Sinhgad Institutes

Name of the Student:_____Roll no:_____

CLASS:- T.E.[I.T] **Division: A**

Course: - 2019

Subject: 314457: Data Science and Big Data Analytics Laboratory

PART_ B _Assignment No. 02

Marks: _____/10

Date of Performance: ___/____/_____ **Sign with Date: _____**

Part- B

ASSIGNMENT NO: 02

TITLE:

To Perform operations on data using Python on the Air quality and Heart Diseases data sets.

AIM:

To Perform the following operations using Python on the Air quality and Heart Diseases data sets

1. Data cleaning
2. Data integration
3. Data transformation
4. Error correcting
5. Data model building

OBJECTIVE:

1. To understand and apply the Analytical concept of Big data using Python.
2. To apply the Analytical concept of Big data using Python.

Software used: IDLE Shell 3.9.6 (Python 3.9.6)

THEORY:

Data Cleaning: Data cleaning is one part of data quality. The aim of Data Quality (DQ) is to have the following:

- Accuracy (data is recorded correctly)
- Completeness (all relevant data is recorded)
- Uniqueness (no duplicated data record)
- Timeliness (the data is not old)
- Consistency (the data is coherent)

Data cleaning attempts to fill in missing values, smooth out noise while identifying outliers, and correct inconsistencies in the data. Data cleaning is usually an iterative two-step process consisting of discrepancy detection and data transformation.

The process of data mining contains two steps in most situations. They are as follows:

- The first step is to perform audition on the source dataset to find the discrepancy.
- The second step is to choose the transformation to fix (based on the accuracy of the attribute to be modified and the closeness of the new value to the original value). This is followed by applying the transformation to correct the discrepancy.

Data Integration: Data integration combines data from multiple sources to form a coherent data store. The common issues here are as follows:

- Heterogeneous data: This has no common key
- Different definition: This is intrinsic, that is, same data with different definition, such as a different database schema
- Time synchronization: This checks if the data is gathered under same time periods
- Legacy data: This refers to data left from the old system
- Sociological factors: This is the limit of data gathering

There are several approaches that deal with the above issues:

- Entity identification problem: Schema integration and object matching are tricky. This referred to as the entity identification problem.
- Redundancy and correlation analysis: Some redundancies can be detected by correlation analysis. Given two attributes, such an analysis can measure how strongly one attribute implies the other, based on the available data.
- Tuples Duplication: Duplication should be detected at the tuple level to detect redundancies between attributes
- Data value conflict detection and resolution: Attributes may differ on the abstraction level, where an attribute...

Data Transformation: Data transformation is an approach to transform the original data to preferable data format for the input of certain data mining algorithms before the processing.

Data transformation routines convert the data into appropriate forms for mining. They're shown as follows:

- Smoothing: This uses binning, regression, and clustering to remove noise from the data
- Attribute construction: In this routine, new attributes are constructed and added from the given set of attributes
- Aggregation: In this summary or aggregation, operations are performed on the data
- Normalization: Here, the attribute data is scaled so as to fall within a smaller range
- Discretization: In this routine, the raw values of a numeric attribute are replaced by interval label or conceptual label
- Concept hierarchy generation for nominal data: Here, attributes can be generalized to higher level concepts

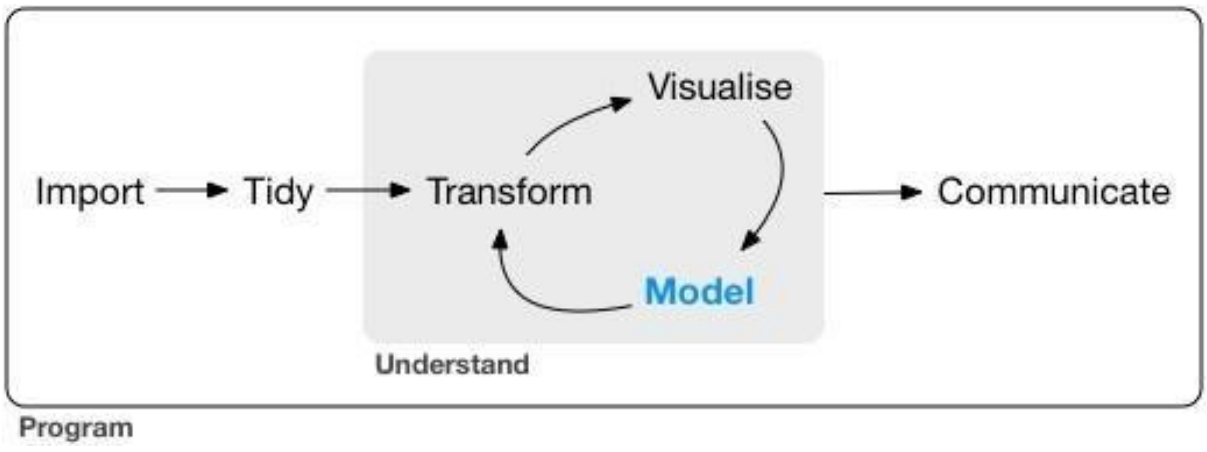
Code in R

```
data_dat$trans_Y<- sqrt(data_dat$Y)
```

The above code tells R to create a new variable (or column in the data_dat dataset) named "trans_Y" that is equal to the square root of the original response variable Y.

Data model building:

The goal of a model is to provide a simple low-dimensional summary of a dataset. Ideally, the model will capture true "signals" (i.e. patterns generated by the phenomenon of interest), and ignore "noise" (i.e. random variation that you're not interested in). Here we only cover "predictive" models, which, as the name suggests, generate predictions and "data discovery" models. These models don't make



predictions, but instead help you discover interesting relationships within your data. (These two categories of models are sometimes called supervised and unsupervised).

If you are serious about doing a confirmatory analysis, one approach is to split your data into three pieces before you begin the analysis:

- 60% of your data goes into a training (or exploration) set. You're allowed to do anything you like with this data: visualize it and fit tons of models to it.
- 20% goes into a query set. You can use this data to compare models or visualizations by hand, but you're not allowed to use it as part of an automated process.
- 20% is held back for a test set. You can only use this data ONCE, to test your final model.

This partitioning allows you to explore the training data, occasionally generating candidate hypotheses that you check with the query set. When you are confident you have the right model, you can check it once with the test data.

There are two parts to a model:

1. First, you define a family of models that express a precise, but generic, pattern that you want to capture. For example, the pattern might be a straight line, or a quadatric curve. You will express the model family as an equation like y = a_1 * x + a_2 or y = a_1 * x ^ a_2. Here, x and y are known variables from your data, and a_1 and a_2 are parameters that can vary to capture different patterns.
2. Next, you generate a fitted model by finding the model from the family that is the closest to your data. This takes the generic model family and makes it specific, like y = 3 * x + 7 or y = 9 * x ^ 2.

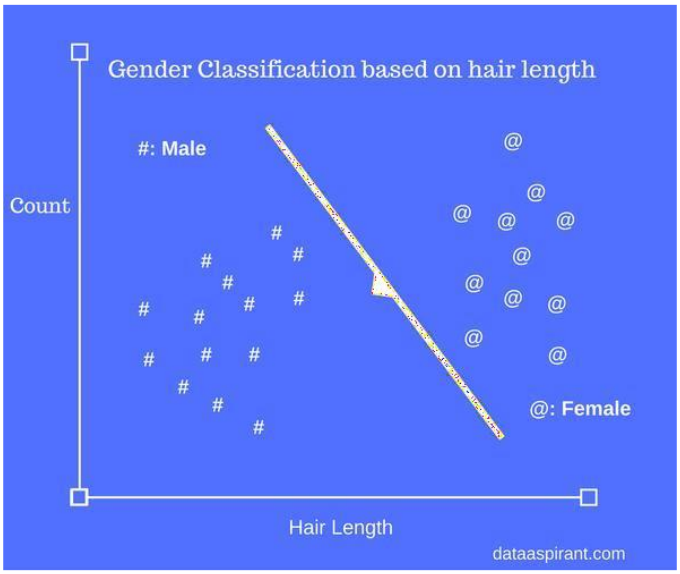
For model building any technique from machine learning can be used such as,

CLASSIFICATION:

The main goal of classification is to predict the target class (Yes/ No). If the trained model is for predicting any of two target classes. It is known as binary classification. Considering the student profile to predict whether the student will pass or fail. Considering the customer, transaction details to predict whether he will buy the new product or not. These kind problems will be addressed with binary classification. If we have to predict more the two target classes it is known as multi- classification. Considering all subject details of a student to predict which subject the student will

score more. Identifying the object in an image. These kind problems are known as multi-classification problems.

User Profile



Classification Algorithms:

- Linear classifiers
  - Logistic regression
  - Naive Bayes classifier
  - Fisher's linear discriminant
- Support vector machines
  - Least squares support vector machines
- Quadratic classifiers
- Kernel estimation
  - k-nearest neighbor (KNN)
- Decision trees o Random forests
- Neural networks
- Learning vector quantization

Classification Example:

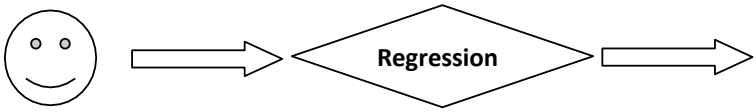
Suppose from your past data (train data) you come to know that your best friend likes the above movies. Now one new movie (test data) released. Hopefully, you want to know your best friend like it or not. If you strongly confirmed about the chances of your friend like the move. You can take your friend to a movie this weekend.

If you clearly observe the problem it is just whether your friend like or not. Finding a solution to this type of problem is called as classification. This is because we are classifying the things to their belongings (yes or no, like or dislike). Here we are forecasting target class (classification) and the other thing this classification belongs to supervised learning. This is because you are learning this from your train data.

In this case, the problem is a binary classification in which we have to predict whether output belongs to class 1 or class 2 (class 1: yes, class 2: no) We can use classification for predicting more classes too. Like (Colour Prediction: RED, GREEN, BLUE, YELLOW, And ORANGE)

REGRESSION

The main goal of regression algorithms is the predict the discrete or a continues value. In some cases, the predicted value can be used to identify the linear relationship between the attributes. Suppose the increase in the product advantage budget will increase the product sales. Based on the problem difference regression algorithms can be used. Some of the basic regression algorithms are linear regression, polynomial regression, and multiple regressions.



Predicting Student marks

PERCENTAGE

User Profile

Regression Example:

Suppose from your past data (train data) you come to know that your best friend likes some movies. You also know how many times each particular movie seen by your friend. Now one new movie (test data) released. Now you're are going to find how many times this newly released movie will your friend watch. It could be 5 times, 6 times, 10

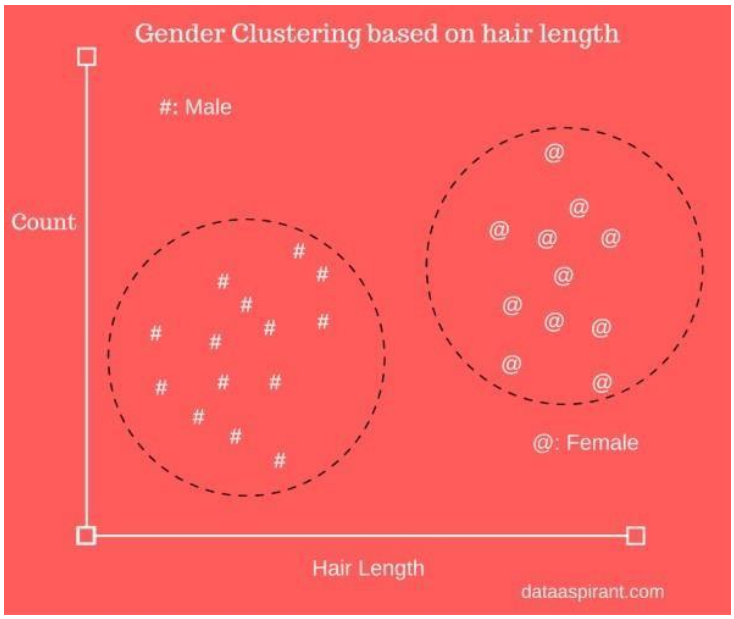
times etc…

If you clearly observe the problem is about finding the count, sometimes we can say this as predicting the value. Here we are forecasting a value (Prediction) and the other thing this prediction also belongs to supervised learning. This is because you are learning this from you train data.

- If forecasting target class (Classification)
- If forecasting a value (Regression)

CLUSTRING:

In clustering the idea is not to predict the target class as like classification, it's more ever trying to group the similar kind of things by considering the most satisfied condition all the items in the same group should be similar and no two different group items should not be similar. To group the similar kind of items in clustering, different similarity measures could be used.



Group items Examples:

- While grouping similar language type documents (Same language documents are one group.)
- While categorizing the news articles (Same news category(Sport) articles are one group)

With clustering genders based on hair length example. To determine gender, different similarity measure could be used to categorize male and female genders. This could be

done by finding the similarity between two hair lengths and keep them in the same group if the similarity is less (Difference of hair length is less). The same process could continue until all the hair length properly grouped into two categories.

CLUSTERING ALGORITHMS

Clustering algorithms can be classified into two main categories Linear clustering algorithms and Non-linear clustering algorithms.

Linear clustering algorithm

- k-means clustering algorithm
- Fuzzy c-means clustering algorithm
- Hierarchical clustering algorithm
- Gaussian (EM) clustering algorithm
- Quality threshold clustering algorithm

Non-linear clustering algorithm

- MST based clustering algorithm
- kernel k-means clustering algorithm
- Density-based clustering algorithm

Application of Clustering Algorithms

- Recommender systems
- Anomaly detection
- Human genetic clustering
- Genom Sequence analysis
- Analysis of antimicrobial activity
- Grouping of shopping items
- Search result grouping
- Slippy map optimization
- Crime analysis
- Climatology

CONCLUSION:

After the study of this assignment we are familiar with the data modeling operations using Python.

Write Short Answers for Following Questions

1. What do you understand from the term data cleaning?
2. What is Data Integration?
3. What are the benefits of data integration?
4. Is Data integration And ETL programming is same?
5. Mention what is the responsibility of a Data analyst?
6. Mention what is data cleansing?

Viva Questions

1. List of some best tools that can be useful for data-analysis?
2. List out some common problems faced by data analyst?
3. Which functions are included in package caret, e1071, catools, class and gmodels name it?
4. How to handle missing values?
5. How do you create log linear models in R language?
6. What is meant by K-nearest neighbor explain with example?
7. Write a function in R language to replace the missing value in a vector with the mean of that vector.
8. Which function is used to create histogram visualization in R programming language?

Python Program

Data Cleaning

 # Data cleaning 1

```
punctuations = '''!()-{}[];:"'\,<>./?@#$%^&*_~`'''
my_str = "Hello!!!, he said ...and went."
no_punct = ""
for char in my_str:
   if(char not in punctuations):
      no_punct = no_punct + char

print(no_punct)
```

 # Data cleaning 2

```
import re
s = "Hello!!!, he said ...and went."
s = re.sub(r'[^\w\s]','',s)
# not of word and space character
print(s)
```

 # Tokanization

```
import nltk
nltk.download('punkt')
from nltk.tokenize import sent_tokenize, word_tokenize
example_text = "Hello Mr. Pravin, how are you doing today? The weather in lonavala is rainy. The sky is full of cloud."
print(sent_tokenize(example_text))
print(word_tokenize(example_text))
```

 # Stopwords

```
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
stop_words = stopwords.words('english')
print(stop_words)
```

 # Steming

```
import nltk
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
stemmer = PorterStemmer()
input_str = "There are several types of stemming algorithms."
input_str = nltk.word_tokenize(input_str)
print (input_str)
for word in input_str:
   print(stemmer.stem(word))
```

 # Lamitization

```
import nltk
nltk.download('wordnet')
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
lemmatizer = WordNetLemmatizer()
input_str = "There are several cities with mice."
input_str = nltk.word_tokenize(input_str)
print (input_str)
for word in input_str:
   print(lemmatizer.lemmatize(word))
```

Operation with missing value

```
import pandas as pd
import numpy as np
#importing the dataset
df = pd.read_csv("records.csv")
df.head()
print(df)
#Missing values are usually represented in the form of Nan or null or None in the dataset.
df.info()
#Let's try fitting the data using logistic regression.
```

#1. Deleting the column with missing data

```
updated_df = df.dropna(axis=1)
print(updated_df)
updated_df.info()
```

#2. Deleting the row with missing data

```
updated_df = df.dropna(axis=0)
print(updated_df)
updated_df.info()
```

#3. Filling the Missing Values – Imputation

```
updated_df = df
updated_df['Salary']=updated_df['Salary'].fillna(updated_df['Salary'].mean())
print(updated_df)
```

#4. Imputation with an additional column

```
updated_df = df
updated_df['Salaryismissing'] = updated_df['Salary'].isnull()
print(updated_df)
```

#5. Filling with a Regression Model

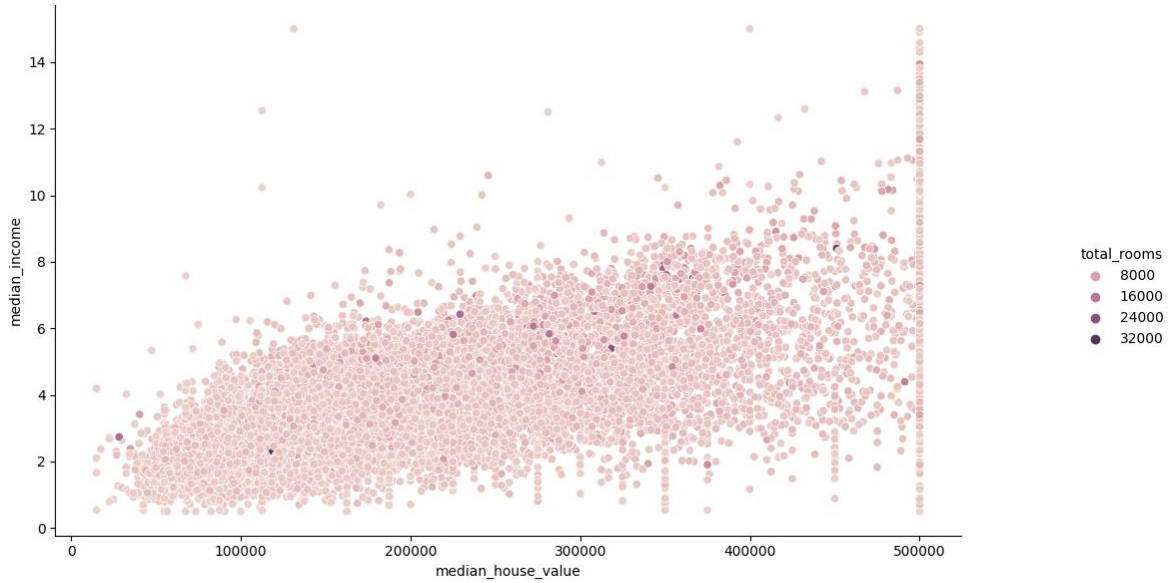
```
testdf = df[df['Salary'].isnull()==True]
traindf = df[df['Salary'].isnull()==False]
traindf.drop("Salary",axis=1,inplace=True)
testdf.drop("Salary",axis=1,inplace=True)
print(traindf)
```

Data Model

House Prize Prediction

```
#Data model_prediction_house_prize
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

data=pd.read_csv('house.csv')

print(data.columns) # This will show all the column names
print(data.head(10)) # Show first 10 records of dataframe
print(data.describe()) #You can look at summary of numerical fields by using describe()
function
print(data.shape)
print(data.isnull().sum())

#sns.relplot(x='median_house_value', y= 'total_bedrooms', data=data)
#sns.relplot(x='median_house_value', y= 'total_rooms', data=data)
#sns.relplot(x='median_house_value', y= 'population', data=data)
#sns.relplot(x='median_house_value', y= 'median_income', data=data)
#sns.relplot(x='median_house_value', y= 'households', data=data)
sns.relplot(x='median_house_value', y= 'median_income', hue= 'total_rooms', data=data)
plt.show()

#Model
train =data.drop(['median_house_value','longitude','latitude','housing_median_age'],
axis=1)
test =data['median_house_value']
x_train, x_test, y_train, y_test = train_test_split(train, test, test_size=0.3,
random_state=2)
```

```
regr= LinearRegression()
regr.fit(x_train, y_train)
pred = regr.predict(x_test)
print(pred)
print(regr.score(x_test, y_test))
```



House Prize Vs Income and total rooms plot

```
Index(['longitude', 'latitude', 'housing_median_age', 'total_rooms',
       'total_bedrooms', 'population', 'households', 'median_income',
       'median_house_value'],
      dtype='object')
```

	longitude	latitude	...	median_income	median_house_value
0	-114.31	34.19	...	1.4936	66900
1	-114.47	34.40	...	1.8200	80100
2	-114.56	33.69	...	1.6509	85700
3	-114.57	33.64	...	3.1917	73400
4	-114.57	33.57	...	1.9250	65500
5	-114.58	33.63	...	3.3438	74000
6	-114.58	33.61	...	2.6768	82400
7	-114.59	34.83	...	1.7083	48500
8	-114.59	33.61	...	2.1782	58400

```
9   -114.60    34.83  ...       2.1908           48100

[10 rows x 9 columns]

       longitude     latitude  ...  median_income  median_house_value
count  17000.000000  17000.000000  ...  17000.000000       17000.000000
mean    -119.562108     35.625225  ...      3.883578      207300.912353
std        2.005166      2.137340  ...      1.908157      115983.764387
min     -124.350000     32.540000  ...      0.499900       14999.000000
25%     -121.790000     33.930000  ...      2.566375      119400.000000
50%     -118.490000     34.250000  ...      3.544600      180400.000000
75%     -118.000000     37.720000  ...      4.767000      265000.000000
max     -114.310000     41.950000  ...     15.000100      500001.000000

[8 rows x 9 columns]
(17000, 9)

longitude                 0
latitude                  0
housing_median_age        0
total_rooms               0
total_bedrooms            0
population                0
households                0
median_income             0
median_house_value        0
dtype: int64

[153579.02594907 221485.91195112  91641.96769418 ... 184433.07649541
 224004.49656511 185189.89991136]

0.541520003241859
```

House Prize prediction model in Power BI