# Robust Multiview feature selection via View Weighted (RMvFS)

Tanishq Shrivastava (CS171088)

December 13, 2020

# Contents

# Chapter 1
# Introduction

## 1.1 Multi-view Learning

Multi-view learning means taking features from different perspectives for data collection. For example, for the task of identifying whether a web page is about any advertisement or not. For this task the features can be taken from two views i.e. the height and width of the advertisement window and different parameters used in the URL. Thus in this way the information becomes rich and it helps in predicting whether the web page is about any advertisement or not with more accuracy. Thus in this way Multi-view learning finds various applications like data integration, feature extraction, feature selection etc.

There are three main domains through which multi-view learning is carried out, they are:

1) Co-training: In this approach we train a model each for a view alternately and information between them is exchanged for perpetual learning.
2) Multi-kernel learning: Here data is mapped to different feature spaces with different kernel and then all are combined from all space.
3) Subspace learning: Here all views are represented in same latent space and shared information is used for learning.

In the referred research paper Subspace method is used for feature selection.

## 1.2 Feature Selection

'Curse of Dimensionality' is often used term in Machine Learning. This simply means that the high dimensional data is difficult to process and also requires high computational resources for training. Although high dimensional data is very common in many practical machine learning tasks. High Dimensional data also increases the probability of over-fitting. Thus it's important to eliminate the redundant and irrelevant features. There are basically two methods for it viz. i)

Feature Extraction and ii) Feature Selection. Feature Extraction reconstructs new features which are less in numbers but are best in predicting the labels. This method thus doesn't keep the original values of the features. Whereas in Feature Selection the subset of relevant features is selected thus keeping the original data values intact.

## 1.3 View Weighted Approach

The referred research paper [1] discusses a view weighted approach for enhanced feature selection from multiple views. This can be summarized as:

1) Each view will be imposed with $l_{2,1}$ –norm penalty to enhance robustness of the model. Also two loss terms are introduced which are for ensuring complementary nature and specificity of views.
2) In order to eliminate redundant features, $G_{2,1}$ –norm sparse regularizer is used to preserve the important views and $l_{2,1}$ –norm sparse regularizer to select the discriminative features from the important views.
3) The high dimensional problem is splitted into several low-dimensional problems in the process of optimizing the objective function, which reduces the computational complexity and the proposed iterative algorithm achieves a fast convergence rate.

# Chapter 2
# Mathematical Formulation

## 2.1 Formulation

The matrix $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N] \in R^{l \times N}$ represents the set of samples belonging to $c$ classes, where $l$ is the dimension of each sample, $N$ is the total number of samples, and $\mathbf{x}_i \in R^{l \times 1}$ represents the $i$th sample. $X$ can be expressed in the form of $m$ views and denoted as $X = [X_1^\top, X_2^\top, \cdots, X_m^\top]^\top \in R^{l \times N}$, where $X_v = [\mathbf{x}_{1v}, \mathbf{x}_{2v}, \ldots, \mathbf{x}_{Nv}] \in R^{l_v \times N}$ represents the $v$th view of the samples. Denote $Y = [\mathbf{y}^{1\top}, \mathbf{y}^{2\top}, \ldots, \mathbf{y}^{N\top}]^\top \in \{0, 1\}^{N \times c}$ the label matrix of $N$ samples. If the $i$th sample belongs to the $j$th category, then the $j$th entry of $\mathbf{y}_i$ equals to 1 and the rest ones equal to 0. $W$ is the projection matrix made up of sub-blocks $W_v \in R^{l_v \times c}$, $v = 1, \cdots, m$, that is, $W = [W_1^\top, W_2^\top, \cdots, W_m^\top]^\top \in R^{l \times c}$.

The referred research presents the following Objective function:

$$\min_{W, \theta_v} \sum_{v=1}^{m} (\theta_v)^p \|X_v^\top W_v - Y\|_{2,1} + \lambda_1 \sum_{v=1}^{m} \|W_v\|_{2,1} + \lambda_2 \sum_{v=1}^{m} \|W_v\|_F$$

$$\text{s.t.} \quad \sum_{v=1}^{m} \theta_v = 1, \; 0 \le \theta_v \le 1$$

Here,

λ1 , λ2, θv => parameters for adjusting the above three terms.

θv  for adjusting the importance of a view.

Formula for $l_{2,1}$ norm is :

$$\|\mathbf{A}\|_F \equiv \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$$

Here, r and p are row and column count of matrix A.

Formula for $G_{2,1}$ norm is:

Here A is a matrix having m and n as number of rows and columns.

Now the above objective function can be optimized using Augmented Lagrangian Multiplier method as below:

$$\begin{aligned}\min_{W_v,\theta_v,E_v} \quad & \sum_{v=1}^{m}(\theta_v)^p\|E_v\|_{2,1}+\lambda_1\sum_{v=1}^{m}\|W_v\|_{2,1}+\lambda_2\sum_{v=1}^{m}\|W_v\|_F \\ \text{s.t.} \quad & X_v^\top W_v - Y - E_v = 0,\ v=1,2,\ldots,m \\ & \sum_{v=1}^{m}\theta_v = 1,\ 0\le\theta_v\le 1\end{aligned}$$

The corresponding augmented Lagrangian function is:
Here $E_v$ is a slack variable,

$$\begin{aligned}L(W_v,E_v,\theta_v) = & \sum_{v=1}^{m}(\theta_v)^p\|E_v\|_{2,1}+\lambda_1\sum_{v=1}^{m}\|W_v\|_{2,1}+\lambda_2\sum_{v=1}^{m}\|W_v\|_F \\ & +\sum_{v=1}^{m} tr|\Lambda_v^\top(X_v^\top W_v - Y - E_v)| \\ & +\sum_{v=1}^{m}\frac{\mu}{2}\|X_v^\top W_v - Y - E_v\|_F^2\end{aligned}$$

where $\Lambda_v$ is the Lagrange multiplier, and $\mu$ is a penalty parameter. Since

$$\begin{aligned}&\sum_{v=1}^{m} tr|\Lambda_v^\top(X_v^\top W_v - Y - E_v)| + \sum^{m}\frac{\mu}{}\|X_v^\top W_v - Y - E_v\|_F^2 \\ =&\sum_{v=1}^{m}\frac{\mu}{2}\|X_v^\top W_v - Y \qquad \|A\|_{2,1} = \sum_{i=1}^{r}\sqrt{\sum_{j=1}^{p}A_{ij}^2}. \qquad (5)\end{aligned}$$

the augmented Lagrangian function is equivalent to the following form:

$$\begin{aligned}L(W_v,E_v,\theta_v) = & \sum_{v=1}^{m}(\theta_v)^p\|E_v\|_{2,1}+\lambda_1\sum_{v=1}^{m}\|W_v\|_{2,1}+\lambda_2\sum_{v=1}^{m}\|W_v\|_F \\ & +\sum_{v=1}^{m}\frac{\mu}{2}\|X_v^\top W_v - Y - E_v + \frac{1}{\mu}\Lambda_v\|_F^2 - \sum_{v=1}^{m}\frac{\mu}{2}\|\frac{1}{\mu}\Lambda_v\|_F^2 \qquad (6)\end{aligned}$$

Next, the specific iterative process of solving (3) is given.

**First step**: Fix $E_v$ and $\theta_v$, update $W_v$

When $E_v$ and $\theta_v$ are fixed, $W_v$ is the only variable, and its solution can be obtained by solving the following optimization problem:

$$\min_{W} \lambda_1 \sum_{v=1}^{m} \|W_v\|_{2,1} + \lambda_2 \sum_{v=1}^{m} \|W_v\|_F + \sum_{v=1}^{m} \frac{\mu}{2} \|X_v^\top W_v - Y - E_v + \frac{1}{\mu}\Lambda_v\|_F^2 \quad (7)$$

Setting the derivative with respect to $W_v$ to zero, we have

$$\lambda_1 D_{1v} W_v + \lambda_2 D_{2v} W_v + \mu X_v (X_v^\top W_v - Y - E_v + \frac{1}{\mu}\Lambda_v) = 0$$

where $D_{1v} \in R^{l_v \times l_v}$ is a diagonal matrix, $\mathbf{w}_v^i$ is the $i$th row of $W_v$, and $D_{2v} \in$ diagonal matrix. (9)

$$D_{1v} = \begin{bmatrix} \frac{1}{2\|\mathbf{w}_v^1\|_2} & & \\ & \ddots & \\ & & \frac{1}{2\|\mathbf{w}_v^{l_v}\|_2} \end{bmatrix} \quad D_{2v} = \begin{bmatrix} \frac{1}{2\|W_v\|_F} & & \\ & \ddots & \\ & & \frac{1}{2\|W_v\|_F} \end{bmatrix}$$

According to (8), we obtain

$$W_v = \mu(\lambda_1 D_{1v} + \lambda_2 D_{2v} + \mu X_v X_v^\top)^{-1} X_v (Y + E_v - \frac{1}{\mu}\Lambda_v)$$

**Second step**: Fix $W_v$ and $\theta_v$, update $E_v$

When $W_v$ and $\theta_v$ are fixed, $E_v$ is the only variable, and its solution can be obtaine solving the following optimization problem:

$$\min_{E_v} \sum_{v=1}^{m} \left( \frac{(\theta_v)^p}{\mu} \|E_v\|_{2,1} + \frac{1}{2} \|X_v^\top W_v - Y - E_v + \frac{1}{\mu}\Lambda_v\|_F^2 \right)$$

Define $J_v = \frac{(\theta_v)^p}{\mu}\|E_v\|_{2,1} + \frac{1}{2}\|X_v^\top W_v - Y - E_v + \frac{1}{\mu}\Lambda_v\|_F^2$. Since $J_v$ is only relat the $v$th view, the problem (11) can be decomposed into $m$ sub-optimization problem

$$\min_{E_v} J_v, \; v = 1, 2, \ldots, m$$

Denote $G_v = X_v^\top W_v - Y + \frac{1}{\mu}\Lambda_v$, then (12) can be rewritten as

$$\min_{E_v} \frac{(\theta_v)^p}{\mu}\|E_v\|_{2,1} + \frac{1}{2}\|E_v - G_v\|_F^2$$

Furthermore, (13) can be converted as

$$\min_{\mathbf{e}_v^i} \sum_{i=1}^{N} \left( \frac{(\theta_v)^p}{\mu} \|\mathbf{e}_v^i\|_2 + \frac{1}{2}\|\mathbf{e}_v^i - \mathbf{g}_v^i\|_2^2 \right)$$

where $\mathbf{e}_v^i$ and $\mathbf{g}_v^i$ are the $i$th row of matrix $E_v$ and $G_v$, respectively. And according to the solution to (14) is

$$\mathbf{e}_v^i = \begin{cases} \left(1 - \frac{(\theta_v)^p}{\mu\|\mathbf{g}_v^i\|_2}\right)\mathbf{g}_v^i, & \|\mathbf{g}_v^i\|_2 > \frac{(\theta_v)^p}{\mu} \\ \mathbf{0}, & \|\mathbf{g}_v^i\|_2 \le \frac{(\theta_v)^p}{\mu} \end{cases}$$

**Third step**: Fix $E_v$ and $W_v$, update $\theta_v$

When $E_v$ and $W_v$ are fixed, $\theta_v$ is the only variable, and its solution can be obtaine solving the following problem:

$$\min_{\theta_v} \sum_{v=1}^{m} (\theta_v)^p \|E_v\|_{2,1}$$
$$\text{s.t.} \sum_{v=1}^{m} \theta_v = 1,\ 0 \le \theta_v \le 1$$

The Lagrangian function of (16) is

$$L(\theta_v, \eta) = \sum_{v=1}^{m} (\theta_v)^p \|E_v\|_{2,1} - \eta \left( \sum_{v=1}^{m} \theta_v - 1 \right)$$

where $\eta$ is the Lagrange multiplier. By setting the derivatives of (17) with respect t and $\eta$ to 0, respectively, we have

$$\begin{cases} p(\theta_v)^{p-1} \|E_v\|_{2,1} - \eta = 0 \\ \sum_{v=1}^{m} \theta_v = 1 \end{cases}$$

Thus $\theta_v$ can be obtained by the following equation:

$$\theta_v = \frac{\left( \frac{1}{\|E_v\|_{2,1}} \right)^{\frac{1}{p-1}}}{\sum_{v=1}^{m} \left( \frac{1}{\|E_v\|_{2,1}} \right)^{\frac{1}{p-1}}}$$

Thus using this we find parameters W and θv which will give us the threshold values for each feature in each view according to W matrix. There are tunable parameters like p, λ1 , λ2 etc to improve performance but this can be given by the user and accordingly results will be there.

# Chapter 3

# Algorithm

## 3.1 Feature selection Algorithm:

**Algorithm 1** The algorithm of solving the objective function (2).

**Input:** The data matrix $X$, the label matrix $Y$, the tolerance $\varepsilon = 10^{-3}$, the maximum number of iterations T = 20, $p > 1$, the positive parameters $\lambda_1$, $\lambda_2$

**Output:** Projection matrix $W$, view weight vector $\theta$

**Initialization**

Set $t = 0$, $\theta_v^0 = \frac{1}{m}$, $v = 1, \cdots, m$. Set $W^0 \in 1^{l \times c}$. Set $E^0 \in 1^{N \times c}$. Set $\Lambda^0 \in 1^{N \times c}$. Set $\mu^0$=0.1, $\zeta$=1.2

**while not converged do**

**1.** Compute the diagonal matrices $D_{1v}^t$ and $D_{2v}^t$ by (9), $v = 1, \cdots, m$

**2.** Update the projection matrix $W_v^{t+1} = \mu^t(\lambda_1 D_{1v}^t + \lambda_2 D_{2v}^t + \mu^t X_v X_v^\top)^{-1} X_v(Y + E_v^t - \frac{1}{\mu^t}\Lambda_v^t)$, $v = 1, \cdots, m$

**3.** Calculate $G_v^{t+1} = X_v^\top W_v^{t+1} - Y + \frac{1}{\mu^t}\Lambda_v^t$, $v = 1, \cdots, m$

**4.** Update the slack variable matrix $E_v^{t+1}$ by (15),$v = 1, \cdots, m$

**5.** Update the view weight $\theta_v^{t+1}$ by (19), $v = 1, \cdots, m$

**6.** Update the Lagrangian multiplier matrix $\Lambda_v^{t+1} = \Lambda_v^t + \mu^t(X_v^\top W_v^{t+1} - Y - E_v^{t+1})$, $v = 1, \cdots, m$

**7.** Update $\mu^{t+1} = \zeta\mu^t$

**8.** Calculate $er = |Obj(t+1) - Obj(t)|$, where $Obj(t) = \sum_{v=1}^m (\theta_v^t)^p \|X_v^\top W_v^t - Y\|_{2,1} + \lambda_1 \sum_{v=1}^m \|W_v^t\|_{2,1} + \lambda_2 \sum_{v=1}^m \|W_v^t\|_F$

**9.** $t = t + 1$

**Until** $er < \varepsilon$ or $t \geq$ T

# Chapter 4
# Documentation of API

## 4.1 Package organization

cca.feature_selection.RMvFS

class cca.feature_selection.RMvFS(tolerance=0.001, max_iter=20, p=10, lam1=0.01, lam2=0.1, threshold=0.01)

**Parameters**

tolerance: float, default: 0.001

controls the conversion point of algorithm. Tolerable error amount.

max_iter: int, default: 20

no of iterations algorithm can take.

p: int, default: 10

exponent power of θv

lam1: float, default: 0.01

controls the robustness by tuning on each view

lam2: float, default: 0.1

controls the robustness by tuning on each features from each view

threshold: float, default: 0.01

threshold for features selection

**Attributes**

v: int

total number of views passed

X:array, float

3d matrix contains each views' samples

Y: array, float

Sparse 2d matrix, has c columns and N rows, where c is no of classes and N is sample size. Contains 1 where there is label for particular sample else 0.

N: int

Total number of samples

lv: array, int

array of no of features of each view

theta: array,float

array of θv for each view, which controls the weight of each view

c: int

number of categories or classes which are unique in labels.

W: array, float

3d matrix which keeps projection matrix of each view.

E: array, int

3d matrix, Lagrangian Slack variable for each view.

LAMBADA:array, int

3d matrix, Lagrangian Multiplier for each view

mu: float

a penalty parameter

zeta: float

ratio of increasing 'mu' with each iteration

## 4.2 Methods

**fit(X1,*Xn,Y)**

fit model to data.

**Parameters**

X1: array, float

2d matrix, samples of all features of first view

*Xn: array of arrays, float

Array of 2d matrics which are representation of different views. Can be empty.

Y: array, float

2d matrix, array of labels for the samples.

**Returns** Nothing

**transform(X1,*Xn)**

using threshold value filters the features across different views that are to be selected.

**Parameters**

X1: array, float

2d matrix, samples of all features of first view

*Xn: array of arrays, float

Array of 2d matrics which are representation of different views. Can be empty.

**Returns** array of feature that must be selected

# Chapter 5
# Example

## 5.1 Example 1

```
from cca.feature_selection import RMvFS
X1 = [[1,2,3],[4,5,6], [7,8,9]]
Y = [‘a’,’b’,’a’]
Rmvfs = RMvFS()
Rmvfs.fit(X1,Y)
X1 = Rmvfs.transform(X1)
```

# Chapter 6

# Learning Outcome

6.1 Norms Regularization:

There various norms like $G_{2,1,}$ $l_{2,1,}$ L1 and L2 reguarization. These help in calculating the penalty terms. The standard norms are implemented in python's numpy.linalg library. G-norm is calculated over full matrix, L-norm is calculated over rows and L1 and L2 is calculated over rows also.

6.2 Feature Selection:

Feature selection works using a threshold value over some matrix( like projection matrix here ) which filters the most contributing features.

6.3 Fit and Transform methods:

How different models' fit transform works and how fit methods calculate some parameters.

6.4 Augmented Lagrangian Multiplier:

Its use in optimizing any function and use of slack variables.

6.5 Regularization:

Regularization is helpful in preventing the model from over-fitting. Here we used this concept as finding the best features among many.

# References

[1] https://link.springer.com/article/10.1007%2Fs11042-020-09617-8