

# CS-E3210

## Machine Learning :Basic Principle

**Group members:** Haseeb Shehzad(601483), Naufal Khalid (546687)

**Report:** Term Project

**Teacher:** Alex Jung

**Date:** 25 November, 2016

# Yelp review prediction using different classification and regression techniques

## **1. Abstract:**

In this project we try to find out the prediction of Yelp review dataset using different classification and regression algorithms. We would apply the different classification and regression techniques to predict the Yelp reviews. In regression problem we predict the number of effective votes the review got characterised by different words. In classification we indicate that whether the review got the desired rating or not. We implement the methods and state our findings. We conclude the explanations of the results and improvements.

## **2. Introduction:**

In this term project, we are given a set of 5000 review for regression. A set of 50 keywords selected by bag of words model are given from the reviews and our task is to predict the number of effective and useful votes the review got and how many people find it useful. In classification we have to determine if the user gave the 3 stars(1) or not(0) from the given binary output data. We apply different machine learning method for the purpose of method evaluation choose the best method. We train the methods with the given data set and test it and find results by applying the additional test data.

We visualise the data, understand the features and form different hypothesis and select the best methods. We calculate and explain the errors obtained in the process.

### 3. Methods:

#### 3.1 Regression:

##### 3.1.1 Multivariate Regression:

Multivariate regression used when we have more features more than one. In Multivariate regression we used the polynomial of different degrees. In any data analysis challenge goal is to extract information or prediction from the raw data and learn input to output relation.

$$\mathbf{x} \rightarrow r$$

We approximate the output by predictor

$$g(\mathbf{x}|\mathbf{w}) = \mathbf{w}^T \mathbf{x}$$

$$\text{where } r = g(\mathbf{x}|\mathbf{w}) + \varepsilon = \mathbf{w}^T \mathbf{x} + \varepsilon$$

We choose the best  $\mathbf{w}$  by minimising the empirical error  $\varepsilon$ .

$$\mathbf{w}_{opt} = \underset{\mathbf{w}}{\operatorname{argmin}} E(\mathbf{w}|\mathcal{D}) = \sum_{t=1}^N (r^t - \mathbf{w}^T \mathbf{x}^t)^2$$

By taking derivative of  $E(\mathbf{w}|\mathcal{D})$  and putting it equal to zero we get the orthogonal condition of

$$\mathbf{X}^T (\mathbf{r} - \underbrace{\mathbf{X} \mathbf{w}_{opt}}_{\mathbf{g}_{opt}}) = \mathbf{0}$$

##### 3.1.2 K-d trees

This is nonparametric method for regression. Regression trees grow similar to the classification trees the different is the learning criteria for splitting the nodes.

Error incurred by leaf is

$$\mathcal{E}_m = (1/N_m) \sum_{\mathbf{x}^t \in \mathcal{D}_m} (r^t - g_m)^2$$

error after split the leaf,

$$\mathcal{E}_s = (1/N_m) \sum_{\mathbf{x}^t \in \mathcal{D}_{m,y}} (r^t - g_{m,y})^2 + (1/N_m) \sum_{\mathbf{x}^t \in \mathcal{D}_{m,n}} (r^t - g_{m,n})^2$$

information gained by converting leaf  $m$  to decision node is

$$\text{Gain}(m \rightarrow j) := \underbrace{\mathcal{H}(\mathcal{D}_m)}_{\text{entropy of leaf } m} - \underbrace{\left( \frac{|\mathcal{D}_{m,y}|}{|\mathcal{D}_m|} \mathcal{H}(\mathcal{D}_{m,y}) + \frac{|\mathcal{D}_{m,n}|}{|\mathcal{D}_m|} \mathcal{H}(\mathcal{D}_{m,n}) \right)}_{\text{entropy of leaf nodes } m_y \text{ and } m_n}$$

## 3.2 Classification:

### 3.2.1 Multivariate Gaussian Distribution:

We assume that the data follow the Gaussian Probability distribution. It is actually the generalisation of one dimensional normal distribution to higher dimensions.

Probability density function is given as

$$p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2} |\boldsymbol{\Sigma}|^{1/2}} \exp \left( -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right)$$

The mean vector  $\boldsymbol{\mu}$  is the expectation of  $\mathbf{x}$ :

$$\boldsymbol{\mu} = \mathbb{E}[\mathbf{x}]$$

The covariance matrix  $\boldsymbol{\Sigma}$  is the expectation of the deviation of  $\mathbf{x}$  from the mean:

$$\boldsymbol{\Sigma} = \mathbb{E}[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T]$$

### Logistic Regression:

This technique involves the probabilistic view of classification. we consider a two-outcome probability space.

Parametrisation of  $P(r = 1 | \mathbf{x}; \mathbf{w})$

$$\log \frac{P(r = 1 | \mathbf{x}; \mathbf{w})}{P(r = 0 | \mathbf{x}; \mathbf{w})} = \log \frac{P(r = 1 | \mathbf{x}; \mathbf{w})}{1 - P(r = 1 | \mathbf{x}; \mathbf{w})} := \mathbf{w}^T \mathbf{x}$$

Optimum weight  $\mathbf{w}$  maximises conditional log likelihood

$$\begin{aligned} \log P(\{r^t\} | \{\mathbf{x}^t\}; \mathbf{w}) &= \log \prod_{t=1}^N P(r^t | \mathbf{x}^t; \mathbf{w}) \\ &= \sum_{t=1}^N (r^t \log y^t + (1 - r^t) \log (1 - y^t)) \end{aligned}$$

logistic regression amounts to solving

$$\mathbf{w}_{opt} = \underset{\mathbf{w} \in \mathbb{R}^d}{\operatorname{argmin}} E(\mathbf{w}|\mathcal{D})$$

We start the initial guess  $\mathbf{w}_0$  for optimal weight and iteratively update the guess by this equation

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \alpha \nabla E(\mathbf{w}_k|\mathcal{D})$$

## 4. Experiments

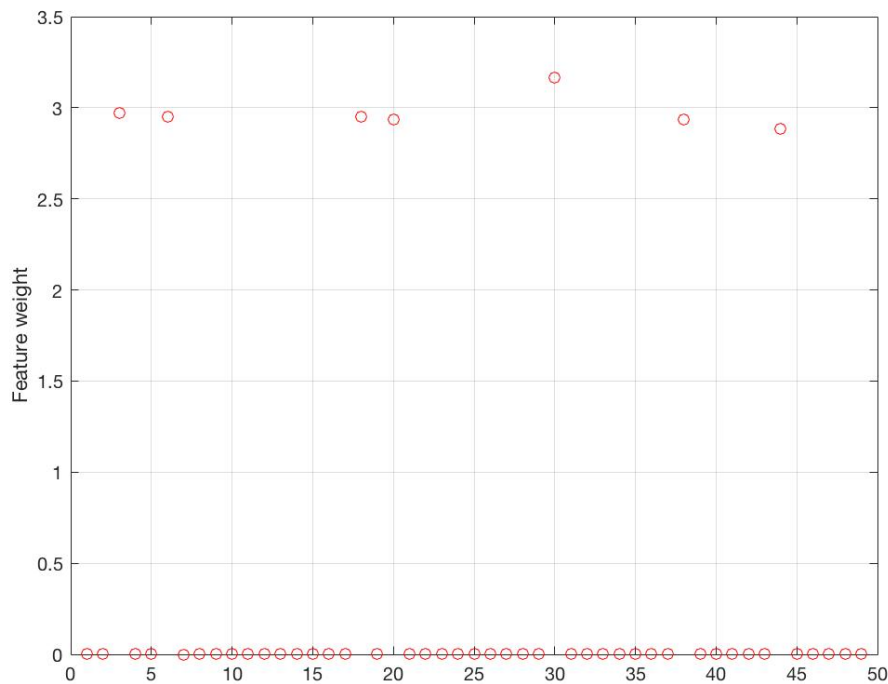
First step in data analysis is the data visualisation. We try to find any patterns if possible. Another important step is to select the features if we have more features. In Feature selection we reduce the dimensionality of data and by transforming data into new features.

### 4.1 Neighbourhood Component Analysis:

The feature selection technique learns the feature weights using a diagonal adaptation of neighbourhood components analysis (NCA). The goal is to find a weighting vector  $\mathbf{w}$  that lends itself to select the feature

subset optimising nearest neighbour classification.

In figure (4.a) plot is given between feature index and feature weight. The higher the weight, we should select the feature for the classification. We can see that the features 1, 11, 21, 23, 35, 41, 46, 47, 48 have more weight and the in determining the class they are more. Fig. 4.1.a



important than other features. When applied to the classification with above mentioned features against all the features there was not much difference. This feature selection actually gives us the effective features on which the data is depending.

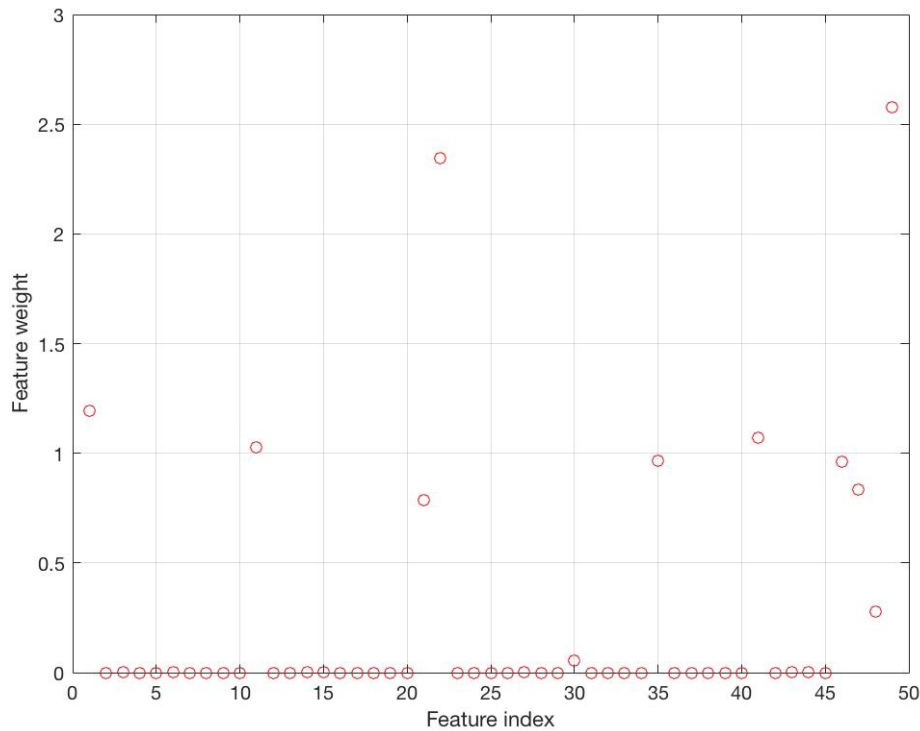


Fig. 4.1.b

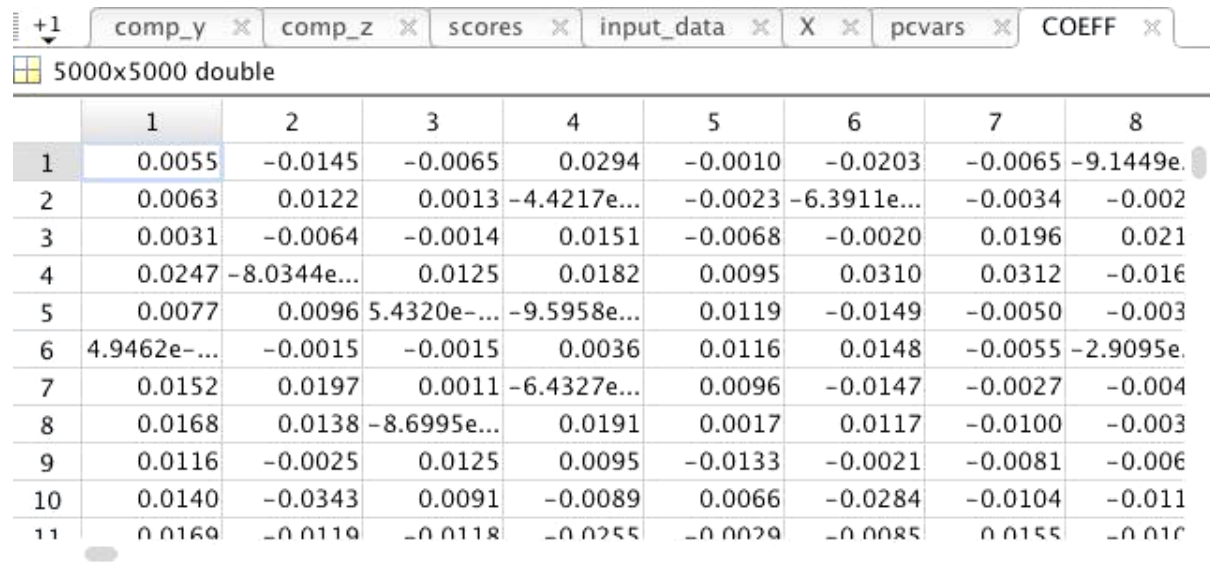
We also apply the NCA in the regression data and find the features 3,6,18,21,30,38,44 with the highest weight. (fig 4.b)

## 4.2 Principle Component Analysis (PCA) :

Principal component analysis is a technique, used for feature extraction to analyse and visualise data having multivariate features. We select the features that form driving forces to govern the behaviour of the system. This method generates the new sets of components which form principal components.

We used the function in MATLAB “`princomp(X')`”. It takes the training data as an argument and extract principal component features. We take the transpose of the input data matrix. It generates the three outputs [COEFF, scores, pcvars ].

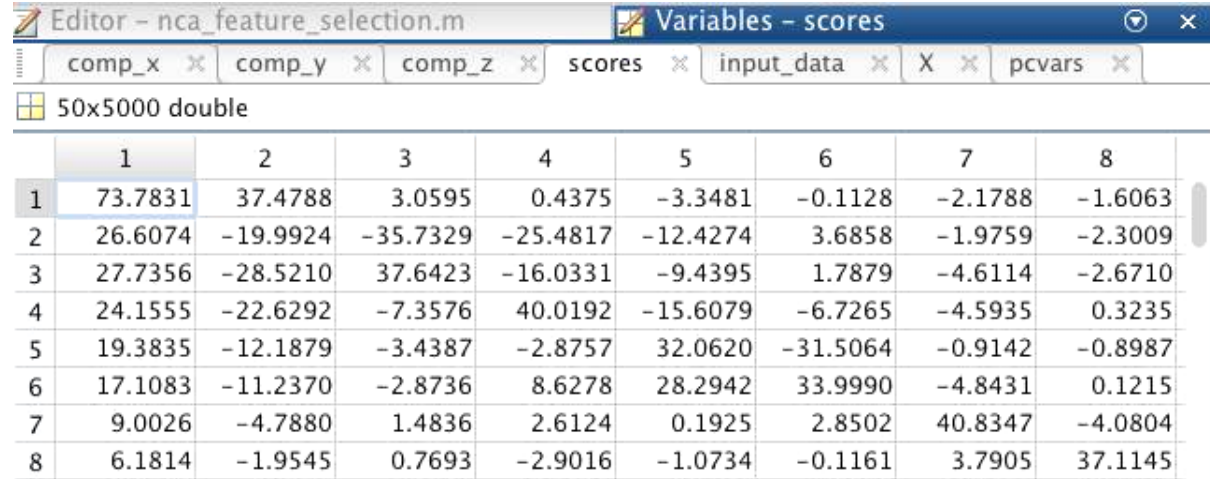
COEFF is a matrix where each column represent coefficients for one principal component in the order of decreasing components of variance. This tells us how are the features correlated.



	1	2	3	4	5	6	7	8
1	0.0055	-0.0145	-0.0065	0.0294	-0.0010	-0.0203	-0.0065	-9.1449e...
2	0.0063	0.0122	0.0013	-4.4217e...	-0.0023	-6.3911e...	-0.0034	-0.002
3	0.0031	-0.0064	-0.0014	0.0151	-0.0068	-0.0020	0.0196	0.021
4	0.0247	-8.0344e...	0.0125	0.0182	0.0095	0.0310	0.0312	-0.016
5	0.0077	0.0096	5.4320e-...	-9.5958e...	0.0119	-0.0149	-0.0050	-0.003
6	4.9462e-...	-0.0015	-0.0015	0.0036	0.0116	0.0148	-0.0055	-2.9095e...
7	0.0152	0.0197	0.0011	-6.4327e...	0.0096	-0.0147	-0.0027	-0.004
8	0.0168	0.0138	-8.6995e...	0.0191	0.0017	0.0117	-0.0100	-0.003
9	0.0116	-0.0025	0.0125	0.0095	-0.0133	-0.0021	-0.0081	-0.006
10	0.0140	-0.0343	0.0091	-0.0089	0.0066	-0.0284	-0.0104	-0.011
11	0.0169	-0.0119	-0.0118	-0.0255	-0.0029	-0.0085	0.0155	-0.016

Fig.4.2.a

The scores matrix corresponds to the scores of the principal components



	1	2	3	4	5	6	7	8
1	73.7831	37.4788	3.0595	0.4375	-3.3481	-0.1128	-2.1788	-1.6063
2	26.6074	-19.9924	-35.7329	-25.4817	-12.4274	3.6858	-1.9759	-2.3009
3	27.7356	-28.5210	37.6423	-16.0331	-9.4395	1.7879	-4.6114	-2.6710
4	24.1555	-22.6292	-7.3576	40.0192	-15.6079	-6.7265	-4.5935	0.3235
5	19.3835	-12.1879	-3.4387	-2.8757	32.0620	-31.5064	-0.9142	-0.8987
6	17.1083	-11.2370	-2.8736	8.6278	28.2942	33.9990	-4.8431	0.1215
7	9.0026	-4.7880	1.4836	2.6124	0.1925	2.8502	40.8347	-4.0804
8	6.1814	-1.9545	0.7693	-2.9016	-1.0734	-0.1161	3.7905	37.1145

Fig.4.2.b

Pcvars matrix represent the variances of the principal components which is in descending order. It signifies the most influential features in the data. This matrix also tells us the percentages by which the principal components affect the data. We have calculated that using the formula

$$**pcvars(row)/sum(pcvars)*100$$

Fig.4.2.c

+1 comp_y x comp_z x scores x input_data x X x pcvars x COEFF x								
5000x1 double								
	1	2	3	4	5	6	7	8
1	196.2657							
2	72.9441							
3	56.8836							
4	53.2654							
5	47.9677							
6	45.4650							
7	36.5648							
8	30.3562							
9	29.2010							
10	21.6119							
11	18.0662							

We pick different columns in the score matrix to plot the graph which signifies the features that are principal drivers of the system.

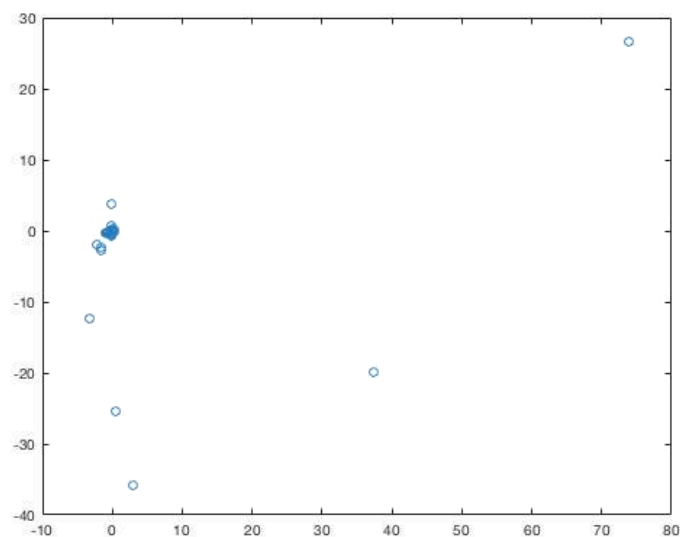
```
[COEFF,scores,pcvars] = princomp(X');
```

```
scores = scores';
```

```
comp_x = scores(:,2);
```

```
comp_y = scores(:,5);
```

```
plot(comp_x,comp_y,'o');
```



Changing the column give representations in the graph

Fig.4.2.d

Selecting feature 1 and 2



Selecting feature 2 and 3

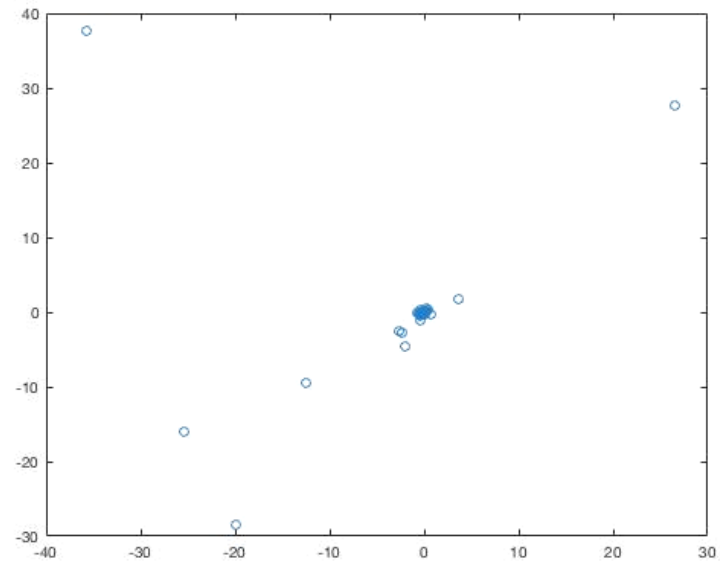


Fig.4.2.e

Selecting feature 3 and 5

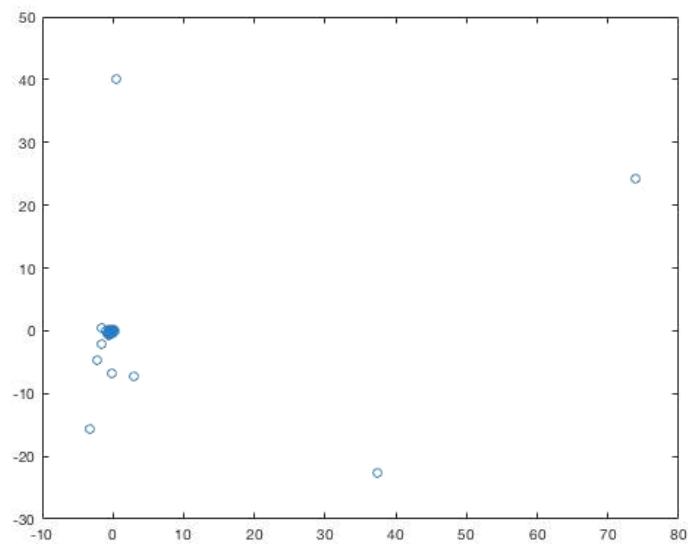


Fig.4.2.f

Selecting feature 1 and 5

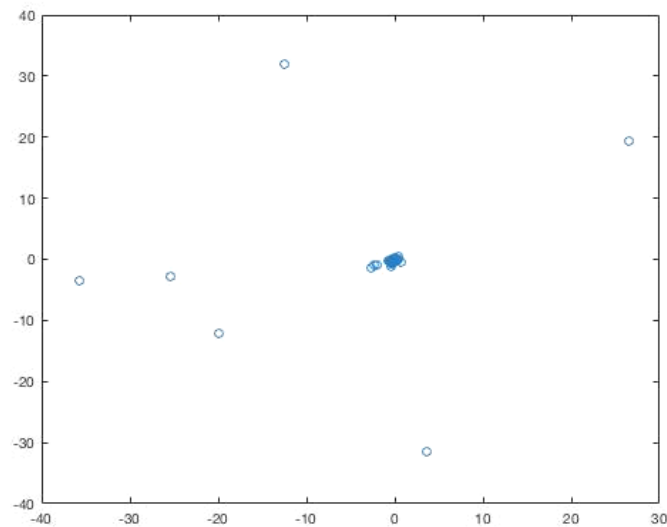


Fig.4.2.g

The most effective features were the top features and when we selected some random features down in the matrix the plot remain the same.

PCA is not effective in our case because the data is not following normal distribution and the features are not correlated with each other

### 4.3 Polynomial curve fitting

In polynomial regression we try to explain our data on different degree polynomials. We select the 30% data as validation set 70% as training set. We select the degree 5 polynomial with the highest accuracy. We also consider the concept of overfitting and under fitting and select the polynomial carefully.

We try the validation technique of leave one out and select the 3 degree polynomial by trying different degrees.

### 4.4 K-d Trees

We use the Matlab function 'fitrtree' that returns a regression tree given some features.

We can control the depth of trees using the MaxNumSplits, MinLeafSize, or MinParentSize name-value pair parameters. We can build the shallow tree to reduce the complexity.

We used many validation methods ( 'CrossVal', 'CVPartition', 'Holdout', 'KFold' ) but there was not so much difference in the results. Also 'prun' did not affect the results much.

#### 4.5 Multivariate Gaussian Distribution:

We used this method for classification. it involves finding the maximum likelihood function. After finding it we calculate the mean and variance those are class specific.

Ee find the posterior probability

$$P(C | \mathbf{x}) = \frac{P(\mathbf{x} | C) \times P(C)}{P(\mathbf{x})}$$

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}.$$

The accuracy was very low. We would discuss the results in the results portion of this report. It means that maybe the data don't belong to the gaussian distribution maybe. We used the distribution fitting tool to check the distribution but the data don't to any specific distribution. Fig(4.4.b, 4.4.c)

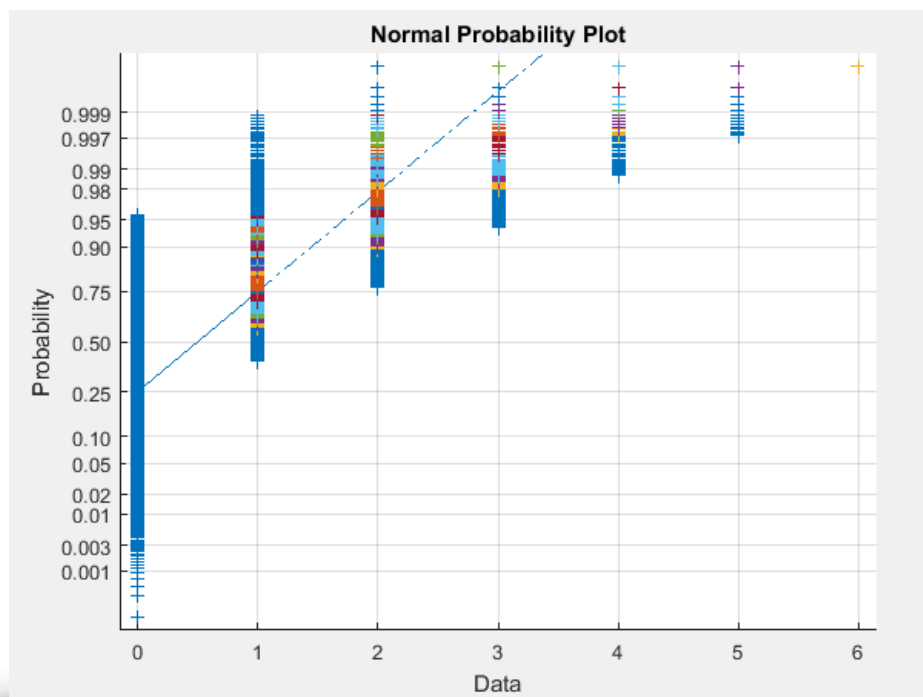
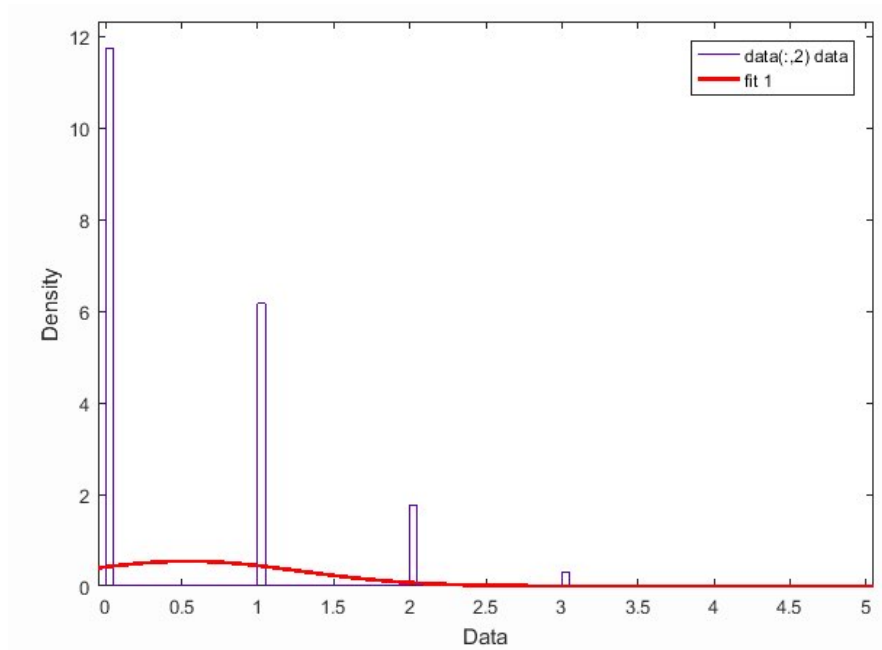
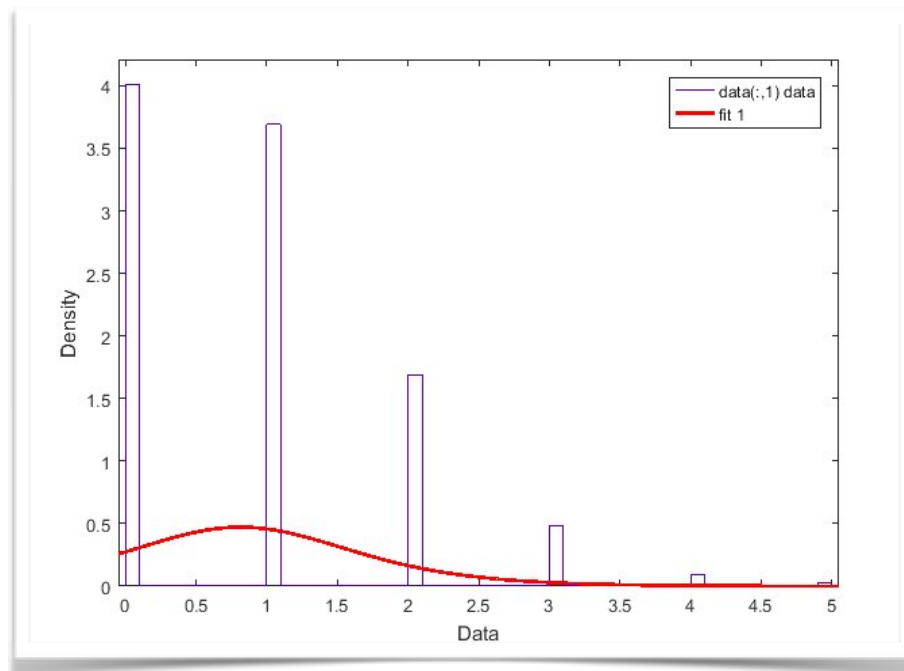


Fig 4.5.a



4.5.b (Feature 2 normal probability distribution plot)



4.5.c (Feature 1, normal probability distribution plot)

## 4.6 Logistic Regression:

We use the logic function in MATLAB for logistic regression. By using this function we find the the co-efficient for logistic regression. We use the k cross validation, leave one out and assess the method. The figure below shows the **K** against accuracy.

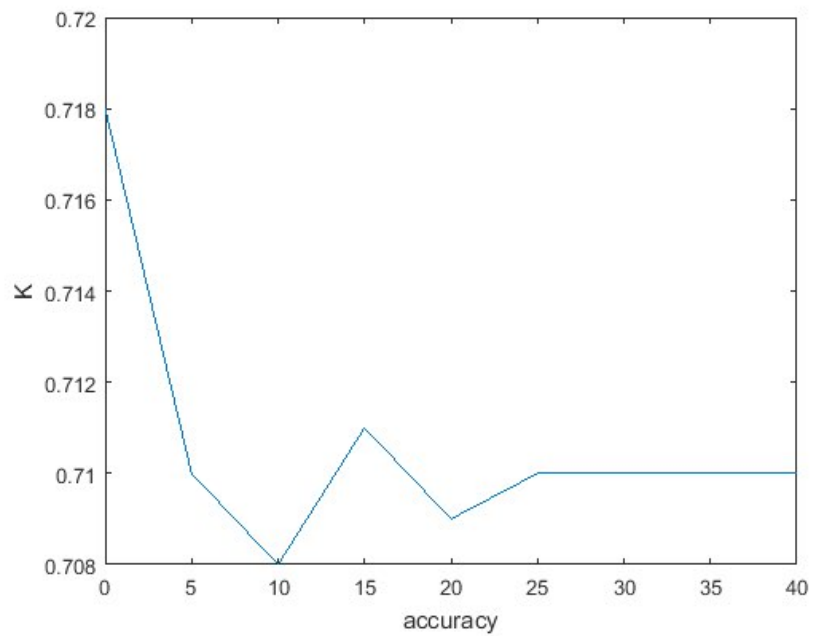


Fig 4.6.a. K against accuracy

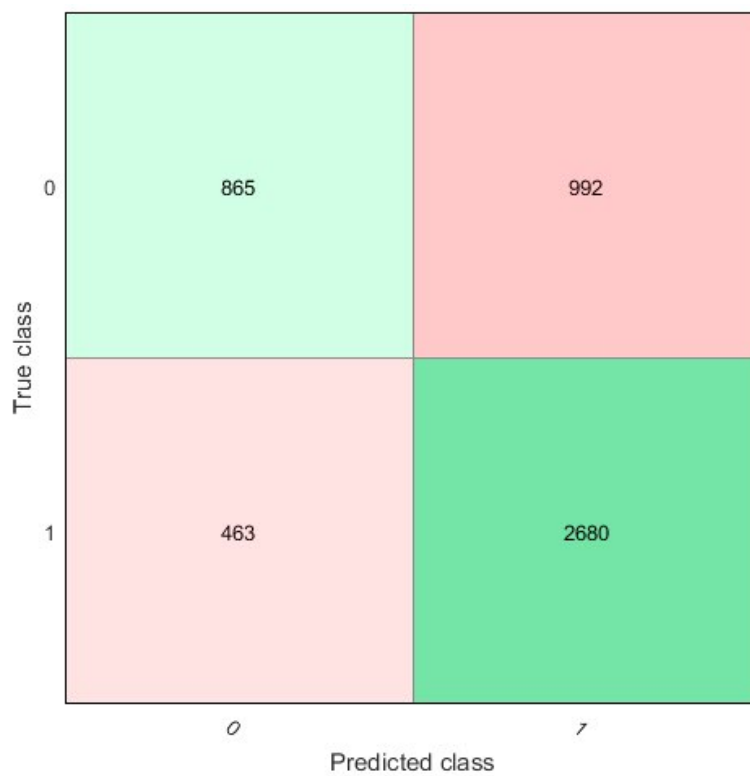


Fig 4.6.b Confusion Matrix with k-fold k=10

## 5 Results:

### 5.1 Regression:

#### 5.1.1 Multivariate Regression

Accuracy on Test Data	Accuracy on Validation Set
83%	93%

#### 5.1.2 K-D Trees:

Accuracy on Test Data	Accuracy on Validation Set
74%	78%

### 5.2 Classification:

#### 5.2.1 Multivariate Gaussian Distribution:

```
Label: ''
Description: ''
ClassLabels: [2×1 double]
GroundTruth: [1000×1 double]
NumberOfObservations: 1000
ControlClasses: 2
TargetClasses: 1
ValidationCounter: 1
SampleDistribution: [1000×1 double]
ErrorDistribution: [1000×1 double]
SampleDistributionByClass: [2×1 double]
ErrorDistributionByClass: [2×1 double]
CountingMatrix: [3×2 double]
CorrectRate: 0.6420
ErrorRate: 0.3580
LastCorrectRate: 0.6420
LastErrorRate: 0.3580
InconclusiveRate: 0
ClassifiedRate: 1
Sensitivity: 0.0457
Specificity: 0.9952
PositivePredictiveValue: 0.8500
NegativePredictiveValue: 0.6378
PositiveLikelihood: 9.5663
NegativeLikelihood: 0.9589
Prevalence: 0.3720
DiagnosticTable: [2×2 double]
```

### 5.2.2 Logistic Regression:

```
Label: ''
Description: ''
ClassLabels: [2×1 double]
GroundTruth: [1000×1 double]
NumberOfObservations: 1000
ControlClasses: 2
TargetClasses: 1
ValidationCounter: 1
SampleDistribution: [1000×1 double]
ErrorDistribution: [1000×1 double]
SampleDistributionByClass: [2×1 double]
ErrorDistributionByClass: [2×1 double]
CountingMatrix: [3×2 double]
CorrectRate: 0.7160
ErrorRate: 0.2840
LastCorrectRate: 0.7160
LastErrorRate: 0.2840
InconclusiveRate: 0
ClassifiedRate: 1
Sensitivity: 0.4919
Specificity: 0.8487
PositivePredictiveValue: 0.6583
NegativePredictiveValue: 0.7382
PositiveLikelihood: 3.2520
NegativeLikelihood: 0.5986
Prevalence: 0.3720
DiagnosticTable: [2×2 double]
```

## 6. Discussions/Conclusions:

The results of the methods that we applied for regression are more accurate than classification. In classification we assume that the data follow the gaussian distribution but the error is 35% which is not very good so we conclude that the data don't follow the distribution. In classification the majority class belongs to the 1, and the accuracy is 62%.

In regression we got pretty much good result with the multivariate regression than the k-d trees. We also confirm the fact that increasing the degree of polynomial we get the less validation error up to certain degree after that the result increases.

## **7. Reference list:**

<https://se.mathworks.com/help/stats/fitrtree.html>

<https://se.mathworks.com/help/stats/classregtree.html>

<https://se.mathworks.com/help/stats/confusionmat.html>

<https://se.mathworks.com/help/stats/princomp.html>

[Introduction to Machine Learning 2nd Edition, Ethem Alpaydin](#)

[Lecture Notes](#)