Machine Learning

Omkar Nitin Pawar

Hawk ID : A20448802

opawar@hawk.iit.edu

Assignment 01

- Histograms
- Boxplots
- K Nearest Neighbors (kNN)

"The code for the assignment is provided in .py file. For notebook, please view the link below. It has all the outputs for a convenient view"

https://www.kaggle.com/opawar600/ml-assignment-1?scriptVersionId=20137497

CS 584-04: Machine Learning.

Fall 2019 Assignment 1

Question 1 (40 points)

Write a Python program to calculate the density estimator of a histogram. Use the field *x* in the NormalSample.csv file.

a) (5 points) According to Izenman (1991) method, what is the recommended bin-width for the histogram of x?

Answer : 0.4

# Code :

```
from scipy.stats import iqr

inter_quartile_range = iqr(df.x)

N = df.x.count()

bin_width = 2*inter_quartile_range*(pow(N,-1/3)) #bin width = 2(IQR)N^(-
1/3)

print(bin_width)
```

b) (5 points) What are the minimum and the maximum values of the field x?

Answer :

Maximum : 35.400000

Minimum : 26.3

# Code:

```
min_x = df.x.min()

max_x = df.x.max()

print("Minimum = ",min_x,"\nMaximum = ",max_x)
```

c) (5 points) Let a be the largest integer less than the minimum value of the field x, and b be the smallest integer greater than the maximum value of the field x. What are the values of a and b?

Answer : a = 26 b = 36

d) (5 points) Use h = 0.1, minimum = a and maximum = b. List the coordinates of the density estimator. Paste the histogram drawn using Python or your favorite graphing tools.

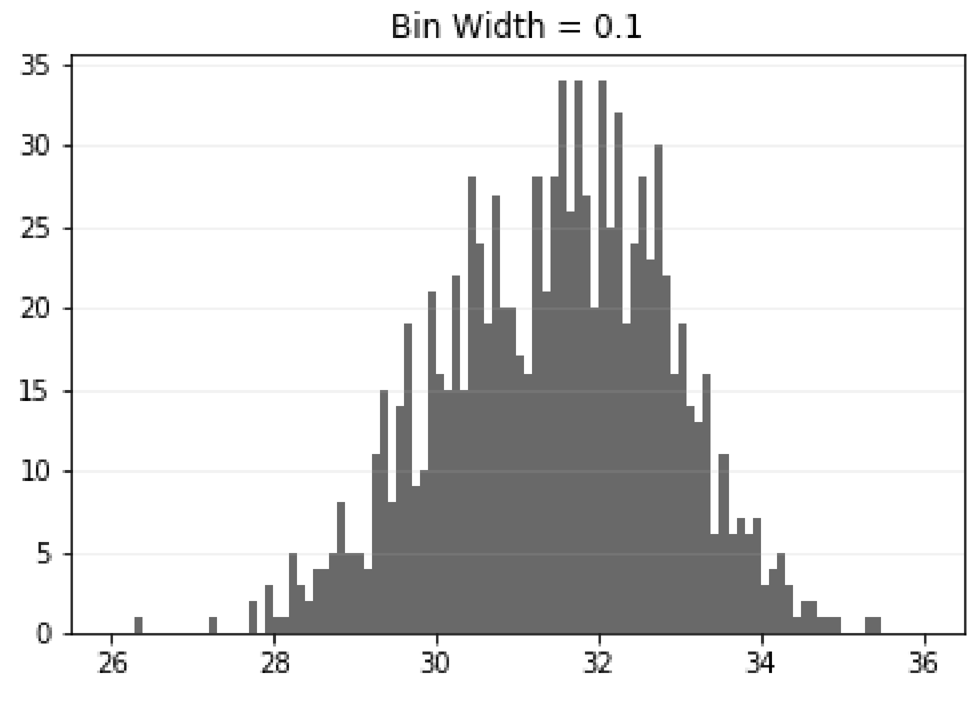
Answer :

NOTE : mi = Midpoints p(mi) = Density Estimate

Number of bins = 100.

The coordinates of density estimator are given below. Only first few rows of the dataframe are shown as the total number of rows are 100 and it is not feasible to put all of them here. To get more details, please refer the code file.

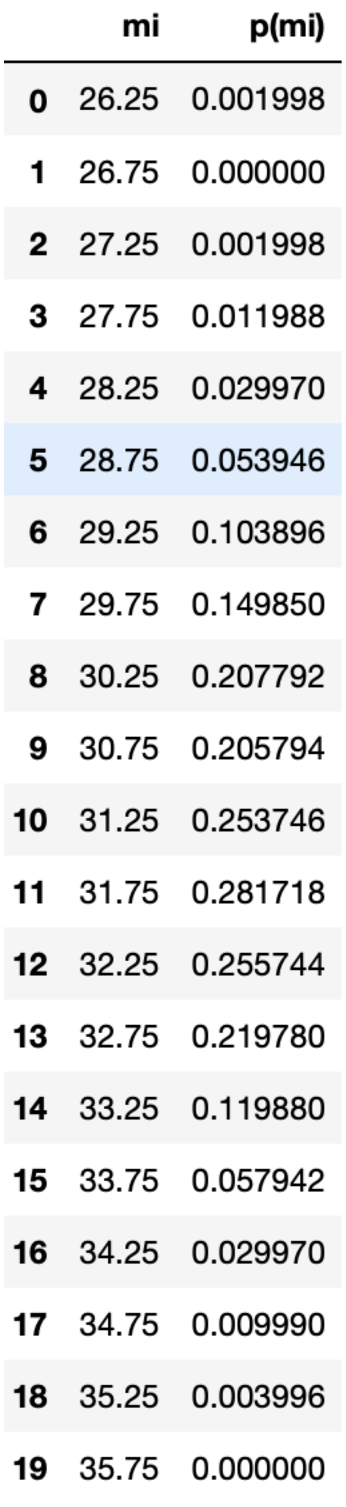
	mi	p(mi)
0	26.05	0.00000
1	26.15	0.00000
2	26.25	0.00999
3	26.35	0.00000
4	26.45	0.00000
5	26.55	0.00000
6	26.65	0.00000
7	26.75	0.00000
8	26.85	0.00000
9	26.95	0.00000
10	27.05	0.00000



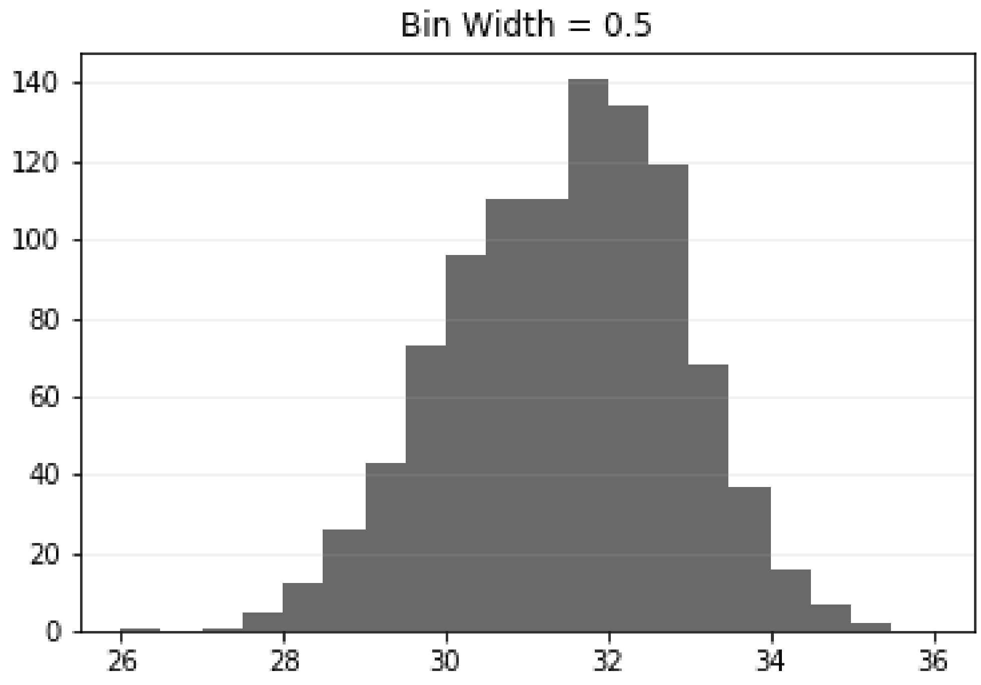
e) (5 points) Use $h = 0.5$, minimum = a and maximum = b. List the coordinates of the density estimator. Paste the histogram drawn using Python or your favorite graphing tools.

Answer :

Number of bins = 20



	mi	p(mi)
0	26.25	0.001998
1	26.75	0.000000
2	27.25	0.001998
3	27.75	0.011988
4	28.25	0.029970
5	28.75	0.053946
6	29.25	0.103896
7	29.75	0.149850
8	30.25	0.207792
9	30.75	0.205794
10	31.25	0.253746
11	31.75	0.281718
12	32.25	0.255744
13	32.75	0.219780
14	33.25	0.119880
15	33.75	0.057942
16	34.25	0.029970
17	34.75	0.009990
18	35.25	0.003996
19	35.75	0.000000

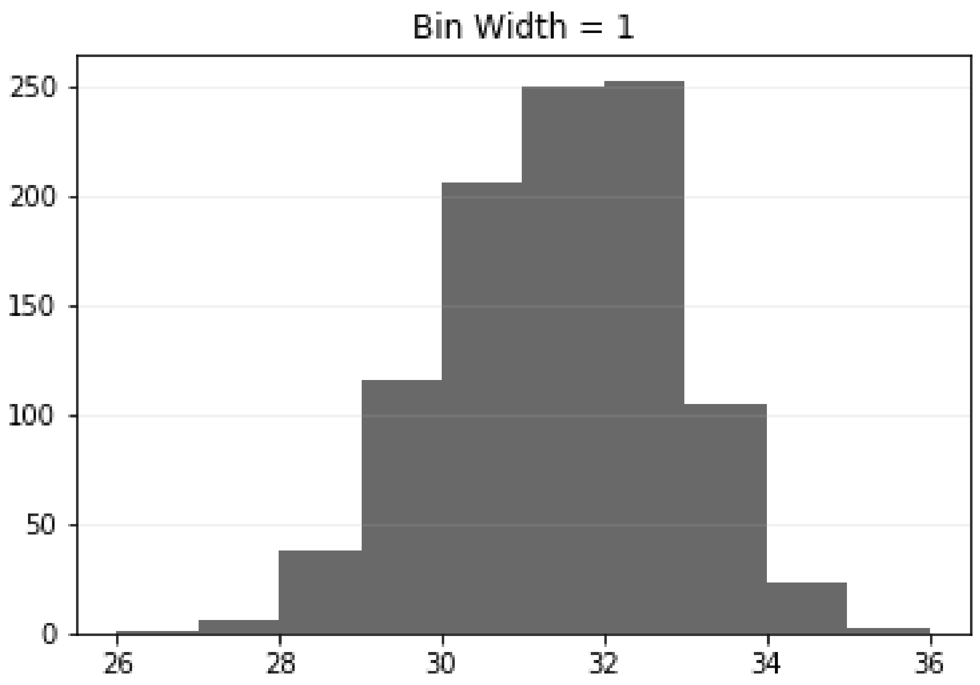


f) (5 points) Use h = 1, minimum = a and maximum = b. List the coordinates of the density estimator. Paste the histogram drawn using Python or your favorite graphing tools.

Answer :

Number of bins = 10

	mi	p(mi)
0	26.5	0.000999
1	27.5	0.006993
2	28.5	0.041958
3	29.5	0.126873
4	30.5	0.206793
5	31.5	0.267732
6	32.5	0.237762
7	33.5	0.088911
8	34.5	0.019980
9	35.5	0.001998

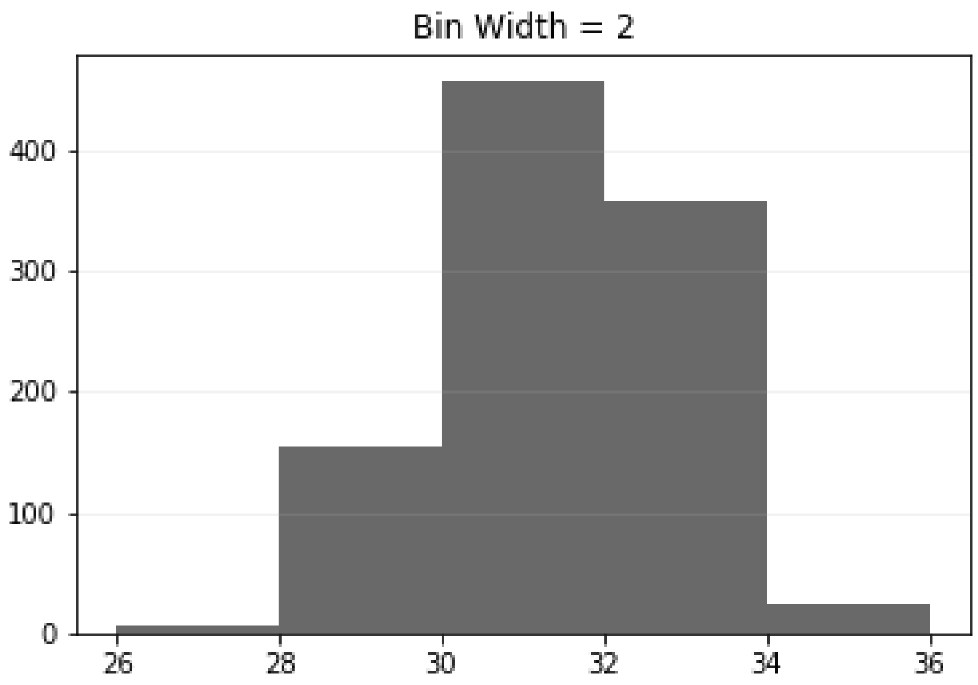


g) (5 points) Use h = 2, minimum = a and maximum = b. List the coordinates of the density estimator. Paste the histogram drawn using Python or your favorite graphing tools.

Answer :

Number of bins = 5

	mi	p(mi)
0	27.0	0.003996
1	29.0	0.084416
2	31.0	0.237263
3	33.0	0.163337
4	35.0	0.010989



h) (5 points) Among the four histograms, which one, in your honest opinions, can best provide your insights into the shape and the spread of the distribution of the field x? Please state your arguments.

Answer :

Among all the histograms, the one with h = 0.5 provides the best insight into the shape and spread of the distribution.

For the first histogram with h = 0.1, we can see a lot of details but can't get the overview of the data.
In histogram with h = 1 and h = 2, we are missing out a lot of information about each interval.
The histogram with h = 0.5 is where we get a good overview of the spread of data (normally distributed) and retain quite a few information too.

Question 2 (20 points)

Use in the NormalSample.csv to generate box-plots for answering the following questions.

a) (5 points) What is the five-number summary of x? What are the values of the 1.5 IQR whiskers?

Answer:

- **5 number summary :**

Sample Minimum :	26.3
First Quartile :	30.4
Median :	31.5
Third Quartile :	32.4
Sample Maximum :	35.4

The 1.5 IQR values are calculated as

Lower Whisker = q1 - 1.5*inter_quartile_range

Upper Whisker = q3 + 1.5*inter_quartile_range

- **1.5 IQR whiskers**

  Lower Whisker = 27.4 Upper Whisker = 35.4

# Code :

```
df = pd.read_csv("/kaggle/input/normal/NormalSample.csv")

df.describe()
```

	i	group	x
count	1001.000000	1001.000000	1001.000000
mean	500.000000	0.685315	31.414585
std	289.108111	0.464623	1.397672
min	0.000000	0.000000	26.300000
25%	250.000000	0.000000	30.400000
50%	500.000000	1.000000	31.500000
75%	750.000000	1.000000	32.400000
max	1000.000000	1.000000	35.400000

```
#FOR calculating Inter Quartile Range

from scipy.stats import iqr

inter_quartile_range = iqr(df.x)

q1 = np.percentile(df.x,25)

q3 = np.percentile(df.x,75)

l_whisker = q1 - 1.5*inter_quartile_range

u_whisker = q3 + 1.5*inter_quartile_range

print ("Lower Whisker = ",l_whisker,"\nUpper Whisker = ",u_whisker)
```

b) 5 points) What is the five-number summary of x for each category of the group? What are the values of the 1.5 IQR whiskers for each category of the group?

Answer :

- **Group 0:**
  Five-number summary

Sample Minimum :	26.3
First Quartile :	29.4

Median : 30

Third Quartile : 30.6

Sample Maximum : 32.2

1.5 IQR Whiskers

Lower Whisker = 27.59 Upper Whisker = 32.4

- **Group 1 :**

  Five-number summary

  Sample Minimum : 29.1

  First Quartile : 31.4

  Median : 32.1

  Third Quartile : 32.7

  Sample Maximum : 35.4

  1.5 IQR Whiskers

  Lower Whisker = 29.49 Upper Whisker = 34.65

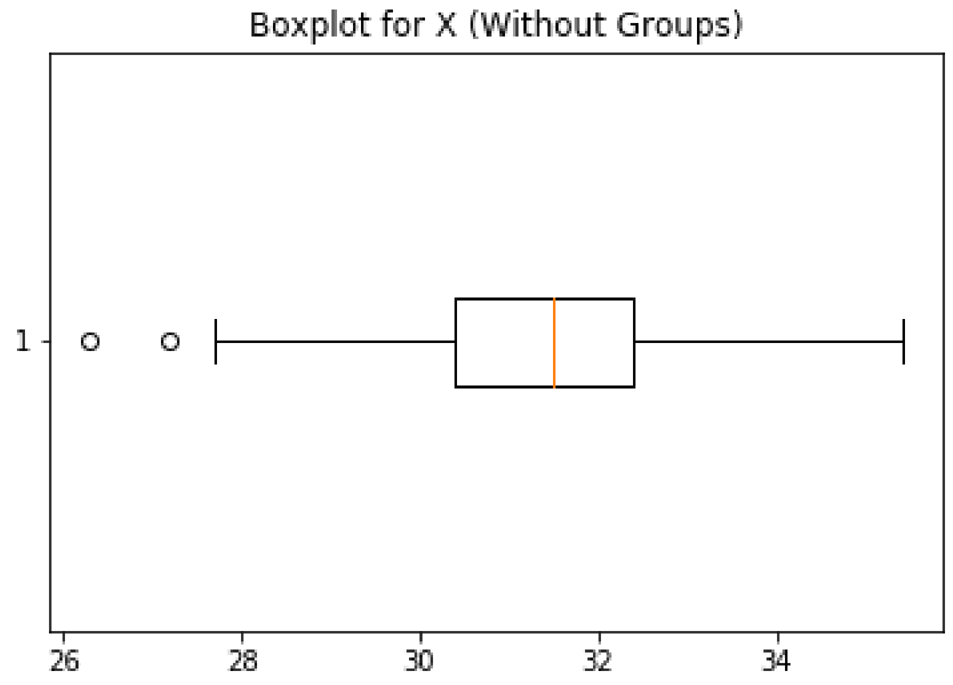
c) (5 points) Draw a boxplot of x (without the group) using the Python boxplot function. Can you tell if the Python's boxplot has displayed the 1.5 IQR whiskers correctly?

Answer :

In the following figure, it can be seen that the boxplot has displayed the 1.5 IQR whiskers correctly. To plot the boxplot, matplotlib library is used.

# Code :

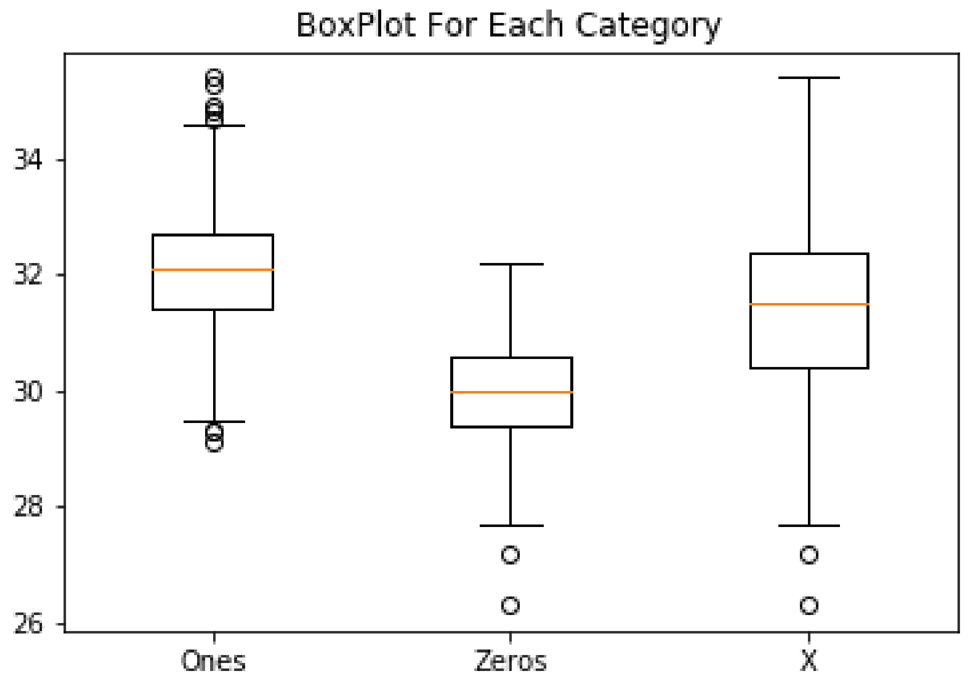
```
plt.boxplot(df.x,vert = False)

plt.title("Boxplot of X")
```



d) (5 points) Draw a graph where it contains the boxplot of x, the boxplot of x for each category of Group (i.e., three boxplots within the same graph frame). Use the 1.5 IQR whiskers, identify the outliers of x, if any, for the entire data and for each category of the group.
Hint: Consider using the CONCAT function in the PANDA module to append observations.

Answer :

The following figure shows the boxplot for each category. The 1.5 IQR whiskers are displayed correctly as per the data above. Outliers are the values that lie beyond the range of 1.5 IQR whisker value. In boxplot, they are represented by small circles outside the upper and lower whisker. There are a lot of outliers for group Ones as compared to zeros and X.


BoxPlot For Each Category
34
32
30
28
26
Ones
Zeros
X

Question 3 (40 points)

The data, FRAUD.csv, contains results of fraud investigations of 5,960 cases. The binary variable FRAUD indicates the result of a fraud investigation: 1 = Fraudulent, 0 = Otherwise. The other interval variables contain information about the cases.

1. TOTAL_SPEND: Total amount of claims in dollars
2. DOCTOR_VISITS: Number of visits to a doctor
3. NUM_CLAIMS: Number of claims made recently
4. MEMBER_DURATION: Membership duration in number of months
5. OPTOM_PRESC: Number of optical examinations
6. NUM_MEMBERS: Number of members covered

You are asked to use the Nearest Neighbors algorithm to predict the likelihood of fraud.

a) (5 points) What percent of investigations are found to be fraudulent? Please give your answer up to 4 decimal places.

Answer : 19.9497 which is 19.9497%

# Code :

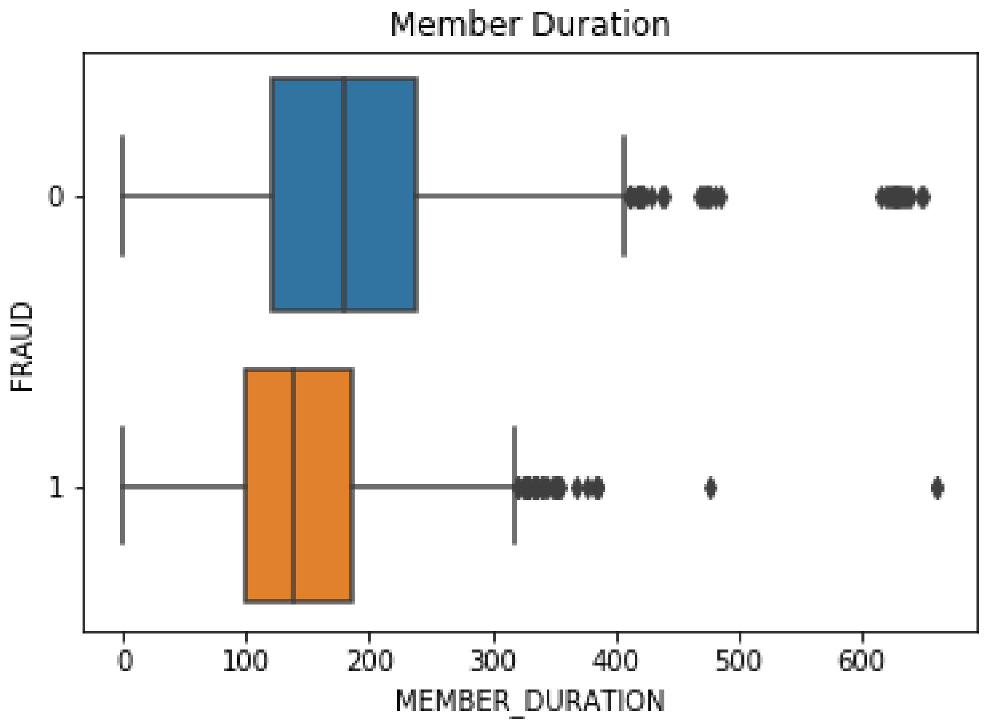
```
num_no_frauds = fraud_df.FRAUD.value_counts()[0]

num_of_frauds = fraud_df.FRAUD.value_counts()[1]

count = fraud_df.FRAUD.count()

fraud_percent = num_of_frauds/count*100

print(round(fraud_percent,4))
```

b) (5 points) Use the BOXPLOT function to produce horizontal box-plots. For each interval variable, one box-plot for the fraudulent observations, and another box-plot for the non-fraudulent observations. These two box-plots must appear in the same graph for each interval variable.

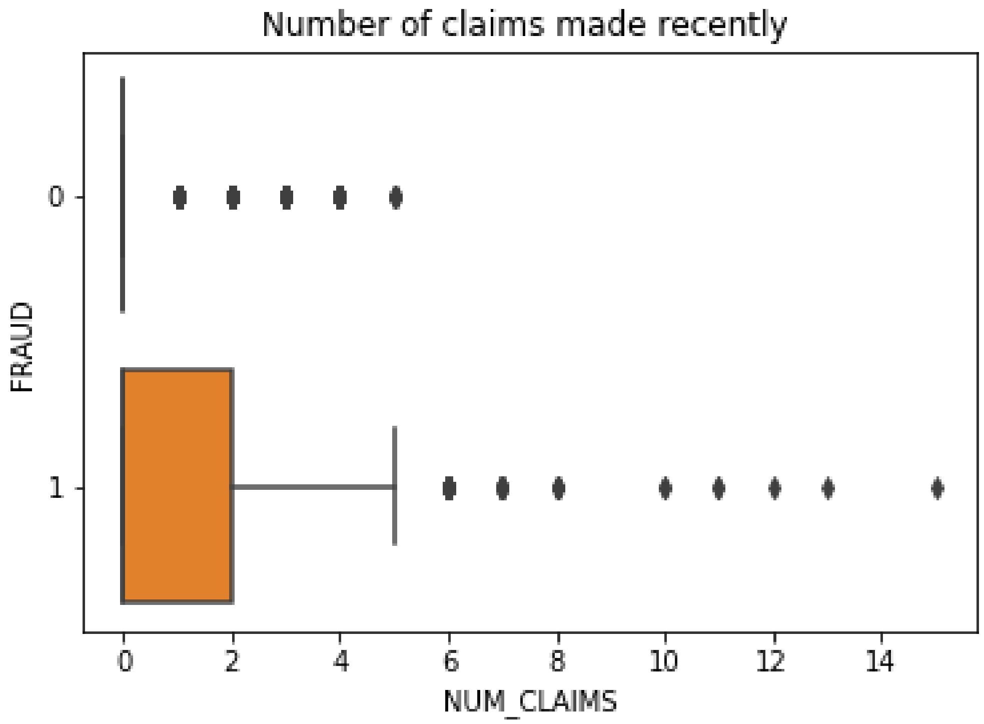
Answer :

#Code :

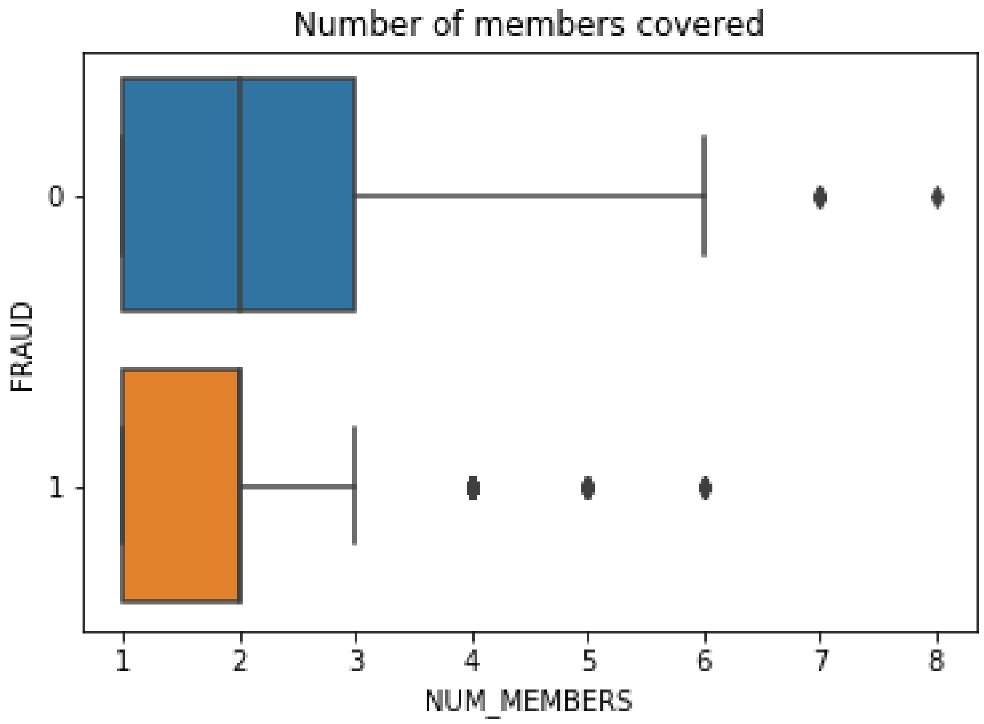
```
sns.boxplot(data = fraud_df , x = 'MEMBER_DURATION' , y =
'FRAUD', orient = 'h')
plt.title("Member Duration")
```



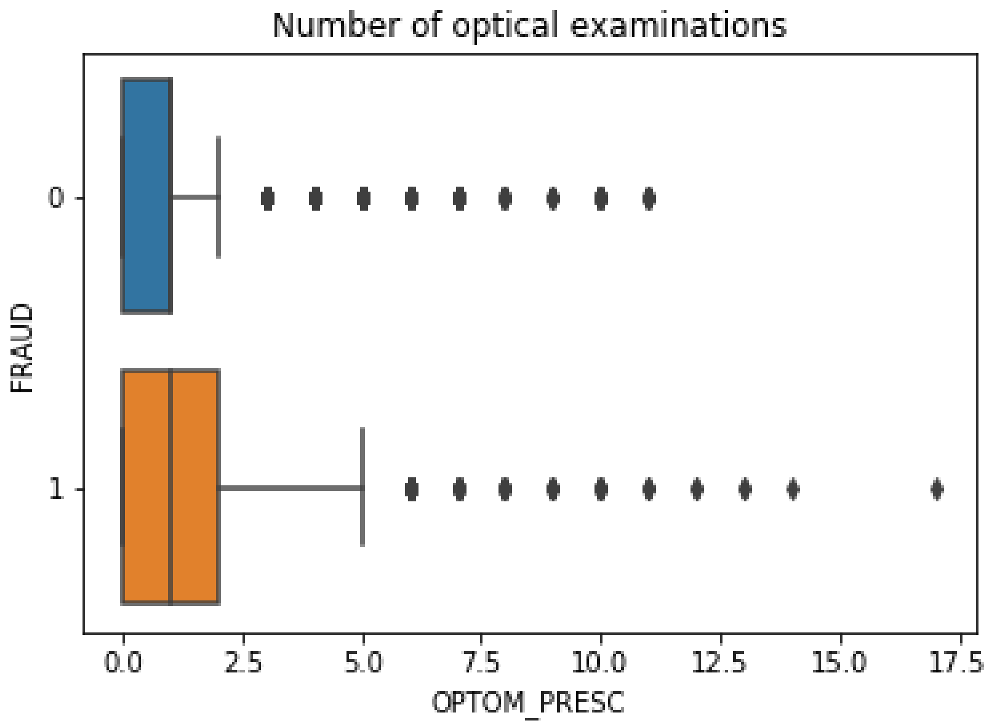
```
sns.boxplot(data = fraud_df , x = 'NUM_CLAIMS' , y =
'FRAUD',orient = 'h')
plt.title("Number of claims made recently ")
```



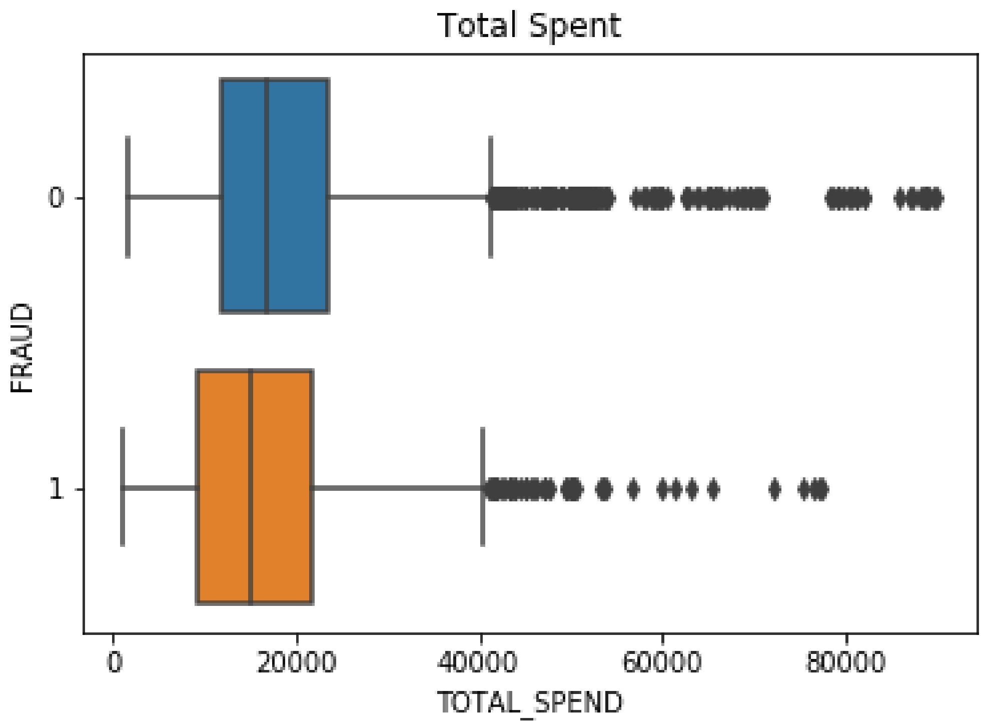
```
sns.boxplot(data = fraud_df , x = 'NUM_MEMBERS' , y =
'FRAUD',orient = 'h')
plt.title("Number of members covered")
```



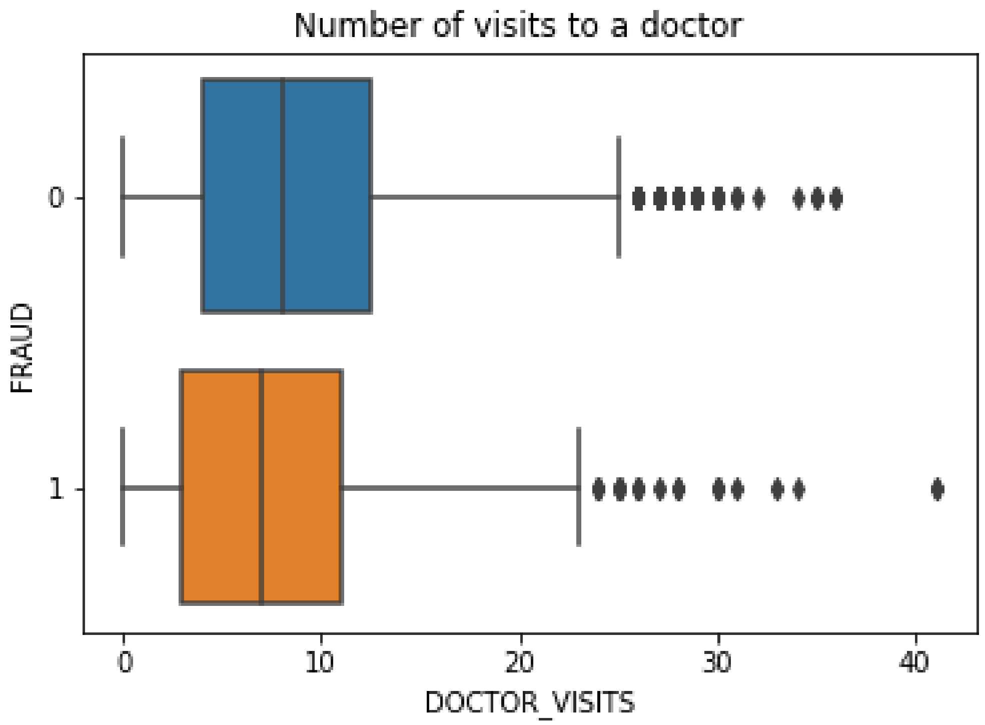
```
sns.boxplot(data = fraud_df , x = 'OPTOM_PRESC' , y = 'FRAUD' ,
orient = 'h')
plt.title("Number of optical examinations")
```



```
sns.boxplot(data = fraud_df , x = 'TOTAL_SPEND' , y =
'FRAUD',orient = 'h')
plt.title("Total Spent")
```



```
sns.boxplot(data = fraud_df , x = 'DOCTOR_VISITS' , y = 'FRAUD'
,orient = 'h')
plt.title("Number of visits to a doctor  ")
```



c) (10 points) Orthonormalize interval variables and use the resulting variables for the nearest neighbor analysis. Use only the dimensions whose corresponding eigenvalues are greater than one.

i. (5 points) How many dimensions are used?

ii. (5 points) Please provide the transformation matrix? You must provide proof that the resulting variables are actually orthonormal.

Answer :

i. A total of dimensions **6** are used here. Dimensions of dataset are nothing but the total number of unique columns that are present in the data. Here, we have removed FRAUD column from our dataframe as it is our target variable and CASE_ID is removed because it does not provide any useful information about the datapoint. Hence we are left with just 6 columns which are ["TOTAL_SPEND","DOCTOR_VISITS","NUM_CLAIMS","MEMBER_DURATION","OPTOM_PRESC","NUM_MEMBERS"].

ii. The transformation matrix is calculated by the code shown below.

   To prove that the resulting variables are actually orthonormal, an Identity matrix is provided.

# Code :

```
x = np.matrix(fraud_df)

xtx = x.transpose() * x

print("t(x) * x = \n", xtx)


# Eigenvalue decomposition

evals, evecs = LA.eigh(xtx)

print("Eigenvalues of x = \n", evals)

print("Eigenvectors of x = \n",evecs)


# Here is the transformation matrix

transf = evecs * LA.inv(np.sqrt(np.diagflat(evals)));

print("Transformation Matrix = \n", transf)
```

Transformation Matrix :

```
[[-6.49862374e-08 -2.41194689e-07  2.69941036e-07 -2.42525871e-07
  -7.90492750e-07  5.96286732e-07]
 [ 7.31656633e-05 -2.94741983e-04  9.48855536e-05  1.77761538e-03
   3.51604254e-06  2.20559915e-10]
 [-1.18697179e-02  1.70828329e-03 -7.68683456e-04  2.03673350e-05
   1.76401304e-07  9.09938972e-12]
 [ 1.92524315e-06 -5.37085514e-05  2.32038406e-05 -5.78327741e-05
   1.08753133e-04  4.32672436e-09]
 [ 8.34989734e-04 -2.29964514e-03 -7.25509934e-03  1.11508242e-05
   2.39238772e-07  2.85768709e-11]
 [ 2.10964750e-03  1.05319439e-02 -1.45669326e-03  4.85837631e-05
   6.76601477e-07  4.66565230e-11]}
```

Identity Matrix :

```
[[ 1.00000000e+00 -2.87703888e-15  1.90299165e-15  7.06552872e-15
   1.16226473e-15 -1.35308431e-16]
 [-2.87703888e-15  1.00000000e+00 -1.37216627e-15 -1.98244199e-14
  -6.59194921e-16  7.21644966e-16]
 [ 1.90299165e-15 -1.37216627e-15  1.00000000e+00  4.96366728e-15
  -6.24500451e-17 -1.17961196e-16]
 [ 7.06552872e-15 -1.98244199e-14  4.96366728e-15  1.00000000e+00
   1.10432496e-14 -4.20496971e-15]
 [ 1.16226473e-15 -6.59194921e-16 -6.24500451e-17  1.10432496e-14
   1.00000000e+00 -6.66133815e-16]
 [-1.35308431e-16  7.21644966e-16 -1.17961196e-16 -4.20496971e-15
  -6.66133815e-16  1.00000000e+00]]
```

d) (10 points) Use the NearestNeighbors module to execute the Nearest Neighbors algorithm using exactly five neighbors and the resulting variables you have chosen in c). The KNeighborsClassifier module has a score function.

i. (5 points) Run the score function, provide the function return value

ii. (5 points) Explain the meaning of the score function return value.

Answer :

i. 0.8778. The accuracy score therefore is 87.78%

# Code :

```
from sklearn import metrics
knn = KNeighborsClassifier(n_neighbors=5, metric = "euclidean")
knn.fit(transf_x,target)
predictions = knn.predict(transf_x)
print(metrics.accuracy_score(target,predictions))
```

ii. The return value of the score function gives use the accuracy of the KNN Model that is trained over the given dataset. The accuracy of approximately 88% is pretty good. This value shows that the model is able to classify 88 % of the outputs correctly.

# Code :

```
from sklearn import metrics
knn = KNeighborsClassifier(n_neighbors=5, metric = "euclidean")
knn.fit(transf_x,target)
predictions = knn.predict(transf_x)
print(metrics.accuracy_score(target,predictions))
```

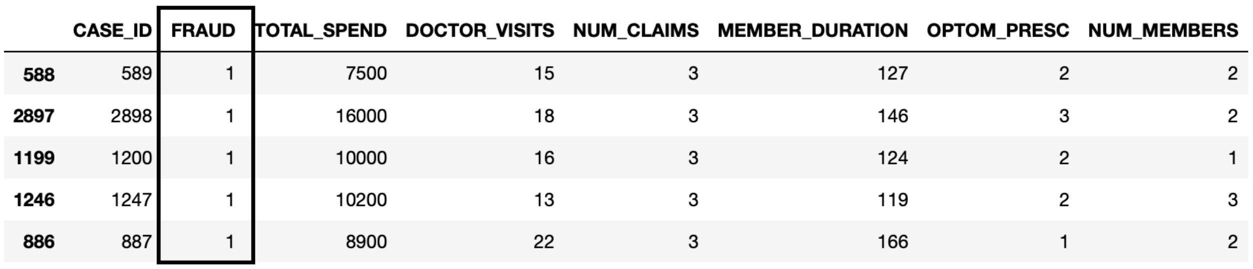
e) (5 points) For the observation which has these input variable values: TOTAL_SPEND = 7500, DOCTOR_VISITS = 15, NUM_CLAIMS = 3, MEMBER_DURATION = 127, OPTOM_PRESC = 2, and NUM_MEMBERS = 2, find its **five** neighbors. Please list their input variable values and the target values. *Reminder: transform the input observation using the results in c) before finding the neighbors*.

Answer :

The 5 neighbors of the given input datapoint are listed below. The target variable "FRAUD" is highlighted with a box. The index of 5 neighbors are [588, 2897, 1199, 1246, 886].

# Code :

```
test = [[7500,15,3,127,2,2]] * transf;
neighs = knn.kneighbors(test,return_distance=False)
fraudulent_df = pd.read_csv("/kaggle/input/fraudknn/Fraud.csv")
fraudulent_df.iloc[neighs[0][0:]]
```



	CASE_ID	FRAUD	TOTAL_SPEND	DOCTOR_VISITS	NUM_CLAIMS	MEMBER_DURATION	OPTOM_PRESC	NUM_MEMBERS
588	589	1	7500	15	3	127	2	2
2897	2898	1	16000	18	3	146	3	2
1199	1200	1	10000	16	3	124	2	1
1246	1247	1	10200	13	3	119	2	3
886	887	1	8900	22	3	166	1	2

# Code :

```
test = [[7500,15,3,127,2,2]] * transf;
# The input observation is transformed.
neighs = knn.kneighbors(test,return_distance=False)
# Neighbors are found and stored.
```

f) (5 points) Follow-up with e), what is the predicted probability of fraudulent (i.e., FRAUD = 1)? If your predicted probability is greater than or equal to your answer in a), then the observation will be classified as fraudulent. Otherwise, non-fraudulent. Based on this criterion, will this observation be misclassified?

Answer:

The predicted probability for the input datapoint will be 1.
This results due to the fact that each neighbor that is found in (e) has a label 1 i.e "FRAUD".
Hence the label for new datapoint is "FRAUD"

The predicted probability is calculated as follows :
Total number of neighbors with label "Fraud" / Total number of neighbors = 5/5 = 1

Not a single neighbor has different label. Hence, based on the given criterion, the observation is not misclassified as 1 > 0.199497 (0. 199497 is the answer of Q3(a)).