

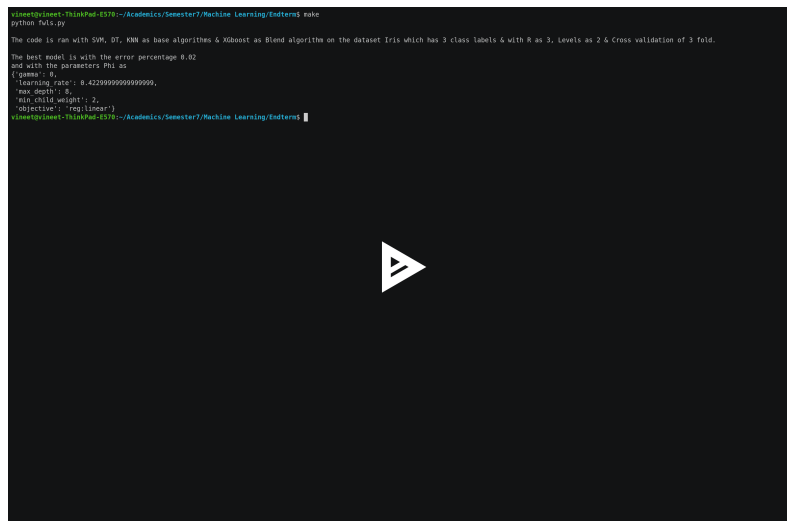
# MACHINE LEARNING END TERM SOLUTIONS FALL 2017

D.Nithish - IMT2014016  
R Vineet Reddy - IMT2014045  
V keerthi Chandra IMT2014064

---

## Question 1

The implementation of the Feature Weighted Linear Stacking algorithm can be found in the file **fwls.py**. To execute the file run **make** all or just **make**. To know how to run the code click on the below image [1](#) or click [here](#).



```
vineet@vineet-ThinkPad-E570:~/Academics/Semester7/Machine Learning/Endterms$ make
python fwls.py
The code is run with SVM, DT, KNN as base algorithms & XGBoost as Blend algorithm on the dataset Iris which has 3 class labels & with R as 3, Levels as 2 & Cross validation of 3 fold.
The best model is with the error percentage 0.02
and with the parameters Phi as
{ 'gamma': 0,
  'learning_rate': 0.42299999999999999,
  'max_depth': 6,
  'min_child_weight': 2,
  'objective': 'reg:linear' }
vineet@vineet-ThinkPad-E570:~/Academics/Semester7/Machine Learning/Endterms$
```

Figure 1: Snippet of how to run the code.

## Question 2

Let  $\epsilon$  be the error matrix of dimensions of  $n$  rows &  $d+1$  columns to make it homogeneous, where every  $\epsilon_i$  is I.I.D and has co-variance matrix  $\sigma^2 I$ .

$$W^* = \underset{W}{\operatorname{argmin}} E_{\epsilon}[(Y - (X + \epsilon)W)^2]$$

Simplifying the expectation term with respect to attribute noise

$$\begin{aligned} &= E_{\epsilon}[(Y - (X + \epsilon)W)^T(Y - (X + \epsilon)W)] \\ &= E_{\epsilon}[(Y^T - W^T X^T - W^T \epsilon^T)(Y - XW - \epsilon W)] \\ &= E_{\epsilon}[Y^T Y - Y^T XW - Y^T \epsilon W - W^T X^T Y + W^T X^T XW + W^T X^T \epsilon W \\ &\quad - W^T \epsilon^T Y + W^T \epsilon^T XW + W^T \epsilon^T \epsilon W] \\ &= E_{\epsilon}[Y^T Y - 2W^T X^T Y - 2W^T \epsilon^T Y + 2W^T X^T \epsilon W + W^T X^T XW \\ &\quad + W^T \epsilon^T \epsilon W] \\ &= Y^T Y - 2W^T X^T Y + W^T X^T XW - 2E_{\epsilon}[W^T \epsilon^T Y] \\ &\quad + 2E_{\epsilon}[W^T X^T \epsilon W] + E_{\epsilon}[W^T \epsilon^T \epsilon W] \end{aligned} \tag{1}$$

And  $W^T \epsilon^T Y$  is nothing but  $\sum_{i=1}^n \sum_{j=1}^{d+1} w_j \epsilon_{ij} y_i$

$W^T X^T \epsilon W$  is nothing but  $\sum_{i=1}^{d+1} \sum_{j=1}^{d+1} \sum_{k=1}^n w_i w_j x_{ik} \epsilon_{kj}$

$W^T \epsilon^T \epsilon W$  is nothing but  $\sum_{i=1}^{d+1} \sum_{j=1}^{d+1} \sum_{k=1}^n w_i w_j (\epsilon_{ik} \cdot \epsilon_{kj})$

Now computing the expectations of these 3 terms with respect to the attribute noise

$$\begin{aligned} E_{\epsilon}[W^T \epsilon^T Y] &= E_{\epsilon}\left[\sum_{i=1}^n \sum_{j=1}^{d+1} w_j \epsilon_{ij} y_i\right] \\ &= \sum_{i=1}^n \sum_{j=1}^{d+1} w_j y_i E_{\epsilon}[\epsilon_{ij}] \quad \because \text{By using the linearity of expectations} \\ &\therefore E_{\epsilon}[W^T \epsilon^T Y] = 0 \quad \because E_{\epsilon}[\epsilon_{ij}] = 0 \end{aligned} \tag{2}$$

Now

$$\begin{aligned} E_{\epsilon}[W^T X^T \epsilon W] &= E_{\epsilon}\left[\sum_{i=1}^{d+1} \sum_{j=1}^{d+1} \sum_{k=1}^n w_i w_j x_{ik} \epsilon_{kj} y_i\right] \\ &= \sum_{i=1}^{d+1} \sum_{j=1}^{d+1} \sum_{k=1}^n w_i w_j x_{ik} E_{\epsilon}[\epsilon_{kj}] \quad \because \text{By using the linearity of expectations} \\ &\therefore E_{\epsilon}[W^T X^T \epsilon W] = 0 \end{aligned} \tag{3}$$

Now,

$$\begin{aligned}
E_\epsilon[W^T \epsilon^T \epsilon] &= E_\epsilon\left[\sum_{i=1}^{d+1} \sum_{j=1}^{d+1} \sum_{k=1}^n w_i w_j (\epsilon_{ik} \epsilon_{kj})\right] \\
&= \sum_{i=1}^{d+1} \sum_{j=1}^{d+1} \sum_{k=1}^n w_i w_j E_\epsilon[\epsilon_{ik} \cdot \epsilon_{kj}] \quad \because \text{By using the linearity of expectations}
\end{aligned}$$

Since  $\epsilon_i$  is I.I.D and co variance matrix of  $\epsilon_i$  is  $\sigma^2 I$ .

$$\begin{aligned}
E_\epsilon[\epsilon_{ik} \cdot \epsilon_{kj}] &= 0 \quad \text{if } i \neq j \\
&= \sigma^2 \quad \text{if } i = j
\end{aligned}$$

$$\therefore E_\epsilon[W^T \epsilon^T W] = \sigma^2 \quad (4)$$

using (2), (3), (4) in equation (1) We get

$$\begin{aligned}
&Y^T Y - 2W^T X^T Y + W^T X^T X W + \sigma^2 \sum_{i=1}^{d+1} w_i^2 \\
&\text{i.e } Y^T Y - 2W^T X^T Y + W^T X^T X W + \sigma^2 \|W\|_2^2 \\
&\implies Y^T Y - 2W^T X^T Y + W^T X^T X W + W^T \sigma^2 W
\end{aligned}$$

Differentiating with respect to  $W$  and equating it to 0 we get

$$\begin{aligned}
-2X^T Y + 2X^T X W + 2\sigma^2 W &= 0 \\
(X^T X + \sigma^2 I)W &= X^T Y
\end{aligned}$$

$$W = (X^T X + \sigma^2 I)^{-1} X^T Y$$

Comparing this to the solution of ridge regression the  $\lambda$  in ridge regression here is  $\sigma^2$

### Question 3

We know that for a data set of size 'n' and hypothesis class 'H'

For,  $n \leq \delta_H \implies G_H(n) = 2^n$

$n > \delta_H \implies G_H(n) = 2^n$

So, there are  $2^n$  distinct ways to label the 'n' points.

To shatter these data points we should have at least  $2^n$  hypothesis and  $VC(H) = n$

$$\begin{aligned} |H| &\geq 2^n \\ n = \delta_H &\leq \log_2 |H| \end{aligned}$$

## Question 4

### Notions Used

- $H(D)$  = Homogeneity measure of labels
- Chi value - Higher the value of Chi-Square higher the statistical significance of differences between sub-node and Parent node.

$$Chi = \sum_{Classes i Children j} \frac{(N_{ij} - E_{ij})^2}{E_{ij}}$$

1.  $N_{ij}$  = Number of points of class i in node j

2.  $E_{ij}$  = Expected number of points of class i in node j i.e  $N_i * P_j$

- Expected information needed to classify a tuple

$$E(D) = \sum_{i=1}^m p_i \log p_i$$

- Information needed (after using attribute A to split D into v partitions) to classify D

$$E(D, A) = \sum_{j=1}^v \frac{|D_j|}{|D|} E(D_j)$$

- $\text{Gain} = E(D) - E(D, A)$
- $D_1, \dots, D_n$  = partitions
- $D$  = dataset
- $n_i$  = no\_of elements in  $D_i$
- $n$  = no of elements in  $D$  without selected attribute missing
- $\text{probability}(D)$  - gives the probability vector of class labels
- The probability of class labels at leaf node is the product of all the conditional probabilities of all the nodes on the path from root to leaf

---

**Algorithm 1** Decision Tree

---

```
1: procedure PARTITIONS(D,attr)
2:   for x = 1....n do
3:     if x[attr] is none then
4:       Add to Di with probability of  $n_i/n$ 
5:     else
6:       D1,D2,,,,Dn = split(D,best_attr)
7: procedure PROBABILITY(D)
8:   c_i = number of labels of class i in dataset D
9:   p_i = c_i / n
10:  C = [p_1,.....p_k]
11:  return C
12:
13: procedure LEARNING(D)
14:   if H(D) = 1 then
15:     return probability(D)
16:   else if no attributes then
17:     return probability(D)
18:   else
19:     for j=1...d do
20:       D_1,D_2...D_n = partition(D,j)
21:       Gain = E(D) - E(D,i)
22:       best_value = max(Gain,best_value)
23:     new_attributes = actual - best_attribute
24:     for item in partitions do
25:       add_sub_tree(learning(item,new_attributes))
26:     for each leaf do
27:       find probability(leaf)
28:     if node.all_leaves then
29:       if significant_chi then
30:         return make_leaf(probability(D))
31:
32: procedure CLASSIFY(point,tree)
33:   if point.results != None then
34:     return point.results
35:   else
36:     v = values[tree.columns]
37:     if v == None then
38:       classify(point, true_branch)
39:       classify(point, false_branch)
40:       modify result[k] with corresponding probabilities ▷ k class label
41:     else
42:       if v == tree.value then
43:         branch = true_branch
44:       else
45:         branch = false_branch
46:     return classify(point,branch)
```

---

## Question 5

To show that 'k' is a valid kernel it is sufficient to show that there exists a feature transformation  $\phi$ .

Let  $\mu$  be the center of mass in the transformed feature space. So  $\mu$  is the average of all the data points in new feature space

$$\therefore \mu = \frac{1}{n} \sum_{i=1}^n \phi(x_i)$$

The square Euclidean distance from  $\mu$  to  $\phi(x)$  would be

$$\begin{aligned} \|\mu - \phi(x)\|^2 &= \langle \mu - \phi(x), \mu - \phi(x) \rangle \\ &= \langle \frac{1}{n} \sum_i \phi(x_i) - \phi(x), \frac{1}{n} \sum_i \phi(x_i) - \phi(x) \rangle \\ &= \frac{1}{n^2} \sum_{i,j} \langle \phi(x_i), \phi(x_j) \rangle - \frac{2}{n} \sum_i \langle \phi(x_i), \phi(x) \rangle + \langle \phi(x), \phi(x) \rangle \\ &= \frac{1}{n^2} \sum_{i,j} k(x_i, x_j) - \frac{2}{n} \sum_i k(x, x_i) + k(x, x) \\ \therefore \frac{1}{n} \|\mu - \phi(x)\|^2 &= \frac{1}{n} \left( \frac{1}{n^2} \sum_{i,j} k(x_i, x_j) - \frac{2}{n} \sum_i k(x, x_i) + k(x, x) \right) \\ &= \frac{1}{n^3} \sum_{i,j} k(x_i, x_j) - \frac{2}{n^2} \sum_i k(x, x_i) + \frac{1}{n} k(x, x) \end{aligned}$$

Closure properties of kernels,

1. If  $k_1$  &  $k_2$  are kernel functions then  $k = k_1 + k_2$  is also a kernel function.

Let Kernel or Gram matrix  $K_1$  correspond to Kernel function  $k_1$  & Kernel or Gram matrix  $K_2$  correspond to Kernel function  $k_2$ .  $K_1$  and  $K_2$  are semi positive definite. So,

$$\begin{aligned} \alpha^T K_1 \alpha &\geq 0 \quad \text{and} \quad \alpha^T K_2 \alpha \geq 0 \\ \alpha^T K_1 \alpha + \alpha^T K_2 \alpha &\geq 0 \\ \alpha^T K \alpha &\geq 0 \end{aligned}$$

2. Multiplication of two kernel functions gives an kernel function

$$\begin{aligned}k(x, y) &= k_1(x, y) \cdot k_2(x, y) \\&= \sum_{i=1} \phi_i(x) \cdot \phi_i(y) \cdot \sum_{j=1} \psi_j(x) \cdot \psi_j(y) \\&= \sum_i \sum_j [\phi_i(x) \cdot \psi_j(x)] \cdot [\phi_i(y) \cdot \psi_j(y)] \\&= \sum_i \sum_j p_{ij}(x) \cdot p_{ij}(y) \quad \text{where } p \text{ is a feature transformation} \\&= p(x)^T \cdot p(y)\end{aligned}$$

By these two properties we can say that above transformation is an feature transformation.

\_\_\_\_\_ End of Document \_\_\_\_\_