## **Satyam Sharma**

## **Student#300145438**

Question 1) PCA

Utilizing principal component analysis decreases the dimensionality of a set of information or when variables are highly correlated with each other in a regression model.

Applications:.

* In **yield curve analysis**, running PCA with 10 components on a swap rates with maturities 2y, 3y, 4y, 5y, 7y, 10y, 15y, 20y, 30y generates three principal components that describe most of the variability in the swap rates. Conveniently, the first factor can be interpreted as a level factor, the second is a slope factor and the third as a curvature factor. These three factors are orthogonal or uncorrelated.
* A customer items business wishes to evaluate client reactions to numerous qualities of a brand-new hair shampoo: color, odor, texture, tidiness, shine, volume, quantity had to soap, and cost. They carry out a principal elements analysis to figure out whether they can form a smaller sized variety of uncorrelated variables that can predict the **most desired attributes**. The outcomes can recognize the following patterns:

– Smell, color, and texture form a “Shampoo quality” component.

– Cleanliness, shine, and volume form an “Effect on hair” component.

– Amount required per use and cost form a “Value” component.

2) Regression

Applications:

A model to predict the effect of revenue, number of patents and CEO compensation on stock prices of a company

* A model that predicts effect of students weight, years of education and ethnicity on IQ or standardized test scores(GMAT)

3) Clustering

Applications:

* Creating different clusters of stocks based on their variances/betas and selecting stocks from different clusters to achieve **portfolio diversification**
* Determining average price of a house different parts of the city through clustering-based variables such as size of house, location, property taxes etc.

4) Classification

Applications:

* **sentiment analysis** using classifications/spam filtering
* Face Detection: It classifies the parts of the image as face and non-face. It contains training data of n x n pixels with a two-class face (+1) and non-face (-1). Then it extracts features from each pixel as face or non-face. Creates a square boundary around faces on the basis of pixel brightness and classifies each image by using the same process
* Detection **fradulent financial statements** using decision trees

Question 2:

> x<-matrix(nrow=100,ncol=2)

> for (i in 1:100){x[i,1]=i}

> print(x)

OUTPUT

[,1] [,2]

[1,] 1 NA

[2,] 2 NA

[3,] 3 NA

[4,] 4 NA

[5,] 5 NA

[6,] 6 NA

[7,] 7 NA

[8,] 8 NA

[9,] 9 NA

[10,] 10 NA

[11,] 11 NA

[12,] 12 NA

[13,] 13 NA

[14,] 14 NA

[15,] 15 NA

[16,] 16 NA

[17,] 17 NA

[18,] 18 NA

[19,] 19 NA

[20,] 20 NA

[21,] 21 NA

[22,] 22 NA

[23,] 23 NA

[24,] 24 NA

[25,] 25 NA

[26,] 26 NA

[27,] 27 NA

[28,] 28 NA

[29,] 29 NA

[30,] 30 NA

[31,] 31 NA

[32,] 32 NA

[33,] 33 NA

[34,] 34 NA

[35,] 35 NA

[36,] 36 NA

[37,] 37 NA

[38,] 38 NA

[39,] 39 NA

[40,] 40 NA

[41,] 41 NA

[42,] 42 NA

[43,] 43 NA

[44,] 44 NA

[45,] 45 NA

[46,] 46 NA

[47,] 47 NA

[48,] 48 NA

[49,] 49 NA

[50,] 50 NA

[51,] 51 NA

[52,] 52 NA

[53,] 53 NA

[54,] 54 NA

[55,] 55 NA

[56,] 56 NA

[57,] 57 NA

[58,] 58 NA

[59,] 59 NA

[60,] 60 NA

[61,] 61 NA

[62,] 62 NA

[63,] 63 NA

[64,] 64 NA

[65,] 65 NA

[66,] 66 NA

[67,] 67 NA

[68,] 68 NA

[69,] 69 NA

[70,] 70 NA

[71,] 71 NA

[72,] 72 NA

[73,] 73 NA

[74,] 74 NA

[75,] 75 NA

[76,] 76 NA

[77,] 77 NA

[78,] 78 NA

[79,] 79 NA

[80,] 80 NA

[81,] 81 NA

[82,] 82 NA

[83,] 83 NA

[84,] 84 NA

[85,] 85 NA

[86,] 86 NA

[87,] 87 NA

[88,] 88 NA

[89,] 89 NA

[90,] 90 NA

[91,] 91 NA

[92,] 92 NA

[93,] 93 NA

[94,] 94 NA

[95,] 95 NA

[96,] 96 NA

[97,] 97 NA

[98,] 98 NA

[99,] 99 NA

[100,] 100 NA

Part 2)

> for (i in 1:100){x[i,2]=x[i,1]\*x[i,1]}

> print(x)

OUTPUT

[,1] [,2]

[1,] 1 1

[2,] 2 4

[3,] 3 9

[4,] 4 16

[5,] 5 25

[6,] 6 36

[7,] 7 49

[8,] 8 64

[9,] 9 81

[10,] 10 100

[11,] 11 121

[12,] 12 144

[13,] 13 169

[14,] 14 196

[15,] 15 225

[16,] 16 256

[17,] 17 289

[18,] 18 324

[19,] 19 361

[20,] 20 400

[21,] 21 441

[22,] 22 484

[23,] 23 529

[24,] 24 576

[25,] 25 625

[26,] 26 676

[27,] 27 729

[28,] 28 784

[29,] 29 841

[30,] 30 900

[31,] 31 961

[32,] 32 1024

[33,] 33 1089

[34,] 34 1156

[35,] 35 1225

[36,] 36 1296

[37,] 37 1369

[38,] 38 1444

[39,] 39 1521

[40,] 40 1600

[41,] 41 1681

[42,] 42 1764

[43,] 43 1849

[44,] 44 1936

[45,] 45 2025

[46,] 46 2116

[47,] 47 2209

[48,] 48 2304

[49,] 49 2401

[50,] 50 2500

[51,] 51 2601

[52,] 52 2704

[53,] 53 2809

[54,] 54 2916

[55,] 55 3025

[56,] 56 3136

[57,] 57 3249

[58,] 58 3364

[59,] 59 3481

[60,] 60 3600

[61,] 61 3721

[62,] 62 3844

[63,] 63 3969

[64,] 64 4096

[65,] 65 4225

[66,] 66 4356

[67,] 67 4489

[68,] 68 4624

[69,] 69 4761

[70,] 70 4900

[71,] 71 5041

[72,] 72 5184

[73,] 73 5329

[74,] 74 5476

[75,] 75 5625

[76,] 76 5776

[77,] 77 5929

[78,] 78 6084

[79,] 79 6241

[80,] 80 6400

[81,] 81 6561

[82,] 82 6724

[83,] 83 6889

[84,] 84 7056

[85,] 85 7225

[86,] 86 7396

[87,] 87 7569

[88,] 88 7744

[89,] 89 7921

[90,] 90 8100

[91,] 91 8281

[92,] 92 8464

[93,] 93 8649

[94,] 94 8836

[95,] 95 9025

[96,] 96 9216

[97,] 97 9409

[98,] 98 9604

[99,] 99 9801

[100,] 100 10000

Question 3)

> i<-length((trees$Volume[(trees$Height >= 70)]))

> j<-sum(trees$Volume[(trees$Height >= 70)])

> print(j/i)

OUTPUT:

[1] 32.80769