MFDS Report

Load the data

```
z = readtable('IPL_Twitter_MissingData.csv');
data = table2array(z);
data_old = data;
data_new = [];
for i = 1:1000
    x = isnan(data(i,:)) ;
    if sum(x) ==0
        data_new = [data_new; data(i,:)];
    end
end
```

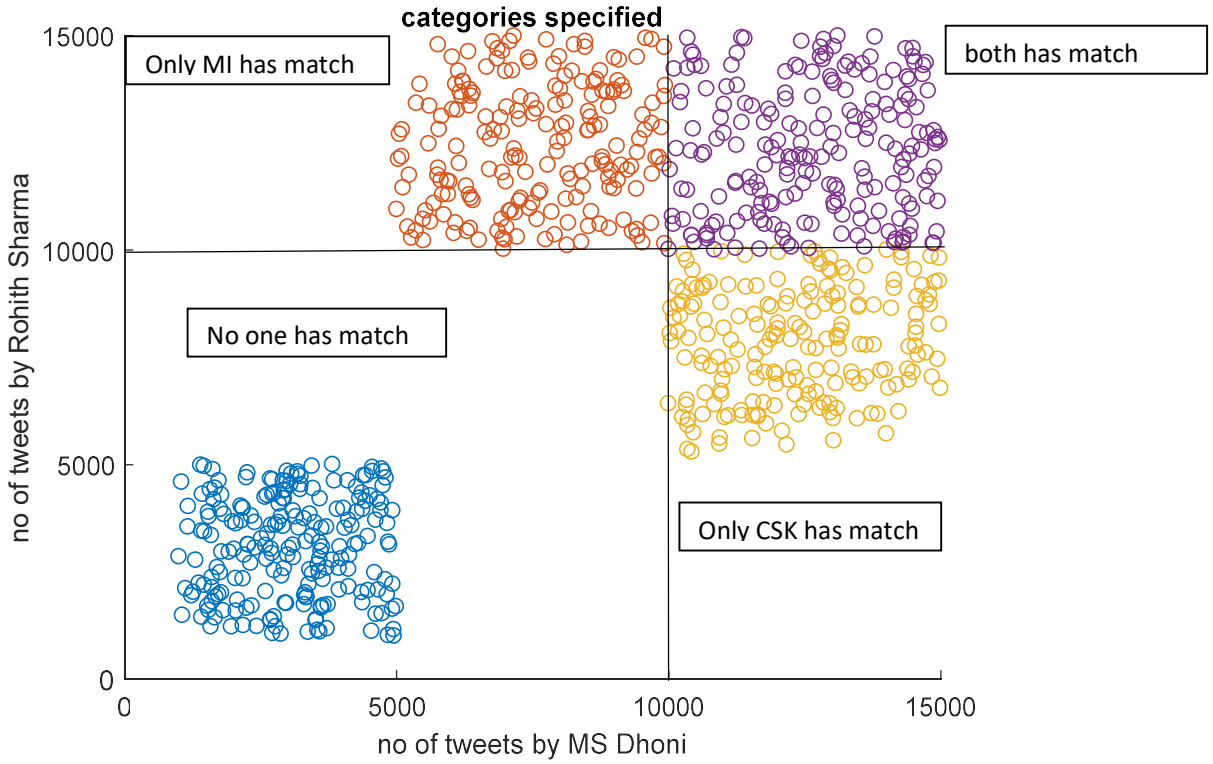
Part-A:

```
disp('Data samples have missing values is given by shape of the data_new matrix')
disp('No of missing samples is')
n = 1000 - size(data_new,1);
disp(n)
```

```
Data samples have missing values is given by shape of the data_new matrix
No of missing samples is = 238
```

Part-B:

Data visualization



MFDS Report

Sol: There are four categories in the data

1) When both MI and CSK has the match,
2) When both MI and CSK do not have the match,
3) When MI has the match and CSK do not have the match,
4) When CSK has the match and MI do not have the match.

Data in different categories

Cat_1: When there is no match for both CSK and MI

Cat_2: When there is no match for CSK but only for MI

Cat_3: When there is no match for MI but only for CSK

Cat_4: When there is match between CSK and MI

```
cat_1 = [];
cat_2 = [];
cat_3 = [];
cat_4 = [];
for a = 1:size(data_new,1)
    if data_new(a,1) == 0 && data_new(a,2) ==0
        cat_1 = [cat_1; data_new(a,3:6)];
    end
    if data_new(a,1) == 0 && data_new(a,2) ==1
        cat_2 = [cat_2; data_new(a,3:6)];
    end
    if data_new(a,1) == 1 && data_new(a,2) ==0
        cat_3 = [cat_3; data_new(a,3:6)];
    end
    if data_new(a,1) == 1 && data_new(a,2) ==1
        cat_4 = [cat_4; data_new(a,3:6)];
    end
end
maximum = [max(cat_1);max(cat_2);max(cat_3);max(cat_4)];
minimum = [min(cat_1);min(cat_2);min(cat_3);min(cat_4)];
```

Part-C(3a):

```
Z = data_new(:,3:6);
[sol_a,b] = TLS(Z);
disp('Linear regression including the intercept:')
disp('My regression model is a1X1 + a2X2 + a3X3 +a4X4 = b where')
fprintf('a1 : %6.4f\n',sol_a(1,1))
fprintf('a2 : %6.4f\n',sol_a(2,1))
fprintf('a3 : %6.4f\n',sol_a(3,1))
fprintf('a4 : %6.4f\n',sol_a(4,1))
fprintf('b : %6.4f\n',b)
```

```
Linear regression including the intercept:
My regression model is a1X1 + a2X2 + a3X3 +a4X4 = b where
a1 : -0.8700
a2 : -0.1231
```

```
a3 : 0.1201
a4 : 0.4621
b : -1287.3468
```

Part-C(3b):

```
[sol_b,b_b] = TLS(cat_1);
disp('Linear regression including the intercept:')
disp('My regression model for cat_1 is a1X1 + a2X2 + a3X3 +a4X4 = b where')
fprintf('a1 : %6.4f\n',sol_b(1,1))
fprintf('a2 : %6.4f\n',sol_b(2,1))
fprintf('a3 : %6.4f\n',sol_b(3,1))
fprintf('a4 : %6.4f\n',sol_b(4,1))
fprintf('b : %6.4f\n',b_b)
```

```
Linear regression including the intercept:
My regression model for cat_1 is a1X1 + a2X2 + a3X3 +a4X4 = b where
a1 : 0.4436
a2 : 0.6337
a3 : -0.5070
a4 : -0.3802
b : 0.0313
```

Part-D(3c):

```
[sol_c,b_c]= TLS(cat_2);
disp('Linear regression for cat_2 including the intercept:')
disp('My regression model is a1X1 + a2X2 + a3X3 +a4X4 = b where')
fprintf('a1 : %6.4f\n',sol_c(1,1))
fprintf('a2 : %6.4f\n',sol_c(2,1))
fprintf('a3 : %6.4f\n',sol_c(3,1))
fprintf('a4 : %6.4f\n',sol_c(4,1))
fprintf('b : %6.4f\n',b_c)
```

```
Linear regression for cat_2 including the intercept:
My regression model is a1X1 + a2X2 + a3X3 +a4X4 = b where
a1 : 0.1691
a2 : -0.6765
a3 : 0.2368
a4 : 0.6765
b : -0.1316
```

Part-D(3d):

```
[sol_d,b_d]= TLS(cat_3);
disp('Linear regression for cat_3 including the intercept:')
disp('My regression model is a1X1 + a2X2 + a3X3 +a4X4 = b where')
fprintf('a1 : %6.4f\n',sol_d(1,1))
fprintf('a2 : %6.4f\n',sol_d(2,1))
fprintf('a3 : %6.4f\n',sol_d(3,1))
fprintf('a4 : %6.4f\n',sol_d(4,1))
fprintf('b : %6.4f\n',b_d)
```

```
Linear regression for cat_3 including the intercept:
My regression model is a1X1 + a2X2 + a3X3 +a4X4 = b where
a1 : -0.2292
a2 : -0.3057
a3 : 0.8024
a4 : -0.4585
b : 0.1166
```

Part-D(3e):

```
[sol_e,b_e]= TLS(cat_4);
disp('Linear regression for cat_4 including the intercept:')
disp('My regression model is a1X1 + a2X2 + a3X3 +a4X4 = b where')
fprintf('a1 : %6.4f\n',sol_e(1,1))
fprintf('a2 : %6.4f\n',sol_e(2,1))
fprintf('a3 : %6.4f\n',sol_e(3,1))
fprintf('a4 : %6.4f\n',sol_e(4,1))
fprintf('b : %6.4f\n',b_e)
```

```
Linear regression for cat_4 including the intercept:
My regression model is a1X1 + a2X2 + a3X3 +a4X4 = b where
a1 : -0.5334
a2 : -0.4103
a3 : 0.4103
a4 : 0.6155
b : 0.1231
```

Note: When CSK and MI has the match, see the regression coefficients, they are almost close.

Final summary of results for different data sets are:

Model is of the form:

a1X1 + a2X2 + a3X3 + a4X4 = b(constant term)

Cases	Coefficient of X1	Coefficient of X2	Coefficient of X3	Coefficient of X4	Constant term
Case-a	0.8700	0.1231	-0.1201	-0.4621	-1287.3
Case-b	-0.4436	-0.6337	0.5070	0.3802	-0.0313
Case-c	0.1691	-0.6765	0.2368	0.6765	-0.1316
Case-d	-0.2292	-0.3507	0.8024	-0.4585	0.1166
Case-e	-0.5334	-0.4103	0.4103	0.6155	0.1231

Part-D: Impute missing values using algorithm

Implementation technique:

MFDS Report

In imputing the data, we can see from the data there are two cases.

Case-1: When both the columns Q1 and Q2 are completely filled

Step-1: Search among the four categories mentioned in the Part-b

Step-2: Then search for missing values in X1, X2, X3, X4

Step-3: Wherever the value is missing, missing value is filled by following way

Value = (max + min)/2

Note: We can use a standard normal distribution with (max + min)/2 as mean with variance of 10% mean

Case-2: When all the X1, X2, X3, X4 are given, then filling back the columns Q1,Q2

Step-1: First pick each data sample

Step-2: Then verify whether your sample filled with X1 to X4

Step3: Check which column among Q1 and Q2 has missing the value

Step-4: If Q1 has the missing value that means we need to confirm whether CSK has match or not, we can do by seeing the no of tweets on M S Dhoni that day, so check column X2

Step-5: So, you need to check whether tweets are greater than 1000(found from max matrix)

Step-6: If Q2 has the missing value that means we need to confirm whether MI has match or not, we can do by seeing the no of tweets on Rohit Sharma that day, so check column X3

Step-7: If Q1 and Q2 have the missing values then we need to check the no of tweets on M S Dhoni and Rohit Sharma, so check both the X2 and X3 columns

Note: Threshold for no of tweets to exceed is 1000 found from the max and min matrix at those instances

The resulting data matrix (imputed) is stored in "data.csv" file.

```
%Case-1: Which is when Both Q1 and Q2 are filled
for i = 1:1000
    p = double(isnan(data(i,:)));
    if p(1,1)==0 && p(1,2)==0
        if data(i,1)==0 && data(i,2)==0
            for j = 3:6
                if p(1,j)==1
                    data(i,j) = (maximum(1,j-2)+minimum(1,j-2))/2;
                end
            end
        end
        if data(i,1)==0 && data(i,2)==1
            for j = 3:6
                if p(1,j)==1
                    data(i,j) = (maximum(2,j-2)+minimum(2,j-2))/2;
                end
            end
        end
        if data(i,1)==1 && data(i,2)==0
            for j = 3:6
```

```
                if p(1,j)==1
                    data(i,j) = (maximum(3,j-2)+minimum(3,j-2))/2;
                end
            end
        end
        if data(i,1)==1 && data(i,2)==1
            for j = 3:6
                if p(1,j)==1
                    data(i,j) = (maximum(4,j-2)+minimum(4,j-2))/2;
                end
            end
        end
    end
end

%Case-2 when all X1, X2, X3, X4, are given
for i = 1:1000
    q = double(isnan(data(i,:)));
    if q(1,3:6) == 0
        if q(1,1)==1 && q(1,2) == 0
            if data(i,4) >= 10000
                data(i,1) = 1;
            else
                data(i,1) = 0;
            end
        end
        if q(1,1)==0 && q(1,2) == 1
            if data(i,5) >= 10000
                data(i,2) = 1;
            else
                data(i,2) = 0;
            end
        end
        if q(1,1)==1 && q(1,2) == 1
            if data(i,4) >= 10000
                data(i,1) = 1;
            else
                data(i,1) = 0;
            end
            if data(i,5) >= 10000
                data(i,2) = 1;
            else
                data(i,2) = 0;
            end
        end
    end
end
```

Function to be used to find regression coefficients in Part-C

```
function [solution,lin_coeff] = TLS(z)
zs = z - mean(z);
% cov_zs = cov(zs);
% [vec,eigen] = eig(cov_zs);
[u,s,v] = svd(zs,'econ');
solution = v(:,end);
```

```
lin_coeff = sum(solution.*mean(z)');
end
```

Published with MATLAB® R2018b

Problem 2:

Introduction :

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labelled training data (*supervised learning*), **the algorithm outputs an optimal hyperplane which categorizes** new examples. In two dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.

Use of Kernels :

Kernel functions offer the user this option of **transforming nonlinear spaces into linear ones**. Most packages which offer SVM will include several non linear kernels ranging from simple polynomial basis functions to sigmoid functions.

Types :

Polynomial kernel

It is popular in image processing.
Equation is:

$$k(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j} + 1)^d$$

Polynomial kernel equation

where d is the degree of the polynomial.

Linear kernel

Equation is:

$$k(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j} + 1)$$

Linear kernel equation

where d is the degree of the polynomial.

Gaussian kernel

It is a general-purpose kernel; used when there is no prior knowledge about the data.
Equation is:

$$k(x, y) = \exp\left(-\frac{\|x -$$

Gaussian kernel equation

Problem 2 Solution :

Data representation :

The "**q2_data_matrix.csv**" is read into a data frame and then converted to a data matrix the data matrix is split into train "**X_train** ". and validation data "**X_valid** ". In the split ratio of 70:30 .

Columns of X_train represent age, transaction amount, total monthly transactions, annual income, gender respectively. Each row represents a sample.

The "**q2_labels.csv**" is read into a column vector called "**Y_train**" and "**Y_valid**".

Results obtained :

(i) Linear Kernel :

Confusion matrix for train data:

Train value\prediction	Pred = 0	Pred = 1
Y_train = 0	416	27
Y_train = 1	31	226

F1 score for train data:

Y = 0	0.93
Y = 1	0.89
average	0.92

Confusion matrix for validation data:

Train value\prediction	Pred = 0	Pred = 1
Y_train = 0	169	22
Y_train = 1	19	90

F1 score for validation data:

Y = 0	0.89
Y = 1	0.81
average	0.86

Polynomial Kernel :

Confusion matrix for train data:

Train value\prediction	Pred = 0	Pred = 1
Y_train = 0	420	24
Y_train = 1	30	227

F1 score for train data:

Y = 0	0.90
Y = 1	0.82
average	0.87

Confusion matrix for validation data:

Train value\prediction	Pred = 0	Pred = 1
Y_train = 0	129	5
Y_train = 1	18	48

F1 score for validation data:

Y = 0	0.92
Y = 1	0.81
average	0.89

Rbf Kernel :

Confusion matrix for train data:

Train value\prediction	Pred = 0	Pred = 1
Y_train = 0	422	21

Y_train = 1	31	226

F1 score for train data:

Y = 0	0.94
Y = 1	0.90
average	0.93

Confusion matrix for validation data:

Train value\prediction	**Pred = 0**	**Pred = 1**
Y_train = 0	176	15
Y_train = 1	19	90

F1 score for validation data:

Y = 0	0.91
Y = 1	0.84
average	0.89

(ii) Accuracies obtained for various kernels used :

Kernel used	Train accuracy	Validation accuracy
Linear	91.7%	86.3%
Poly	90.2%	87.8%
RBF	92.6%	88.6%

RBF kernel is better.

Conclusions :

- The **RBF kernel is better**. On the basis of validation data used.

Observations and Assumptions :

- It takes a lot of time to run svm using polynomial kernel because of its huge complexity.