**Assignment\_3 CSE-805: Machine Learning and Data Mining** **Name**: Robert Akinie

1. Do the Descriptive Statistics of the dataset and List the columns that could have the missing data in terms of zero. **(sometimes zero could be part of the data distribution in the attribute and sometimes not). You need not have to replace the zeros with Means of Columns yet.**

Based on the descriptive statistics below, columns plas, pres, skin, test and mass seem to have missing values in terms of zero, based on their min values and/or 25% percentile. Column plas also has a very large max value, and that could also represent a missing value.

Text, table, Excel

Description automatically generated

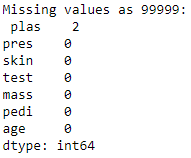
1. Find the indices of the rows that have missing values as 99999. (Paste your Code/Script).



Comment: #Question 4 in Appendix

1. Find how many missing values are present in each column. ( Paste your Code)

Text, table

Description automatically generated

Comment: #Question 5 in Appendix

1. Find the Column\_Names that could have outliers. (Paste your Code/Script). **Hint:-- Outlier, if greater than 3sigma.**



Comment: #Question 6 in Appendix

1. Discuss how do you find the missing Data and Outliers, (previously in # 3, #4, and #5) from the Data Visualizations by Columns Wise. Which data visualization works better as you have huge samples here?.

For missing data, if the label/feature cannot be represented by zero values, then feature values having zero values implies missing data. For example, BMI of a patient cannot be zero, hence having zero values of patients’ BMI indicates missing values. Features can also have values which are very high, of which the mean and deviation of the feature values can tell whether there is a large value skewing the statistics. Those could also be termed as missing values, or outliers. Outlier values of a feature can easily be seen from box plots. Box plots show the degree of spread of the data, and individual outliers, as supposed to histograms and density plots.

Calendar

Description automatically generated

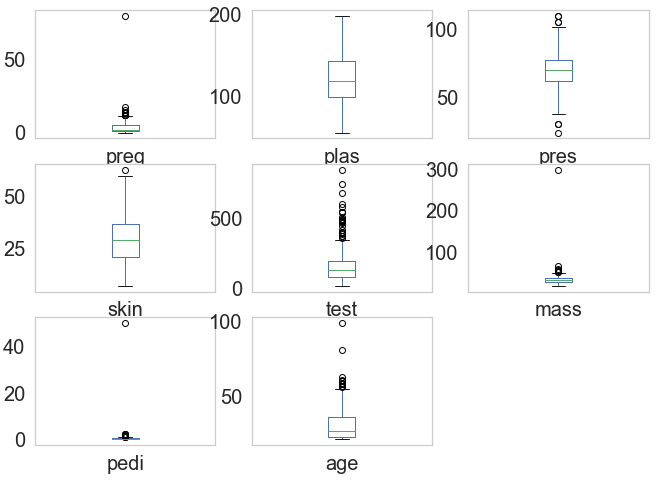
1. Replace missing Values and errors with NaNs and print the descriptive statistics of the dataset (Paste your Code/Script).

Table

Description automatically generated

Comment: #Question 8 in Appendix

1. Remove the rows of the Missing Data and use one of the data visualization method (used previously in # 7) to show the difference?



Comparing the box plots, one can notice the stark differences in the columns plas(Glucose) and skin(Skin Thickness), which initially contain most of the missing data.

Comment: #Question 9 in Appendix

1. If I give you a choice of removing the two attributes that would least contribute to the Machine Learning Classification performance here, What are the two Columns?



Based on the feature importance metric, the least two attribute contributions are test and skin. Based on the original dataset.

Comment: #Question 10 in Appendix

1. Compare the SVM Classification Model against the dataset with Missing Values (Original Dataset) and after the missing values are removed (after #9), and Discuss how it helped the classification model. **(Hint:-- Do not keep the NaNs in the dataset like # 8, the classification models do not work)**



The values represent the accuracies of the datasets in question respectively. From the values, it shows that the dataset with missing values performs poorly against the dataset with removed missing values. The expectation was that a dataset without missing values will perform better, as the predicting capacity of the model is based on the values of features in question, and can introduce bias and skew

Comment: #Question 11 in Appendix

1. Compare the SVM Classification Model against the dataset with Missing Values (Original Dataset) and after the attributes are reduced (after # 10), and Discuss how it helped the classification model.



The values represent the accuracies of the datasets in question respectively. From the values, it shows that the dataset with missing values has the same performance as that of the dataset with least contributing attributes removed. The initial expectation was that that would improve the model a bit, but since the features do not contribute very little, the model focuses on the most important features for classification, and hence little to no change in performance.

1. Replace the missing Values with mean of the columns and compare the classification model accuracy against the original missing dataset.



The values represent the accuracies of the datasets in question respectively. From the values, it shows that the dataset with feature means replacing respective missing values performs better than the original dataset. The preservation of other row data when one corresponding column that has a missing value replaced would affect number of training examples for the model, and hence more data to train on and avoiding likely model underfit.

1. Normalize all attributes, except #7 (pedigree), after cleaning the missing values, and Compare the classification model accuracy against the Original Dataset. (Paste your Code/Script).



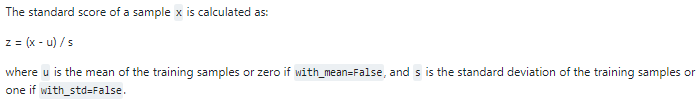
As shown by the respective values, normalization improved the accuracy of the original dataset.

1. Standardize attributes, #2 (plasma) and # 3 (pressure), after cleaning the missing values, and Compare the classification model accuracy against the Original Dataset. (Paste your Code/Script). Write down your standardize equation.



As shown by the respective values, standardization of the above specified features barely improved the accuracy of the original dataset. Initial expectations were that the accuracy would be improved significantly, which is not the case. It could be as a result of not standardizing the whole dataset, as supposed to just selected features.

I used the sklearn library in implementing the standardization. The documentation for the function is below, and it displays the equation used.



Appendix

#Import

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

from sklearn.model\_selection import KFold

from sklearn.model\_selection import cross\_val\_score

from sklearn.svm import SVC

from sklearn.preprocessing import MinMaxScaler

from sklearn.preprocessing import StandardScaler

#Load Dataset

names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']

my\_csv = "pima-indians-diabetes\_Gokaraju\_edited.csv"

pima\_edited = pd.read\_csv(my\_csv, names = names)

#Question 3

pima\_edited.describe()

#Question 4

indice = np.where(pima\_edited == 99999)

print(list(zip(indice[0], indice[1])))

#indice = np.asarray(pima\_edited == 99999).nonzero()

#print(list(zip(indice[0], indice[1])))

#Question 5

pima\_missingData=pima\_edited.drop(columns = ['preg', 'class'])

print("Missing values as zero:\n",(pima\_missingData==0).sum(axis = 0), "\n")

print("Missing values as 99999:\n",(pima\_missingData == 99999).sum(axis = 0))

#Question 6

for (colname, coldata) in pima\_edited.drop(columns = 'class').iteritems():

mean, stdev, threeSigma= np.mean(coldata.values), np.std(coldata.values), 3\*stdev

lower, upper = mean - threeSigma, mean + threeSigma

if (np.nonzero(coldata.values < lower) or np.nonzero(coldata.values > upper)):

print(colname)

#Question 7

pima\_edited.drop(columns = 'class').plot(kind = 'box', subplots = True, layout = (3, 3), fontsize = 20, grid = False)

plt.plot()

#Question 8

pima\_replace0 = pima\_missingData.replace(0, np.nan)

pima\_full\_replace= pima\_replace0.replace(99999, np.nan)

pima\_full\_replace.describe()

#Question 9

pima\_noNan = pima\_edited[~np.isnan(pima\_full\_replace).any(axis = 1)]

pima\_noNan.plot(kind = 'box', subplots = True, layout = (3, 3), fontsize = 20, grid = False)

plt.plot()

#Question 10

array = pima\_edited.values

X, Y = array[:, 0:8], array[:, 8]

model = ExtraTreesClassifier()

model.fit(X,Y)

print(model.feature\_importances\_)

#Question 11

#array = pima\_edited.values

array\_noNan = pima\_noNan.values

#X, Y = array[:, 0:8], array[:, 8]

X\_noNan, Y\_noNan = array\_noNan[:, 0:8], array\_noNan[:, 8]

kfold = KFold(n\_splits = 10, shuffle = True, random\_state = 7)

model = SVC()

results = cross\_val\_score(model, X, Y, cv=kfold)

results\_noNan = cross\_val\_score(model, X\_noNan, Y\_noNan, cv=kfold)

print(results.mean(), results\_noNan.mean())

#Question 12

pima\_att\_removed = pima\_edited.drop(columns = ['test', 'skin'])

array\_att\_removed = pima\_att\_removed.values

X\_att\_removed, Y\_att\_removed = array\_att\_removed[:, 0:6], array\_att\_removed[:, 6]

results\_att\_removed = cross\_val\_score(model, X\_att\_removed, Y\_att\_removed, cv=kfold)

print(results.mean(), results\_att\_removed.mean())

#Question 13

pima\_edited1= pima\_edited.replace(99999, 0)

pima\_mean = pima\_edited1

pima\_mean['plas'].replace(0, pima\_edited1['plas'].mean(), inplace = True)

pima\_mean['pres'].replace(0, pima\_edited1['pres'].mean(), inplace = True)

pima\_mean['skin'].replace(0, pima\_edited1['skin'].mean(), inplace = True)

pima\_mean['test'].replace(0, pima\_edited1['test'].mean(), inplace = True)

pima\_mean['mass'].replace(0, pima\_edited1['mass'].mean(), inplace = True)

array\_mean = pima\_edited4.values

X\_mean, Y\_mean = array\_mean[:, 0:8], array\_mean[:, 8]

results\_mean = cross\_val\_score(model, X\_mean, Y\_mean, cv=kfold)

print(results.mean(), results\_mean.mean())

#Question 14

scaler = MinMaxScaler()

pima\_edited4 = pima\_mean.drop(columns = ('pedi'))

scaler.fit(pima\_edited4)

edited\_4 = scaler.transform(pima\_edited4)

pima\_normalized = pd.DataFrame(edited\_4, columns = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'age', 'class'])

pima\_normalized.insert(6, 'pedi', np.asarray(pima\_mean.pedi))

array\_norm = pima\_normalized.values

X\_norm, Y\_norm = array\_norm[:, 0:8], array\_norm[:, 8]

results\_norm = cross\_val\_score(model, X\_norm, Y\_norm, cv=kfold)

print(results.mean(), results\_norm.mean())

#Question 15

scaler2 = StandardScaler()

scalr = scaler2.fit(pima\_mean[['plas', 'pres']])

scaled = scalr.transform(pima\_mean[['plas', 'pres']])

pima\_scaled = pd.DataFrame(scaled, columns = ['plas', 'pres'])

drop\_plas = pima\_mean.drop(columns = 'plas')

pima\_standard = drop\_plas.drop(columns = 'pres')

pima\_standard.insert(1, 'plas', np.asarray(pima\_scaled.plas))

pima\_standard.insert(2, 'pres', np.asarray(pima\_scaled.pres))

array\_std = pima\_standard.values

X\_std, Y\_std = array\_std[:, 0:8], array\_std[:, 8]

results\_std = cross\_val\_score(model, X\_std, Y\_std, cv=kfold)

print(results.mean(), results\_std.mean())