



National University of Computer & Emerging Sciences,
Karachi



Fall 2024, lab manual – 01
Introduction to machine learning

Course code :	Machine learning
Instructor :	Usama Bin Umar

Objective

1. Introduction to machine learning, ai and traditional programming
2. Prerequisite to be an expert in machine learning (python)
3. Fundamental knowledge of python programming
4. How AI / ML is differ from traditional programming?

Artificial intelligence

The term Artificial Intelligence was first coined decades ago in the year 1956 by John McCarthy at the Dartmouth conference. (Rajaraman, 2014) He defined AI as: “The science and engineering of making intelligent machines.” In other words, Artificial Intelligence is the science of getting machines to think and make decisions like humans. In the recent past, AI has been able to accomplish this by creating machines and robots that have been used in a wide range of fields including healthcare, robotics, marketing, business analytics and many more.



How to make machine intelligent?

We can make intelligent machine using the concept of AI. There are different approaches that can help us develop intelligent machines. Artificial intelligence is the concept to make our system intelligent to take decision just like human. So to make the system intelligent there are various ways like machine learning, deep learning, computer vision, NLP. These are the domains of AI

Machine learning:

Machine learning is a field of artificial intelligence that allows systems to learn and improve from experience without being explicitly programmed.

Deep learning:

Deep learning is a method in artificial intelligence (ai) that teaches computers to process data in a way that is inspired by the human brain

Computer vision:

Computer vision applications use input from sensing devices, artificial intelligence, machine learning, and deep learning to replicate the way the human vision.

Natural language processing:

Natural language processing (NLP) is a form of artificial intelligence (ai) that allows computers to understand human language.

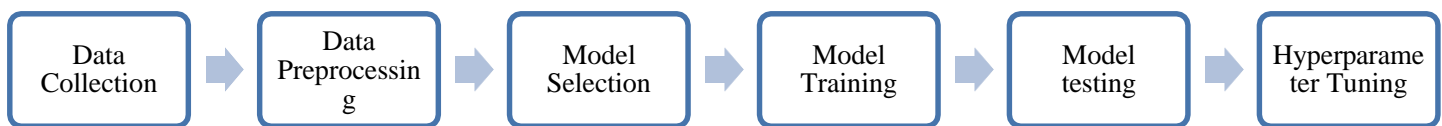
Process of machine learning

Figure 1 : Life cycle of Machine Learning

Basic python data for storing collection

To store collection, we either use list, tuple, set, range and dictionary in python.

List, tuple, set, and range stores 1d data. Dictionary is used to store key and value pairs.

Basic data type	Collection data type
<pre>int_data = 17 float_data= 12.6 complex_data = 14 + 5j string_data = 'Machine Learning'</pre> <p>Output</p> <pre>Data : 17, Type : <class 'int'> Data : 12.6, Type : <class 'float'> Data : (14+5j), Type : <class 'complex'> Data : Machine Learning, Type : <class 'str'></pre>	<pre>list_data = [11,12,13,14,15, 'orange'] tuple_data = (12,15,17 , 'kiran') set_data = {'ali', 'hassan', 'ayesha' , 12 ,18.6} range_data = range(1,11)</pre> <p>Output</p> <pre>List Data : [11, 12, 13, 14, 15, 'orange'] Tuple Data : (12, 15, 17, 'kiran') Set Data : {18.6, 'hassan', 'ali', 'ayesha', 12} Range Data : range(1, 11)</pre>

In machine learning, we perform exploratory data analysis or preprocess our data so often we require it to perform on single column or on multiple columns.

Pandas allows users to manipulate and analyze data.

Pandas provides two data structures that shape data into a readable form: 1) series 2) data frame

We know that python stored their collections in the form of list, tuple, set and dictionary. So why we need to create or work on series and dataframe. What its relation with basic data structures.

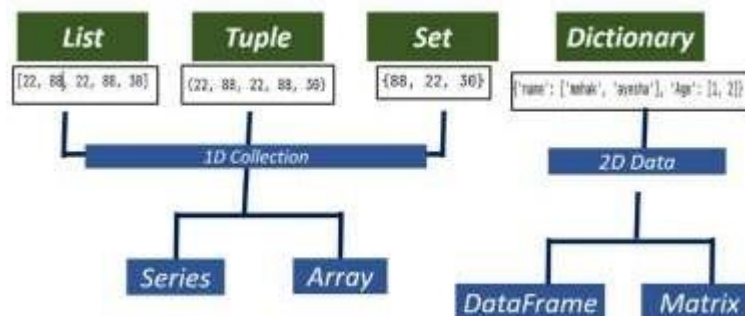


Figure 2 : 1D and 2D data

The basic data types of python includes integer, float and string to store some value. But when it comes to store a collection, collection of names, collection of number, collection of year etc, we used some additional data types like dictionary, list, set, and tuple. To store a collection of one dimensional information we mostly use list, tuple and set. Dictionary is used to store key value pair when we have 2d data, when there are multiple keys or attributes associated in our collection data. We can store 1d data in the form of list, tuple, set and array. 2d data in the form of dictionary, Dataframe and matrix.

List	Tuple	Set	Series
<pre>[1] l=[1,2,3,4,5] print(l)</pre> <p>[1, 2, 3, 4, 5]</p>	<pre>[2] t=[1,2,3,4,5] print(t)</pre> <p>[1, 2, 3, 4, 5]</p>	<pre>[3] s=[1,2,3,4,5] print(s)</pre> <p>[1, 2, 3, 4, 5]</p>	<pre>[9] import pandas as pd df=pd.Series(t) print(df)</pre> <pre>0 1 1 2 2 3 3 4 4 5 dtype: int64</pre>

Dictionary	Dataframe
<pre>d={'name': ['mehak', 'ali'], 'age': [22, 25]} print(d)</pre> <pre>{'name': ['mehak', 'ali'], 'age': [22, 25]}</pre>	<pre>data=pd.DataFrame(d) print(data)</pre> <pre> name age 0 mehak 22 1 ali 25</pre>

We can perform numerical operation and data preprocessing easily on series and dataframe but it takes a more bit challenging to perform advance computation on basic python data types.

Series

A panda's series is a one-dimensional data structure that comprises of a key-value pair. It is similar to a python dictionary. To initialize a series, use pandas.series():

How to create series

Program	Output
<pre>import pandas as pd data = [1,2,66,77,'mehak'] series = pd.Series(data) print(series)</pre>	<pre>0 1 1 2 2 66 3 77 4 mehak dtype: object</pre>

Dataframe:

A pandas Dataframe is a two-dimensional data-structure that can be thought of as a spreadsheet. A Dataframe can also be thought of as a combination of two or more series. To initialize a Dataframe, use pandas.dataframe

How to create Dataframe?

Program	Output
<pre>df = {'Name': ['mehak', 'ali', 'hassan'], 'Age' : [50,62,75]} dataframe = pd.DataFrame(df) print(dataframe)</pre>	<pre> Name Age 0 mehak 50 1 ali 62 2 hassan 75</pre>

How to read data?

Tabular data is available in different formats like csv, xlsx, json etc. We can read data to perform data analysis to make it useful to prepare machine learning model. There is a read method available in pandas that is used to read different file formats.

```
data = pd.read_csv('heart.csv')
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
...
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

How to store dataset?

We can store our data after applying any transformation or after creating a manual dataframe using “to_csv” or “to” function.

```
dataframe.to_csv('newdata.csv')
```

Data preprocessing

Data preprocessing is an important step in the data mining process. It refers to the cleaning, transforming, and integrating of data in order to make it ready for analysis. The goal of data preprocessing is to improve the quality of the data and to make it more suitable for a machine learning model.

A) displaying top most and bottom records

Top records

```
data.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

Bottom records

```
data.tail()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

We can display desired number of top most and bottom most records by declaring total records length in tail and head method.

Top 3 records														
<pre>data.head(3)</pre>														
	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0

Bottom 4 records

```
data.tail(4)
```

B) how to check column names in data and total number of records

We can check features/attributes names using columns method in pandas.

Column names
<pre>data.columns</pre>
<pre>Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'], dtype='object')</pre>

Checking total number of records and columns`data.shape``(1025, 14)`**C) Checking data statistics**

We can explore our dataset using different methods. We can check data null values , mean , median , standard deviation , percentiles , type of data and non null count.

Checking non null count	Checking null values	Checking data type
<code>data.count()</code>	<code>data.isnull().sum()</code>	<code>data.dtypes</code>
<pre> age 1025 sex 1025 cp 1025 trestbps 1025 chol 1025 fbs 1025 restecg 1025 thalach 1025 exang 1025 oldpeak 1025 slope 1025 ca 1025 thal 1025 target 1025 dtype: int64 </pre>	<pre> age 0 sex 0 cp 0 trestbps 0 chol 0 fbs 0 restecg 0 thalach 0 exang 0 oldpeak 0 slope 0 ca 0 thal 0 target 0 dtype: int64 </pre>	<pre> age int64 sex int64 cp int64 trestbps int64 chol int64 fbs int64 restecg int64 thalach int64 exang int64 oldpeak float64 slope int64 ca int64 thal int64 target int64 dtype: object </pre>

Count method is used to count the total number of filled records that contains information in each attribute. Isnull method is used to check total number of empty or null records, it gives output in true and false so we combine it with sum() method that will count total true and false results and finally yields missing records in numbers. Dtypes method is used to check the data type of attributes. Dtype and isnull method to explore whether there is any categorical feature or missing record is available that we need to handle it before passing this data to machine learning model.

Info method	Checking null values
<code>data.info()</code>	<code>data.describe()</code>

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
#   Column      Non-Null Count  Dtype
---  ---
0    age        1025 non-null   int64
1    sex         1025 non-null   int64
2    cp          1025 non-null   int64
3    trestbps    1025 non-null   int64
4    chol        1025 non-null   int64
5    fbs         1025 non-null   int64
6    restecg     1025 non-null   int64
7    thalach     1025 non-null   int64
8    exang       1025 non-null   int64
9    oldpeak     1025 non-null   float64
10   slope       1025 non-null   int64
11   ca          1025 non-null   int64
12   thal        1025 non-null   int64
13   target      1025 non-null   int64
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

	age	sex	cp	trestbps	chol	fbs
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000
mean	54.434146	0.695610	0.942439	131.611707	246.000000	0.149268
std	9.072290	0.460373	1.029641	17.516718	51.59251	0.356527
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000
25%	48.000000	0.000000	0.000000	120.000000	211.000000	0.000000
50%	56.000000	1.000000	1.000000	130.000000	240.000000	0.000000
75%	61.000000	1.000000	2.000000	140.000000	275.000000	0.000000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000

Info method gives information regarding Dataframe class, number of records, and total #of columns, count, dtypes used in dataset and memory usage. Describe function gives information of mean, median, maximum, minimum, and standard deviation, count 25%, 50% and 75%.

D) How to remove columns or rows in a data

Removing columns

```
cdata= data.drop(['thal',axis=1])
cdata
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	0
...

Removing rows

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

Drop method is used to remove records and columns in a dataset , by declaring axis = 1 we drop columns and by declaring axis =0 we remove records at particular position

E) How to add columns or rows in a data

Adding columns

```
total_5_records = data.head(5)
names = ['mehak' , 'areej' , 'ali' , 'hadi' , 'sara']
total_5_records.insert(0,'name',names)
total_5_records
```

	name	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	mehak	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	areej	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	ali	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	hadi	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	sara	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

Adding rows

```
#adding rows
new_row = {'age':25, 'sex':1, 'cp':3,'trestbps':125 , 'chol':170, 'fbs':12,'restecg':1,
           'thalach': 160, 'exang':1, 'oldpeak':1.2 , 'slope': 1, 'ca': 2, 'thal':2 , 'target':1}

data = data.append(new_row, ignore_index=True) #ignore_index by-default=False. If True,
data
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52.0	1.0	0.0	125.0	212.0	0.0	1.0	168.0	0.0	1.0	2.0	2.0	3.0	0.0
1	53.0	1.0	0.0	140.0	203.0	1.0	0.0	155.0	1.0	3.1	0.0	0.0	3.0	0.0
2	70.0	1.0	0.0	145.0	174.0	0.0	1.0	125.0	1.0	2.6	0.0	0.0	3.0	0.0
3	61.0	1.0	0.0	148.0	203.0	0.0	1.0	161.0	0.0	0.0	2.0	1.0	3.0	0.0
4	62.0	0.0	0.0	138.0	294.0	1.0	1.0	106.0	0.0	1.9	1.0	3.0	2.0	0.0
...
1021	60.0	1.0	0.0	125.0	268.0	0.0	0.0	141.0	1.0	2.8	1.0	1.0	3.0	0.0
1022	47.0	1.0	0.0	110.0	275.0	0.0	0.0	118.0	1.0	1.0	1.0	1.0	2.0	0.0
1023	50.0	0.0	0.0	110.0	254.0	0.0	0.0	159.0	0.0	0.0	2.0	0.0	2.0	1.0
1024	54.0	1.0	0.0	120.0	186.0	0.0	1.0	113.0	0.0	1.4	1.0	1.0	3.0	0.0
1025	25.0	1.0	3.0	125.0	170.0	12.0	1.0	160.0	1.0	1.2	1.0	2.0	2.0	1.0

1026 rows x 14 columns

How to reset and set index?

Re-setting index

```
data.reset_index(drop=False,inplace=True)
```

	level_0	index	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	0	0	52.0	1.0	0.0	125.0	212.0	0.0	1.0	168.0	0.0	1.0	2.0	2.0	3.0	0.0
1	1	1	53.0	1.0	0.0	140.0	203.0	1.0	0.0	155.0	1.0	3.1	0.0	0.0	3.0	0.0
2	2	2	70.0	1.0	0.0	145.0	174.0	0.0	1.0	125.0	1.0	2.6	0.0	0.0	3.0	0.0
3	3	3	61.0	1.0	0.0	148.0	203.0	0.0	1.0	161.0	0.0	0.0	2.0	1.0	3.0	0.0
4	4	4	62.0	0.0	0.0	138.0	294.0	1.0	1.0	106.0	0.0	1.9	1.0	3.0	2.0	0.0

Setting index

```
total_5_records = total_5_records.set_index('name')
total_5_records
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
name														
mehak	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
areej	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
ali	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
hadi	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
sara	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

How to sort values

```
data.sort_values('age',inplace=True)
data.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
1025	25.0	1.0	3.0	125.0	170.0	12.0	1.0	160.0	1.0	1.2	1.0	2.0	2.0	1.0
64	29.0	1.0	1.0	130.0	204.0	0.0	0.0	202.0	0.0	0.0	2.0	0.0	2.0	1.0
668	29.0	1.0	1.0	130.0	204.0	0.0	0.0	202.0	0.0	0.0	2.0	0.0	2.0	1.0
60	29.0	1.0	1.0	130.0	204.0	0.0	0.0	202.0	0.0	0.0	2.0	0.0	2.0	1.0
118	29.0	1.0	1.0	130.0	204.0	0.0	0.0	202.0	0.0	0.0	2.0	0.0	2.0	1.0

Sorting the values with respect to age , the dataset has been sorted by ascending order of ages. We can sort our dataset based on different attributes.

Changing values of records at particular column

```
data['sex'][data['sex'] == 0] = 'female'
data['sex'][data['sex'] == 1] = 'male'

data['slope'][data['slope'] == 0] = 'upsloping'
data['slope'][data['slope'] == 1] = 'flat'
data['slope'][data['slope'] == 2] = 'downsloping'
data.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52.0	male	0.0	125.0	212.0	0.0	1.0	168.0	0.0	1.0	flat	2.0	3.0	0.0
1	53.0	male	0.0	140.0	203.0	1.0	0.0	155.0	1.0	3.1	upsloping	0.0	3.0	0.0
2	70.0	male	0.0	145.0	174.0	0.0	1.0	125.0	1.0	2.6	upsloping	0.0	3.0	0.0
3	61.0	male	0.0	148.0	203.0	0.0	1.0	161.0	0.0	0.0	flat	1.0	3.0	0.0
4	62.0	female	0.0	138.0	294.0	1.0	1.0	106.0	0.0	1.9	upsloping	3.0	2.0	0.0

Extracting specific columns

Extracting single column	Extracting more than one column	Extracting using pop
<code>data['age']</code>	<code>data[['age', 'ca']]</code>	<code>y=data.pop('target')</code> <code>y</code>
<pre>0 52.0 1 53.0 2 70.0 3 61.0 4 62.0 ... 1021 60.0 1022 47.0 1023 50.0 1024 54.0 1025 25.0</pre>	<pre> age ca 0 52.0 2.0 1 53.0 0.0 2 70.0 0.0 3 61.0 1.0 4 62.0 3.0 1021 60.0 1.0 1022 47.0 1.0 1023 50.0 0.0 1024 54.0 1.0 1025 25.0 2.0</pre> <p>1026 rows x 2 columns</p>	<pre>0 0.0 1 0.0 2 0.0 3 0.0 4 0.0 ... 1021 0.0 1022 0.0 1023 1.0 1024 0.0 1025 1.0</pre> <p>Name: target,</p>

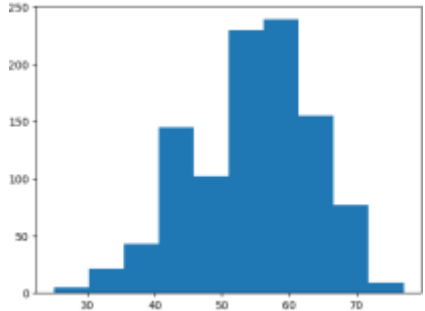
Extracting desired rows and columns

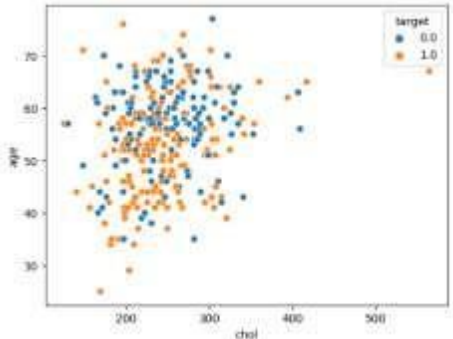
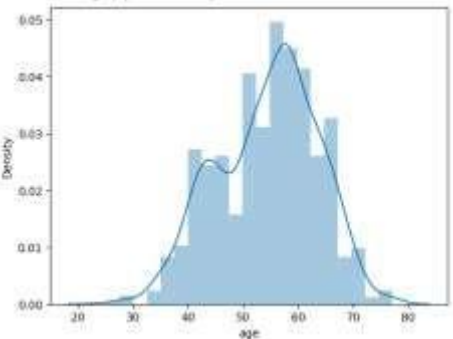
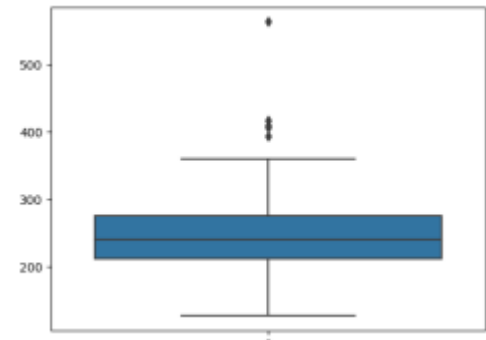
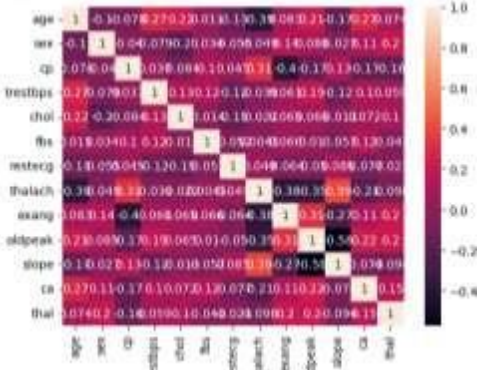
Iloc method	Loc method																																																																																		
<code>data.iloc[0:6 ,0:5]</code>	<code>data.loc[0:6 , 'age': 'trestbps']</code>																																																																																		
<table><tr><th></th><th>age</th><th>sex</th><th>cp</th><th>trestbps</th><th>chol</th></tr><tr><td>0</td><td>52.0</td><td>male</td><td>0.0</td><td>125.0</td><td>212.0</td></tr><tr><td>1</td><td>53.0</td><td>male</td><td>0.0</td><td>140.0</td><td>203.0</td></tr><tr><td>2</td><td>70.0</td><td>male</td><td>0.0</td><td>145.0</td><td>174.0</td></tr><tr><td>3</td><td>61.0</td><td>male</td><td>0.0</td><td>148.0</td><td>203.0</td></tr><tr><td>4</td><td>62.0</td><td>female</td><td>0.0</td><td>138.0</td><td>294.0</td></tr><tr><td>5</td><td>58.0</td><td>female</td><td>0.0</td><td>100.0</td><td>248.0</td></tr></table>		age	sex	cp	trestbps	chol	0	52.0	male	0.0	125.0	212.0	1	53.0	male	0.0	140.0	203.0	2	70.0	male	0.0	145.0	174.0	3	61.0	male	0.0	148.0	203.0	4	62.0	female	0.0	138.0	294.0	5	58.0	female	0.0	100.0	248.0	<table><tr><th></th><th>age</th><th>sex</th><th>cp</th><th>trestbps</th></tr><tr><td>0</td><td>52.0</td><td>male</td><td>0.0</td><td>125.0</td></tr><tr><td>1</td><td>53.0</td><td>male</td><td>0.0</td><td>140.0</td></tr><tr><td>2</td><td>70.0</td><td>male</td><td>0.0</td><td>145.0</td></tr><tr><td>3</td><td>61.0</td><td>male</td><td>0.0</td><td>148.0</td></tr><tr><td>4</td><td>62.0</td><td>female</td><td>0.0</td><td>138.0</td></tr><tr><td>5</td><td>58.0</td><td>female</td><td>0.0</td><td>100.0</td></tr><tr><td>6</td><td>58.0</td><td>male</td><td>0.0</td><td>114.0</td></tr></table>		age	sex	cp	trestbps	0	52.0	male	0.0	125.0	1	53.0	male	0.0	140.0	2	70.0	male	0.0	145.0	3	61.0	male	0.0	148.0	4	62.0	female	0.0	138.0	5	58.0	female	0.0	100.0	6	58.0	male	0.0	114.0
	age	sex	cp	trestbps	chol																																																																														
0	52.0	male	0.0	125.0	212.0																																																																														
1	53.0	male	0.0	140.0	203.0																																																																														
2	70.0	male	0.0	145.0	174.0																																																																														
3	61.0	male	0.0	148.0	203.0																																																																														
4	62.0	female	0.0	138.0	294.0																																																																														
5	58.0	female	0.0	100.0	248.0																																																																														
	age	sex	cp	trestbps																																																																															
0	52.0	male	0.0	125.0																																																																															
1	53.0	male	0.0	140.0																																																																															
2	70.0	male	0.0	145.0																																																																															
3	61.0	male	0.0	148.0																																																																															
4	62.0	female	0.0	138.0																																																																															
5	58.0	female	0.0	100.0																																																																															
6	58.0	male	0.0	114.0																																																																															

We can extract specific columns using square brackets approach `data['columnname']` . This approach extract the desired information using attributes name. But if you want to extract data based on records and features then we use loc and iloc method. **Loc** method is used to extract rows and columns using their names. Whereas **iloc** method extract data based on integer or indexing of row and columns. **Pop** method extract desired column by removing it from original dataset and storing into another location.

Data visualization

Data visualization is a graphical representation of quantitative information and data by using visual elements like graphs, charts, and maps. Data visualization is used to explore data in visual form.

Graph	Program	Output
Hist	<pre>import matplotlib.pyplot as plt plt.hist(data['age']) plt.show()</pre>	

Scatterplot	<pre>import seaborn as sn x1=data['chol'] x2=data['age'] sn.scatterplot(x=x1,y=x2,hue=data['target'])</pre>	
Distplot	<pre>sn.distplot(data['age'])</pre>	
Boxplot	<pre>sn.boxplot(data['chol'])</pre>	
Corr plot	<pre>sn.heatmap(data=data.corr(),annot=True) plt.figure(figsize=(14,14))</pre>	

Reference

- RAJARAMAN, V. 2014. JohnMcCarthy—Father of artificial intelligence. *Resonance*, 19, 198-207.

LAB TASKS

Task # 1

Create a dataset for representing 5 attributes that reflects the cause of a failure in examination (using an approach other than List to List and Dic to List.). Three most common attributes are ('test_score', 'writing skills', 'reading skill') decide two attributes by yourself. Fill at least 10 records. Come up with 2 more different idea to Create Data Frame

Task # 2

How to organize your code (Create a Text block for Importing Libraries in next cell import required library now Create a Heading for Data Preprocessing and the add a new cell for Data analysis. Create headings for each cell)

- A. Download the dataset from [Here](#)
- B. Import your dataset in Cola or Jupyter notebook
- C. Find out the unique category in target variable
- D. Find out the total numbers of columns in entire dataset, Check out the first 50 records of the dataset + bottom 30 records. Find its Information related to memory size , its dimension and Explore the size and dimension of your dataset
- E. Find the numbers of columns and total number of samples
- F. Find out the number of records for each unique category of target variable using numerical and graphical visualization.
- G. Extract sample from 550th sample to 900th sample and take these features Gender to Coughing of Blood using Loc Method and store in a new variable. Save this dataset in a new place and apply statistical analysis on these extracted records In statistical analysis you have to find out the mean , median , mode , standard deviation , minimum an maximum
- H. Extract sample of last 20 records , take 5 features in consideration from Shortness of Breath to Frequent cold column using iLoc Method Save this dataset in a new place and apply statistical analysis on these extracted records
- I. Extract sample of these records 110,220,360,440,656,778,202,889. Take all features in consideration. Check out the average of alcohol use in these records.

Task # 3

- A. Perform Graphical Visualization
- B. Display a correlation graph
- C. Display the distribution of **Age** and **Frequent Cold**
- D. Find out outliers in Obesity feature by using Graphical Visualization