Machine Intelligence Assignment

PES1UG20CS452 Week Zero Swapnil S Nair

An Intro to Numpy and Pandas

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


# Do not change the function defination 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
   w = np.ones(np.shape(shape))
   return w

#input: tuple (x,y)    x,y:int
def create_numpy_zeros_array(shape):
   #return a numpy array with zeros at all index
   w = np.zeros(np.shape(shape))
   return w

#input: int
def create_identity_numpy_array(order):
   #return a identity numpy array of the defined order
   w = np.identity(order)
   return w

#input: numpy array
def matrix_cofactor(array):
   #return cofactor matrix of the given array
   w = np.linalg.det(array) * np.linalg.inv(array)
   return np.transpose(w)
```

```python
#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
   np.random.seed(seed1)
   W1 = np.random.rand(*shape1)
   #print(W1,'\n',X1,'\n')
   np.random.seed(seed2)
   W2 = np.random.rand(*shape2)
   #print(W2,'\n',X2,'\n')

   X1 = np.matmul(W1,X1 ** coef1)
   X2 = np.matmul(W2,X2 ** coef2)
   if( np.shape(X1) != np.shape(X2) ):
       return -1
   X = X1 + X2

   np.random.seed(seed3)
   b = np.random.rand(*np.shape(X))
   ans = X + b
   return ans
```

```python
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)
    #print(df.head(14))
    df[column] = df[column].fillna(df[column].mode()[0])
    #print(df.head(14))
    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] = df[column].fillna(df.groupby(group)[column].transform('mean'))
    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.
    """
    df2 = df[ df[column] >= df[column].mean() ]
    return df2
```

Machine Intelligence Assignment

PES1UG20CS452 Week Zero Swapnil S Nair

OUTPUT :