Week_08_Quiz-qm2162

November 10, 2021

1 Week 8 Quiz

1.1 Qi Meng - qm2162

1.1.1 Due Sun. Nov 14, 11:59pm ET

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

[2]:
```
# Read in data from data/week8_housing_data.csv and store as dataframe df.
# This data includes a column datetaime 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: 1000 entries, 0 to 999
Data columns (total 3 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   DocumentDate  1000 non-null   datetime64[ns]
 1   PropertyType  956 non-null    object
 2   SqFtLot       975 non-null    float64
dtypes: datetime64[ns](1), float64(1), object(1)
memory usage: 23.6+ KB
```

[3]:
```
# If we run df.duplicated() we get a vector of booleans that indicate␣
 ↪duplicated rows.
# Print df.duplicated() and .sum() to get the number of duplicated rows (there␣
 ↪should be 9).
print(df.duplicated())
print(df.duplicated().sum())
```

```
0      False
1      False
2      False
```

```
3      False
4      False
       …
995    False
996    False
997    False
998    False
999    False
Length: 1000, dtype: bool
9
```

[4]:
```
# Use drop_duplicates() to drop the duplicated rows inplace.
# Check the entire row (subset=None) and keep the first duplicate (keep='first')
df.drop_duplicates(subset=None, keep='first', inplace=True)

# Print df.shape to confirm rows were dropped (num rows should now be 991).
df.shape
```

[4]:
```
(991, 3)
```

[5]:
```
# From the .info() above, we see there are missing values in SqFtLot.
# Before we fill this column, create a new 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 sum of 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
```

[6]:
```
# Now fill the missing values in df.SqFtLot with the mean of the SqFtLot column.
# Use .fillna() and .mean()
# Be sure to either use inplace or 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
```

[7]:
```
# Standardize the SqFtLot column using the sklearn StandardScaler

# Import StandardScaler from sklearn.preprocessing
from sklearn.preprocessing import StandardScaler
```

```
# Use fit_transform() on the SqFtLot column only.
# NOTE: fit_transform requires a 2D matrix. Use df[['SqFtLot']] to pass a␣
 ↪dataframe instead of a series.
# Store the transformed values back into a new column 'SqFtLot_scaled' in df.
df['SqFtLot_scaled'] = StandardScaler().fit_transform(df[['SqFtLot']])

# Call .agg(['mean','std']).round(2) on SqFtLot and SqFtLot_scaled columns
#   to confirm the scaling has operated as expected.
df[['SqFtLot', 'SqFtLot_scaled']].agg(['mean', 'std']).round(2)
```

[7]:

	SqFtLot	SqFtLot_scaled
mean	16335.78	-0.0
std	43717.98	1.0

[8]:
```
# 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'.
# Be sure to either use inplace or 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.
df.PropertyType.value_counts()
```

[8]:
```
Single Family    906
MISSING           44
Townhouse         27
Multiplex         14
Name: PropertyType, dtype: int64
```

[9]:
```
# 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
```

[10]:
```
# 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 back into df
df = df.join(pd.get_dummies(df.PropertyType, prefix='PropertyType'))
# Print out the first 3 rows of df to see the result.
df.head(3)
```

[10]:

	DocumentDate	PropertyType	SqFtLot	SqFtLot_missing	SqFtLot_scaled	\
0	2006-11-21	Single Family	34840.0	0	0.423477	
1	2014-03-20	Single Family	8428.0	0	-0.180973	
2	2011-07-26	Single Family	6000.0	0	-0.236539	

```
   PropertyType_MISSING  PropertyType_Multiplex  PropertyType_Single Family  \
0                     0                       0                           1
1                     0                       0                           1
2                     0                       0                           1

   PropertyType_Townhouse
0                       0
1                       0
2                       0
```