

### Practical Number: 1A

**Aim:** Design a simple machine learning model to train the training instances and test the same using Python.

**Code:**

```
# python library to generate random numbers
from random import randint

# the limit within which random numbers are generated
TRAIN_SET_LIMIT = 1000

# to create exactly 100 data items
TRAIN_SET_COUNT = 100

# list that contains input and corresponding output
TRAIN_INPUT = list()
TRAIN_OUTPUT = list()

# loop to create 100 data items with three columns each
for i in range(TRAIN_SET_COUNT):
    a = randint(0, TRAIN_SET_LIMIT)
    b = randint(0, TRAIN_SET_LIMIT)
    c = randint(0, TRAIN_SET_LIMIT)

# creating the output for each data item
op = a + (2 * b) + (3 * c)
TRAIN_INPUT.append([a, b, c])

# adding each output to output list
TRAIN_OUTPUT.append(op)

# Sk-Learn contains the linear regression model
# To install sklearn package in python type in command prompt: pip3 install sklearn
from sklearn.linear_model import LinearRegression

# Initialize the linear regression model
# n_jobs int, default=None : The number of jobs to use for the computation.
# This will only provide speedup in case of sufficiently large problems.
# -1 means using all processors
predictor = LinearRegression(n_jobs=-1)

# Fill the Model with the Data
predictor.fit(X = TRAIN_INPUT, y = TRAIN_OUTPUT)

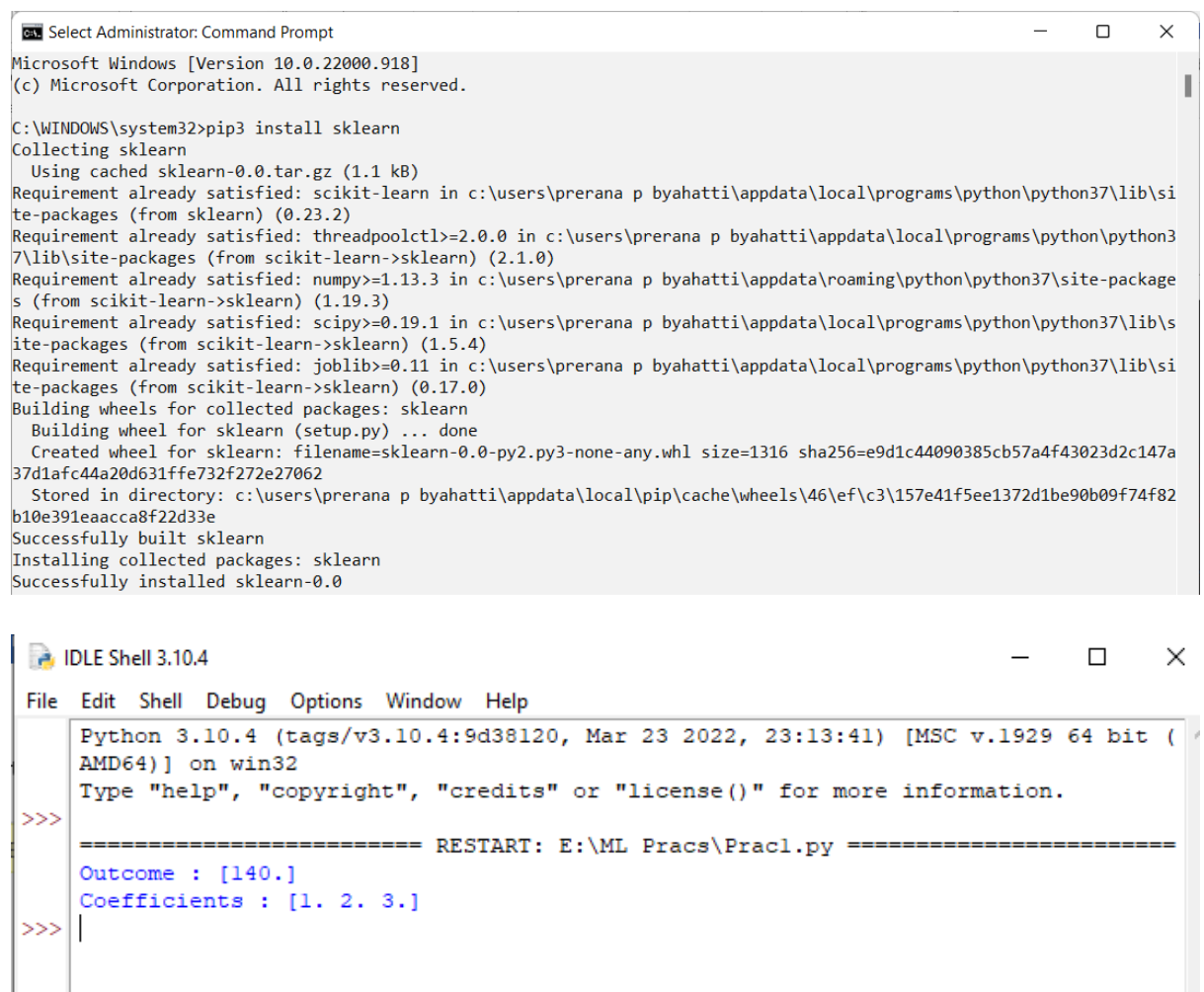
# Random Test data
X_TEST = [[ 10, 20, 30 ]]
```

```
# Predict the result of X_TEST which holds testing data
# predict() function enables us to predict the labels of the data values on the basis of
the trained model.
outcome = predictor.predict(X = X_TEST)

# Predict the coefficients
coefficients = predictor.coef_

# Print the result obtained for the test data
print('Outcome : {}\nCoefficients : {}'.format(outcome, coefficients))
```

### Output:



The first screenshot shows a Windows Command Prompt window titled "Select Administrator: Command Prompt". It displays the command `pip3 install sklearn` and its output, which includes the installation of sklearn-0.0 and its dependencies: scikit-learn, numpy, scipy, and joblib. The second screenshot shows an IDLE Shell window titled "IDLE Shell 3.10.4". It displays the output of a Python script, showing the outcome as `[140.]` and the coefficients as `[1. 2. 3.]`.

```
Microsoft Windows [Version 10.0.22000.918]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>pip3 install sklearn
Collecting sklearn
  Using cached sklearn-0.0.tar.gz (1.1 kB)
Requirement already satisfied: scikit-learn in c:\users\prerana p byahatti\appdata\local\programs\python\python37\lib\site-packages (from sklearn) (0.23.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\prerana p byahatti\appdata\local\programs\python\python37\lib\site-packages (from scikit-learn->sklearn) (2.1.0)
Requirement already satisfied: numpy>=1.13.3 in c:\users\prerana p byahatti\appdata\roaming\python\python37\site-packages (from scikit-learn->sklearn) (1.19.3)
Requirement already satisfied: scipy>=0.19.1 in c:\users\prerana p byahatti\appdata\local\programs\python\python37\lib\site-packages (from scikit-learn->sklearn) (1.5.4)
Requirement already satisfied: joblib>=0.11 in c:\users\prerana p byahatti\appdata\local\programs\python\python37\lib\site-packages (from scikit-learn->sklearn) (0.17.0)
Building wheels for collected packages: sklearn
  Building wheel for sklearn (setup.py) ... done
    Created wheel for sklearn: filename=sklearn-0.0-py2.py3-none-any.whl size=1316 sha256=e9d1c44090385cb57a4f43023d2c147a37d1afc44a20d631ffe732f272e27062
    Stored in directory: c:\users\prerana p byahatti\appdata\local\pip\cache\wheels\46\ef\c3\157e41f5ee1372d1be90b09f74f82b10e391eaacca8f22d33e
Successfully built sklearn
Installing collected packages: sklearn
Successfully installed sklearn-0.0

Python 3.10.4 (tags/v3.10.4:9d38120, Mar 23 2022, 23:13:41) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: E:\ML Pracs\Prac1.py =====
Outcome : [140.]
Coefficients : [1. 2. 3.]
>>>
```

### Practical Number: 1B

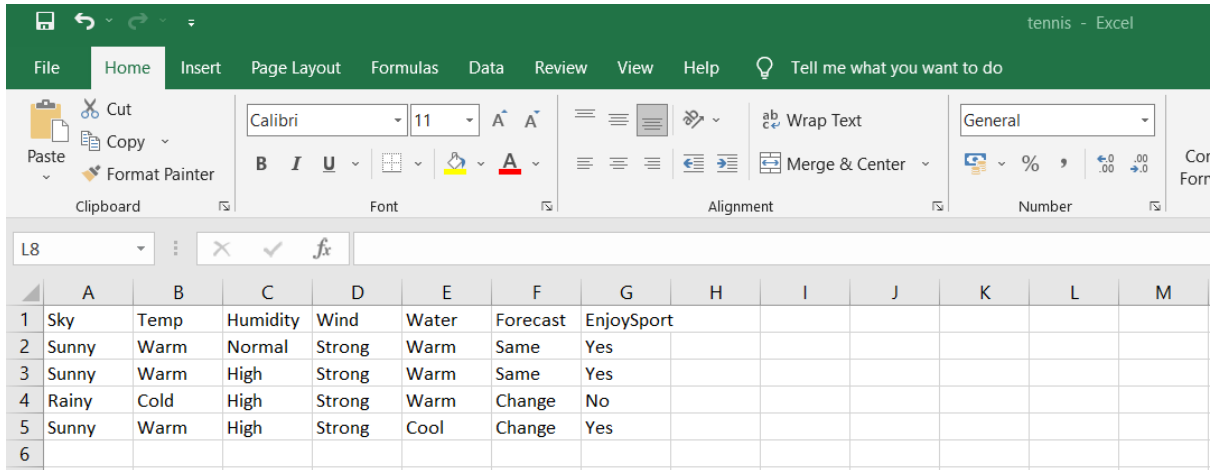
**Aim:** Implement and demonstrate the FIND-S algorithm for finding the most specific hypothesis based on a given set of training data samples. Read the training data from a .CSV file.

**Code:**

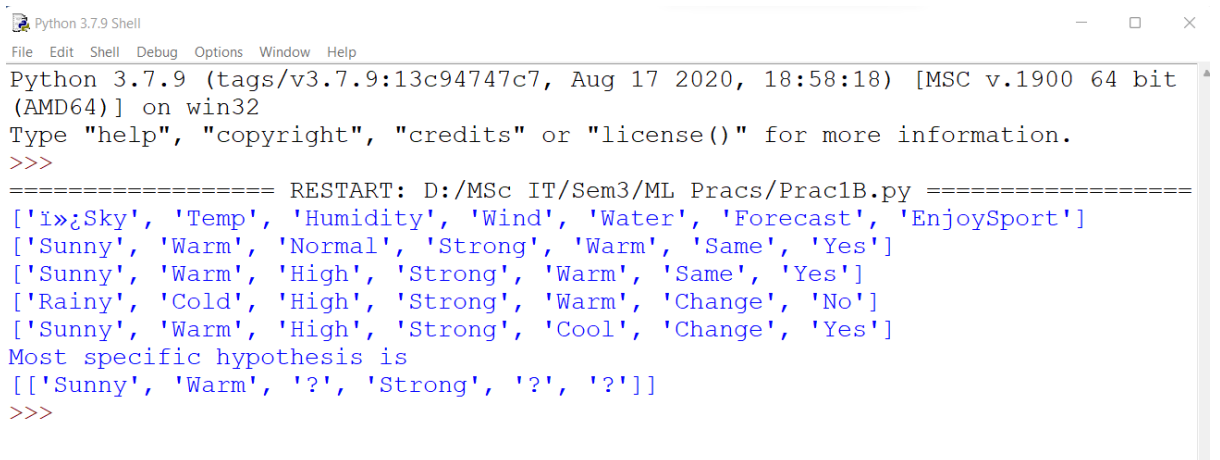
```
import csv

with open('tennis.csv', 'r') as f:
    reader = csv.reader(f)
    your_list = list(reader)
h = [['0', '0', '0', '0', '0', '0']]
for i in your_list:
    print(i)
    if i[-1]=="Yes":
        j=0
        for x in i:
            if x!="Yes":
                if x!=h[0][j] and h[0][j]=='0':
                    h[0][j]=x
                elif x!=h[0][j] and h[0][j]!='0':
                    h[0][j]='?'
            else:
                pass
        j=j + 1
print("Most specific hypothesis is")
print(h)
```

**tennis.csv file**



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Sky	Temp	Humidity	Wind	Water	Forecast	EnjoySport						
2	Sunny	Warm	Normal	Strong	Warm	Same	Yes						
3	Sunny	Warm	High	Strong	Warm	Same	Yes						
4	Rainy	Cold	High	Strong	Warm	Change	No						
5	Sunny	Warm	High	Strong	Cool	Change	Yes						
6													

**Output:**


```
Python 3.7.9 Shell
File Edit Shell Debug Options Window Help
Python 3.7.9 (tags/v3.7.9:13c94747c7, Aug 17 2020, 18:58:18) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: D:/MSc IT/Sem3/ML Pracs/Prac1B.py =====
['i>Sky', 'Temp', 'Humidity', 'Wind', 'Water', 'Forecast', 'EnjoySport']
['Sunny', 'Warm', 'Normal', 'Strong', 'Warm', 'Same', 'Yes']
['Sunny', 'Warm', 'High', 'Strong', 'Warm', 'Same', 'Yes']
['Rainy', 'Cold', 'High', 'Strong', 'Warm', 'Change', 'No']
['Sunny', 'Warm', 'High', 'Strong', 'Cool', 'Change', 'Yes']
Most specific hypothesis is
[['Sunny', 'Warm', '?', 'Strong', '?', '?']]
>>>
```

1. Describe the forward selection algorithm for implementing subset selection procedure for dimensionally refunction
2. Describe the backward selection algorithm for implementing subset selection procedure for dimensionally refunction