Week 8 Quiz

Clarissa Tai - rt2822

Due Tues. Nov 1st, 11:59pm ET

In this quiz we'll practice some data transformations.

Instructions

Replace the Name and UNI in cell above and the notebook filename

Replace all '__' below using the instructions provided.

When completed,

1. make sure you've replaced Name and UNI in the first cell and filename
2. Kernel -> Restart & Run All to run all cells in order
3. Print Preview -> Print (Landscape Layout) -> Save to pdf
4. post pdf to GradeScope

In [1]:
```
# import numpy as np and pandas as pd
import numpy as np
import pandas as pd
```

In [2]:
```
# Read in data from data/week8_housing_data.csv and store as dataframe df.
#   This data includes a datetime column DocumentDate.
#   Use parse_dates to parse this column into datetimes
#   Print df.info() to see the number of rows, column names, column datatypes and amount of missing data.
df = pd.read_csv('../data/week8_housing_data.csv',parse_dates=['DocumentDate'])
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype         
---  ------         --------------  -----         
 0   DocumentDate   500 non-null    datetime64[ns]
 1   PropertyType   478 non-null    object        
 2   SqFtLot        489 non-null    float64       
dtypes: datetime64[ns](1), float64(1), object(1)
memory usage: 11.8+ KB
```

In [3]:
```
# If we run df.duplicated() we get a vector of booleans that indicate duplicated rows.
# Use df.duplicated() with default values and .sum() to assert that there are 9 duplicated rows
assert df.duplicated().sum() == 3
```

In [4]:
```
# Use drop_duplicates() to drop the duplicated rows.
#   Compare the entire row (subset=None) and keep the first duplicate (keep='first') (the defaults)
# Store back into df
df = df.drop_duplicates(subset=None)

# Confirm that the correct number of rows have been dropped
assert df.shape[0] == 497
```

In [5]:
```
# Before continuing, note the is a missing SqFtLot in the first row
df.head(3)
```

Out[5]:

	DocumentDate	PropertyType	SqFtLot
0	2006-11-21	Single Family	NaN
1	2007-04-16	Townhouse	937.0
2	2006-01-18	NaN	13167.0

In [6]:
```
# From the .info() above, we see there are missing values in SqFtLot.
#   Before we fill this column, create a new dummy column 'SqFtLot_missing' in df.
#   This column should contain integers, 1 for missing, 0 for not missing.
# Use .isna() and .astype(int) to create the 'SqFtLot_missing' column.
df['SqFtLot_missing'] = df.SqFtLot.isna().astype(int)

# Assert that the number of 1's in the SqFtLot_missing column equals the number of missing values in SqFtLot
assert df['SqFtLot_missing'].sum() == df.SqFtLot.isna().sum()
```

```
# Assert that the dtype of SqFtLot_missing is int
assert df.SqFtLot_missing.dtype == int
```

In [7]:
```
# Now fill the missing values in df.SqFtLot with the mean of the SqFtLot column.
#   Use .fillna() and .mean()
# Store back into the existing SqFtLot column.
df['SqFtLot'] = df['SqFtLot'].fillna(df.SqFtLot.mean())

# Assert that the SqFtLot column no longer contains any missing values (number of missing values == 0)
assert df.SqFtLot.isna().sum() == 0
```

In [8]:
```
# The missing SqFtLot should now be filled
df.head(3)
```

Out[8]:

	DocumentDate	PropertyType	SqFtLot	SqFtLot_missing
0	2006-11-21	Single Family	13801.04321	1
1	2007-04-16	Townhouse	937.00000	0
2	2006-01-18	NaN	13167.00000	0

In [9]:
```
# There are also missing values in PropertyType.
#   Since 'PropertyType' is categorical, let's treat MISSING as another category.
#   Fill the empty values in PropertyType with the string 'MISSING'.
# Store back into the existing PropertyType column.
df['PropertyType'] = df['PropertyType'].fillna('MISSING')

# Call .value_counts() on the PropertyType column
#   to see how many of each category exist in the dataframe.
# We should see 22 MISSING values
df.PropertyType.value_counts()
```

Out[9]:
```
Single Family    455
MISSING           22
Townhouse         12
Multiplex          8
Name: PropertyType, dtype: int64
```

In [10]:
```
# Confirm we have no missing data by asserting that the sum of df.isna() over rows and columns is equal to 0.
assert df.isna().sum().sum() == 0
```

```
# Print df.info() to visualy confirm there are no missing values as well
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 497 entries, 0 to 499
Data columns (total 4 columns):
 #   Column           Non-Null Count  Dtype         
---  ------           --------------  -----         
 0   DocumentDate     497 non-null    datetime64[ns]
 1   PropertyType     497 non-null    object        
 2   SqFtLot          497 non-null    float64       
 3   SqFtLot_missing  497 non-null    int64         
dtypes: datetime64[ns](1), float64(1), int64(1), object(1)
memory usage: 19.4+ KB
```

In [11]:
```
# Before we one hot enocode PropertyType, print the first 3 rows of df and note their PropertyType values
df.head(3)
```

Out[11]:

	DocumentDate	PropertyType	SqFtLot	SqFtLot_missing
0	2006-11-21	Single Family	13801.04321	1
1	2007-04-16	Townhouse	937.00000	0
2	2006-01-18	MISSING	13167.00000	0

In [12]:
```
# Transform the categorical feature PropertyType using pd.get_dummies().
#   Note that we can call get_dummies on the entire dataframe and only categorical features will be transformed.
# Store the result of get_dummies into df_new
df_new = pd.get_dummies(df.PropertyType)

# Print out the first 3 rows of df_new to see the result.
df_new.head(3)
```

Out[12]:

	MISSING	Multiplex	Single Family	Townhouse
0	0	0	1	0
1	0	0	0	1
2	1	0	0	0

In [ ]:
```

```