Final Project - Categorization-of-Houses-into-Price-Range

Categorization of Houses into Different Price Range using ML Algorithms from American Housing Survey 2017 Dataset

The main goal of this project is to predict the range of selling price of house with a high degree of predictive accuracy using various Machine Learning methods. Given house sale data or explanatory variable such as number of bedrooms, number of bathrooms in unit, housing cost, annual commuting cost etc, we build our model. Next, the model is evaluated with respect to test data, and plot the prediction and coefficients.

Importing Libraries

In [1]:
```
import pandas as pd
import numpy as np
from numpy import argmax
import re
import copy
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler, LabelEncoder, OneHotEncoder,
MinMaxScaler
from sklearn.model_selection import train_test_split
#pd.set_option('display.max_rows', 1000)
#pd.set_option('display.max_columns', 1000)
import math
from subprocess import call
from IPython.display import Image
from IPython.display import display
import warnings; warnings.simplefilter('ignore')

# Learning Libraries
from sklearn.metrics import accuracy_score, roc_curve, auc, confusion_matrix
#from sklearn import tree
from sklearn.model_selection import GridSearchCV
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LinearRegression
```

Data Input

We are using American Housing Survey 2017 data https://www.census.gov/programs-surveys/ahs/data/2017/ahs-2017-public-use-file--puf-/ahs-2017-national-public-use-file--puf-.html (household.csv in AHS 2017 National PUF v3.0 CSV.zip). Since the dataset is very big, I am just providing the link. It could not be uploaded in github repo. There is another csv file called AHSDICT_15NOV19_21_17_31_97_S.csv that consist of the mapping information of each feature name to their actual meaning and data type information. This file is already present in github repo. In the AHS microdata, the basic unit is an individual housing unit. Each record shows most of the information associated with a specific housing unit or individual, except for data items that could be used to personally identify that housing unit or individual. Our dataset comprises of housing data features like TOTROOMS(Number of rooms in unit), PERPOVLVL(Household income as percent of poverty threshold (rounded)), COMCOST(Total annual commuting cost), JBATHROOMS(Number of bathrooms in unit), UNITSF(Square footage of unit), JGARAGE(Flag indicating unit has a garage or carport), JFIREPLACE(Flag indicating unit has a useable fireplace) etc., and target column as MARKETVAL(Current market value of unit) to evaluate model and also check which amongst all features is the most correlated feature for price predication.

In [2]:
```
# Loading the dataset
data = pd.read_csv("household.csv")
headings = pd.read_csv("AHSDICT_15NOV19_21_17_31_97_S.csv", encoding = "ISO-88
59-1")
```

In [3]:
```
data.head()
```

Out[3]:

	CONTROL	TOTROOMS	PERPOVLVL	COMTYPE	COMCOST	JACPRIMARY	JACSECNDRY	J/
0	'11000001'	8	501	'-6'	-6	'0'	'0'	
1	'11000002'	7	501	'-6'	-6	'0'	'0'	
2	'11000005'	8	501	'-6'	-6	'0'	'0'	
3	'11000006'	5	361	'-6'	-6	'0'	'0'	
4	'11000007'	8	501	'1'	5564	'0'	'0'	

5 rows × 1090 columns

Data Cleaning

In [4]:
```
# Converting dataset into a format that can be processed further
col_to_check = data.columns
data[col_to_check] = data[col_to_check].replace({'\'':''}, regex=True)
```

In [5]:
```
# The column CONTROL is not relevant to our problem, and all values of JRENT i
s NaN, so we can remove that
col_to_remo = ['CONTROL','JRENT']
data = data.drop(col_to_remo, axis = 1)
```

In [6]:
```
# Replace all Not Applicable/No Response values with Nan for further processin
g
L = ['-6', -6, '-9', -9, 'M', 'N']
data = data.replace(L, np.nan)
```

In [7]:
```
# Getting rid of non relevant values
for c in list(data.columns):
    nan = (len(data) - data[c].count())/(len(data))
    if nan >= 0.85:
        del data[c]
```

In [8]:
```
# Target column
data['MARKETVAL'].describe()
```

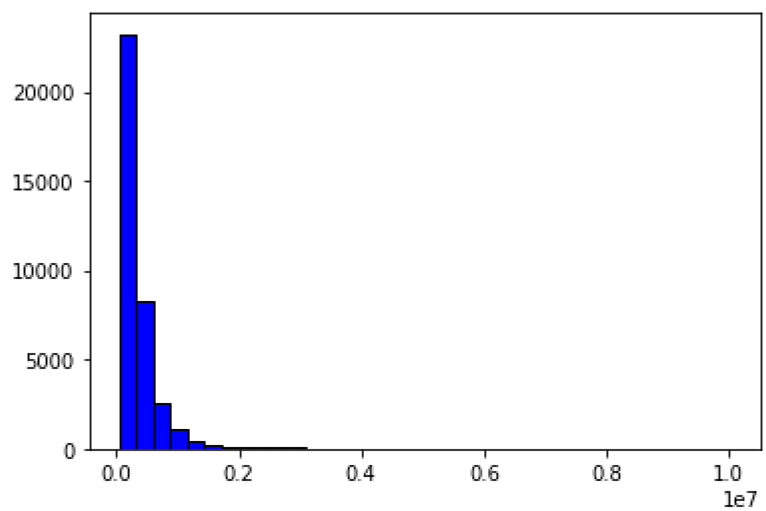
Out[8]:
```
count    3.995100e+04
mean     3.382110e+05
std      5.246493e+05
min      1.000000e+00
25%      1.210225e+05
50%      2.265120e+05
75%      3.958310e+05
max      9.999998e+06
Name: MARKETVAL, dtype: float64
```

In [9]:
```
data = data[pd.notnull(data['MARKETVAL'])]
```

In [10]:
```
indexNames = data[ data['MARKETVAL'] < 50000 ].index
# Delete these row indexes from dataFrame
data.drop(indexNames , inplace=True)
```

In [11]:
```
# Checking distribution of data
plt.hist(data['MARKETVAL'], bins = int(180/5), color = 'blue', edgecolor = 'bl
ack')
```

Out[11]:
```
(array([2.323e+04, 8.323e+03, 2.557e+03, 1.060e+03, 3.870e+02, 2.360e+02,
        1.350e+02, 9.000e+01, 6.700e+01, 4.900e+01, 4.300e+01, 2.200e+01,
        1.700e+01, 1.200e+01, 1.900e+01, 1.100e+01, 8.000e+00, 1.000e+01,
        9.000e+00, 1.000e+00, 0.000e+00, 5.000e+00, 5.000e+00, 3.000e+00,
        2.000e+00, 0.000e+00, 4.000e+00, 1.000e+00, 0.000e+00, 1.000e+00,
        2.000e+00, 2.000e+00, 3.000e+00, 6.000e+00, 7.000e+00, 3.100e+01]),
 array([  50001.        ,  326389.80555556,  602778.61111111,
         879167.41666667, 1155556.22222222, 1431945.02777778,
        1708333.83333333, 1984722.63888889, 2261111.44444444,
        2537500.25      , 2813889.05555556, 3090277.86111111,
        3366666.66666667, 3643055.47222222, 3919444.27777778,
        4195833.08333333, 4472221.88888889, 4748610.69444444,
        5024999.5       , 5301388.30555556, 5577777.11111111,
        5854165.91666667, 6130554.72222222, 6406943.52777778,
        6683332.33333333, 6959721.13888889, 7236109.94444444,
        7512498.75      , 7788887.55555556, 8065276.36111111,
        8341665.16666667, 8618053.97222222, 8894442.77777778,
        9170831.58333333, 9447220.38888889, 9723609.19444444,
        9999998.        ]),
 <a list of 36 Patch objects>)
```



In [12]:
```
# Dividing the dataset into numerical and categorical features
col_o = list(data.columns)
numeric = []
categorical = []
e = []
for c in col_o:
    j = 0
    if c[0]=='J':
        j = 1
        c = c[1:]
    h = headings.loc[headings['Variable']== c]['TYPE'].tolist()
    if h != []:
        if (h[0] == 'Character'):
            if j == 0:
                categorical.append(c)
            elif j == 1:
                categorical.append('J' + c)
        elif (h[0] == 'Numeric'):
            if j == 0:
                numeric.append(c)
            elif j == 1:
                numeric.append('J' + c)
        else:
            if j == 0:
                e.append(c)
            elif j == 1:
                e.append('J' + c)
```

Numeric Columns

In [13]:
```
# Defining data_numeric which only has numerical features
data_numeric = data.drop(categorical, axis = 1)
```

In [14]:
```
# Getting rid of all NaN entries in data_numeric
data_numeric = data_numeric.fillna(data_numeric.mean())
for i in numeric:
    if math.isnan(float(data_numeric[i].mean())):
        data_numeric = data_numeric.drop(i, axis = 1)
    else:
        data_numeric[i] = data_numeric[i].fillna(data_numeric[i].mean())

print(data_numeric.isnull().sum())
```

```
TOTROOMS     0
PERPOVLVL    0
COMCOST      0
JBEDROOMS    0
JCARPOOL     0
            ..
MORTAMT      0
HINCP        0
FINCP        0
REMODAMT     0
TOTHCAMT     0
Length: 612, dtype: int64
```

Categorical Columns

In [15]:
```
# Defining data_categorical which only has categorical features
data_categorical = data.drop(numeric, axis = 1)
```

In [16]:
```
# Getting rid of all NaN entries in data_catagorical
for i in categorical:
    # dict to store counts of each unique value occurring for each feature
    freq = {}
    for j in data_categorical[i]:
        if (j in freq):
            freq[j] += 1
        else:
            freq[j] = 1
    freq_sorted = sorted(freq, key=freq.get, reverse=True)

    # if the most frequent value is Nan
    if math.isnan(float(freq_sorted[0])):
        # if Nan is not the only value for that feature, then use the next mos
t frequent value to replace Nan
        if len(freq_sorted) > 1:
            mode_val = freq_sorted[1]
            data_categorical[i] = data_categorical[i].fillna(mode_val)
        # if Nan is the only value for that feature, then drop the column
        else:
            # drop the unnecessary columns
            print("Dropping the Column: ", i)
            data_categorical = data_categorical.drop(i, axis = 1)
    else:
        mode_val = freq_sorted[0]
        data_categorical[i] = data_categorical[i].fillna(mode_val)

print(data_categorical.isnull().sum())
```

```
Dropping the Column:  DBEVICLK
Dropping the Column:  DBEVICTHT
Dropping the Column:  RENTSUB
Dropping the Column:  DBMISSRENT
Dropping the Column:  DBEVICWHERE
Dropping the Column:  MGRONSITE
Dropping the Column:  HUDSUB
COMTYPE         0
JACPRIMARY      0
JACSECNDRY      0
JADEQUACY       0
JBATHEXCLU      0
               ..
SP2REPWGT157    0
SP2REPWGT158    0
SP2REPWGT159    0
SP2REPWGT160    0
FIRSTHOME       0
Length: 874, dtype: int64
```

In [17]:
```
# Concatenate numerical and categorical data
clean_data = pd.concat([data_numeric, data_categorical], axis=1, sort=False)
```

In [18]:
```
clean_data.describe()
```

Out[18]:

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	(
count	36358.000000	36358.000000	36358.000000	36358.000000	36358.000000	36358.000000	363
mean	6.393476	364.573773	3728.401992	0.616481	0.372325	1.970625	
std	1.628565	144.970111	1762.354679	0.529572	0.524527	1.031114	
min	1.000000	1.000000	0.000000	0.000000	0.000000	1.000000	
25%	5.000000	266.000000	3728.401992	0.000000	0.000000	1.000000	
50%	6.000000	374.000000	3728.401992	1.000000	0.000000	2.000000	
75%	7.000000	501.000000	3728.401992	1.000000	1.000000	3.000000	
max	13.000000	501.000000	43706.000000	2.000000	2.000000	7.000000	

8 rows × 1023 columns

In [19]:
```
print(clean_data.isnull().sum())
```

```
TOTROOMS        0
PERPOVLVL       0
COMCOST         0
JBEDROOMS       0
JCARPOOL        0
               ..
SP2REPWGT157    0
SP2REPWGT158    0
SP2REPWGT159    0
SP2REPWGT160    0
FIRSTHOME       0
Length: 1486, dtype: int64
```

In [20]:
```
# Remove duplicate columns after concatenation
clean_data = clean_data.iloc[:,~clean_data.columns.duplicated()]
clean_data.shape
```

Out[20]:
```
(36358, 1006)
```

Correlation Matrix

Correlation matrix to check which amongst all features is the most correlated feature for price prediction.

In [21]:
```
corr_matrix=clean_data.corr()
corr_matrix["MARKETVAL"].sort_values(ascending=False)
```

Out[21]:
```
MARKETVAL       1.000000
PROTAXAMT       0.490285
INSURAMT        0.402006
TOTHCAMT        0.297566
TOTBALAMT       0.260077
                  ...
REPWEIGHT154   -0.095058
REPWEIGHT41    -0.097701
REPWEIGHT1     -0.099587
REPWEIGHT61    -0.100648
WEIGHT         -0.113222
Name: MARKETVAL, Length: 543, dtype: float64
```

In [22]:
```
corr_matrix.style.background_gradient(cmap='coolwarm').set_precision(2)
```

Out[22]:

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
TOTROOMS	1	0.2	0.021	0.58	0.24	0.1	-0.035
PERPOVLVL	0.2	1	0.024	0.089	0.087	0.13	-0.0015
COMCOST	0.021	0.024	1	0.012	0.0072	-0.0097	0.086
DINING	0.58	0.089	0.012	1	0.048	0.061	-0.013
LAUNDY	0.24	0.087	0.0072	0.048	1	-0.045	-0.013
STORIES	0.1	0.13	-0.0097	0.061	-0.045	1	-0.016
COMDAYS	-0.035	-0.0015	0.086	-0.013	-0.013	-0.016	1
DIST	0.022	0.027	0.69	0.016	0.016	-0.016	-0.0059
RATINGHS	0.12	0.05	0.0012	0.063	0.068	0.024	-0.029
RATINGNH	0.11	0.049	0.0014	0.057	0.053	0.033	-0.018
CELLPHONE	0.21	0.16	0.011	0.078	0.063	0.025	0.013
WEIGHT	-0.036	-0.054	-0.023	-0.051	0.056	-0.028	-0.00089
SP1WEIGHT	-0.013	-0.02	-0.022	-0.02	0.024	-0.0096	0.00047
SP2WEIGHT	-0.019	-0.026	4.4e-17	-0.028	0.034	-0.016	1.6e-16
HHAGE	-0.063	-0.21	-0.034	0.0044	-0.027	-0.046	-0.084
HHMOVE	0.021	0.17	0.033	-0.0098	0.069	0.017	0.046
HHINUSYR	0.0079	0.022	0.0026	-0.0033	0.0051	0.024	0.02
NUMELDERS	-0.039	-0.16	-0.033	0.0072	-0.015	-0.041	-0.074
NUMADULTS	0.22	0.084	0.0038	0.089	0.03	-0.0076	0.022
NUMNONREL	-0.013	0.024	0.0082	-0.0093	-0.015	0.006	0.016
HHYNGKIDS	0.072	0.013	0.014	0.011	0.028	0.032	0.00017
HHOLDKIDS	0.18	-0.023	0.028	0.041	0.049	0.041	0.026
NUMVETS	0.024	0.018	-0.0068	0.018	0.035	-0.035	-0.025
NUMYNGKIDS	0.076	-0.014	0.016	0.01	0.025	0.024	0.0037
NUMOLDKIDS	0.18	-0.056	0.025	0.04	0.042	0.032	0.025
NUMSUBFAM	0.034	-0.084	-0.0026	0.003	-0.015	-0.021	0.0079
NUMSECFAM	0.0045	-0.012	0.0083	-0.0057	0.0053	-9.2e-05	0.0058
NUMPEOPLE	0.26	0.017	0.022	0.082	0.051	0.021	0.03
HHADLTKIDS	0.12	-0.026	-0.0024	0.054	-0.0019	-0.009	0.021
UFINROOMS	0.073	0.028	-0.00062	0.01	0.082	0.056	-0.0057
FINROOMS	0.73	0.2	0.0049	0.21	0.26	0.15	-0.047
YRBUILT	0.14	0.09	0.044	0.0016	0.18	-0.09	-0.0039
UNITFLOORS	0.37	0.17	0.0043	0.18	0.031	0.67	-0.019
BEDROOMS	0.82	0.14	0.026	0.28	0.18	0.011	-0.016
KITCHENS	0.12	0.00028	0.0044	0.04	0.019	0.0095	0.0011

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
DWNPAYPCT	0.021	-0.0041	-0.02	0.018	0.023	0.049	-0.045
ELECAMT	0.33	0.12	0.03	0.15	0.087	-0.038	-0.0032
GASAMT	0.23	0.098	0.0033	0.1	0.047	0.12	-0.0096
OILAMT	0.038	0.019	0.0032	0.029	-0.05	0.097	0.0032
OTHERAMT	0.0022	-0.0059	0.0051	-0.0069	0.0048	0.021	-0.00049
TRASHAMT	0.11	0.039	0.0073	0.064	0.044	-0.14	0.0037
WATERAMT	0.16	0.084	0.017	0.074	0.018	0.026	-0.015
UTILAMT	0.42	0.17	0.029	0.2	0.084	0.031	-0.0098
REMODJOBS	0.098	0.099	0.0013	0.024	0.069	0.0055	-0.022
MVG1PER	0.07	-0.039	0.015	0.015	0.011	0.00037	0.0023
CARPOOL	0.0026	-0.0039	-0.044	-0.0043	0.00021	-0.011	0.0076
TAXI	0.01	0.014	0.087	0.011	-0.0072	0.036	-0.025
FERRY	-0.0039	0.0085	0.05	-0.0076	0.0052	0.0082	-0.0028
DRIVEALL	-0.011	-0.033	0.21	-0.0082	0.0019	-0.062	0.32
PARKING	0.024	0.022	0.24	0.013	-0.00024	0.026	-0.0096
TOLL	0.017	0.014	0.37	0.013	-0.0041	0.0097	0.01
POVLVLINC	0.25	0.014	0.025	0.078	0.046	0.023	0.036
REPWEIGHT1	-0.034	-0.047	-0.023	-0.039	0.044	-0.025	0.0014
REPWEIGHT2	-0.032	-0.041	-0.016	-0.044	0.038	-0.023	0.0034
REPWEIGHT3	-0.025	-0.032	-0.022	-0.039	0.045	-0.021	-0.0066
REPWEIGHT4	-0.028	-0.043	-0.016	-0.037	0.042	-0.029	0.0033
REPWEIGHT5	-0.021	-0.037	-0.015	-0.045	0.048	-0.018	0.0031
REPWEIGHT6	-0.028	-0.035	-0.02	-0.037	0.033	-0.016	-0.0084
REPWEIGHT7	-0.032	-0.044	-0.021	-0.044	0.038	-0.03	0.0014
REPWEIGHT8	-0.024	-0.038	-0.019	-0.034	0.042	-0.021	-0.002
REPWEIGHT9	-0.025	-0.04	-0.022	-0.032	0.04	-0.018	-0.0035
REPWEIGHT10	-0.028	-0.039	-0.013	-0.037	0.039	-0.019	-0.0032
REPWEIGHT11	-0.02	-0.037	-0.019	-0.034	0.035	-0.019	0.0046
REPWEIGHT12	-0.025	-0.04	-0.016	-0.038	0.043	-0.019	-0.0019
REPWEIGHT13	-0.031	-0.043	-0.018	-0.044	0.043	-0.014	-0.0052
REPWEIGHT14	-0.023	-0.039	-0.017	-0.034	0.037	-0.01	-0.0016
REPWEIGHT15	-0.031	-0.034	-0.021	-0.044	0.047	-0.017	0.0017
REPWEIGHT16	-0.03	-0.041	-0.014	-0.036	0.043	-0.022	0.00012
REPWEIGHT17	-0.027	-0.041	-0.015	-0.039	0.042	-0.015	0.0042
REPWEIGHT18	-0.018	-0.041	-0.017	-0.034	0.042	-0.019	-0.0068
REPWEIGHT19	-0.033	-0.036	-0.023	-0.039	0.034	-0.018	0.00094

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
REPWEIGHT20	-0.026	-0.035	-0.018	-0.037	0.044	-0.021	-0.007
REPWEIGHT21	-0.028	-0.041	-0.027	-0.048	0.047	-0.021	0.001
REPWEIGHT22	-0.026	-0.039	-0.016	-0.04	0.036	-0.018	-0.0017
REPWEIGHT23	-0.021	-0.037	-0.022	-0.037	0.045	-0.023	0.0019
REPWEIGHT24	-0.026	-0.042	-0.013	-0.032	0.046	-0.027	-0.0065
REPWEIGHT25	-0.026	-0.034	-0.013	-0.04	0.037	-0.014	-0.0017
REPWEIGHT26	-0.033	-0.043	-0.019	-0.044	0.042	-0.034	-0.0038
REPWEIGHT27	-0.03	-0.035	-0.017	-0.034	0.045	-0.025	-0.00089
REPWEIGHT28	-0.023	-0.045	-0.016	-0.041	0.037	-0.026	-0.0076
REPWEIGHT29	-0.026	-0.038	-0.014	-0.045	0.05	-0.016	0.0027
REPWEIGHT30	-0.032	-0.045	-0.013	-0.041	0.044	-0.025	-0.0018
REPWEIGHT31	-0.024	-0.037	-0.017	-0.034	0.046	-0.024	-0.0032
REPWEIGHT32	-0.031	-0.045	-0.021	-0.042	0.039	-0.02	-0.0011
REPWEIGHT33	-0.028	-0.046	-0.019	-0.035	0.048	-0.027	0.00035
REPWEIGHT34	-0.028	-0.044	-0.013	-0.04	0.037	-0.021	-0.0063
REPWEIGHT35	-0.029	-0.041	-0.019	-0.042	0.038	-0.022	0.0076
REPWEIGHT36	-0.028	-0.043	-0.017	-0.036	0.043	-0.018	-0.00013
REPWEIGHT37	-0.035	-0.04	-0.017	-0.045	0.044	-0.026	0.0003
REPWEIGHT38	-0.029	-0.044	-0.0086	-0.042	0.051	-0.02	-0.0056
REPWEIGHT39	-0.03	-0.042	-0.015	-0.034	0.035	-0.025	0.003
REPWEIGHT40	-0.017	-0.036	-0.018	-0.037	0.048	-0.021	-0.0087
REPWEIGHT41	-0.032	-0.048	-0.02	-0.038	0.048	-0.028	-0.002
REPWEIGHT42	-0.027	-0.041	-0.018	-0.036	0.041	-0.019	0.0029
REPWEIGHT43	-0.021	-0.035	-0.02	-0.038	0.041	-0.018	-0.0064
REPWEIGHT44	-0.025	-0.042	-0.014	-0.04	0.036	-0.022	-0.0032
REPWEIGHT45	-0.022	-0.041	-0.018	-0.04	0.05	-0.016	0.0021
REPWEIGHT46	-0.029	-0.036	-0.025	-0.045	0.043	-0.021	0.00028
REPWEIGHT47	-0.023	-0.035	-0.015	-0.038	0.039	-0.012	-0.0049
REPWEIGHT48	-0.028	-0.045	-0.016	-0.043	0.041	-0.03	-0.0053
REPWEIGHT49	-0.015	-0.038	-0.02	-0.033	0.044	-0.011	-0.0035
REPWEIGHT50	-0.025	-0.044	-0.019	-0.031	0.038	-0.019	-0.0019
REPWEIGHT51	-0.02	-0.04	-0.013	-0.034	0.045	-0.02	0.0057
REPWEIGHT52	-0.024	-0.042	-0.017	-0.042	0.043	-0.015	-0.012
REPWEIGHT53	-0.026	-0.04	-0.016	-0.041	0.038	-0.017	0.0035
REPWEIGHT54	-0.022	-0.041	-0.014	-0.033	0.046	-0.017	-0.005
REPWEIGHT55	-0.026	-0.048	-0.0084	-0.037	0.043	-0.022	-0.0066

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
REPWEIGHT56	-0.027	-0.043	-0.013	-0.036	0.049	-0.017	0.0027
REPWEIGHT57	-0.026	-0.04	-0.017	-0.041	0.048	-0.019	-0.00062
REPWEIGHT58	-0.02	-0.042	-0.015	-0.033	0.039	-0.021	-0.0073
REPWEIGHT59	-0.028	-0.033	-0.011	-0.037	0.037	-0.018	-0.0024
REPWEIGHT60	-0.022	-0.042	-0.016	-0.039	0.043	-0.021	0.00053
REPWEIGHT61	-0.026	-0.047	-0.024	-0.038	0.048	-0.025	-0.0026
REPWEIGHT62	-0.032	-0.04	-0.019	-0.044	0.039	-0.025	-0.004
REPWEIGHT63	-0.026	-0.04	-0.016	-0.036	0.04	-0.022	-0.00044
REPWEIGHT64	-0.016	-0.042	-0.018	-0.033	0.039	-0.019	-0.0018
REPWEIGHT65	-0.032	-0.041	-0.011	-0.038	0.04	-0.024	0.0027
REPWEIGHT66	-0.016	-0.038	-0.019	-0.041	0.038	-0.02	0.001
REPWEIGHT67	-0.032	-0.042	-0.015	-0.042	0.045	-0.021	-0.0039
REPWEIGHT68	-0.029	-0.041	-0.015	-0.045	0.052	-0.02	-0.0015
REPWEIGHT69	-0.025	-0.042	-0.019	-0.04	0.039	-0.023	-0.0053
REPWEIGHT70	-0.019	-0.04	-0.012	-0.03	0.046	-0.016	0.0058
REPWEIGHT71	-0.018	-0.042	-0.021	-0.033	0.054	-0.017	-0.0032
REPWEIGHT72	-0.032	-0.039	-0.024	-0.042	0.042	-0.021	-0.00052
REPWEIGHT73	-0.029	-0.037	-0.013	-0.034	0.044	-0.027	0.0011
REPWEIGHT74	-0.032	-0.047	-0.015	-0.037	0.038	-0.023	-0.0022
REPWEIGHT75	-0.032	-0.043	-0.018	-0.049	0.046	-0.026	0.0035
REPWEIGHT76	-0.023	-0.043	-0.013	-0.034	0.04	-0.016	0.0045
REPWEIGHT77	-0.029	-0.038	-0.02	-0.032	0.041	-0.021	-0.00041
REPWEIGHT78	-0.021	-0.032	-0.015	-0.039	0.042	-0.015	-0.0055
REPWEIGHT79	-0.024	-0.034	-0.008	-0.037	0.047	-0.01	0.0041
REPWEIGHT80	-0.024	-0.041	-0.014	-0.039	0.046	-0.026	-0.0014
REPWEIGHT81	-0.034	-0.049	-0.018	-0.042	0.041	-0.02	0.0052
REPWEIGHT82	-0.036	-0.034	-0.013	-0.046	0.034	-0.025	0.0029
REPWEIGHT83	-0.026	-0.039	-0.025	-0.038	0.043	-0.021	-0.00037
REPWEIGHT84	-0.026	-0.042	-0.011	-0.035	0.044	-0.017	0.011
REPWEIGHT85	-0.026	-0.04	-0.02	-0.04	0.043	-0.027	-0.002
REPWEIGHT86	-0.028	-0.036	-0.015	-0.044	0.042	-0.019	-0.00049
REPWEIGHT87	-0.029	-0.037	-0.016	-0.037	0.041	-0.024	1.1e-05
REPWEIGHT88	-0.027	-0.037	-0.023	-0.037	0.04	-0.019	-0.0041
REPWEIGHT89	-0.032	-0.043	-0.02	-0.04	0.042	-0.027	0.0051
REPWEIGHT90	-0.029	-0.041	-0.016	-0.035	0.04	-0.018	0.0015
REPWEIGHT91	-0.027	-0.04	-0.014	-0.039	0.04	-0.024	0.0026

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
REPWEIGHT92	-0.024	-0.035	-0.019	-0.034	0.044	-0.019	-0.0019
REPWEIGHT93	-0.029	-0.043	-0.019	-0.044	0.04	-0.021	-0.0063
REPWEIGHT94	-0.021	-0.032	-0.017	-0.033	0.041	-0.013	0.0036
REPWEIGHT95	-0.025	-0.037	-0.014	-0.028	0.039	-0.021	0.0026
REPWEIGHT96	-0.025	-0.028	-0.014	-0.037	0.038	-0.016	0.0051
REPWEIGHT97	-0.025	-0.042	-0.02	-0.037	0.034	-0.024	-0.00058
REPWEIGHT98	-0.028	-0.035	-0.024	-0.037	0.036	-0.022	0.0028
REPWEIGHT99	-0.028	-0.048	-0.017	-0.039	0.035	-0.018	-0.0015
REPWEIGHT100	-0.026	-0.032	-0.016	-0.037	0.043	-0.017	0.0035
REPWEIGHT101	-0.028	-0.043	-0.019	-0.049	0.056	-0.019	0.0016
REPWEIGHT102	-0.027	-0.037	-0.018	-0.04	0.042	-0.016	-0.0019
REPWEIGHT103	-0.03	-0.042	-0.02	-0.04	0.043	-0.018	-7e-05
REPWEIGHT104	-0.025	-0.044	-0.021	-0.035	0.046	-0.024	0.0038
REPWEIGHT105	-0.032	-0.043	-0.014	-0.039	0.035	-0.016	-0.0016
REPWEIGHT106	-0.032	-0.037	-0.019	-0.037	0.053	-0.032	0.0036
REPWEIGHT107	-0.023	-0.034	-0.016	-0.032	0.035	-0.016	-0.0013
REPWEIGHT108	-0.028	-0.037	-0.019	-0.036	0.038	-0.026	0.0016
REPWEIGHT109	-0.033	-0.046	-0.022	-0.038	0.04	-0.021	-0.0013
REPWEIGHT110	-0.034	-0.043	-0.018	-0.047	0.05	-0.021	-0.0023
REPWEIGHT111	-0.03	-0.04	-0.02	-0.044	0.037	-0.024	-0.00095
REPWEIGHT112	-0.026	-0.049	-0.022	-0.039	0.042	-0.021	0.0046
REPWEIGHT113	-0.03	-0.039	-0.014	-0.041	0.041	-0.032	0.0021
REPWEIGHT114	-0.026	-0.042	-0.02	-0.039	0.05	-0.024	0.00056
REPWEIGHT115	-0.026	-0.042	-0.02	-0.036	0.039	-0.024	0.002
REPWEIGHT116	-0.026	-0.038	-0.022	-0.035	0.04	-0.014	-0.0024
REPWEIGHT117	-0.037	-0.041	-0.019	-0.048	0.045	-0.029	0.0031
REPWEIGHT118	-0.03	-0.043	-0.016	-0.04	0.042	-0.024	0.001
REPWEIGHT119	-0.034	-0.048	-0.013	-0.037	0.038	-0.026	0.0018
REPWEIGHT120	-0.019	-0.032	-0.026	-0.029	0.042	-0.018	0.00046
REPWEIGHT121	-0.031	-0.05	-0.015	-0.042	0.046	-0.024	0.0019
REPWEIGHT122	-0.028	-0.041	-0.02	-0.042	0.036	-0.019	0.0076
REPWEIGHT123	-0.022	-0.035	-0.018	-0.037	0.043	-0.0084	0.00048
REPWEIGHT124	-0.029	-0.047	-0.015	-0.038	0.033	-0.025	0.0077
REPWEIGHT125	-0.028	-0.044	-0.022	-0.041	0.048	-0.014	-0.0024
REPWEIGHT126	-0.026	-0.035	-0.018	-0.039	0.047	-0.026	0.001
REPWEIGHT127	-0.019	-0.034	-0.019	-0.038	0.045	-0.0089	-0.0061

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
REPWEIGHT128	-0.03	-0.039	-0.02	-0.044	0.044	-0.02	-0.0045
REPWEIGHT129	-0.028	-0.045	-0.015	-0.042	0.04	-0.019	0.0031
REPWEIGHT130	-0.023	-0.045	-0.022	-0.033	0.044	-0.02	-0.0074
REPWEIGHT131	-0.033	-0.042	-0.017	-0.037	0.043	-0.024	0.0043
REPWEIGHT132	-0.025	-0.039	-0.017	-0.039	0.046	-0.015	-0.0062
REPWEIGHT133	-0.02	-0.043	-0.019	-0.041	0.039	-0.018	-0.0011
REPWEIGHT134	-0.026	-0.04	-0.013	-0.032	0.043	-0.022	0.007
REPWEIGHT135	-0.025	-0.043	-0.015	-0.027	0.039	-0.023	-0.0059
REPWEIGHT136	-0.022	-0.041	-0.012	-0.035	0.043	-0.017	0.0044
REPWEIGHT137	-0.024	-0.04	-0.026	-0.041	0.04	-0.019	-0.0064
REPWEIGHT138	-0.034	-0.04	-0.011	-0.041	0.035	-0.028	-0.0017
REPWEIGHT139	-0.021	-0.042	-0.02	-0.032	0.037	-0.016	0.002
REPWEIGHT140	-0.032	-0.042	-0.012	-0.043	0.041	-0.018	-0.00098
REPWEIGHT141	-0.023	-0.049	-0.015	-0.036	0.057	-0.021	-0.00016
REPWEIGHT142	-0.035	-0.033	-0.014	-0.045	0.04	-0.022	-0.0036
REPWEIGHT143	-0.033	-0.041	-0.013	-0.037	0.03	-0.015	0.0052
REPWEIGHT144	-0.021	-0.043	-0.022	-0.036	0.047	-0.016	0.0051
REPWEIGHT145	-0.036	-0.043	-0.017	-0.041	0.04	-0.027	-0.0016
REPWEIGHT146	-0.021	-0.037	-0.011	-0.037	0.043	-0.02	-0.0023
REPWEIGHT147	-0.029	-0.032	-0.03	-0.041	0.043	-0.016	-0.0078
REPWEIGHT148	-0.036	-0.041	-0.018	-0.048	0.044	-0.026	0.0017
REPWEIGHT149	-0.026	-0.048	-0.015	-0.034	0.036	-0.023	0.0019
REPWEIGHT150	-0.023	-0.033	-0.02	-0.034	0.046	-0.012	-0.0016
REPWEIGHT151	-0.027	-0.039	-0.02	-0.039	0.047	-0.018	-0.0044
REPWEIGHT152	-0.033	-0.046	-0.018	-0.048	0.043	-0.026	0.0067
REPWEIGHT153	-0.03	-0.035	-0.021	-0.046	0.043	-0.025	-0.0024
REPWEIGHT154	-0.03	-0.046	-0.015	-0.036	0.043	-0.031	-0.0021
REPWEIGHT155	-0.029	-0.038	-0.02	-0.04	0.047	-0.023	0.0072
REPWEIGHT156	-0.024	-0.036	-0.017	-0.033	0.038	-0.017	0.00051
REPWEIGHT157	-0.029	-0.041	-0.019	-0.045	0.04	-0.024	-0.0073
REPWEIGHT158	-0.032	-0.039	-0.017	-0.045	0.043	-0.019	0.0049
REPWEIGHT159	-0.025	-0.036	-0.019	-0.031	0.042	-0.017	-0.0066
REPWEIGHT160	-0.021	-0.042	-0.018	-0.033	0.051	-0.02	0.0055
SP1REPWGT1	-0.012	-0.019	-0.024	-0.015	0.018	-0.0088	0.0021
SP1REPWGT2	-0.014	-0.018	-0.018	-0.019	0.016	-0.012	0.0038
SP1REPWGT3	-0.011	-0.014	-0.024	-0.016	0.022	-0.0058	-0.0057

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
SP1REPWGT4	-0.014	-0.014	-0.018	-0.022	0.019	-0.013	0.0044
SP1REPWGT5	-0.0088	-0.015	-0.018	-0.02	0.024	-0.002	0.0052
SP1REPWGT6	-0.0099	-0.014	-0.022	-0.018	0.016	-0.008	-0.0087
SP1REPWGT7	-0.015	-0.015	-0.024	-0.023	0.018	-0.0098	0.004
SP1REPWGT8	-0.0082	-0.015	-0.02	-0.015	0.018	-0.01	-0.0018
SP1REPWGT9	-0.011	-0.017	-0.024	-0.016	0.021	-0.0028	-0.0017
SP1REPWGT10	-0.014	-0.016	-0.015	-0.02	0.02	-0.0083	-0.0026
SP1REPWGT11	-0.0047	-0.018	-0.02	-0.015	0.015	-0.0053	0.0055
SP1REPWGT12	-0.01	-0.017	-0.019	-0.017	0.021	-0.0052	-0.0014
SP1REPWGT13	-0.013	-0.018	-0.02	-0.019	0.019	-0.0066	-0.0046
SP1REPWGT14	-0.013	-0.017	-0.021	-0.017	0.02	0.0042	-0.0019
SP1REPWGT15	-0.014	-0.016	-0.023	-0.02	0.022	-0.0097	0.0021
SP1REPWGT16	-0.018	-0.02	-0.016	-0.019	0.021	-0.0045	0.00015
SP1REPWGT17	-0.012	-0.014	-0.016	-0.019	0.019	-0.0092	0.0055
SP1REPWGT18	-0.0069	-0.016	-0.02	-0.014	0.023	-0.0058	-0.0064
SP1REPWGT19	-0.013	-0.012	-0.024	-0.016	0.016	-0.0091	0.0022
SP1REPWGT20	-0.014	-0.014	-0.022	-0.02	0.023	-0.0013	-0.0067
SP1REPWGT21	-0.011	-0.017	-0.029	-0.021	0.023	-0.0092	0.0017
SP1REPWGT22	-0.012	-0.017	-0.019	-0.015	0.016	-0.0056	-0.0013
SP1REPWGT23	-0.0068	-0.013	-0.023	-0.018	0.022	-0.0091	0.0036
SP1REPWGT24	-0.0085	-0.018	-0.015	-0.0089	0.024	-0.011	-0.0058
SP1REPWGT25	-0.01	-0.015	-0.014	-0.02	0.015	-0.0048	-0.00016
SP1REPWGT26	-0.014	-0.016	-0.021	-0.024	0.024	-0.016	-0.0041
SP1REPWGT27	-0.012	-0.013	-0.017	-0.016	0.02	-0.012	0.0013
SP1REPWGT28	-0.01	-0.019	-0.018	-0.02	0.014	-0.0065	-0.0075
SP1REPWGT29	-0.0078	-0.018	-0.016	-0.021	0.026	-0.0093	0.0034
SP1REPWGT30	-0.014	-0.018	-0.014	-0.019	0.023	-0.011	9.2e-05
SP1REPWGT31	-0.012	-0.015	-0.019	-0.013	0.021	-0.011	-0.002
SP1REPWGT32	-0.014	-0.018	-0.022	-0.021	0.017	-0.0079	-0.001
SP1REPWGT33	-0.0081	-0.021	-0.021	-0.013	0.021	-0.01	0.0027
SP1REPWGT34	-0.016	-0.02	-0.015	-0.022	0.018	-0.0089	-0.0071
SP1REPWGT35	-0.011	-0.021	-0.02	-0.018	0.018	-0.0079	0.009
SP1REPWGT36	-0.013	-0.014	-0.02	-0.016	0.019	-0.0088	0.00021
SP1REPWGT37	-0.014	-0.018	-0.018	-0.02	0.022	-0.014	0.0017
SP1REPWGT38	-0.012	-0.019	-0.0086	-0.02	0.026	-0.0072	-0.0045
SP1REPWGT39	-0.011	-0.018	-0.017	-0.015	0.017	-0.0094	0.0044

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
SP1REPWGT40	-0.0051	-0.016	-0.02	-0.021	0.022	-0.0077	-0.0094
SP1REPWGT41	-0.012	-0.017	-0.02	-0.017	0.02	-0.0096	-0.0013
SP1REPWGT42	-0.011	-0.017	-0.02	-0.016	0.021	-0.0037	0.0033
SP1REPWGT43	-0.0078	-0.011	-0.021	-0.019	0.017	-0.0067	-0.0056
SP1REPWGT44	-0.0086	-0.016	-0.015	-0.017	0.02	-0.0097	-0.0022
SP1REPWGT45	-0.0063	-0.016	-0.021	-0.019	0.024	-0.0064	0.0029
SP1REPWGT46	-0.011	-0.015	-0.027	-0.021	0.021	-0.0082	0.00085
SP1REPWGT47	-0.01	-0.011	-0.016	-0.018	0.02	-0.0039	-0.0035
SP1REPWGT48	-0.011	-0.019	-0.018	-0.02	0.017	-0.012	-0.0057
SP1REPWGT49	-0.0047	-0.017	-0.022	-0.015	0.025	-0.006	-0.0023
SP1REPWGT50	-0.0083	-0.018	-0.02	-0.01	0.017	-0.0075	-0.00078
SP1REPWGT51	-0.0074	-0.016	-0.014	-0.017	0.021	-0.0077	0.0071
SP1REPWGT52	-0.008	-0.019	-0.019	-0.017	0.022	-0.0049	-0.013
SP1REPWGT53	-0.013	-0.019	-0.018	-0.019	0.017	-0.012	0.0056
SP1REPWGT54	-0.0028	-0.018	-0.016	-0.011	0.022	-0.0046	-0.0054
SP1REPWGT55	-0.011	-0.019	-0.0087	-0.018	0.019	-0.011	-0.0068
SP1REPWGT56	-0.012	-0.021	-0.016	-0.014	0.026	-0.0076	0.0045
SP1REPWGT57	-0.011	-0.015	-0.018	-0.021	0.021	-0.007	0.00064
SP1REPWGT58	-0.0084	-0.018	-0.016	-0.018	0.02	-0.0087	-0.0066
SP1REPWGT59	-0.012	-0.013	-0.013	-0.02	0.018	-0.0075	-0.0023
SP1REPWGT60	-0.0059	-0.021	-0.017	-0.015	0.022	-0.013	0.0013
SP1REPWGT61	-0.011	-0.019	-0.025	-0.016	0.022	-0.0093	-0.0016
SP1REPWGT62	-0.015	-0.018	-0.021	-0.025	0.02	-0.013	-0.0039
SP1REPWGT63	-0.0091	-0.014	-0.017	-0.013	0.019	-0.011	0.0016
SP1REPWGT64	-0.0074	-0.021	-0.02	-0.018	0.024	-0.0056	-0.00088
SP1REPWGT65	-0.014	-0.017	-0.011	-0.018	0.019	-0.0088	0.0049
SP1REPWGT66	-0.0075	-0.015	-0.022	-0.02	0.018	-0.0079	0.0014
SP1REPWGT67	-0.014	-0.018	-0.017	-0.021	0.022	-0.0094	-0.0024
SP1REPWGT68	-0.012	-0.019	-0.017	-0.021	0.023	-0.0052	-0.00085
SP1REPWGT69	-0.011	-0.017	-0.021	-0.019	0.022	-0.01	-0.0053
SP1REPWGT70	-0.0043	-0.015	-0.014	-0.014	0.024	-0.0069	0.0074
SP1REPWGT71	-0.0064	-0.017	-0.022	-0.013	0.024	-0.0044	-0.0016
SP1REPWGT72	-0.015	-0.019	-0.026	-0.021	0.025	-0.009	0.00068
SP1REPWGT73	-0.011	-0.015	-0.014	-0.014	0.016	-0.011	0.0033
SP1REPWGT74	-0.013	-0.022	-0.017	-0.016	0.02	-0.0094	-0.0034
SP1REPWGT75	-0.013	-0.021	-0.02	-0.024	0.024	-0.013	0.0051

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
SP1REPWGT76	-0.01	-0.018	-0.014	-0.018	0.018	-0.0065	0.0066
SP1REPWGT77	-0.012	-0.015	-0.022	-0.016	0.025	-0.0088	0.0008
SP1REPWGT78	-0.0053	-0.014	-0.017	-0.017	0.022	-0.0072	-0.0044
SP1REPWGT79	-0.014	-0.015	-0.01	-0.019	0.024	-0.0022	0.005
SP1REPWGT80	-0.0083	-0.018	-0.015	-0.017	0.023	-0.0099	-0.00025
SP1REPWGT81	-0.011	-0.02	-0.018	-0.016	0.017	-0.0076	0.0064
SP1REPWGT82	-0.014	-0.016	-0.015	-0.021	0.016	-0.014	0.0041
SP1REPWGT83	-0.011	-0.017	-0.027	-0.017	0.019	-0.01	0.0012
SP1REPWGT84	-0.011	-0.015	-0.013	-0.018	0.018	-0.0042	0.013
SP1REPWGT85	-0.01	-0.017	-0.022	-0.016	0.02	-0.012	-0.00073
SP1REPWGT86	-0.0093	-0.017	-0.017	-0.02	0.023	-0.0058	0.00041
SP1REPWGT87	-0.014	-0.012	-0.018	-0.02	0.018	-0.01	0.0019
SP1REPWGT88	-0.0096	-0.015	-0.025	-0.017	0.023	-0.0075	-0.0036
SP1REPWGT89	-0.012	-0.022	-0.023	-0.018	0.019	-0.01	0.0068
SP1REPWGT90	-0.013	-0.016	-0.018	-0.017	0.02	-0.0089	0.002
SP1REPWGT91	-0.0088	-0.019	-0.017	-0.018	0.019	-0.0082	0.0041
SP1REPWGT92	-0.007	-0.014	-0.02	-0.013	0.022	-0.0055	-0.0015
SP1REPWGT93	-0.011	-0.018	-0.021	-0.022	0.019	-0.0092	-0.0059
SP1REPWGT94	-0.0079	-0.012	-0.018	-0.016	0.022	-0.0047	0.0041
SP1REPWGT95	-0.0078	-0.016	-0.016	-0.0089	0.018	-0.011	0.0038
SP1REPWGT96	-0.012	-0.012	-0.015	-0.021	0.021	-0.0067	0.0066
SP1REPWGT97	-0.01	-0.017	-0.023	-0.017	0.011	-0.012	-0.00079
SP1REPWGT98	-0.0093	-0.013	-0.026	-0.014	0.02	-0.0052	0.0044
SP1REPWGT99	-0.013	-0.02	-0.019	-0.019	0.014	-0.0086	-0.00051
SP1REPWGT100	-0.0081	-0.0095	-0.018	-0.015	0.023	-0.0067	0.0049
SP1REPWGT101	-0.012	-0.018	-0.02	-0.021	0.028	-0.0082	0.0027
SP1REPWGT102	-0.012	-0.016	-0.021	-0.019	0.023	-0.0093	-0.00035
SP1REPWGT103	-0.012	-0.019	-0.022	-0.018	0.018	-0.0057	0.0007
SP1REPWGT104	-0.0084	-0.02	-0.022	-0.012	0.022	-0.011	0.0053
SP1REPWGT105	-0.013	-0.02	-0.016	-0.02	0.016	-0.0033	-0.0012
SP1REPWGT106	-0.012	-0.015	-0.022	-0.016	0.029	-0.015	0.0048
SP1REPWGT107	-0.0093	-0.014	-0.017	-0.014	0.016	-0.0063	-0.0005
SP1REPWGT108	-0.0094	-0.016	-0.022	-0.017	0.02	-0.0097	0.002
SP1REPWGT109	-0.013	-0.02	-0.024	-0.017	0.019	-0.0082	-0.00046
SP1REPWGT110	-0.015	-0.02	-0.021	-0.022	0.022	-0.011	-0.0016
SP1REPWGT111	-0.013	-0.018	-0.021	-0.022	0.02	-0.011	0.0012

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
SP1REPWGT112	-0.01	-0.018	-0.024	-0.018	0.02	-0.008	0.0052
SP1REPWGT113	-0.012	-0.02	-0.015	-0.021	0.018	-0.014	0.0037
SP1REPWGT114	-0.011	-0.016	-0.023	-0.017	0.027	-0.011	0.00089
SP1REPWGT115	-0.011	-0.02	-0.021	-0.017	0.02	-0.013	0.0039
SP1REPWGT116	-0.01	-0.015	-0.025	-0.014	0.018	-0.005	-0.003
SP1REPWGT117	-0.015	-0.019	-0.021	-0.025	0.022	-0.013	0.0049
SP1REPWGT118	-0.011	-0.019	-0.017	-0.017	0.02	-0.0082	0.0014
SP1REPWGT119	-0.012	-0.019	-0.014	-0.017	0.022	-0.01	0.0038
SP1REPWGT120	-0.0082	-0.016	-0.029	-0.016	0.019	-0.0062	0.00054
SP1REPWGT121	-0.012	-0.018	-0.015	-0.018	0.02	-0.0085	0.0031
SP1REPWGT122	-0.0095	-0.018	-0.022	-0.018	0.019	-0.0071	0.0095
SP1REPWGT123	-0.01	-0.013	-0.02	-0.018	0.022	-0.0015	0.002
SP1REPWGT124	-0.013	-0.02	-0.016	-0.018	0.015	-0.0086	0.0099
SP1REPWGT125	-0.01	-0.021	-0.024	-0.016	0.022	-0.002	-0.0019
SP1REPWGT126	-0.0089	-0.015	-0.02	-0.02	0.023	-0.01	0.0021
SP1REPWGT127	-0.007	-0.013	-0.021	-0.016	0.024	-0.0018	-0.0061
SP1REPWGT128	-0.011	-0.019	-0.022	-0.019	0.022	-0.0086	-0.0036
SP1REPWGT129	-0.012	-0.019	-0.017	-0.02	0.02	-0.0059	0.0038
SP1REPWGT130	-0.0092	-0.019	-0.024	-0.01	0.018	-0.0061	-0.0072
SP1REPWGT131	-0.013	-0.018	-0.02	-0.018	0.024	-0.01	0.0055
SP1REPWGT132	-0.011	-0.019	-0.017	-0.02	0.024	-0.0037	-0.0064
SP1REPWGT133	-0.0078	-0.018	-0.02	-0.017	0.017	-0.0089	0.00067
SP1REPWGT134	-0.012	-0.021	-0.014	-0.015	0.024	-0.005	0.0085
SP1REPWGT135	-0.0081	-0.017	-0.016	-0.0093	0.023	-0.0095	-0.0043
SP1REPWGT136	-0.0096	-0.018	-0.014	-0.016	0.021	-0.0035	0.005
SP1REPWGT137	-0.0087	-0.017	-0.029	-0.02	0.019	-0.0074	-0.0058
SP1REPWGT138	-0.013	-0.015	-0.011	-0.017	0.015	-0.0083	-0.00023
SP1REPWGT139	-0.011	-0.021	-0.023	-0.017	0.019	-0.0097	0.0031
SP1REPWGT140	-0.014	-0.019	-0.014	-0.02	0.021	-0.0065	0.00015
SP1REPWGT141	-0.0095	-0.02	-0.015	-0.014	0.028	-0.0057	0.0011
SP1REPWGT142	-0.011	-0.013	-0.016	-0.018	0.02	-0.01	-0.0021
SP1REPWGT143	-0.015	-0.017	-0.015	-0.017	0.013	-0.0062	0.0078
SP1REPWGT144	-0.0083	-0.019	-0.024	-0.016	0.024	-0.0035	0.0063
SP1REPWGT145	-0.015	-0.017	-0.019	-0.018	0.019	-0.013	-0.0011
SP1REPWGT146	-0.0074	-0.018	-0.013	-0.017	0.025	-0.0093	-0.0013
SP1REPWGT147	-0.012	-0.016	-0.033	-0.019	0.02	-0.0065	-0.0072

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
SP1REPWGT148	-0.018	-0.017	-0.02	-0.024	0.023	-0.012	0.0027
SP1REPWGT149	-0.01	-0.022	-0.017	-0.015	0.018	-0.012	0.0032
SP1REPWGT150	-0.0054	-0.0099	-0.022	-0.011	0.023	-0.0031	-0.0012
SP1REPWGT151	-0.01	-0.017	-0.022	-0.017	0.026	-0.0049	-0.0034
SP1REPWGT152	-0.012	-0.021	-0.02	-0.026	0.022	-0.011	0.0086
SP1REPWGT153	-0.013	-0.014	-0.022	-0.021	0.018	-0.013	-0.0011
SP1REPWGT154	-0.0077	-0.017	-0.017	-0.014	0.025	-0.012	-0.0026
SP1REPWGT155	-0.01	-0.016	-0.021	-0.016	0.023	-0.0088	0.0092
SP1REPWGT156	-0.0073	-0.014	-0.019	-0.017	0.019	-0.0074	0.00094
SP1REPWGT157	-0.012	-0.015	-0.021	-0.021	0.02	-0.012	-0.0066
SP1REPWGT158	-0.011	-0.016	-0.02	-0.02	0.023	-0.0074	0.0069
SP1REPWGT159	-0.01	-0.015	-0.022	-0.013	0.022	-0.0054	-0.0063
SP1REPWGT160	-0.0086	-0.02	-0.02	-0.016	0.026	-0.0093	0.006
SP2REPWGT1	-0.022	-0.025	1.7e-16	-0.024	0.031	-0.016	2.4e-17
SP2REPWGT2	-0.018	-0.021	2.8e-17	-0.027	0.031	-0.012	1.3e-16
SP2REPWGT3	-0.015	-0.016	-1.7e-16	-0.026	0.029	-0.016	1.9e-17
SP2REPWGT4	-0.015	-0.027	1.8e-16	-0.018	0.031	-0.019	2e-16
SP2REPWGT5	-0.013	-0.021	-7.6e-17	-0.029	0.032	-0.017	-1.1e-16
SP2REPWGT6	-0.018	-0.019	1.3e-16	-0.023	0.025	-0.0097	1.5e-16
SP2REPWGT7	-0.018	-0.027	-4.5e-17	-0.024	0.028	-0.022	-2.2e-16
SP2REPWGT8	-0.016	-0.023	2.9e-16	-0.022	0.031	-0.013	1.9e-16
SP2REPWGT9	-0.014	-0.022	-7.5e-17	-0.02	0.028	-0.015	-1.4e-16
SP2REPWGT10	-0.015	-0.021	6.4e-17	-0.019	0.027	-0.013	-5.2e-17
SP2REPWGT11	-0.014	-0.019	6.2e-17	-0.021	0.027	-0.016	-3.3e-17
SP2REPWGT12	-0.016	-0.021	-9.5e-17	-0.024	0.029	-0.014	2.1e-16
SP2REPWGT13	-0.018	-0.023	-1.8e-17	-0.027	0.033	-0.0088	1.4e-16
SP2REPWGT14	-0.012	-0.021	7e-17	-0.021	0.026	-0.014	1.1e-16
SP2REPWGT15	-0.019	-0.017	-1.3e-16	-0.028	0.034	-0.008	-1.9e-16
SP2REPWGT16	-0.014	-0.021	8.2e-18	-0.02	0.032	-0.019	6.3e-18
SP2REPWGT17	-0.015	-0.026	-3.1e-17	-0.022	0.031	-0.0076	-5.1e-17
SP2REPWGT18	-0.013	-0.023	-2.6e-17	-0.023	0.027	-0.014	1.9e-16

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
SP2REPWGT19	-0.02	-0.022	1.5e-16	-0.026	0.025	-0.011	7.3e-17
SP2REPWGT20	-0.013	-0.019	-4.5e-17	-0.022	0.03	-0.02	-1.5e-16
SP2REPWGT21	-0.016	-0.021	4.5e-17	-0.029	0.03	-0.013	2.9e-16
SP2REPWGT22	-0.015	-0.019	1.7e-16	-0.028	0.028	-0.013	2.6e-16
SP2REPWGT23	-0.014	-0.024	-1.8e-16	-0.021	0.031	-0.015	-2.1e-17
SP2REPWGT24	-0.018	-0.024	-2.8e-17	-0.026	0.03	-0.018	1.5e-16
SP2REPWGT25	-0.017	-0.017	-1.9e-16	-0.023	0.029	-0.011	8.6e-17
SP2REPWGT26	-0.019	-0.027	-9.3e-17	-0.025	0.027	-0.021	5.6e-17
SP2REPWGT27	-0.018	-0.02	-1.7e-16	-0.022	0.034	-0.016	-8.3e-18
SP2REPWGT28	-0.013	-0.027	-5.8e-17	-0.024	0.03	-0.021	7.2e-17
SP2REPWGT29	-0.018	-0.019	3.8e-17	-0.026	0.032	-0.0078	1.9e-16
SP2REPWGT30	-0.018	-0.026	-2.2e-17	-0.025	0.03	-0.015	-7.9e-17
SP2REPWGT31	-0.013	-0.021	-1.3e-16	-0.024	0.033	-0.016	9.8e-17
SP2REPWGT32	-0.018	-0.027	2e-17	-0.024	0.029	-0.013	3e-16
SP2REPWGT33	-0.021	-0.023	-2.6e-17	-0.024	0.036	-0.019	1.6e-17
SP2REPWGT34	-0.014	-0.023	1e-16	-0.022	0.026	-0.013	5e-17
SP2REPWGT35	-0.019	-0.021	-1.6e-16	-0.027	0.026	-0.015	1.3e-16
SP2REPWGT36	-0.016	-0.029	2.2e-17	-0.024	0.032	-0.012	-2e-16
SP2REPWGT37	-0.022	-0.021	6.4e-17	-0.027	0.028	-0.015	-1.2e-17
SP2REPWGT38	-0.018	-0.025	-3.2e-17	-0.026	0.034	-0.015	-6.9e-17
SP2REPWGT39	-0.019	-0.023	-1e-16	-0.022	0.027	-0.016	3e-17
SP2REPWGT40	-0.012	-0.019	-2.5e-17	-0.021	0.032	-0.014	1.5e-16
SP2REPWGT41	-0.019	-0.028	-3.7e-18	-0.023	0.033	-0.019	2.7e-16
SP2REPWGT42	-0.016	-0.023	1.2e-16	-0.024	0.028	-0.016	-1.5e-16
SP2REPWGT43	-0.013	-0.021	1.4e-16	-0.023	0.031	-0.013	1.7e-16
SP2REPWGT44	-0.017	-0.026	-3.6e-16	-0.026	0.024	-0.015	-5.8e-17
SP2REPWGT45	-0.015	-0.022	-1.1e-16	-0.024	0.036	-0.01	-1.6e-16
SP2REPWGT46	-0.018	-0.019	2e-18	-0.027	0.031	-0.015	8.5e-17
SP2REPWGT47	-0.013	-0.022	-1.8e-16	-0.023	0.027	-0.0088	1.6e-16
SP2REPWGT48	-0.017	-0.026	-5.3e-16	-0.028	0.032	-0.021	7.2e-17
SP2REPWGT49	-0.0096	-0.021	7.2e-18	-0.021	0.027	-0.0064	1.5e-16

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
SP2REPWGT50	-0.017	-0.025	3.6e-17	-0.022	0.03	-0.013	3e-16
SP2REPWGT51	-0.014	-0.024	1.5e-17	-0.021	0.03	-0.014	1.7e-17
SP2REPWGT52	-0.017	-0.023	1.1e-18	-0.028	0.03	-0.012	-1.2e-16
SP2REPWGT53	-0.014	-0.021	4.6e-17	-0.026	0.029	-0.0075	9.9e-17
SP2REPWGT54	-0.019	-0.023	-1.9e-17	-0.023	0.032	-0.013	5.6e-17
SP2REPWGT55	-0.016	-0.03	3.7e-17	-0.023	0.031	-0.014	9.5e-17
SP2REPWGT56	-0.017	-0.022	-5.3e-16	-0.025	0.032	-0.011	-2.6e-17
SP2REPWGT57	-0.015	-0.023	-7.9e-17	-0.022	0.033	-0.013	-2.4e-17
SP2REPWGT58	-0.012	-0.023	-2.8e-17	-0.019	0.027	-0.015	2.4e-16
SP2REPWGT59	-0.016	-0.019	1.4e-16	-0.02	0.028	-0.011	1.1e-16
SP2REPWGT60	-0.018	-0.021	-1.5e-16	-0.029	0.029	-0.012	-1.1e-16
SP2REPWGT61	-0.014	-0.025	1e-16	-0.024	0.032	-0.016	2.3e-16
SP2REPWGT62	-0.016	-0.021	-9.8e-17	-0.023	0.028	-0.014	9.1e-17
SP2REPWGT63	-0.017	-0.024	1.4e-16	-0.024	0.027	-0.012	8e-17
SP2REPWGT64	-0.0092	-0.021	1.3e-16	-0.019	0.024	-0.014	2.9e-16
SP2REPWGT65	-0.019	-0.022	1.1e-16	-0.024	0.027	-0.017	1e-16
SP2REPWGT66	-0.0083	-0.023	-1.4e-16	-0.025	0.029	-0.014	-6.7e-17
SP2REPWGT67	-0.019	-0.023	1.3e-16	-0.024	0.032	-0.013	2.3e-17
SP2REPWGT68	-0.018	-0.022	-8.3e-17	-0.028	0.037	-0.016	2e-16
SP2REPWGT69	-0.015	-0.024	2e-17	-0.024	0.025	-0.015	-1.1e-16
SP2REPWGT70	-0.013	-0.026	-5.7e-17	-0.018	0.03	-0.012	1.5e-16
SP2REPWGT71	-0.013	-0.024	-5.6e-17	-0.022	0.037	-0.014	1.8e-16
SP2REPWGT72	-0.019	-0.02	4.1e-17	-0.024	0.026	-0.013	2.8e-16
SP2REPWGT73	-0.017	-0.019	5.1e-17	-0.022	0.037	-0.018	2.6e-18
SP2REPWGT74	-0.02	-0.026	1.4e-16	-0.024	0.026	-0.015	7.9e-18
SP2REPWGT75	-0.019	-0.021	1.6e-16	-0.029	0.029	-0.016	1.2e-16
SP2REPWGT76	-0.014	-0.025	-3.5e-17	-0.021	0.029	-0.011	-1.5e-16
SP2REPWGT77	-0.019	-0.021	-2.6e-17	-0.019	0.024	-0.015	1.9e-16
SP2REPWGT78	-0.016	-0.017	1.7e-16	-0.026	0.029	-0.0088	1.2e-16

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
SP2REPWGT79	-0.01	-0.018	-6.4e-17	-0.02	0.031	-0.0074	2.4e-16
SP2REPWGT80	-0.016	-0.023	1.5e-16	-0.025	0.031	-0.019	-2.8e-17
SP2REPWGT81	-0.022	-0.026	-1.5e-16	-0.027	0.03	-0.013	-4.8e-17
SP2REPWGT82	-0.023	-0.018	-6.5e-17	-0.03	0.025	-0.014	2.2e-16
SP2REPWGT83	-0.015	-0.019	1.3e-16	-0.024	0.031	-0.012	-1.3e-17
SP2REPWGT84	-0.016	-0.025	-2.8e-17	-0.02	0.032	-0.014	-2e-17
SP2REPWGT85	-0.017	-0.024	-2.6e-17	-0.028	0.03	-0.016	-2.3e-17
SP2REPWGT86	-0.019	-0.018	2.1e-16	-0.027	0.027	-0.013	2.3e-16
SP2REPWGT87	-0.017	-0.024	1e-17	-0.021	0.03	-0.016	6.4e-17
SP2REPWGT88	-0.018	-0.02	4.1e-17	-0.022	0.025	-0.013	1.6e-16
SP2REPWGT89	-0.021	-0.021	-8.6e-18	-0.026	0.032	-0.018	-8.1e-17
SP2REPWGT90	-0.017	-0.023	-4.3e-17	-0.022	0.028	-0.01	-2.3e-17
SP2REPWGT91	-0.018	-0.021	-1.6e-17	-0.023	0.029	-0.018	2.2e-16
SP2REPWGT92	-0.017	-0.018	-4.3e-17	-0.025	0.03	-0.015	6.1e-17
SP2REPWGT93	-0.019	-0.025	-1.2e-16	-0.024	0.028	-0.012	3.7e-17
SP2REPWGT94	-0.014	-0.021	1.2e-16	-0.022	0.028	-0.011	2.2e-17
SP2REPWGT95	-0.019	-0.018	-8.6e-17	-0.023	0.028	-0.012	2e-16
SP2REPWGT96	-0.013	-0.016	-1.5e-16	-0.02	0.026	-0.011	3.6e-18
SP2REPWGT97	-0.015	-0.025	-6.4e-17	-0.023	0.03	-0.014	-4.4e-17
SP2REPWGT98	-0.018	-0.021	-5.6e-16	-0.025	0.023	-0.017	1.2e-16
SP2REPWGT99	-0.017	-0.026	-1.2e-16	-0.025	0.029	-0.011	-3e-17
SP2REPWGT100	-0.02	-0.02	7.8e-17	-0.026	0.029	-0.012	2.5e-16
SP2REPWGT101	-0.016	-0.022	1.7e-16	-0.03	0.035	-0.01	-1.9e-16
SP2REPWGT102	-0.016	-0.019	-6.5e-17	-0.024	0.027	-0.0085	2.4e-16
SP2REPWGT103	-0.018	-0.022	2.6e-17	-0.025	0.033	-0.014	1.7e-16
SP2REPWGT104	-0.016	-0.022	-1.1e-18	-0.026	0.031	-0.016	1.1e-16
SP2REPWGT105	-0.02	-0.021	7.1e-17	-0.022	0.026	-0.012	6.8e-17
SP2REPWGT106	-0.021	-0.021	-2.1e-17	-0.025	0.034	-0.02	7e-17
SP2REPWGT107	-0.014	-0.019	5.3e-17	-0.019	0.026	-0.011	1.4e-17
SP2REPWGT108	-0.018	-0.021	1.4e-16	-0.022	0.026	-0.018	9.9e-17
SP2REPWGT109	-0.02	-0.024	3.1e-17	-0.023	0.027	-0.014	1.8e-16

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
SP2REPWGT110	-0.02	-0.021	3.7e-17	-0.029	0.038	-0.013	3.2e-17
SP2REPWGT111	-0.017	-0.021	-2.9e-18	-0.026	0.023	-0.013	6.6e-17
SP2REPWGT112	-0.015	-0.03	5.3e-18	-0.024	0.03	-0.016	1e-16
SP2REPWGT113	-0.018	-0.02	-4e-16	-0.023	0.031	-0.02	-2.4e-16
SP2REPWGT114	-0.016	-0.024	-2.9e-16	-0.026	0.032	-0.014	-1.1e-17
SP2REPWGT115	-0.015	-0.02	5.7e-17	-0.021	0.027	-0.014	1.2e-16
SP2REPWGT116	-0.015	-0.023	6e-17	-0.023	0.029	-0.01	-1.6e-16
SP2REPWGT117	-0.023	-0.021	-8.7e-17	-0.027	0.032	-0.018	8e-17
SP2REPWGT118	-0.02	-0.024	1.6e-16	-0.026	0.03	-0.017	1.2e-16
SP2REPWGT119	-0.023	-0.027	9e-17	-0.022	0.026	-0.019	-3.1e-17
SP2REPWGT120	-0.01	-0.015	8.5e-17	-0.016	0.031	-0.012	1.6e-16
SP2REPWGT121	-0.018	-0.029	4.4e-18	-0.025	0.031	-0.015	7.3e-17
SP2REPWGT122	-0.02	-0.022	1.2e-16	-0.029	0.025	-0.015	3e-16
SP2REPWGT123	-0.012	-0.019	1.4e-16	-0.022	0.028	-0.0077	-6.8e-17
SP2REPWGT124	-0.018	-0.028	-5.3e-16	-0.024	0.024	-0.018	1.4e-16
SP2REPWGT125	-0.017	-0.021	1.5e-16	-0.028	0.035	-0.013	-9e-17
SP2REPWGT126	-0.018	-0.018	1.2e-16	-0.023	0.032	-0.017	5.9e-17
SP2REPWGT127	-0.011	-0.019	-7e-17	-0.025	0.029	-0.007	1.6e-16
SP2REPWGT128	-0.021	-0.02	-4.1e-16	-0.028	0.03	-0.012	-2.3e-16
SP2REPWGT129	-0.017	-0.025	1e-16	-0.025	0.029	-0.015	4.5e-17
SP2REPWGT130	-0.015	-0.026	-3e-17	-0.025	0.033	-0.015	2.1e-16
SP2REPWGT131	-0.02	-0.023	-6.6e-17	-0.022	0.026	-0.014	-2.6e-17
SP2REPWGT132	-0.014	-0.021	5e-17	-0.023	0.03	-0.013	9.6e-18
SP2REPWGT133	-0.013	-0.024	2.3e-17	-0.027	0.028	-0.0099	6.5e-17
SP2REPWGT134	-0.015	-0.019	-4.8e-17	-0.021	0.028	-0.018	9.4e-17
SP2REPWGT135	-0.018	-0.026	7.5e-17	-0.021	0.024	-0.016	2.9e-16
SP2REPWGT136	-0.012	-0.023	4.6e-17	-0.022	0.028	-0.015	1.3e-16
SP2REPWGT137	-0.015	-0.022	-4.2e-17	-0.024	0.029	-0.011	-1.8e-16
SP2REPWGT138	-0.021	-0.024	1e-16	-0.027	0.026	-0.023	-5.6e-17

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
SP2REPWGT139	-0.013	-0.02	-2.8e-17	-0.019	0.027	-0.0087	2.1e-16
SP2REPWGT140	-0.019	-0.023	-1.3e-16	-0.028	0.027	-0.013	2.8e-17
SP2REPWGT141	-0.013	-0.027	3.1e-17	-0.023	0.035	-0.014	2.1e-16
SP2REPWGT142	-0.024	-0.02	1e-16	-0.031	0.026	-0.015	4.2e-17
SP2REPWGT143	-0.019	-0.021	6.5e-17	-0.023	0.024	-0.0094	2.7e-16
SP2REPWGT144	-0.013	-0.024	-5.3e-17	-0.022	0.03	-0.012	1.8e-16
SP2REPWGT145	-0.022	-0.026	3.9e-17	-0.027	0.03	-0.015	9.7e-17
SP2REPWGT146	-0.013	-0.019	1.7e-16	-0.023	0.027	-0.012	5.9e-18
SP2REPWGT147	-0.017	-0.014	-5e-16	-0.024	0.032	-0.011	-2.7e-17
SP2REPWGT148	-0.019	-0.022	-5.4e-16	-0.028	0.029	-0.016	5.3e-17
SP2REPWGT149	-0.016	-0.027	8.5e-17	-0.021	0.026	-0.013	-7.5e-17
SP2REPWGT150	-0.018	-0.021	-7.4e-17	-0.025	0.032	-0.011	3e-17
SP2REPWGT151	-0.016	-0.021	2e-16	-0.024	0.029	-0.014	1e-16
SP2REPWGT152	-0.022	-0.024	-3.8e-17	-0.026	0.03	-0.016	-3.5e-17
SP2REPWGT153	-0.018	-0.02	-6.3e-16	-0.028	0.032	-0.014	-7.4e-17
SP2REPWGT154	-0.022	-0.03	6.8e-17	-0.024	0.028	-0.02	8.3e-17
SP2REPWGT155	-0.02	-0.02	2.5e-17	-0.028	0.032	-0.016	2e-16
SP2REPWGT156	-0.016	-0.02	1.2e-17	-0.018	0.026	-0.01	1.3e-16
SP2REPWGT157	-0.018	-0.024	3.3e-17	-0.028	0.028	-0.014	2.8e-16
SP2REPWGT158	-0.021	-0.022	-1.1e-16	-0.029	0.028	-0.013	-2.3e-16
SP2REPWGT159	-0.016	-0.018	-9.4e-17	-0.021	0.03	-0.013	1.4e-16
SP2REPWGT160	-0.012	-0.022	6.8e-17	-0.019	0.034	-0.012	-8.9e-17
MARKETVAL	0.19	0.14	0.0044	0.088	0.011	0.15	-0.027
TOTBALAMT	0.15	0.14	0.018	0.067	0.026	0.074	-0.028
PROTAXAMT	0.24	0.19	0.017	0.12	0.0018	0.18	-0.032
INSURAMT	0.23	0.15	0.014	0.12	0.051	0.038	-0.027
HOAAMT	-0.097	0.029	-0.0042	-0.029	-0.032	0.22	-0.0031
MAINTAMT	0.16	0.12	0.021	0.094	0.0047	0.047	-0.0068
MORTAMT	0.081	0.07	0.0098	0.037	0.0045	0.046	-0.015
HINCP	0.22	0.44	0.017	0.089	0.055	0.13	-0.015

	TOTROOMS	PERPOVLVL	COMCOST	DINING	LAUNDY	STORIES	COMDAYS
FINCP	0.22	0.43	0.016	0.09	0.057	0.13	-0.016
REMODAMT	0.12	0.11	0.00092	0.04	0.048	0.029	-0.026
TOTHCAMT	0.16	0.17	0.023	0.075	0.019	0.1	-0.009

The dataset was cleaned to make it free from erroneous or irrelevant data. By filling up missing values, removing rows and reducing data size, the final dataset was (36358 rows X 1006 columns).

Dataset Split

Now we will separate the Test Data and Train Data. Will keep 30% of the data for Testing purpose and rest for training purpose.

In [23]:
```
# Separating out the target
y = clean_data['MARKETVAL']

# Separating out the features
x = clean_data.drop('MARKETVAL', axis = 1)

X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, rando
m_state=0)
```

In [24]:
```
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

```
(25450, 1005) (10908, 1005) (25450,) (10908,)
```

In [25]:
```
# Dividing the dataset into numerical and categorical features
col_o = list(x.columns)
numeric = []
categorical = []
e = []
for c in col_o:
    j = 0
    if c[0]=='J':
        j = 1
        c = c[1:]
    h = headings.loc[headings['Variable']== c]['TYPE'].tolist()
    if h != []:
        if (h[0] == 'Character'):
            if j == 0:
                categorical.append(c)
            elif j == 1:
                categorical.append('J' + c)
        elif (h[0] == 'Numeric'):
            if j == 0:
                numeric.append(c)
            elif j == 1:
                numeric.append('J' + c)
        else:
            if j == 0:
                e.append(c)
            elif j == 1:
                e.append('J' + c)
```

In [26]:
```
# Training and Test Data for numeric features
X_train_numeric = X_train.drop(categorical, axis = 1)
X_test_numeric = X_test.drop(categorical, axis = 1)
```

Now, we will stadardize and normalyze the data.

In [27]:
```
# Scalar Transform the Numeric Variable
scaler = MinMaxScaler()#StandardScaler()
scaler.fit(X_train_numeric)

# Apply transform to both the training set and the test set.
X_train_numeric_trans = pd.DataFrame(scaler.transform(X_train_numeric))
X_test_numeric_trans = pd.DataFrame(scaler.transform(X_test_numeric))

X_train_numeric_trans.columns = X_train_numeric.columns
X_test_numeric_trans.columns = X_test_numeric.columns
```

In [28]:
```
# Training and Test Data for catagorical features
X_train_categorical = X_train.drop(numeric, axis = 1)
X_test_categorical = X_test.drop(numeric, axis = 1)
```

In [29]:
```
# Resetting the index to avoid nan on concatenation
X_train_numeric_trans.reset_index(drop=True, inplace=True)
X_train_categorical.reset_index(drop=True, inplace=True)

X_test_numeric_trans.reset_index(drop=True, inplace=True)
X_test_categorical.reset_index(drop=True, inplace=True)
```

In [30]:
```
# Concatenating numeric and catagorical features
X_train = pd.concat([X_train_numeric_trans, X_train_categorical], axis=1, sort
=False)
X_test = pd.concat([X_test_numeric_trans, X_test_categorical], axis=1, sort=Fa
lse)
```

In [31]:
```
# Remove duplicate columns after concatenation
X_train = X_train.iloc[:,~X_train.columns.duplicated()]
X_train.shape
```

Out[31]: (25450, 1005)

In [32]:
```
# Remove duplicate columns after concatenation
X_test = X_test.iloc[:,~X_test.columns.duplicated()]
X_test.shape
```

Out[32]: (10908, 1005)

Price Range Encoding

Since, we want to classify houses into different price ranges, we will need to perform feature encoding for various price ranges.

In [33]:
```
# Final formatting before applying algorithms
y_train = pd.DataFrame(y_train)
y_train = y_train.reset_index(drop=True)
y_train_int = y_train.astype(int)

y_test = pd.DataFrame(y_test)
y_test = y_test.reset_index(drop=True)
y_test_int = y_test.astype(int)
```

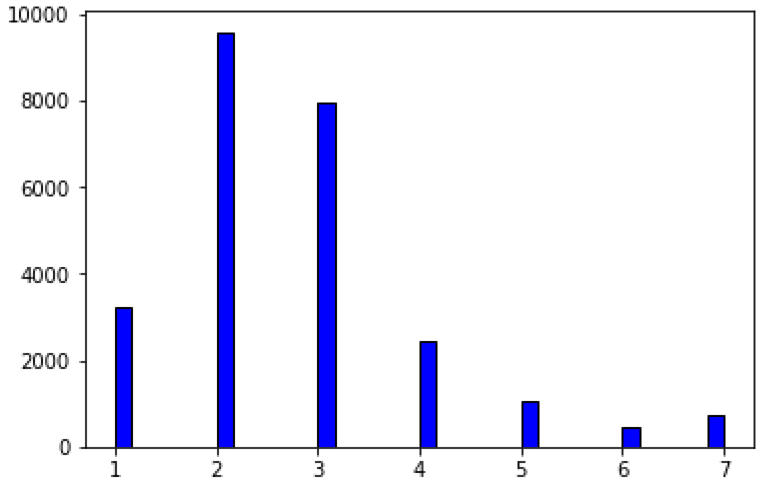
In [34]:
```
# Function to assign code for different price ranges
def price_range(price):
    if price < 100000:
        return 1
    elif price >= 100000 and price < 250000:
        return 2
    elif price >= 250000 and price < 500000:
        return 3
    elif price >= 500000 and price < 750000:
        return 4
    elif price >= 750000 and price < 1000000:
        return 5
    elif price >= 1000000 and price < 1250000:
        return 6
    elif price >= 1250000:
        return 7
    else:
        print(price)
        return 13
```

In [35]:
```
# Get the price range encoded field in y dataset
y_train['MARKETVAL'] = y_train_int['MARKETVAL'].apply(price_range)
y_test['MARKETVAL'] = y_test_int['MARKETVAL'].apply(price_range)
```

From below histograms it can be seen that most houses fall in the MARKETVAL range of 100000 to 250000

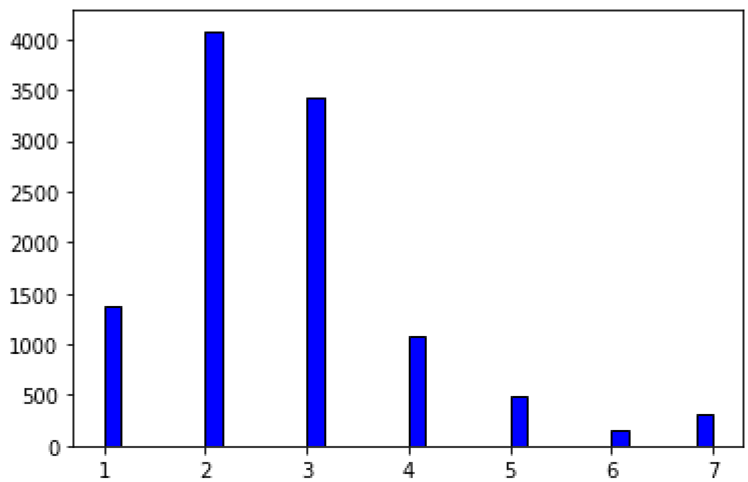
In [36]:
```
# Distribution of data in y training dataset
plt.hist(y_train['MARKETVAL'], bins = int(180/5), color = 'blue', edgecolor =
'black')
```

Out[36]:
```
(array([3232.,    0.,    0.,    0.,    0.,    0., 9582.,    0.,    0.,
           0.,    0.,    0., 7952.,    0.,    0.,    0.,    0.,    0.,
        2464.,    0.,    0.,    0.,    0.,    0., 1078.,    0.,    0.,
           0.,    0.,    0.,  433.,    0.,    0.,    0.,    0.,  709.]),
 array([1.        , 1.16666667, 1.33333333, 1.5       , 1.66666667,
        1.83333333, 2.        , 2.16666667, 2.33333333, 2.5       ,
        2.66666667, 2.83333333, 3.        , 3.16666667, 3.33333333,
        3.5       , 3.66666667, 3.83333333, 4.        , 4.16666667,
        4.33333333, 4.5       , 4.66666667, 4.83333333, 5.        ,
        5.16666667, 5.33333333, 5.5       , 5.66666667, 5.83333333,
        6.        , 6.16666667, 6.33333333, 6.5       , 6.66666667,
        6.83333333, 7.        ]),
 <a list of 36 Patch objects>)
```



In [37]:
```
# Distribution of data in y testing dataset
plt.hist(y_test['MARKETVAL'], bins = int(180/5), color = 'blue', edgecolor =
'black')
```

Out[37]:
```
(array([1377.,    0.,    0.,    0.,    0.,    0., 4080.,    0.,    0.,
           0.,    0.,    0., 3422.,    0.,    0.,    0.,    0.,    0.,
        1071.,    0.,    0.,    0.,    0.,    0.,  496.,    0.,    0.,
           0.,    0.,    0.,  157.,    0.,    0.,    0.,    0.,  305.]),
 array([1.        , 1.16666667, 1.33333333, 1.5       , 1.66666667,
        1.83333333, 2.        , 2.16666667, 2.33333333, 2.5       ,
        2.66666667, 2.83333333, 3.        , 3.16666667, 3.33333333,
        3.5       , 3.66666667, 3.83333333, 4.        , 4.16666667,
        4.33333333, 4.5       , 4.66666667, 4.83333333, 5.        ,
        5.16666667, 5.33333333, 5.5       , 5.66666667, 5.83333333,
        6.        , 6.16666667, 6.33333333, 6.5       , 6.66666667,
        6.83333333, 7.        ]),
 <a list of 36 Patch objects>)
```



In [38]:
```
# One Hot Encoding to represent categorical variables as binary vectors
le = LabelEncoder()
le.fit(y_train)
y_train_le = le.transform(y_train['MARKETVAL'])#.reshape(-1, 1)
#y_test_le = le.transform(y_train['MARKETVAL'])#.reshape(-1, 1)

oh = OneHotEncoder(sparse=False)
y_train_le = y_train_le.reshape(len(y_train_le), 1)
oh.fit(y_train_le)

y_train_oh = oh.transform(y_train_le)
```

Performance Measures

The function accuracy is used to calculate accuracy scores for both training and testing dataset for different ML models. The function train_model is used to train and fit the data for different classifier models.

In [39]:
```
# Calculating accuracy score for training and testing datasets
def accuracy(X_train, X_test, y_train, y_test, model):
    y_pred_train = model.predict(X_train)
    train_accuracy = accuracy_score(y_train.values, y_pred_train)
    print(f"Train accuracy: {train_accuracy:0.2%}")

    y_pred_test = model.predict(X_test)
    test_accuracy = accuracy_score(y_test.values, y_pred_test)
    print(f"Test accuracy: {test_accuracy:0.2%}")

    # For comparison of models later
    return test_accuracy
```

In [40]:
```
# Function to train data based on different classifiers
def train_model(X_train, X_test, classifier, **kwargs):
    model = classifier(**kwargs)
    model.fit(X_train, y_train)
    return model
```

Algorithms Implemented

In this project, my aim was to implement algorithms which will be able to learn and classify the new observations to correct house price ranges. I decided to use below machine learning algorithms for the same-
• Random Forest (RandomForestClassifier)
• Logistic Regression (LogisticRegression)
• K-Nearest Neighbor (KNeighborsClassifier)
• Decision Tree (DecisionTreeClassifier)

In [41]:
```
# Test accuracy for all models for comparison later
accuracy_val = []
# List of Algorithms Mames
classifiers = ['Random Forest', 'Logistic Regression', 'Knn (7 Neighbors)', 'D
ecision Tree']
```

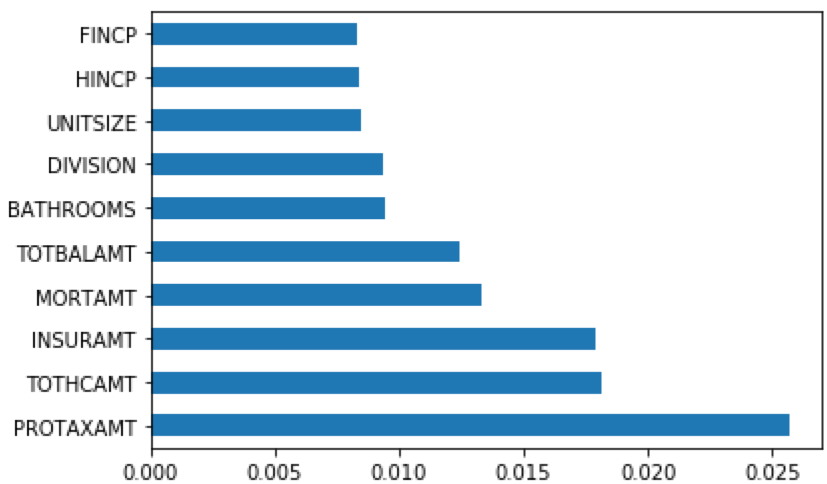
Random Forest Classifier

The random forest is a model made up of many decision trees. Rather than just simply averaging the prediction of trees, this model uses two key concepts that gives it the name random:
• Random sampling of training data points when building trees
• Random subsets of features considered when splitting nodes
The random forest combines hundreds or thousands of decision trees, trains each one on a slightly different set of the observations, splitting nodes in each tree considering a limited number of the features. The final predictions of the random forest are made by averaging the predictions of each individual tree.

In [42]:
```
# Using Random Forest Classifier
model = train_model(X_train, y_train_oh, RandomForestClassifier, n_estimators=
200, random_state=20)
test_accuracy_val = accuracy(X_train, X_test, y_train, y_test, model)
accuracy_val.append(test_accuracy_val)
# Top 10 features that determine price
pd.Series(model.feature_importances_, x.columns).sort_values(ascending=True).n
largest(10).plot.barh(align='center')
```

```
Train accuracy: 100.00%
Test accuracy: 55.90%
```

Out[42]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x178b7729240>
```



Results: With RandomForestClassifier, the accuracy score were as below:
Training Accuracy – 100.00%
Testing Accuracy – 55.90%

I also plotted a bar graph representing the top 10 features based on their importance in determining the house price range.

Logistic Regression

Logistic regression is one of the most fundamental and widely used Machine Learning Algorithms. Logistic regression is not a regression algorithm but a probabilistic classification model. Multi class classification is implemented by training multiple logistic regression classifiers, one for each of the K classes in the training dataset.

In [43]:
```
# Using Logistic Regression
model = train_model(X_train, y_train, LogisticRegression,solver='lbfgs')
test_accuracy_val = accuracy(X_train, X_test, y_train, y_test, model)
accuracy_val.append(test_accuracy_val)
```

```
Train accuracy: 47.08%
Test accuracy: 46.68%
```

Results: With LogisticRegression, the accuracy score were as below:
Training Accuracy – 47.08%
Testing Accuracy – 46.68%

k-Nearest Neighbor

KNN or k-nearest neighbours is the simplest classification algorithm. This classification algorithm does not depend on the structure of the data. Whenever a new example is encountered, its k nearest neighbours from the training data are examined. Distance between two examples can be the euclidean distance between their feature vectors. The majority class among the k nearest neighbours is taken to be the class for the encountered example.

In [44]:
```
# Using kNN Classifier
model = train_model(X_train, y_train, KNeighborsClassifier, n_neighbors=7)
test_accuracy_val = accuracy(X_train, X_test, y_train, y_test, model)
accuracy_val.append(test_accuracy_val)
```

```
Train accuracy: 60.04%
Test accuracy: 46.89%
```

Results: With KNeighborsClassifier, the accuracy score were as below:
Training Accuracy – 60.04%
Testing Accuracy – 46.89%

Decision Tree Classifier

Decision tree classifier is a systematic approach for multiclass classification. It poses a set of questions to the dataset (related to its attributes/features). The decision tree classification algorithm can be visualized on a binary tree. On the root and each of the internal nodes, a question is posed and the data on that node is further split into separate records that have different characteristics. The leaves of the tree refer to the classes in which the dataset is split.

In [45]:
```
# Using Decision Tree Classifier
model = train_model(X_train, y_train, DecisionTreeClassifier, max_depth=8)
test_accuracy_val = accuracy(X_train, X_test, y_train, y_test, model)
accuracy_val.append(test_accuracy_val)
```

```
Train accuracy: 65.47%
Test accuracy: 59.87%
```

Results: With DecisionTreeClassifier, the accuracy score were as below:
Training Accuracy – 65.47%
Testing Accuracy – 59.87%

Conclusion

The purpose of this project was correlate and compare the above mentioned ML algorithms in order to check their performances.

In [46]:
```
# Create a dataframe from accuracy results
summary = pd.DataFrame({'Test Accuracy':accuracy_val}, index=classifiers)
summary
```

Out[46]:

	Test Accuracy
Random Forest	0.559039
Logistic Regression	0.466813
Knn (7 Neighbors)	0.468922
Decision Tree	0.598735

For this particular problem, the algorithm with best accuracy value is DecisionTreeClassifier with test accuracy score of 59.87% and therefore it can be considered as a good classifier algorithm for house price range prediction problem. Also, the RandomForestClassifier is close enough with 55.90% accuracy score. I have tried tuning each algorithm with different hyper-parameter values and finally kept the best results for each. In this project we can say that in machine learning problems data processing and tuning makes the model more accurate and efficient compare to non processed data. It also makes simple models quite accurate.

In [ ]:
```
```