Nearest Neighbor Classifiers versus Random Forests and Support Vector Machines

Saket Sathe, Charu C. Aggarwal
IBM Research AI

Motivation – What is the difference between NN-Classifiers and RF/SVM?

- **Nearest Neighbor** classifiers are lazy learners $\Longrightarrow$ very little work is done during training and most of the work is done during testing.
- **Random Forest (RF) and Support Vector Machines (SVMs)** are eager learners $\Longrightarrow$ very little work is done during testing and most of the work is done during training
- Nearest neighbor methods can discover complex decision boundaries and can provide theoretically bounded error with an infinite amount of data.
- In spite of this theoretical promises nearest neighbor classifiers have not come close to achieving the empirical performance of RF and SVMs
- **Key Reason:** Lazy learners do not understand which data points are important for prediction $\Longrightarrow$ High variance decision boundary + poor generalization

Objective: Can we connect NN $\Leftrightarrow$ RF/SVM and do better?

Weighted Nearest Neighbors (KNN)

- We are given a labeled data set $\mathcal{D} = \{\overline{X}_i, y_i\}$ for $i \in (1, \ldots, n)$
- Let w_i be the weight of the ith data point $\overline{X_i}$.
- Then, the predicted label of test data point $\overline{Z}$ (denoted $\hat{y}(\overline{Z})$) is the weighted nearest neighbor method is computed as follows:

$$\hat{y}(\overline{Z}) = \text{sign}\left(\sum_{i=1}^{n} y_i w_i(\overline{Z})\right). \tag{1}$$

- k-**nearest neighbors classifier** is a special case where the weights of the k closest examples are set to 1, and all others to 0.

Support Vector Machines (SVM)

- Support vector machines are weighted nearest neighbor classifiers where:

$$w_i(\overline{Z}) = \lambda_i S(\overline{X_i}, \overline{Z}). \tag{2}$$

  $S(\overline{X_i}, \overline{Z})$ is Gaussian kernel similarity Gaussian kernel and $\lambda_i \geq 0$ is a global weight learned from data.

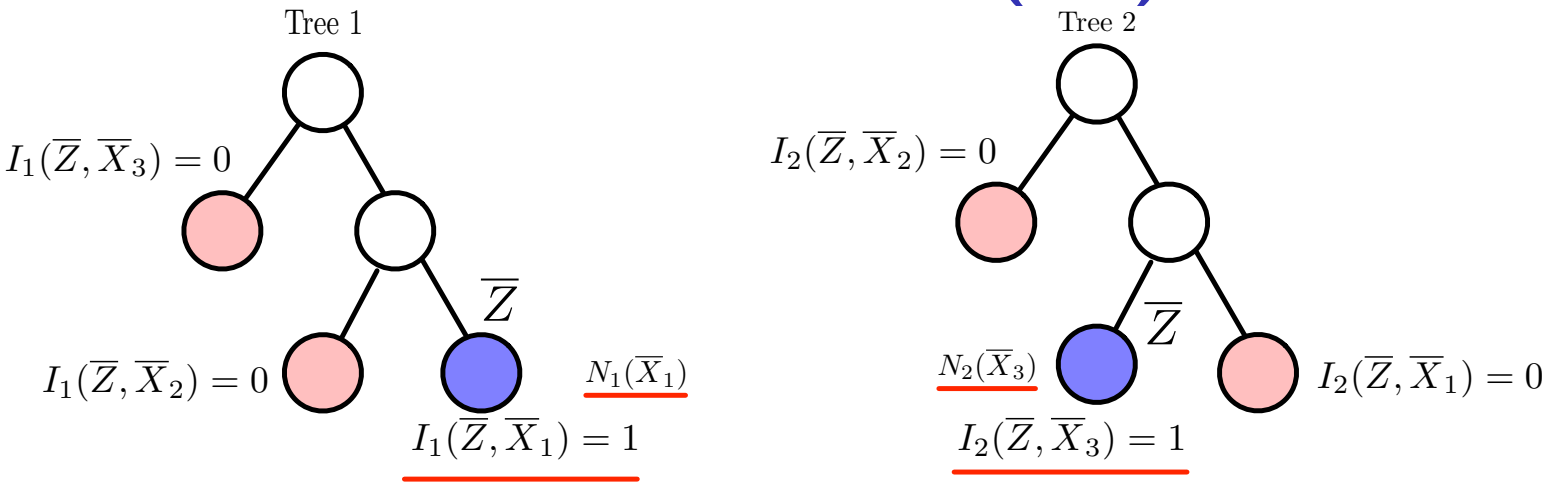
- **Optimization model for learning λ_i:**

$$\text{Maximize } \sum_{i=1}^{n} \lambda_i - \overbrace{\sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j S(\overline{X_i}, \overline{X_j}) y_i y_j}^{\text{influences weights } \lambda_i} \tag{3}$$

$$\text{subject to: } \sum_{i=1}^{n} \lambda_i y_i = 0 \text{ and } 0 \leq \lambda_i \leq C \tag{4}$$

λ_i increases for instances near other classes ("difficult instances"), and tends to set λ_i to 0 otherwise ("easy instances").

Random Forests (RF)



- Then the weight of the training instance $\overline{X_i}$ is defined as follows:

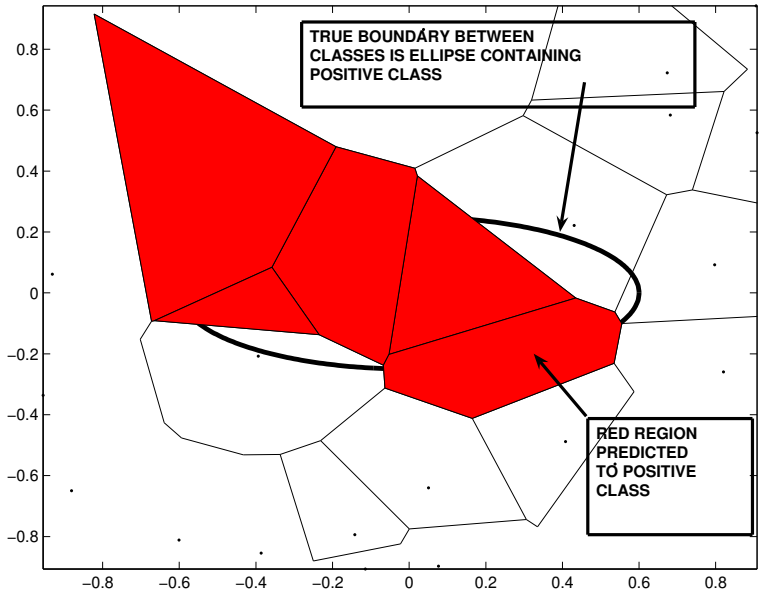
$$w_i(\overline{Z}) = S(\overline{X_i}, \overline{Z}) = \frac{1}{T} \sum_{j=1}^{T} \frac{I_j(\overline{Z}, \overline{X_i})}{N_j(\overline{X_i})} \tag{5}$$

- Substitute Eq. 5 in Eq. 1, and change order of summation:

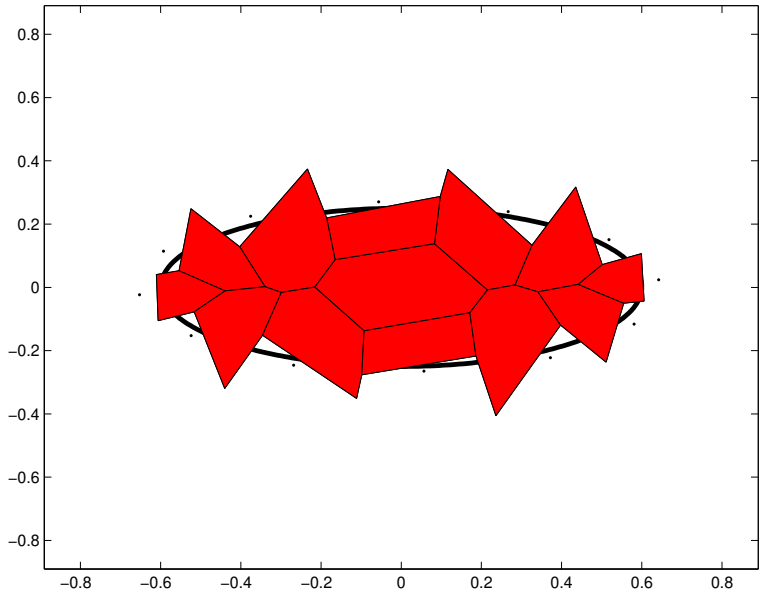
$$\hat{y}(\overline{Z}) = \text{sign} \left(\underbrace{\frac{1}{T} \sum_{j=1}^{T}}_{\text{ensemble}} \overbrace{\sum_{i=1}^{n} y_i \frac{I_j(\overline{Z}, \overline{X_i})}{N_j(\overline{X_i})}}^{\text{prediction of tree } j} \right) . \tag{6}$$

Voronoi Classifier (VC) - Motivation

- **Adaptive nearest neighbor method** that has connections with **random forests** and **support vector machines**.
- Ensemble-centric approach, where overfitting is encouraged in each ensemble component (like in all ensemble-centric approaches)
- **Voronoi Anchors:** Subset of carefully chosen data points (similar to support vectors of SVM) for classification by using the induced Voronoi partitioning.



(a) randomly chosen anchors



(b) carefully chosen anchors

Voronoi Classifier (cont'd) (VC)

Steps of a single ensemble component:

1. **Sub-sampling**: Randomly sample S training points from data set $\mathcal{D}$
2. **Anchor Selection:** Randomly sample A voronoi anchors from S, $|A| = \lfloor f \cdot n \rfloor$
3. **Label Prediction:** Predict label of each point in S as the label in the Voronoi cell in which it lies and compute accuracy α
4. **Iterative Anchor Set Refinement**
    1. Select a misclassified point $\overline{X}$ from S and promote it to become an anchor
    2. Demote an existing anchor $\overline{Y}$ to become non-anchor
    3. Compute α' accuracy with the new anchor set. If $\alpha' > \alpha$ keep the promotion, otherwise revert
    4. Stop if no improvement in L consecutive tries

Soft Voronoi Classifier (SVC)

- As the name suggests, the SVC is a "soft" replacement version of VC.
- Instead of promoting non-anchor $\overline{X}$ and demoting anchor $\overline{Y}$, SVC computed a third point $\overline{Z}$ as follows:

$$\overline{Z} = \beta\overline{X} + (1-\beta)\overline{Y}. \qquad \beta \in U(0,1). \tag{7}$$

- $\overline{Z}$ is assigned $\overline{X}$'s label if $\beta \geq 0.5$, otherwise it gets $\overline{Y}$'s label.
- $\overline{Z}$ is promoted instead of $\overline{X}$, and $\overline{Y}$ is demoted (as before)
- The decision boundary is gradually expanded (or contracted) and encourages even more overfitting as compared to VC

Experiments - Classification Accuracy

Data Sets: UCI Machine Learning Repository[1], LibSVM repository[2] or from Fernandez-Delgado *et. al.*[3]

Table: Test data accuracy (%). The top-2 methods are shown in boldface.

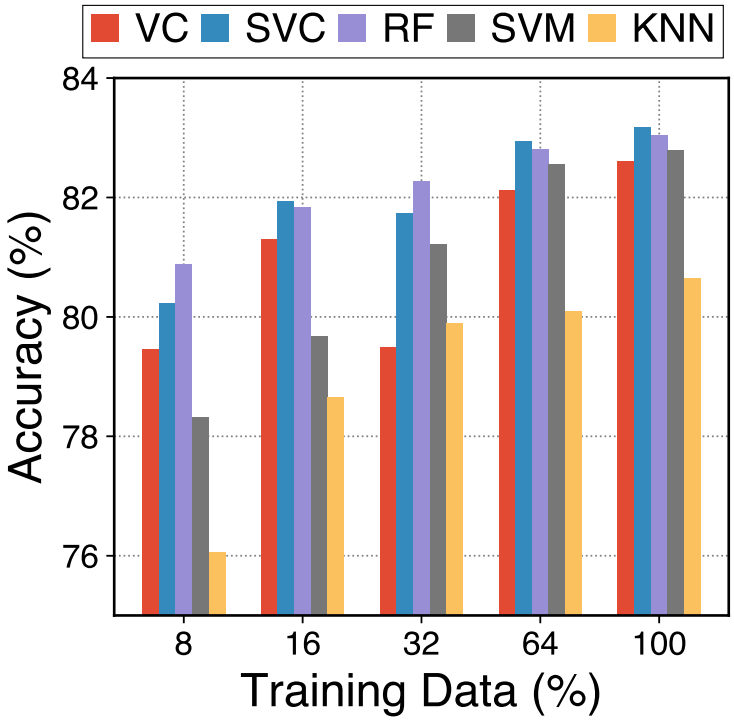
Data set	SVC	VC	RF	SVM	KNN
Parkinsons	**71.79**	**71.79**	71.54	63.08	**73.85**
Congress-Voting	**60.29**	58.20	59.77	**58.39**	54.92
Breast-Cancer	**96.93**	96.86	**97.58**	95.97	95.40
Annealing	**92.22**	92.10	**95.97**	91.97	92.00
Optical	97.65	97.63	**97.88**	**98.37**	97.65
a1a	**83.17**	82.61	**83.04**	82.78	80.65

[1] `http://archive.ics.uci.edu/ml/index.php`

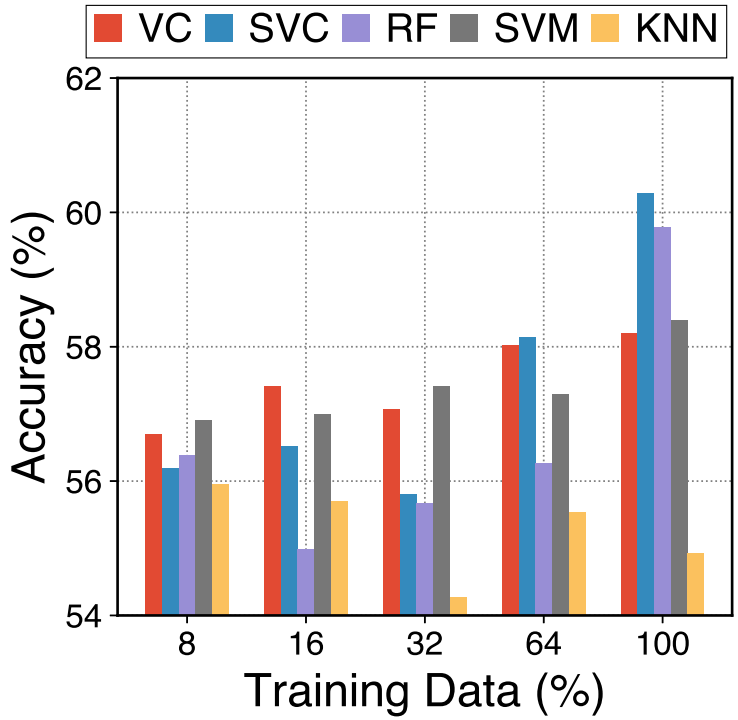
[2] `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`

[3] M. Fernandez-Delgado, *et. al.*. Do we Need Hundreds of Classifiers to Solve Real World Classification Problems? JMLR, 15(1), pp. 3133–3181, 2014.

Experiments - Effect of Increasing Data Size



(a) a1a



(b) Congress-Voting

Figure: Comparing the performance of SVC and VC with the other classifiers as the data set size increases. Note the scale on the x-axis.

Average improvement in accuracy over RF, SVM, and KNN is 0.59%, 0.67% and 3.78%

Summary and Conclusion

- We showed the relationship between SVMs, random forests, and nearest neighbor classifiers.
- We demonstrate that it is possible for eager versions of nearest neighbor classifiers to match and even outperform random forests or SVMs.