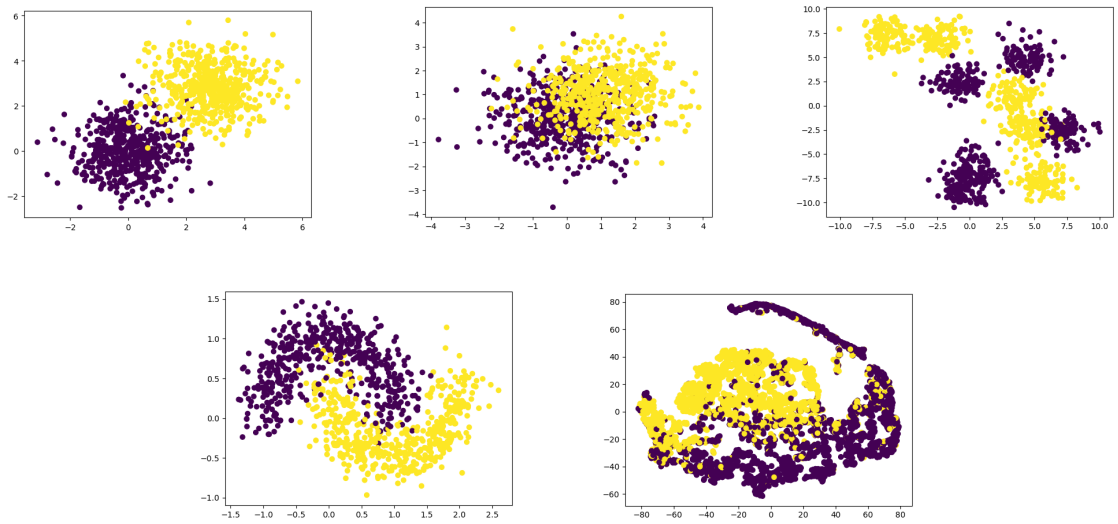


Figure 1: Different types of Data. (a) Almost Linearly Separable (b) Inseparable (c)Locally Linearly Separable (d) Moons Dataset (e) Twitter Dataset

We consider 4 type of synthetic spatial data

- **Almost Linearly Separable Data**: Two isotropic normal distributions $\mathcal{N}_1, \mathcal{N}_2$ at a distance of 2σ ($|\mu_1 - \mu_2| = 2\sigma$) have nearly will overlap in nearly 5% of data (95% of points in a normal distribution are contained in a radius of 2σ). We use this fact to generate nearly seperable data.
- **Badly Non Linearly Separable Data**: Following the same procedure as above, this time we are keeping a low separation between the centers of the clusters so that there is sufficient overlap between the clusters to make it badly non linearly separable.
- **Locally Linearly Separable Data**: In this case, the data is nearly linearly separable locally but not as a whole. This is achieved by having points generated by k gaussian distributions with random centres and random labels.
- **Moons dataset**: This is a 2-dimensional dataset that consists of two interleaving half circles. The dataset is not seperable by a linear boundary but radial basis kernels have success classifying this data.

Additionally we also run our method on the **Twitter dataset** that is part of the competitive project to assess the performance of our method on a real-world dataset.

3.2 Training local classifiers

The aim is to train k linear classifiers (i.e. whose decision boundary is given by $\mathbf{w^T x} + b$) such that each classifier is "anchored" in a small region of the instance space. The idea is that even an highly non-linear dataset can be viewed as a combination of multiple locally linear sub-regions (this idea is also used in Locally Linear Embeddings (LLE), a non-linear dimensionality reduction method).

The anchoring is achieved by training k linear classifiers on weighted copies of the dataset. The weights are learned by fitting a Gaussian Mixture Model (GMM) G, and the weights of a datapoint are then given by $G(\mathbf{x}) = \{g_1(\mathbf{x}), g_2(\mathbf{x}), \dots g_k(\mathbf{x})\}$. This gives us a set of weak local classifiers (somewhat similar to the weak learners in AdaBoost, with the difference being that each weak learner is possibly a strong learner for a small region but a weak learner with respect to the entire instance space).

Following describes the components of the model in more detail.

3.2.1 Gaussian Mixture Model

We use Gaussian Mixture Model (GMM) to perform the clustering of the data points and probabilistically assign each point to cluster centers. For an observation x and a Gaussian Mixture Model with k clusters denote the cluster probability for a point x as $P_x = \{p_1(x), p_2(x), ..., p_k(x)\}$. The vector P_x is the likelihood of x w.r.t the means of each of the k components of the GMM. The number of components k is a hyper-parameter and can be tuned using cross-validation.

3.2.2 Local linear classifier

Let $X \in \mathcal{R}^{n \times d}$ be the training data, $Y \in \{-1, 1\}^n$ be the true labels. Let $G_k(X) \in \mathcal{R}^n$ be the likelihood of each n dimensional vector in X w.r.t. the k^{th} gaussian component of the trained gaussian mixture model G.

For $i \in \{1, \dots, k\}$, we train linear classifiers that minimize the following objective:

$$J_i(\mathbf{w}, X, Y) = \mathbf{w^T} G_k(X) Y$$

The above equation is essentially the same objective as the traditional linear classifiers except that we use a weighted version of the input data. This weighing assigns higher weights to all points in vicinity of the k^{th} cluster mean, giving us the desired anchoring effect.

This gives us k linear classifiers $\mathbf{w_1}, \mathbf{w_2}, \dots, \mathbf{w_k}$. Here, each $\mathbf{w_i}$ represents the weight vectors of the i^{th} classifier.

3.3 Inference

Let the test datapoint be x'. Let $P_{x'}$ be the vector of likelihoods of x' with each of the k components of the gaussian mixture model G. The label for this point in our model is given by:

$$y_{pred} = sgn\left(\sum_{i=1}^{k} P_{x'}(i) \ \mathbf{w_i^T} x' \right)$$

The above equation translates to making individual predictions by all k classifiers and then taking a weighted sum of the predictions where the weights are given the likelihood of test point w.r.t the gaussian mixture.

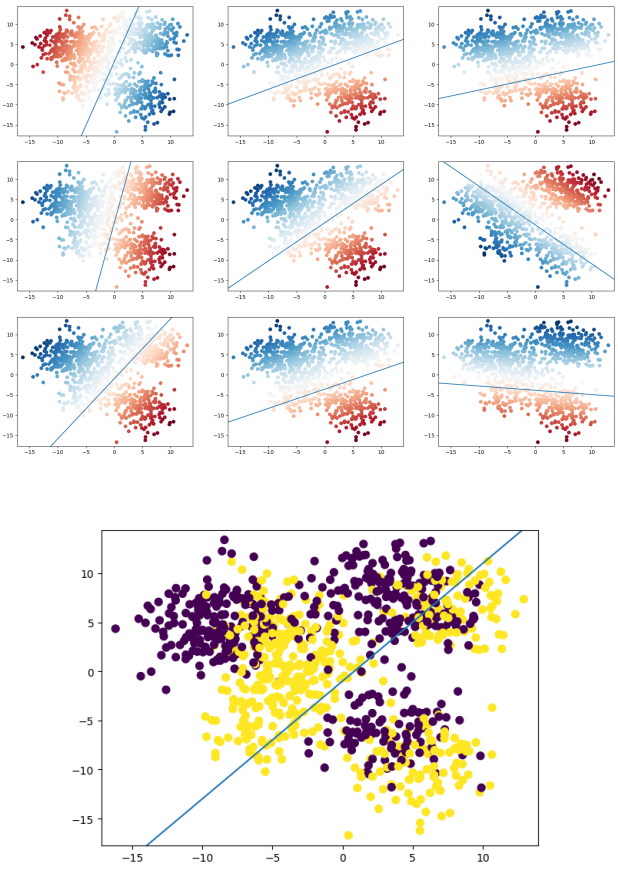


Figure 2: (top) Multiple local classifiers (bottom) Single global classifier

4 Experiments

We use our implementation of the perceptron (PP) and logistic regression (LR) classifiers. We first train a global linear classifier denoted by PP-G (global perceptron) and LR-G (global logistic regression classifier). We compare these against implementation of our locally linear perceptron (PP-LL) and locally linear logistic regression (LR-LL). The number of components k for the GMM [4] is chosen by 5-fold cross validation on the training set. The results in Table 1 are on a 20 dimensional dataset of 5000 training examples and 5000 test examples for locally linearly seperable, almost linearly seperable, inseparable, 2 dimensions and 5000 training examples for Moons dataset and 16 dimensions for Twitter dataset with 20000 training examples and 9000 test examples.

We studied the impact of dimension on both global and locally linear classifiers.

We also report the performance of K-nearest neighbors classification with $k = 10$ on each of the datasets mentioned in 3.1.

5 Results

The columns represent the type of data used to compute the accuracies. In this case, we have used 10000 points in each of the dataset. Here, the rows represents the accuracies obtained by testing on single classifier and an ensemble of linear classifiers.

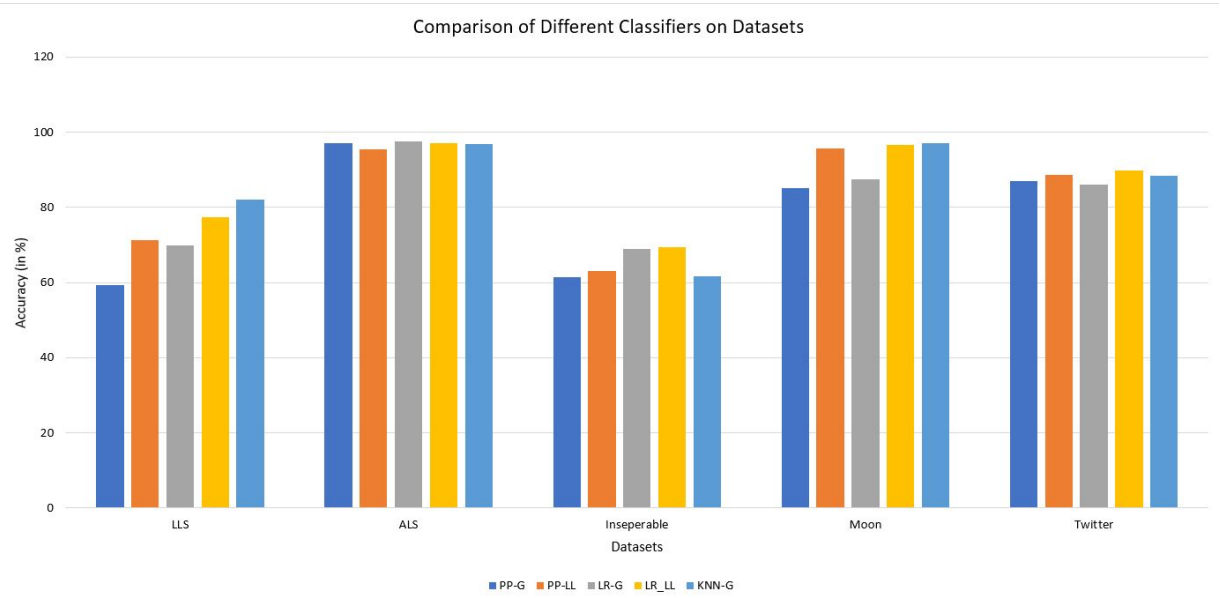
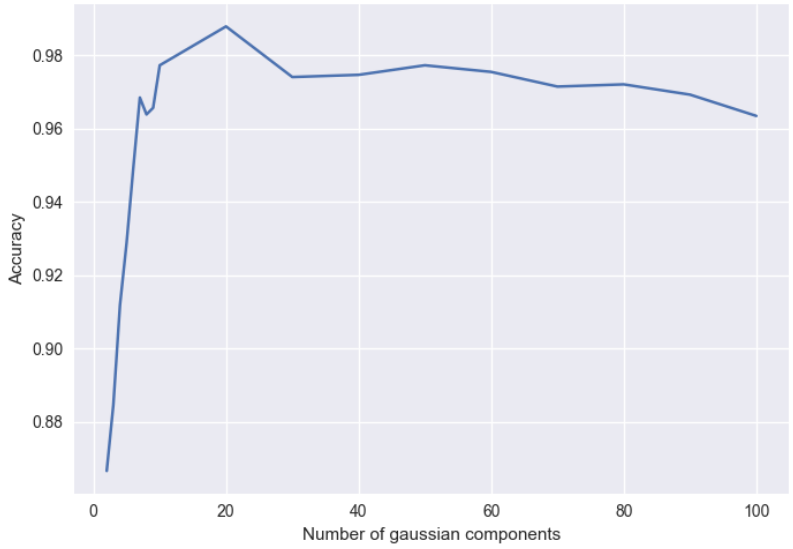


Table 1: Classification results on different datasets

	Classifier	Locally Linearly Separable	Almost Linearly Separable	Inseparable	Moon	Twitter
1	PP-G	59.30%	97.06%	61.38%	85.06%	86.96%
2	PP-LL	**71.36%**	**95.38%**	**62.98%**	**95.66%**	**88.60%**
3	LR-G	69.78%	97.52%	68.86%	87.36%	86.08%
4	LR-LL	**77.44%**	**97.00%**	**69.30%**	**96.70%**	**89.76%**
5	KNN-G	82.10%	96.84%	61.78%	97.00%	88.36%

We also show the effect that parameter k (number of components of the mixture model) has on model performance.



6 Conclusion

From the Table 1, it is clear that if we create an ensemble of linear classifiers for the classification of high-dimensional data, we get a reasonably high accuracy as compared to the sub-optimal accuracy of a single linear classifier. The results hereby demonstrates our theoretical understanding correctly.

7 Future Work

- Using different models like Locality Sensitive Hashing (LSH) in place of Gaussian Mixture Models in the training phase
- Incorporate two different linear classifiers like Perceptron and SVM together in the training phase and combiner them according to the locality of data
- Pose learning weights of each classifier for test points as a learning problem itself - use learning to learn to the weights for each point. (Maybe try out some non-linear combinations of linear classification)
- This paper solely focus on creating ensemble of Linear Classifiers but ensemble of Non-Linear Classifiers can also be developed and used.

8 References

[1] Thomas G Dietterich et al. "Ensemble methods in machine learning". In: *Multiple classifier systems* 1857 (2000), pp. 1–15.

[2] R. P. W. Duin. "The combining classifier: to train or not to train?" In: *Object recognition supported by user interaction for service robots*. Vol. 2. 2002, 765–770 vol.2. DOI: 10.1109/ICPR.2002.1048415.

[3] Hakan Erdogan and Mehmet Umut Sen. "A unifying framework for learning the linear combiners for classifier ensembles". In: *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE. 2010, pp. 2985–2988.

[4] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[5] Moacir P Ponti Jr. "Combining classifiers: from the creation of ensembles to the decision fusion". In: *Graphics, Patterns and Images Tutorials (SIBGRAPI-T), 2011 24th SIBGRAPI Conference on*. IEEE. 2011, pp. 1–10.

[6] Kagan Tumer and Joydeep Ghosh. "Analysis of decision boundaries in linearly combined neural classifiers". In: *Pattern Recognition* 29.2 (1996), pp. 341–348.