Machine Intelligence Assignment 1

Aryansh Bhargavan

PES1UG20CS084

CODE:

```
#This weeks code focuses on understanding basic functions of pandas and numpy
#This will help you complete other lab experiments

# ARYANSH BHARGAVAN

# Do not change the function definations or the parameters
import numpy as np
import pandas as pd

#input: tuple (x,y)    x,y:int
def create_numpy_ones_array(shape):
    #return a numpy array with one at all index
    array=None
    array = np.ones(shape)
    return array

#input: tuple (x,y)    x,y:int
def create_numpy_zeros_array(shape):
    #return a numpy array with zeros at all index
    array=None
    array = np.zeros(shape)
    return array

#input: int
def create_identity_numpy_array(order):
    #return a identity numpy array of the defined order
    array=None
    array = np.identity(order)
    return array

#input: numpy array
def matrix_cofactor(array):
    #return cofactor matrix of the given array
    determinants = np.linalg.det(array)
    cofac = np.linalg.inv(array).T * determinants
    return cofac

#Input: (numpy array, int ,numpy array, int , int , int , int , tuple,tuple)
#tuple (x,y)    x,y:int
def f1(X1,coef1,X2,coef2,seed1,seed2,seed3,shape1,shape2):
    #note: shape is of the forst (x1,x2)
    #return W1 x (X1 ** coef1) + W2 x (X2 ** coef2) +b
    # where W1 is random matrix of shape shape1 with seed1
```

```
    # where W2 is random matrix of shape shape2 with seed2
    # where B is a random matrix of comaptible shape with seed3
    # if dimension mismatch occur return -1
    try:
        np.random.seed(seed1)
        W1=np.random.rand(*shape1)

        np.random.seed(seed2)
        W2=np.random.rand(*shape2)

        np.random.seed(seed3)
        B = np.random.rand(shape1[0], X1.shape[1])

        Y1=X1**coef1
        Y2=X2**coef2

        ans = np.dot(W1, Y1)+np.dot(W2, Y2)+B

    except:
        ans=-1
    return ans



def fill_with_mode(filename, column):
    """
    Fill the missing values(NaN) in a column with the mode of that column
    Args:
        filename: Name of the CSV file.
        column: Name of the column to fill
    Returns:
        df: Pandas DataFrame object.
        (Representing entire data and where 'column' does not contain NaN
values)
        (Filled with above mentioned rules)
    """
    df = pd.read_csv(filename)
    df[column].fillna(df[column].mode()[0], inplace=True)
    return df

def fill_with_group_average(df, group, column):
    """
    Fill the missing values(NaN) in column with the mean value of the
    group the row belongs to.
    The rows are grouped based on the values of another column

    Args:
        df: A pandas DataFrame object representing the data.
        group: The column to group the rows with
        column: Name of the column to fill
    Returns:
        df: Pandas DataFrame object.
        (Representing entire data and where 'column' does not contain NaN
values)
        (Filled with above mentioned rules)
```

```
    """
    df[column].fillna(df.groupby(group)[column].transform('mean'), inplace=True)
    return df


def get_rows_greater_than_avg(df, column):
    """
    Return all the rows(with all columns) where the value in a certain 'column'
    is greater than the average value of that column.

    row where row.column > mean(data.column)

    Args:
        df: A pandas DataFrame object representing the data.
        column: Name of the column to fill
    Returns:
        df: Pandas DataFrame object.
    """
    return df[df[column] > df[column].mean()]
```

OUTPUT:

```
aryansh@poopoo:/mnt/d/SEM5/MI/assignment1_kinda$ python3 SampleTest.py --SRN PES1UG20CS084
Test Case 1 for create_numpy_ones_array PASSED
Test Case 2 for create_numpy_zeros_array PASSED
Test Case 3 for create_identity_numpy_array PASSED
Test Case 4 for matrix_cofactor PASSED
Test Case 5 for f1 PASSED
Test Case 6 for f1 PASSED
Test Case 7 for the function fill_with_mode PASSED
Test Case 8 for the function fill_with_group_average PASSED
Test Case 9 for the function get_rows_greater_than_avg PASSED
aryansh@poopoo:/mnt/d/SEM5/MI/assignment1_kinda$
```