**DS USING PYTHON LAB**

**EXPERIMENT: 03**

**AIM:** Perform data Data Modeling.

**PROBLEM STATEMENT:**

a. Partition the data set, for example 75% of the records are included in the training data

set and 25% are included in the test data set.

b. Use a bar graph and other relevant graphs to confirm your proportions.

c. Identify the total number of records in the training data set.

d. Validate partition by performing a two‐sample Z‐test.

**THEORY:**

**Data Partitioning:**

Data partitioning is the division of the whole data available into two or three non-overlapping sets: the training set , the validation set , and the test set . If the data set is very large, often only a portion of it is selected for the partitions. Partitioning is normally used when the model for the data at hand is being chosen from a broad set of models. The basic idea of data partitioning is to keep a subset of available data out of analysis, and to use it later for verification of the model. Data partitioning is normally used in supervised learning techniques in data mining where a predictive model is chosen from a set of models, using their performance on the training set as the validation of choice.

**Hypothesis Testing:**

Hypothesis Testing is a type of statistical analysis in which you put your assumptions about a population parameter to the test. It is used to estimate the relationship between 2 statistical variables. A hypothesis is often described as an “educated guess” about a specific parameter or population. Once it is defined, one can collect data to determine whether it provides enough evidence that the hypothesis is true.

* Null Hypothesis: The null hypothesis is a statement that the value of a population parameter (such as proportion, mean, or standard deviation) is equal to some claimed value. We either reject or fail to reject the null hypothesis. Null Hypothesis is denoted by H0.
* Alternate Hypothesis: The alternative hypothesis is the statement that the parameter has a value that is different from the claimed value. It is denoted by HA.

**Z-test:**

Z-test is a statistical method to determine whether the distribution of the test statistics can be approximated by a normal distribution. It is the method to determine whether two sample means are approximately the same or different when their variance is known and the sample size is large (should be >= 30).

When to Use Z-test:

* The sample size should be greater than 30. Otherwise, we should use the t-test.
* Samples should be drawn at random from the population.
* The standard deviation of the population should be known.
* Samples that are drawn from the population should be independent of each other.
* The data should be normally distributed, however for large sample sizes, it is assumed to have a normal distribution.

**Scikit-learn:**

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

The train\_test\_split() method from scikit-learn is used to split our data into train and test sets. First, we need to divide our data into features (X) and labels (y). The dataframe gets divided into X\_train, X\_test, y\_train, and y\_test. X\_train and y\_train sets are used for training and fitting the model.

* Train set: The training dataset is a set of data that was utilized to fit the model. The dataset on which the model is trained. This data is seen and learned by the model.
* Test set: The test dataset is a subset of the training dataset that is utilized to give an accurate evaluation of a final model fit.

**Statsmodels:**

statsmodels is a Python library that provides utilities for the estimation of several statistical models and includes extensive results and metrics for each estimator. In particular, statsmodels excel at generalized linear models (GLMs) which are far superior to scikit-learn’s implementation of ordinary least squares.

You can use the ztest() function from the statsmodels package to perform one sample and two sample z-tests in Python.

**IMPLEMENTATION:**

**Selected Dataset :**

<https://www.kaggle.com/datasets/akshaydattatraykhare/diabetes-dataset>

The above dataset consists of information about the diabetics stats of many patients. It has the values for various parameters such as Glucose level, Blood Pressure, Skin Thickness, Insulin level, BMI value and their Age. All the selected patients here are female and hence their number of pregnancies is also given.

1. First of all the dataset is loaded and read using the pandas library of python. Then we can have some basic information and statistics about the dataset using df.describe(), df.info(), df.isnull().sum() where the dataset is stored in df.
2. from sklearn.model\_selection import train\_test\_split

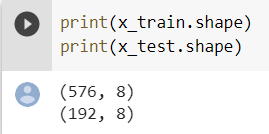
X =df.drop(columns=['Outcome'])

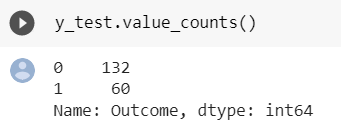
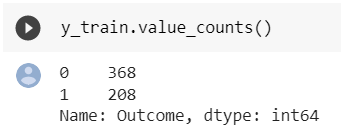
Y=df['Outcome']

x\_train, x\_test, y\_train, y\_test = train\_test\_split(X,Y,test\_size=0.25)

Here, we’ve split the data in train and test sets where Y holds the column ‘Outcome’ while X holds the rest of the data columns. The data is randomly splitted in sizes of 75% and 25% for train and test respectively.

1. Looking at the train and test sets counts.



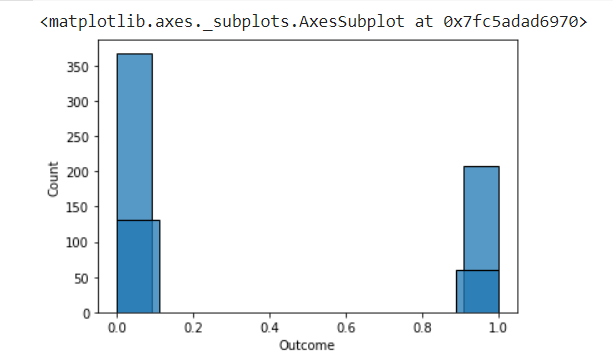
 

1. Bar plotting the Y train and test sets using seaborn

import seaborn as sns

sns.histplot(y\_train)

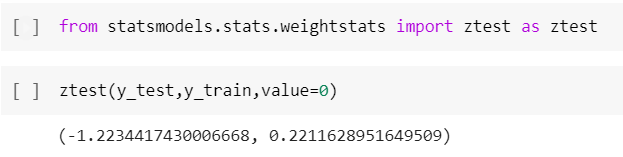
sns.histplot(y\_test)



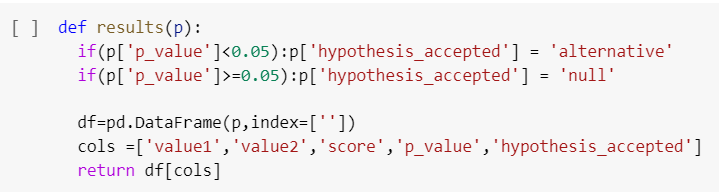
1. To perform the z-test, first we import the z-test function from the statsmodel package.

from statsmodels.stats.weightstats import ztest as ztest

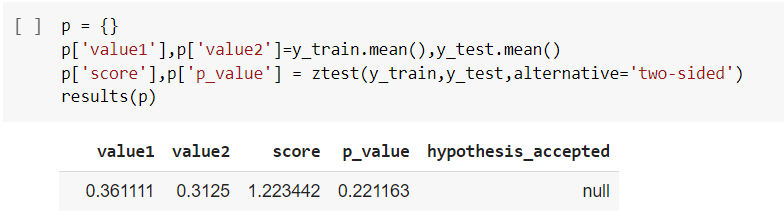
1. Now we’ll pass the train and test sets of Y to the z-test function to get two values. First one is the z-score and other o is the p value



1. Next, we’ve declared a function where we pass the ‘p’ value to determine the result of our hypothesis testing. The function basically takes the ‘p’ value and checks if it is greater than equal to 0.05 or less than 0.05 (0.05 is the level of significance in z-test when not specified). If it is greater or equal to 0.05, then our null hypothesis is accepted, else it is rejected.



1. Finally, we’ve declared ‘p’ as a dictionary where we’ve stored multiple values used to pass it to the ‘results’ function.



Our p value is 0.22 which is greater than 0.05 and hence our null hypothesis is accepted.

**CONCLUSION:**

In this experiment, we partitioned our data into train and test sets using the ‘train\_test\_split() method from scikit-learn’. Next we performed hypothesis testing using z-test on our data with the help of z-test() function from statsmethod package of python.