Advanced Housing Prices-Feature Engineering

We will be doing the following steps in Feature Engineering:

1. Missing Values
2. Temporal variables
3. Categorical variables: remove rare labels
4. Standarise the values of the variables to the same range

In [55]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
# to visualise al the columns in the dataframe
pd.pandas.set_option('display.max_columns', None)
```

In [56]:
```
dataset=pd.read_csv('train.csv')
dataset.head()
```

Out[56]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	Inside
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	FR2
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	Inside
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	Corner
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	FR2

Data fields

Here's a brief version of what you'll find in the data description file.

- SalePrice - the property's sale price in dollars. This is the target variable that you're trying to predict.
- MSSubClass: The building class
- MSZoning: The general zoning classification
- LotFrontage: Linear feet of street connected to property
- LotArea: Lot size in square feet
- Street: Type of road access
- Alley: Type of alley access
- LotShape: General shape of property
- LandContour: Flatness of the property
- Utilities: Type of utilities available
- LotConfig: Lot configuration
- LandSlope: Slope of property
- Neighborhood: Physical locations within Ames city limits
- Condition1: Proximity to main road or railroad
- Condition2: Proximity to main road or railroad (if a second is present)
- BldgType: Type of dwelling

- HouseStyle: Style of dwelling
- OverallQual: Overall material and finish quality
- OverallCond: Overall condition rating
- YearBuilt: Original construction date
- YearRemodAdd: Remodel date
- RoofStyle: Type of roof
- RoofMatl: Roof material
- Exterior1st: Exterior covering on house
- Exterior2nd: Exterior covering on house (if more than one material)
- MasVnrType: Masonry veneer type
- MasVnrArea: Masonry veneer area in square feet
- ExterQual: Exterior material quality
- ExterCond: Present condition of the material on the exterior
- Foundation: Type of foundation
- BsmtQual: Height of the basement
- BsmtCond: General condition of the basement
- BsmtExposure: Walkout or garden level basement walls
- BsmtFinType1: Quality of basement finished area
- BsmtFinSF1: Type 1 finished square feet
- BsmtFinType2: Quality of second finished area (if present)
- BsmtFinSF2: Type 2 finished square feet
- BsmtUnfSF: Unfinished square feet of basement area
- TotalBsmtSF: Total square feet of basement area
- Heating: Type of heating
- HeatingQC: Heating quality and condition
- CentralAir: Central air conditioning
- Electrical: Electrical system
- 1stFlrSF: First Floor square feet
- 2ndFlrSF: Second floor square feet
- LowQualFinSF: Low quality finished square feet (all floors)
- GrLivArea: Above grade (ground) living area square feet
- BsmtFullBath: Basement full bathrooms
- BsmtHalfBath: Basement half bathrooms
- FullBath: Full bathrooms above grade
- HalfBath: Half baths above grade
- Bedroom: Number of bedrooms above basement level
- Kitchen: Number of kitchens
- KitchenQual: Kitchen quality
- TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)
- Functional: Home functionality rating
- Fireplaces: Number of fireplaces
- FireplaceQu: Fireplace quality
- GarageType: Garage location
- GarageYrBlt: Year garage was built
- GarageFinish: Interior finish of the garage
- GarageCars: Size of garage in car capacity
- GarageArea: Size of garage in square feet
- GarageQual: Garage quality

- GarageCond: Garage condition
- PavedDrive: Paved driveway
- WoodDeckSF: Wood deck area in square feet
- OpenPorchSF: Open porch area in square feet
- EnclosedPorch: Enclosed porch area in square feet
- 3SsnPorch: Three season porch area in square feet
- ScreenPorch: Screen porch area in square feet
- PoolArea: Pool area in square feet
- PoolQC: Pool quality
- Fence: Fence quality
- MiscFeature: Miscellaneous feature not covered in other categories
- MiscVal: $Value of miscellaneous feature
- MoSold: Month Sold
- YrSold: Year Sold
- SaleType: Type of sale
- SaleCondition: Condition of sale

In [57]:
```
'''Always remember there way always be a chance of data leakage so we need to split the
Engineering'''

from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(dataset,dataset['SalePrice'],test_size=0.
```

In [58]:
```
X_train.shape, X_test.shape
```

Out[58]:
```
((1314, 81), (146, 81))
```

Missing Values

In [9]:
```
# Capturing all the NaN values
# First handling categorical featues which are missing

features_nan=[feature for feature in dataset.columns if dataset[feature].isnull().sum()>

for feature in features_nan:
    print("{}: {}%missing values".format(feature,np.round(dataset[feature].isnull().mean
```

```
Alley: 0.9377%missing values
MasVnrType: 0.0055%missing values
BsmtQual: 0.0253%missing values
BsmtCond: 0.0253%missing values
BsmtExposure: 0.026%missing values
BsmtFinType1: 0.0253%missing values
BsmtFinType2: 0.026%missing values
FireplaceQu: 0.4726%missing values
GarageType: 0.0555%missing values
GarageFinish: 0.0555%missing values
GarageQual: 0.0555%missing values
GarageCond: 0.0555%missing values
PoolQC: 0.9952%missing values
Fence: 0.8075%missing values
MiscFeature: 0.963%missing values
```

In [59]:
```
#Replacing values with new label

def replace_cat_feature(dataset,features_nan):
    data=dataset.copy()
```

```
    data[features_nan]=data[features_nan].fillna('Missing')
    return data

dataset=replace_cat_feature(dataset,features_nan)

dataset[features_nan].isnull().sum()
```

Out[59]:

```
Alley           0
MasVnrType      0
BsmtQual        0
BsmtCond        0
BsmtExposure    0
BsmtFinType1    0
BsmtFinType2    0
FireplaceQu     0
GarageType      0
GarageFinish    0
GarageQual      0
GarageCond      0
PoolQC          0
Fence           0
MiscFeature     0
dtype: int64
```

In [60]:

```
dataset.head()
```

Out[60]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfi
0	1	60	RL	65.0	8450	Pave	Missing	Reg	Lvl	AllPub	Insid
1	2	20	RL	80.0	9600	Pave	Missing	Reg	Lvl	AllPub	FR
2	3	60	RL	68.0	11250	Pave	Missing	IR1	Lvl	AllPub	Insid
3	4	70	RL	60.0	9550	Pave	Missing	IR1	Lvl	AllPub	Corne
4	5	60	RL	84.0	14260	Pave	Missing	IR1	Lvl	AllPub	FR

In [61]:

```
#Checking numerical variables with missing values

numerical_with_nan=[feature for feature in dataset.columns if dataset[feature].isnull().

## We will print the numerical nan variables and percentage of missing values

for feature in numerical_with_nan:
    print("{}: {}% missing value".format(feature,np.around(dataset[feature].isnull().mea
```

```
LotFrontage: 0.1774% missing value
MasVnrArea: 0.0055% missing value
GarageYrBlt: 0.0555% missing value
```

In [62]:

```
#Replacing with numerical Nan(missing) values

for feature in numerical_with_nan:
    ## We will replace by using median since there are outliers
    median_value=dataset[feature].median()

    ## create a new feature to capture nan values
    dataset[feature+'nan']=np.where(dataset[feature].isnull(),1,0)
    dataset[feature].fillna(median_value,inplace=True)

dataset[numerical_with_nan].isnull().sum()
```

Out[62]:

```
LotFrontage     0
MasVnrArea      0
```

```
GarageYrBlt     0
dtype: int64
```

In [63]:
```
dataset.head(50)
```

Out[63]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotCon
0	1	60	RL	65.0	8450	Pave	Missing	Reg	Lvl	AllPub	Insi
1	2	20	RL	80.0	9600	Pave	Missing	Reg	Lvl	AllPub	F
2	3	60	RL	68.0	11250	Pave	Missing	IR1	Lvl	AllPub	Insi
3	4	70	RL	60.0	9550	Pave	Missing	IR1	Lvl	AllPub	Corr
4	5	60	RL	84.0	14260	Pave	Missing	IR1	Lvl	AllPub	F
5	6	50	RL	85.0	14115	Pave	Missing	IR1	Lvl	AllPub	Insi
6	7	20	RL	75.0	10084	Pave	Missing	Reg	Lvl	AllPub	Insi
7	8	60	RL	69.0	10382	Pave	Missing	IR1	Lvl	AllPub	Corr
8	9	50	RM	51.0	6120	Pave	Missing	Reg	Lvl	AllPub	Insi
9	10	190	RL	50.0	7420	Pave	Missing	Reg	Lvl	AllPub	Corr
10	11	20	RL	70.0	11200	Pave	Missing	Reg	Lvl	AllPub	Insi
11	12	60	RL	85.0	11924	Pave	Missing	IR1	Lvl	AllPub	Insi
12	13	20	RL	69.0	12968	Pave	Missing	IR2	Lvl	AllPub	Insi
13	14	20	RL	91.0	10652	Pave	Missing	IR1	Lvl	AllPub	Insi
14	15	20	RL	69.0	10920	Pave	Missing	IR1	Lvl	AllPub	Corr
15	16	45	RM	51.0	6120	Pave	Missing	Reg	Lvl	AllPub	Corr
16	17	20	RL	69.0	11241	Pave	Missing	IR1	Lvl	AllPub	CulDS
17	18	90	RL	72.0	10791	Pave	Missing	Reg	Lvl	AllPub	Insi
18	19	20	RL	66.0	13695	Pave	Missing	Reg	Lvl	AllPub	Insi
19	20	20	RL	70.0	7560	Pave	Missing	Reg	Lvl	AllPub	Insi
20	21	60	RL	101.0	14215	Pave	Missing	IR1	Lvl	AllPub	Corr
21	22	45	RM	57.0	7449	Pave	Grvl	Reg	Bnk	AllPub	Insi
22	23	20	RL	75.0	9742	Pave	Missing	Reg	Lvl	AllPub	Insi
23	24	120	RM	44.0	4224	Pave	Missing	Reg	Lvl	AllPub	Insi
24	25	20	RL	69.0	8246	Pave	Missing	IR1	Lvl	AllPub	Insi
25	26	20	RL	110.0	14230	Pave	Missing	Reg	Lvl	AllPub	Corr
26	27	20	RL	60.0	7200	Pave	Missing	Reg	Lvl	AllPub	Corr
27	28	20	RL	98.0	11478	Pave	Missing	Reg	Lvl	AllPub	Insi
28	29	20	RL	47.0	16321	Pave	Missing	IR1	Lvl	AllPub	CulDS
29	30	30	RM	60.0	6324	Pave	Missing	IR1	Lvl	AllPub	Insi
30	31	70	C (all)	50.0	8500	Pave	Pave	Reg	Lvl	AllPub	Insi
31	32	20	RL	69.0	8544	Pave	Missing	IR1	Lvl	AllPub	CulDS
32	33	20	RL	85.0	11049	Pave	Missing	Reg	Lvl	AllPub	Corr
33	34	20	RL	70.0	10552	Pave	Missing	IR1	Lvl	AllPub	Insi

34	35	120	RL	60.0	7313	Pave	Missing	Reg	Lvl	AllPub	Insi
35	36	60	RL	108.0	13418	Pave	Missing	Reg	Lvl	AllPub	Insi
36	37	20	RL	112.0	10859	Pave	Missing	Reg	Lvl	AllPub	Corr
37	38	20	RL	74.0	8532	Pave	Missing	Reg	Lvl	AllPub	Insi
38	39	20	RL	68.0	7922	Pave	Missing	Reg	Lvl	AllPub	Insi
39	40	90	RL	65.0	6040	Pave	Missing	Reg	Lvl	AllPub	Insi
40	41	20	RL	84.0	8658	Pave	Missing	Reg	Lvl	AllPub	Insi
41	42	20	RL	115.0	16905	Pave	Missing	Reg	Lvl	AllPub	Insi
42	43	85	RL	69.0	9180	Pave	Missing	IR1	Lvl	AllPub	CulDS
43	44	20	RL	69.0	9200	Pave	Missing	IR1	Lvl	AllPub	CulDS
44	45	20	RL	70.0	7945	Pave	Missing	Reg	Lvl	AllPub	Insi
45	46	120	RL	61.0	7658	Pave	Missing	Reg	Lvl	AllPub	Insi
46	47	50	RL	48.0	12822	Pave	Missing	IR1	Lvl	AllPub	CulDS
47	48	20	FV	84.0	11096	Pave	Missing	Reg	Lvl	AllPub	Insi
48	49	190	RM	33.0	4456	Pave	Missing	Reg	Lvl	AllPub	Insi
49	50	20	RL	66.0	7742	Pave	Missing	Reg	Lvl	AllPub	Insi

In [65]:

```
# TEMPORAL VARIABLES(DATE & TIME VARIABLES)
#We are just making it features with how many years ago. So feature - Year Sold

for feature in ['YearBuilt','YearRemodAdd','GarageYrBlt']:

    dataset[feature]=dataset['YrSold']-dataset[feature]
```

In [66]:

```
dataset.head()
```

Out[66]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfi
0	1	60	RL	65.0	8450	Pave	Missing	Reg	Lvl	AllPub	Insid
1	2	20	RL	80.0	9600	Pave	Missing	Reg	Lvl	AllPub	FR
2	3	60	RL	68.0	11250	Pave	Missing	IR1	Lvl	AllPub	Insid
3	4	70	RL	60.0	9550	Pave	Missing	IR1	Lvl	AllPub	Corne
4	5	60	RL	84.0	14260	Pave	Missing	IR1	Lvl	AllPub	FR

In [67]:

```
dataset[['YearBuilt','YearRemodAdd','GarageYrBlt']].head()
```

Out[67]:

	YearBuilt	YearRemodAdd	GarageYrBlt
0	5	5	5.0
1	31	31	31.0
2	7	6	7.0
3	91	36	8.0
4	8	8	8.0

Numerical Values

Since the numerical variables are skewed data we will perform log normal distribution

In [68]:
```
dataset.head()
```

Out[68]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfi
0	1	60	RL	65.0	8450	Pave	Missing	Reg	Lvl	AllPub	Insid
1	2	20	RL	80.0	9600	Pave	Missing	Reg	Lvl	AllPub	FR
2	3	60	RL	68.0	11250	Pave	Missing	IR1	Lvl	AllPub	Insid
3	4	70	RL	60.0	9550	Pave	Missing	IR1	Lvl	AllPub	Corne
4	5	60	RL	84.0	14260	Pave	Missing	IR1	Lvl	AllPub	FR

In [69]:
```
import numpy as np
num_features=['LotFrontage', 'LotArea', '1stFlrSF', 'GrLivArea', 'SalePrice']

for feature in num_features:
    dataset[feature]=np.log(dataset[feature])
```

In [70]:
```
dataset.head()
```

Out[70]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfi
0	1	60	RL	4.174387	9.041922	Pave	Missing	Reg	Lvl	AllPub	Insic
1	2	20	RL	4.382027	9.169518	Pave	Missing	Reg	Lvl	AllPub	FF
2	3	60	RL	4.219508	9.328123	Pave	Missing	IR1	Lvl	AllPub	Insic
3	4	70	RL	4.094345	9.164296	Pave	Missing	IR1	Lvl	AllPub	Corn
4	5	60	RL	4.430817	9.565214	Pave	Missing	IR1	Lvl	AllPub	FF

Rare Categorical Feature

We will remove the categorical features which are present than 1% of the database

In [71]:
```
categorical_features=[feature for feature in dataset.columns if dataset[feature].dtype==
```

In [72]:
```
categorical_features
```

Out[72]:
```
['MSZoning',
 'Street',
 'Alley',
 'LotShape',
 'LandContour',
 'Utilities',
 'LotConfig',
 'LandSlope',
 'Neighborhood',
 'Condition1',
 'Condition2',
 'BldgType',
 'HouseStyle',
 'RoofStyle',
 'RoofMatl',
```

```
 'Exterior1st',
 'Exterior2nd',
 'MasVnrType',
 'ExterQual',
 'ExterCond',
 'Foundation',
 'BsmtQual',
 'BsmtCond',
 'BsmtExposure',
 'BsmtFinType1',
 'BsmtFinType2',
 'Heating',
 'HeatingQC',
 'CentralAir',
 'Electrical',
 'KitchenQual',
 'Functional',
 'FireplaceQu',
 'GarageType',
 'GarageFinish',
 'GarageQual',
 'GarageCond',
 'PavedDrive',
 'PoolQC',
 'Fence',
 'MiscFeature',
 'SaleType',
 'SaleCondition']
```

In [73]:

```
for feature in categorical_features:
    temp=dataset.groupby(feature)['SalePrice'].count()/len(dataset)
    temp_df=temp[temp>0.01].index
    dataset[feature]=np.where(dataset[feature].isin(temp_df),dataset[feature],'Rare_var'
```

In [74]:

```
dataset.head(100)
```

Out[74]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotCo
0	1	60	RL	4.174387	9.041922	Pave	Missing	Reg	Lvl	AllPub	Ir
1	2	20	RL	4.382027	9.169518	Pave	Missing	Reg	Lvl	AllPub	
2	3	60	RL	4.219508	9.328123	Pave	Missing	IR1	Lvl	AllPub	Ir
3	4	70	RL	4.094345	9.164296	Pave	Missing	IR1	Lvl	AllPub	Cc
4	5	60	RL	4.430817	9.565214	Pave	Missing	IR1	Lvl	AllPub	
...	...	...	...	...	...	...	...	...	...	...	
95	96	60	RL	4.234107	9.186560	Pave	Missing	IR2	Lvl	AllPub	Cc
96	97	20	RL	4.356709	9.236398	Pave	Missing	IR1	Lvl	AllPub	Ir
97	98	20	RL	4.290459	9.298443	Pave	Missing	Reg	HLS	AllPub	Ir
98	99	30	RL	4.442651	9.270965	Pave	Missing	Reg	Lvl	AllPub	Cc
99	100	20	RL	4.343805	9.139918	Pave	Missing	IR1	Lvl	AllPub	Ir

100 rows × 84 columns

In [75]:

```
for feature in categorical_features:
    labels_ordered=dataset.groupby([feature])['SalePrice'].mean().sort_values().index
```

```
    labels_ordered={k:i for i,k in enumerate(labels_ordered,0)}
    dataset[feature]=dataset[feature].map(labels_ordered)
```

In [76]:

```
dataset.head(10)
```

Out[76]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig
0	1	60	3	4.174387	9.041922	1	2	0	1	1	0
1	2	20	3	4.382027	9.169518	1	2	0	1	1	2
2	3	60	3	4.219508	9.328123	1	2	1	1	1	0
3	4	70	3	4.094345	9.164296	1	2	1	1	1	1
4	5	60	3	4.430817	9.565214	1	2	1	1	1	2
5	6	50	3	4.442651	9.554993	1	2	1	1	1	0
6	7	20	3	4.317488	9.218705	1	2	0	1	1	0
7	8	60	3	4.234107	9.247829	1	2	1	1	1	1
8	9	50	1	3.931826	8.719317	1	2	0	1	1	0
9	10	190	3	3.912023	8.911934	1	2	0	1	1	1

In [ ]:

In [77]:

```
scaling_feature=[feature for feature in dataset.columns if feature not in ['Id','SalePer
len(scaling_feature)
```

Out[77]:

```
83
```

In [78]:

```
scaling_feature
```

Out[78]:

```
['MSSubClass',
 'MSZoning',
 'LotFrontage',
 'LotArea',
 'Street',
 'Alley',
 'LotShape',
 'LandContour',
 'Utilities',
 'LotConfig',
 'LandSlope',
 'Neighborhood',
 'Condition1',
 'Condition2',
 'BldgType',
 'HouseStyle',
 'OverallQual',
 'OverallCond',
 'YearBuilt',
 'YearRemodAdd',
 'RoofStyle',
 'RoofMatl',
 'Exterior1st',
 'Exterior2nd',
 'MasVnrType',
 'MasVnrArea',
 'ExterQual',
 'ExterCond',
 'Foundation',
 'BsmtQual',
```

```
 'BsmtCond',
 'BsmtExposure',
 'BsmtFinType1',
 'BsmtFinSF1',
 'BsmtFinType2',
 'BsmtFinSF2',
 'BsmtUnfSF',
 'TotalBsmtSF',
 'Heating',
 'HeatingQC',
 'CentralAir',
 'Electrical',
 '1stFlrSF',
 '2ndFlrSF',
 'LowQualFinSF',
 'GrLivArea',
 'BsmtFullBath',
 'BsmtHalfBath',
 'FullBath',
 'HalfBath',
 'BedroomAbvGr',
 'KitchenAbvGr',
 'KitchenQual',
 'TotRmsAbvGrd',
 'Functional',
 'Fireplaces',
 'FireplaceQu',
 'GarageType',
 'GarageYrBlt',
 'GarageFinish',
 'GarageCars',
 'GarageArea',
 'GarageQual',
 'GarageCond',
 'PavedDrive',
 'WoodDeckSF',
 'OpenPorchSF',
 'EnclosedPorch',
 '3SsnPorch',
 'ScreenPorch',
 'PoolArea',
 'PoolQC',
 'Fence',
 'MiscFeature',
 'MiscVal',
 'MoSold',
 'YrSold',
 'SaleType',
 'SaleCondition',
 'SalePrice',
 'LotFrontagenan',
 'MasVnrAreanan',
 'GarageYrBltnan']
```

In [79]:

```
dataset.head()
```

Out[79]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig
0	1	60	3	4.174387	9.041922	1	2	0	1	1	0
1	2	20	3	4.382027	9.169518	1	2	0	1	1	2
2	3	60	3	4.219508	9.328123	1	2	1	1	1	0
3	4	70	3	4.094345	9.164296	1	2	1	1	1	1
4	5	60	3	4.430817	9.565214	1	2	1	1	1	2

Feature Scaling

In [80]:
```
feature_scale=[feature for feature in dataset.columns if feature not in ['Id','SalePrice

from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler.fit(dataset[feature_scale])
```

Out[80]:
```
MinMaxScaler()
```

In [81]:
```
scaler.transform(dataset[feature_scale])
```

Out[81]:
```
array([[0.23529412, 0.75      , 0.41820812, ..., 0.        , 0.        ,
        0.        ],
       [0.        , 0.75      , 0.49506375, ..., 0.        , 0.        ,
        0.        ],
       [0.23529412, 0.75      , 0.434909  , ..., 0.        , 0.        ,
        0.        ],
       ...,
       [0.29411765, 0.75      , 0.42385922, ..., 0.        , 0.        ,
        0.        ],
       [0.        , 0.75      , 0.434909  , ..., 0.        , 0.        ,
        0.        ],
       [0.        , 0.75      , 0.47117546, ..., 0.        , 0.        ,
        0.        ]])
```

In [82]:
```
# transform the train and test set, and add on the Id and SalePrice variables
data = pd.concat([dataset[['Id', 'SalePrice']].reset_index(drop=True),
                    pd.DataFrame(scaler.transform(dataset[feature_scale]), columns=featu
                    axis=1)
```

In [84]:
```
data.head()
```

Out[84]:

	Id	SalePrice	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities
0	1	12.247694	0.235294	0.75	0.418208	0.366344	1.0	1.0	0.000000	0.333333	1.0
1	2	12.109011	0.000000	0.75	0.495064	0.391317	1.0	1.0	0.000000	0.333333	1.0
2	3	12.317167	0.235294	0.75	0.434909	0.422359	1.0	1.0	0.333333	0.333333	1.0
3	4	11.849398	0.294118	0.75	0.388581	0.390295	1.0	1.0	0.333333	0.333333	1.0
4	5	12.429216	0.235294	0.75	0.513123	0.468761	1.0	1.0	0.333333	0.333333	1.0

In [86]:
```
data.to_csv('X_train.csv',index=False)
```

In [ ]: