Lab Exercise 4

1. Load housing dataset "**housing.csv**" (find the file on this server) into a variable **df**.
2. Display the brief information about this dataset.
3. Display number of rows and features available in this dataset.
4. Find the target variable.
5. Show first few rows of the dataset.
6. Display the summary statistics about all the features of the dataset.
7. Show the histogram plot of each attribute. (use df.hist(bins=50, figsize=(20,25)) , plt.show())
8. Show if there are any missing/Null values in the dataset
9. Show different types of values in categorical attributes along with their frequencies.
10. Fill the missing values with most frequently used value for categorical attribute and for numerical attribute fill median value.
11. Display sum of missing values after filling the values.
12. Transform "median_income" attribute into a new attribute "income_cat" which has 5 levels (1,2,3,4,5) ranging from 0-1.5, 1.5-3.0, 3.0-4.5, 4.5-6.0, 6.0-np.inf respectively.
    { Use pd.cut(housing["attribute name"], bins=[0., 1.5, 3.0, 4.5, 6., np.inf], labels=[1, 2, 3, 4, 5])}
13. Find the distribution based on "income_cat" in the entire dataset.
14. Plot histogram of "income_cat" attributes. (use df['attribute name'].hist())
15. Split the dataset 80% of rows for training, and 20% of rows for testing purpose. Just for the sake of learning take first 80% rows as training and, rest 20% rows as testing respectively. Store these train and test datasets in temp_train and temp_test variables.
16. Check the distribution based on "income_cat" in train and test set that you obtained in above step.
17. Reshufle the dataset to have stratified distribution of 'income_cat' and then split it into train and test. Use following function

    ```
    from sklearn.model_selection import StratifiedShuffleSplit
    split = StratifiedShuffleSplit(n_splits=1, test_size=0.2, random_state=42)
    for train_index, test_index in split.split(housing, housing["attribute name"]):
        train = housing.loc[train_index]
        test = housing.loc[test_index]
    ```

18. Check again the distribution based on "income_cat" in train and test set
19. Find correlation of target attribute with rest of the attributes. Use correlation=df.corr()
    correlation["attribute name"].sort_values()

20. Convert categorical attribute to numeric using ordinal encoder. Use from sklearn.preprocessing import OrdinalEncoder

    ```
    oe=OrdinalEncoder ()
    df_cat_oe =oe.fit_transform(df[["attribute name"]])
    ```

21. Add the new attribute that you have transformed into numeric into dataset df.
22. Drop the attribute which has categorical values from the dataset.
23. Split the dataset. use sklearn.model_selection import train_test_split
    train_set, test_set = train_test_split(housing, test_size=0.2, random_state=42)
24. Separate the target attribute and rest of the attributes from train_set and test_set and store them as train_target, and test_target in two separate variables.
25. Take a linear regression mode and train it. Use
    from sklearn.linear_model import LinearRegression
    reg = LinearRegression()
26. reg.fit(training_dataset_name, training_dataset_target)
27. Predict few values from the dataset. Use predict method and pass some rows from dataset.
28. Compute performance of the model. Use following
    from sklearn.metrics import mean_squared_error
    prediction=reg.predict(df2)
    mse=mean_squared_error(target,prediction)
    performance=np.sqrt(mse)

29. Repeat from 23 to 28 by changing the training set size to 70%, 60%, 50%,40%,30%, and 20% in each attempt. Compare the difference in performance.
30. Add new attributes to the dataset like “rooms_per_household” , “bedrooms_per_room”, and “population_per_household”. Now with new added feature compute the performance of the model.