ASS 1 DSBDA

January 22, 2024

[2]:
```
# 1. Import all the required Python Libraries. >>>>> (numpy, pandas,␣
↪matplotlib, seaborn, ...)
```

[1]:
```
import pandas as pd
import numpy as np
```

[111]:
```
# 2. Locate an open source data from the web (e.g. https://www.kaggle.com)

# https://www.kaggle.com/datasets/rajgupta2019/medical-insurance-dataset -␣
↪Medical Insuarance Dataset
```

[112]:
```
# 3.Loading the dataset into pandas frame
```

[2]:
```
df = pd.read_csv(r'C:\Users\Aditi\Downloads\Test_Data.csv')
```

[114]:
```
# 4.Checking for null or missing values in the dataset
```

[3]:
```
df.head()
```

[3]:

	age	gender	bmi	smoker	region	children
0	40.000000	male	29.900000	no	southwest	2.0
1	47.000000	male	NaN	no	southwest	1.0
2	54.000000	female	28.880000	no	northeast	2.0
3	NaN	male	30.568094	no	northeast	NaN
4	59.130049	male	33.132854	yes	northeast	4.0

[4]:
```
df.isna()                                #checking for null or missing values ␣
↪(True means there is a null value)
```

[4]:

	age	gender	bmi	smoker	region	children
0	False	False	False	False	False	False
1	False	False	True	False	False	False
2	False	False	False	False	False	False
3	True	False	False	False	False	True
4	False	False	False	False	False	False
..	…	…	…	…	…	
487	False	False	False	False	False	False
488	False	False	False	False	False	False

```
489  False   False  False   False   False     False
490  False   False  False   False   False     False
491  False   False  False   False   False     False

[492 rows x 6 columns]
```

[5]:
```
df.isnull()                                    # Another method for checking␣
 ↪null values
```

[5]:
```
       age  gender    bmi  smoker  region  children
0    False   False  False   False   False     False
1    False   False   True   False   False     False
2    False   False  False   False   False     False
3     True   False  False   False   False      True
4    False   False  False   False   False     False
..     …     …    …     …       …       …
487  False   False  False   False   False     False
488  False   False  False   False   False     False
489  False   False  False   False   False     False
490  False   False  False   False   False     False
491  False   False  False   False   False     False

[492 rows x 6 columns]
```

[6]:
```
df.isna().sum()                    # It will give sum of the missing values
                                   # Here we can see that there are 3 missing␣
 ↪values i.e. one in age column and in bmi and children columns
```

[6]:
```
age         1
gender      0
bmi         1
smoker      0
region      0
children    1
dtype: int64
```

[7]:
```
# Fill missing values with the mode of the 'children' column
df['children'].fillna(df['children'].mode()[0], inplace=True)
```

[8]:
```
# Fill missing values with the mean of the 'age' column
df['age'].fillna(df['age'].mean(), inplace=True)
```

[9]:
```
# Fill missing values with the median of the 'bmi' column
df['bmi'].fillna(df['bmi'].median(), inplace=True)
```

[10]:
```
# Getting last 5 records
```

[11]:
```
df.head()                              # Outcome after filling the missing values
```

[11]:

	age	gender	bmi	smoker	region	children
0	40.000000	male	29.900000	no	southwest	2.0
1	47.000000	male	29.959061	no	southwest	1.0
2	54.000000	female	28.880000	no	northeast	2.0
3	38.844276	male	30.568094	no	northeast	1.0
4	59.130049	male	33.132854	yes	northeast	4.0

[12]:
```
# Getting first 5 records
```

[13]:
```
df.tail()
```

[13]:

	age	gender	bmi	smoker	region	children
487	51.000000	male	27.740000	no	northeast	1.0
488	33.000000	male	42.400000	no	southwest	5.0
489	47.769999	male	29.064615	no	northeast	4.0
490	41.530738	female	24.260852	no	southeast	5.0
491	36.000000	male	33.400000	yes	southwest	2.0

[14]:
```
# Getting information about dataset
```

[15]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 492 entries, 0 to 491
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       492 non-null    float64
 1   gender    492 non-null    object
 2   bmi       492 non-null    float64
 3   smoker    492 non-null    object
 4   region    492 non-null    object
 5   children  492 non-null    float64
dtypes: float64(3), object(3)
memory usage: 23.2+ KB
```

[16]:
```
# Getting Dimensions of dataset
```

[17]:
```
df.shape                          # It gives dimensions as ( rows,columns)
```

[17]:
```
(492, 6)
```

[18]:
```
# describing the dataset
```

[19]:
```
df.describe
```

[19]:
```
<bound method NDFrame.describe of            age  gender        bmi smoker
region  children
```

```
0     40.000000    male  29.900000     no  southwest       2.0
1     47.000000    male  29.959061     no  southwest       1.0
2     54.000000  female  28.880000     no  northeast       2.0
3     38.844276    male  30.568094     no  northeast       1.0
4     59.130049    male  33.132854    yes  northeast       4.0
..          …     …         …    …       …       …
487   51.000000    male  27.740000     no  northeast       1.0
488   33.000000    male  42.400000     no  southwest       5.0
489   47.769999    male  29.064615     no  northeast       4.0
490   41.530738  female  24.260852     no  southeast       5.0
491   36.000000    male  33.400000    yes  southwest       2.0

[492 rows x 6 columns]>
```

[20]:
```
# For Getting Size
```

[21]:
```
df.size                 # It will give size of a dataset as 492(rows) x␣
 ↪5(columns) --> 2952
```

[21]:
```
2952
```

[22]:
```
# 5. For getting Datatypes of the variables
```

[23]:
```
df.dtypes
```

[23]:
```
age         float64
gender       object
bmi         float64
smoker       object
region       object
children    float64
dtype: object
```

[24]:
```
# need for conversion --->
''' We need to perform datatype conversions to ensure that our data is in
the appropriate format for analysis and modeling '''
```

[24]:
```
' We need to perform datatype conversions to ensure that our data is in \nthe
appropriate format for analysis and modeling '
```

[25]:
```
# Generally we have Age and no of children are in Integer Datatype
#converting float to int

df=df.astype({"age":int})
df=df.astype({"children":int})

# df['smoker'] = pd.to_numeric(df['smoker'], errors='coerce').fillna(0).
 ↪astype(int) ----> object to int
```

[26]:
```
df.dtypes
```

[26]:
```
age           int32
gender       object
bmi         float64
smoker       object
region       object
children      int32
dtype: object
```

[27]:
```
df.head()
```

[27]:

	age	gender	bmi	smoker	region	children
0	40	male	29.900000	no	southwest	2
1	47	male	29.959061	no	southwest	1
2	54	female	28.880000	no	northeast	2
3	38	male	30.568094	no	northeast	1
4	59	male	33.132854	yes	northeast	4

[28]:
```
# 6. Conversion of categorical data into quantitative(numerical) data
```

[29]:
```
# getting columns under categorical data
df_cat=df.select_dtypes(object)
```

[32]:
```
# Prints only columns under categorical data
df_cat
```

[32]:

	gender	smoker	region
0	male	no	southwest
1	male	no	southwest
2	female	no	northeast
3	male	no	northeast
4	male	yes	northeast
..	…	…	…
487	male	no	northeast
488	male	no	southwest
489	male	no	northeast
490	female	no	southeast
491	male	yes	southwest

[492 rows x 3 columns]

[34]:
```
# getting categorical columns in to the list
categorical_columns = ['gender', 'smoker', 'region']
```

[35]:
```
# Create a new DataFrame with non-categorical columns
df_non_categorical = df.drop(columns=categorical_columns)
```

[36]:
```
# Use pandas get_dummies to perform one-hot encoding on categorical data
df_encoded = pd.get_dummies(df[categorical_columns])
```

[37]:
```
print(df_encoded)
```

```
     gender_female  gender_male  smoker_no  smoker_yes  region_northeast  \
0            False         True       True       False             False
1            False         True       True       False             False
2             True        False       True       False              True
3            False         True       True       False              True
4            False         True      False        True              True
..             …            …          …           …                 …
487          False         True       True       False              True
488          False         True       True       False             False
489          False         True       True       False              True
490           True        False       True       False             False
491          False         True      False        True             False

     region_northwest  region_southeast  region_southwest
0               False             False              True
1               False             False              True
2               False             False             False
3               False             False             False
4               False             False             False
..                …                 …                 …
487             False             False             False
488             False             False              True
489             False             False             False
490             False              True             False
491             False             False              True

[492 rows x 8 columns]
```

[38]:
```
# Convert boolean values to integers (0s and 1s)
df_encoded = df_encoded.astype(int)
```

[39]:
```
# Concatenate the one-hot encoded categorical columns with the non-categorical␣
↪columns
df_merg = pd.concat([df_non_categorical, df_encoded], axis=1)
```

[40]:
```
# Display the resulting DataFrame
print(df_merg)
```

```
     age        bmi  children  gender_female  gender_male  smoker_no  \
0     40  29.900000         2              0            1          1
1     47  29.959061         1              0            1          1
2     54  28.880000         2              1            0          1
3     38  30.568094         1              0            1          1
```

```
4     59  33.132854         4              0            1           0
..   ...        ...       ...            ...          ...         ...
487   51  27.740000         1              0            1           1
488   33  42.400000         5              0            1           1
489   47  29.064615         4              0            1           1
490   41  24.260852         5              1            0           1
491   36  33.400000         2              0            1           0

     smoker_yes  region_northeast  region_northwest  region_southeast  \
0             0                 0                 0                 0
1             0                 0                 0                 0
2             0                 1                 0                 0
3             0                 1                 0                 0
4             1                 1                 0                 0
..          ...               ...               ...               ...
487           0                 1                 0                 0
488           0                 0                 0                 0
489           0                 1                 0                 0
490           0                 0                 0                 1
491           1                 0                 0                 0

     region_southwest
0                   1
1                   1
2                   0
3                   0
4                   0
..                ...
487                 0
488                 1
489                 0
490                 0
491                 1

[492 rows x 11 columns]
```

[ ]:
```
# Alternative method for conversion if there are many categorical data that are not possible to write manually in the list
'''cols = df_cat.columns

def cat_2_num(df, cols):
    for col in cols:
        dummies = pd.get_dummies(df[col], prefix=col, drop_first=True)
        df = pd.concat([df, dummies], axis=1)
        df = df.drop(col, axis=1)
    return df
```

```
df_cat = cat_2_num(df_cat, cols)'''
```