Proyecto 1 IA

November 6, 2020

1 Lectura de la base de datos

[1]:
```
import pandas as pd
```

Se modificaron las variables: BsmtQual, BsmtCond, BsmtExposure, BsmtFinType1, BsmtFinType2, FireplaceQu, GarageTyp, GarageFinish, GarageQual, GarageCond, PoolQC, Fence, MiscFeature. Estas variables tenían una categoría llamada “NA”, sin embargo, Python al momento de leer esa cadena la interpreta como dato faltante, por lo tanto se reemplazó el nombre de esa categoría por “NoA”

[2]:
```
train = pd.read_csv("train.csv", index_col="Id")
test = pd.read_csv("test.csv", index_col="Id")
sample_submission = pd.read_csv("sample_submission.csv", index_col="Id")
```

En cuanto a las variables numéricas, si no se registróa valor no pasa nada si se reemplaza con 0 pues las variables numéricas representan longitudes no afecta.

[3]:
```
train.fillna(0,inplace=True)
test.fillna(0,inplace=True)
```

[4]:
```
train.head()
```

[4]:
```
    MSSubClass MSZoning  LotFrontage  LotArea Street Alley LotShape  \
Id
1           60       RL         65.0     8450   Pave   NoA      Reg
2           20       RL         80.0     9600   Pave   NoA      Reg
3           60       RL         68.0    11250   Pave   NoA      IR1
4           70       RL         60.0     9550   Pave   NoA      IR1
5           60       RL         84.0    14260   Pave   NoA      IR1

   LandContour Utilities LotConfig  ... PoolArea PoolQC Fence MiscFeature  \
Id                                  ...
1          Lvl    AllPub    Inside  ...        0    NoA   NoA         NoA
2          Lvl    AllPub       FR2  ...        0    NoA   NoA         NoA
3          Lvl    AllPub    Inside  ...        0    NoA   NoA         NoA
4          Lvl    AllPub    Corner  ...        0    NoA   NoA         NoA
5          Lvl    AllPub       FR2  ...        0    NoA   NoA         NoA
```

```
   MiscVal MoSold  YrSold  SaleType  SaleCondition  SalePrice
Id
1        0      2    2008        WD         Normal     208500
2        0      5    2007        WD         Normal     181500
3        0      9    2008        WD         Normal     223500
4        0      2    2006        WD        Abnorml     140000
5        0     12    2008        WD         Normal     250000

[5 rows x 80 columns]
```

2 Revisión de la lectura de los datos

Es importante revisar que el tipo de dato de cada variable sea el adecuado y si no, definirlo de forma correcta. También debe verificarse que las celdas no contengan NaN

[5]:
```
print(train.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1460 entries, 1 to 1460
Data columns (total 80 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   MSSubClass     1460 non-null   int64
 1   MSZoning       1460 non-null   object
 2   LotFrontage    1460 non-null   float64
 3   LotArea        1460 non-null   int64
 4   Street         1460 non-null   object
 5   Alley          1460 non-null   object
 6   LotShape       1460 non-null   object
 7   LandContour    1460 non-null   object
 8   Utilities      1460 non-null   object
 9   LotConfig      1460 non-null   object
 10  LandSlope      1460 non-null   object
 11  Neighborhood   1460 non-null   object
 12  Condition1     1460 non-null   object
 13  Condition2     1460 non-null   object
 14  BldgType       1460 non-null   object
 15  HouseStyle     1460 non-null   object
 16  OverallQual    1460 non-null   int64
 17  OverallCond    1460 non-null   int64
 18  YearBuilt      1460 non-null   int64
 19  YearRemodAdd   1460 non-null   int64
 20  RoofStyle      1460 non-null   object
 21  RoofMatl       1460 non-null   object
 22  Exterior1st    1460 non-null   object
 23  Exterior2nd    1460 non-null   object
```

```
24  MasVnrType     1460 non-null   object
25  MasVnrArea     1460 non-null   float64
26  ExterQual      1460 non-null   object
27  ExterCond      1460 non-null   object
28  Foundation     1460 non-null   object
29  BsmtQual       1460 non-null   object
30  BsmtCond       1460 non-null   object
31  BsmtExposure   1460 non-null   object
32  BsmtFinType1   1460 non-null   object
33  BsmtFinSF1     1460 non-null   int64
34  BsmtFinType2   1460 non-null   object
35  BsmtFinSF2     1460 non-null   int64
36  BsmtUnfSF      1460 non-null   int64
37  TotalBsmtSF    1460 non-null   int64
38  Heating        1460 non-null   object
39  HeatingQC      1460 non-null   object
40  CentralAir     1460 non-null   object
41  Electrical     1460 non-null   object
42  1stFlrSF       1460 non-null   int64
43  2ndFlrSF       1460 non-null   int64
44  LowQualFinSF   1460 non-null   int64
45  GrLivArea      1460 non-null   int64
46  BsmtFullBath   1460 non-null   int64
47  BsmtHalfBath   1460 non-null   int64
48  FullBath       1460 non-null   int64
49  HalfBath       1460 non-null   int64
50  BedroomAbvGr   1460 non-null   int64
51  KitchenAbvGr   1460 non-null   int64
52  KitchenQual    1460 non-null   object
53  TotRmsAbvGrd   1460 non-null   int64
54  Functional     1460 non-null   object
55  Fireplaces     1460 non-null   int64
56  FireplaceQu    1460 non-null   object
57  GarageType     1460 non-null   object
58  GarageYrBlt    1460 non-null   float64
59  GarageFinish   1460 non-null   object
60  GarageCars     1460 non-null   int64
61  GarageArea     1460 non-null   int64
62  GarageQual     1460 non-null   object
63  GarageCond     1460 non-null   object
64  PavedDrive     1460 non-null   object
65  WoodDeckSF     1460 non-null   int64
66  OpenPorchSF    1460 non-null   int64
67  EnclosedPorch  1460 non-null   int64
68  3SsnPorch      1460 non-null   int64
69  ScreenPorch    1460 non-null   int64
70  PoolArea       1460 non-null   int64
71  PoolQC         1460 non-null   object
```

```
 72   Fence          1460 non-null   object
 73   MiscFeature    1460 non-null   object
 74   MiscVal        1460 non-null   int64
 75   MoSold         1460 non-null   int64
 76   YrSold         1460 non-null   int64
 77   SaleType       1460 non-null   object
 78   SaleCondition  1460 non-null   object
 79   SalePrice      1460 non-null   int64
dtypes: float64(3), int64(34), object(43)
memory usage: 923.9+ KB
None
```

2.1 Var 00 MSSubClass

Por definición de los datos la variable MSSubClass aunque toma valores enteros realmente es una variable categórica, pues indica el tipo de vivienda involucrada en la venta, por lo tanto, debe trabajarse como variable catégorica y no numérica.

[6]:
```
train['MSSubClass'] = train['MSSubClass'].astype('category')
print(train['MSSubClass'].dtype)
print(train['MSSubClass'].isnull().sum())
# este cambio tambien lo hacemos en test
test['MSSubClass'] = test['MSSubClass'].astype('category')
```

```
category
0
```

2.2 Var 01 MSZoning

Por definición esta variable es categórica, así que **está bien definida**.

[7]:
```
print(train['MSZoning'].dtype)
print(train['MSZoning'].isnull().sum())
```

```
object
0
```

2.3 Var 02 LotFrontage

Esta variable representa una medida en pies de la calle conectada a la propiedad, por lo tanto, como flotante **está bien definida**.

[8]:
```
print(train['LotFrontage'].dtype)
print(train['LotFrontage'].isnull().sum())
```

```
float64
0
```

2.4 Var 03 LotArea

Representa la medida del lote en pies cuadrados, si se identificó como entero significa que ningún registro tiene una medida decimal, lo que es raro pero ya ni modo, **el tipo de dato está bien definido**

[9]:
```
print(train['LotArea'].dtype)
print(train['LotArea'].isnull().sum())
```

```
int64
0
```

2.5 Var 04 Street

Igualmente **está bien definida**, representa si calle está pavimentada o es grava.

[10]:
```
print(train['Street'].dtype)
print(train['Street'].isnull().sum())
```

```
object
0
```

2.6 Var 05 Alley

Igualmente **está bien definida**, representa si el acceso a la calle está pavimentada o es grava.

[11]:
```
print(train['Alley'].dtype)
print(train['Alley'].isnull().sum())
```

```
object
0
```

2.7 Var 06 LotShape

Representa la forma del lote, **está bien definida**.

[12]:
```
print(train['LotShape'].dtype)
print(train['LotShape'].isnull().sum())
```

```
object
0
```

2.8 Var 07 LandContour

Representa la inflación de la propiedad, **está bien definida**.

[13]:
```
print(train['LandContour'].dtype)
print(train['LandContour'].isnull().sum())
```

```
object
0
```

2.9 Var 08 Utilities

Variable categórica que indica los serivios que tiene la propiedad (agua, luz, gas, etc). **Está bien definida**.

[14]:
```
print(train['Utilities'].dtype)
print(train['Utilities'].isnull().sum())
```

```
object
0
```

2.10 Var 09 LotConfig

Representa si la propiedad está en una cerrada, sobre la avenida, en esquina o es un predio dentro de otro, etc. **Está bien definida**.

[15]:
```
print(train['LotConfig'].dtype)
print(train['LotConfig'].isnull().sum())
```

```
object
0
```

2.11 Var 10 LandSlope

Indica si la propiedad está sobre terreno plano o si tiene cierta inclinación. **Está bien definido**.

[16]:
```
print(train['LandSlope'].dtype)
print(train['LandSlope'].isnull().sum())
```

```
object
0
```

2.12 Var 11 Neighborhood

Localización física dentro de los límites de Ames City. **Está bien definida**.

[17]:
```
print(train['Neighborhood'].dtype)
print(train['Neighborhood'].isnull().sum())
```

```
object
0
```

2.13 Var 12 Condition1

Proximidad a varias condiciones (cerca a una avenida principal, etc.) **Está bien definida**.

[18]:
```
print(train['Condition1'].dtype)
print(train['Condition1'].isnull().sum())
```

```
object
0
```

2.14 Var 13 Condition2

Igualmente a la anterior es categórica, **está bien definida**.

[19]:
```
print(train['Condition2'].dtype)
print(train['Condition2'].isnull().sum())
```

```
object
0
```

2.15 Var 14 BldgType

Tipo de vivienda (1 familia, originalmente contruida para 1 familia y adaptada para 2, duplex, etc.). **Está bien definida**.

[20]:
```
print(train['BldgType'].dtype)
print(train['BldgType'].isnull().sum())
```

```
object
0
```

2.16 Var 15 HouseStyle

Estilo de vivienda.**Está bien definida**.

[21]:
```
print(train['HouseStyle'].dtype)
print(train['HouseStyle'].isnull().sum())
```

```
object
0
```

2.17 Var 16 OverallQual

Esta variable representa una calificación en la calidad de los materiales y acabados de la clase, aunque es categórica es una variable ordinal, entonces como entero está bien definida pues la calificación, el valor, sí proporcionan información, el 10 es Muy Excelente y el 1 en Muy Pobre. **está bien**.

[22]:
```
print(train['OverallQual'].dtype)
print(train['OverallQual'].isnull().sum())
```

```
int64
0
```

2.18 Var 17 OverallCond

Esta calificación representa una calificación en la condición de la casa, al igual que la anterior, **está bien definida**.

[23]:
```
print(train['OverallCond'].dtype)
print(train['OverallCond'].isnull().sum())
```

```
int64
0
```

2.19 Var 18 YearBuilt

Año de construcción, **está bien definida**.

[24]:
```
print(train['YearBuilt'].dtype)
print(train['YearBuilt'].isnull().sum())
```

```
int64
0
```

2.20 Var 19 YearRemodAdd

Añe de remodelación, mismo año que construcción sino ha sido remodelada, **está bien definida**.

[25]:
```
print(train['YearRemodAdd'].dtype)
print(train['YearRemodAdd'].isnull().sum())
```

```
int64
0
```

2.21 Var 20 RoofStyle

Tipo de techo, variable categ´rotica, **está bien definida**.

[26]:
```
print(train['RoofStyle'].dtype)
print(train['RoofStyle'].isnull().sum())
```

```
object
0
```

2.22 Var 21 RoofMatl

Material del techo, **está bien definida**.

[27]:
```
print(train['RoofMatl'].dtype)
print(train['RoofMatl'].isnull().sum())
```

```
object
0
```

2.23 Var 22 Exterior1st

Cuvierta del exterior de la casa, variable categórica, **está bien definida**.

[28]:
```
print(train['Exterior1st'].dtype)
print(train['Exterior1st'].isnull().sum())
```

```
object
0
```

2.24 Var 23 Exterior2nd

Si es que tiene otro material la fachada de la casa, variable categórica, **está bien definida**.

[29]:
```
print(train['Exterior2nd'].dtype)
print(train['Exterior2nd'].isnull().sum())
```

```
object
0
```

2.25 Var 24 MasVnrType

Tipo de revestimiento de mamposería, **está bien definida**.

[30]:
```
print(train['MasVnrType'].dtype)
print(train['MasVnrType'].isnull().sum())
```

```
object
0
```

2.26 Var 25 MasVnrArea

Área, en pies cuadrados, de recubriminetos de mampostería. **Está bien definida**.

[31]:
```
print(train['MasVnrArea'].dtype)
print(train['MasVnrArea'].isnull().sum())
```

```
float64
0
```

2.27 Var 26 ExterQual

Variable categorica de la calidad del material exterior. **está bien definida**.

[32]:
```
print(train['ExterQual'].dtype)
print(train['ExterQual'].isnull().sum())
```

```
object
0
```

2.28 Var 27 ExterCond

Evalúa la condición de lo materiales del exterior, variable categótica. **Está bien definida**.

[33]:
```
print(train['ExterCond'].dtype)
print(train['ExterCond'].isnull().sum())
```

```
object
0
```

2.29 Var 28 Foundation

Tipo de fundamento, **está bien definida**.

[34]:
```
print(train['Foundation'].dtype)
print(train['Foundation'].isnull().sum())
```

```
object
0
```

2.30 Var 29 BsmtQual

Evalúa el grosor de los fundamentos, variable categórica, **está bien definida**.

[35]:
```
print(train['BsmtQual'].dtype)
print(train['BsmtQual'].isnull().sum())
```

```
object
0
```

2.31 Var 30 BsmtCond

Condición general de los simientos, **está bien definida**.

[36]:
```
print(train['BsmtCond'].dtype)
print(train['BsmtCond'].isnull().sum())
```

```
object
0
```

2.32 Var 31 BsmtExposure

Se refiere a los miros de la entrada o jardín, **está bien definida**.

[37]:
```
print(train['BsmtExposure'].dtype)
print(train['BsmtExposure'].isnull().sum())
```

```
object
0
```

2.33 Var 32 BsmtFinType1

Calificación de los simientos terminados, **está bien definida**.

[38]:
```
print(train['BsmtFinType1'].dtype)
print(train['BsmtFinType1'].isnull().sum())
```

```
object
0
```

2.34 Var 33 BsmtFinSF1

Metros cuadrados terminado, **está bien definida**.

[39]:
```
print(train['BsmtFinSF1'].dtype)
print(train['BsmtFinSF1'].isnull().sum())
```

```
int64
0
```

2.35 Var 34 BsmtFinType2

rango de los simientos del área terminada, **está bien definida**.

[40]:
```
print(train['BsmtFinType2'].dtype)
print(train['BsmtFinType2'].isnull().sum())
```

```
object
0
```

2.36 Var 35 BsmtFinSF2

pies cuadrados terminados, **está bien definida**.

[41]:
```
print(train['BsmtFinSF2'].dtype)
print(train['BsmtFinSF2'].isnull().sum())
```

```
int64
0
```

2.37 Var 36 BsmtUnfSF

pues cuadrados in terminar de área de simientos, **está bien definida**.

[42]:
```
print(train['BsmtUnfSF'].dtype)
print(train['BsmtUnfSF'].isnull().sum())
```

```
int64
0
```

2.38 Var 37 TotalBsmtSF

pies cuadrados totales de área de simientos, **está bien definida**.

[43]:
```
print(train['TotalBsmtSF'].dtype)
print(train['TotalBsmtSF'].isnull().sum())
```

```
int64
0
```

2.39 Var 38 Heating

tipo de califección, **está bien definida**.

[44]:
```
print(train['Heating'].dtype)
print(train['Heating'].isnull().sum())
```

```
object
0
```

2.40 Var 39 HeatingQC

Calidad de la caleffación, **está bien definida**.

[45]:
```
print(train['HeatingQC'].dtype)
print(train['HeatingQC'].isnull().sum())
```

```
object
0
```

2.41 Var 40 CentralAir

si/no tiene aire acondicionado centra, **está bien definida**.

[46]:
```
print(train['CentralAir'].dtype)
print(train['CentralAir'].isnull().sum())
```

```
object
0
```

2.42 Var 41 Electrical

Tipo de sistema eléctrico, **está bien definida**.

[47]:
```
print(train['Electrical'].dtype)
print(train['Electrical'].isnull().sum())
```

```
object
0
```

2.43 Var 42 1stFlrSF

Pies cuadrados del primer piso, **está bien definida**.

[48]:
```
print(train['1stFlrSF'].dtype)
print(train['1stFlrSF'].isnull().sum())
```

```
int64
0
```

2.44 Var 43 2ndFlrSF

Pies cuadrados del segundo piso, **está bien definida**.

[49]:
```
print(train['2ndFlrSF'].dtype)
print(train['2ndFlrSF'].isnull().sum())
```

```
int64
0
```

2.45 Var 44 LowQualFinSF

pies cuadrados de baja calidad, **está bien definida**.

[50]:
```
print(train['LowQualFinSF'].dtype)
print(train['LowQualFinSF'].isnull().sum())
```

```
int64
0
```

2.46 Var 45 GrLivArea

pies cuadrados de superficie habitable, **está bien definida**.

[51]:
```
print(train['GrLivArea'].dtype)
print(train['GrLivArea'].isnull().sum())
```

```
int64
0
```

2.47 Var 46 BsmtFullBath

Baños completos en el sótano, **está bien definida**.

[52]:
```
print(train['BsmtFullBath'].dtype)
print(train['BsmtFullBath'].isnull().sum())
```

```
int64
0
```

2.48 Var 47 BsmtHalfBath

Medios baños en el sótano, **está bien definida**.

[53]:
```
print(train['BsmtHalfBath'].dtype)
print(train['BsmtHalfBath'].isnull().sum())
```

```
int64
0
```

2.49 Var 48 FullBath

Baños completos, **está bien definida**.

[54]:
```
print(train['FullBath'].dtype)
print(train['FullBath'].isnull().sum())
```

```
int64
0
```

2.50 Var 49 HalfBath

Medios baños, **está bien definida**.

[55]:
```
print(train['HalfBath'].dtype)
print(train['HalfBath'].isnull().sum())
```

```
int64
0
```

2.51 Var 50 BedroomAbvGr

Recamaras sin incluir las de sótano, **está bien definida**.

[56]:
```
print(train['BedroomAbvGr'].dtype)
print(train['BedroomAbvGr'].isnull().sum())
```

```
int64
0
```

2.52 Var 51 KitchenAbvGr

Numero de cocinas en la casa, **está bien definida**.

[57]:
```
print(train['KitchenAbvGr'].dtype)
print(train['KitchenAbvGr'].isnull().sum())
```

```
int64
0
```

2.53 Var 52 KitchenQual

Calidad de la cocina, **está bien definida**.

[58]:
```
print(train['KitchenQual'].dtype)
print(train['KitchenQual'].isnull().sum())
```

```
object
0
```

2.54 Var 53 TotRmsAbvGrd

Habitaciones totales sin incluir baños, **está bien definida**.

[59]:
```
print(train['TotRmsAbvGrd'].dtype)
print(train['TotRmsAbvGrd'].isnull().sum())
```

```
int64
0
```

2.55 Var 54 Functional

Funcionalidad de la casa, **está bien definida**.

[60]:
```
print(train['Functional'].dtype)
print(train['Functional'].isnull().sum())
```

```
object
0
```

2.56 Var 55 Fireplaces

Número de chimeneas, **está bien definida**.

[61]:
```
print(train['Fireplaces'].dtype)
print(train['Fireplaces'].isnull().sum())
```

```
int64
0
```

2.57 Var 56 FireplaceQu

Calidad de las chimeneas, **está bien definida**.

[62]:
```
print(train['FireplaceQu'].dtype)
print(train['FireplaceQu'].isnull().sum())
```

```
object
0
```

2.58 Var 57 GarageType

ubicación del gargae, **está bien definida**.

[63]:
```
print(train['GarageType'].dtype)
print(train['GarageType'].isnull().sum())
```

```
object
0
```

2.59 Var 58 GarageYrBlt

año en que el garage se construyó, **está bien definida**.

[64]:
```
print(train['GarageYrBlt'].dtype)
print(train['GarageYrBlt'].isnull().sum())
```

```
float64
0
```

2.60 Var 59 GarageFinish

estatus del garage, **está bien definida**.

[65]:
```
print(train['GarageFinish'].dtype)
print(train['GarageFinish'].isnull().sum())
```

```
object
0
```

2.61 Var 60 GarageCars

Capacidad de carros en el garage, **está bien definida**.

[66]:
```
print(train['GarageCars'].dtype)
print(train['GarageCars'].isnull().sum())
```

```
int64
0
```

2.62 Var 61 GarageArea

pies cuadrados del garage, **está bien definida**.

[67]:
```
print(train['GarageArea'].dtype)
print(train['GarageArea'].isnull().sum())
```

```
int64
0
```

2.63 Var 62 GarageQual

Calidad del garage, **está bien definida**.

[68]:
```
print(train['GarageQual'].dtype)
print(train['GarageQual'].isnull().sum())
```

```
object
0
```

2.64 Var 63 GarageCond

condición del garage, **está bien definida**.

[69]:
```
print(train['GarageCond'].dtype)
print(train['GarageCond'].isnull().sum())
```

```
object
0
```

2.65 Var 64 PavedDrive

Pavimentado, variable categórica, **estábien definida**.

[70]:
```
print(train['PavedDrive'].dtype)
print(train['PavedDrive'].isnull().sum())
```

```
object
0
```

2.66 Var 65 WoodDeckSF

pies cuadrados de área decorada con madera, **está bien definida**.

[71]:
```
print(train['WoodDeckSF'].dtype)
print(train['WoodDeckSF'].isnull().sum())
```

```
int64
0
```

2.67 Var 66 OpenPorchSF

pies cuadrados de porch abierto, **está bien definida**.

[72]:
```
print(train['OpenPorchSF'].dtype)
print(train['OpenPorchSF'].isnull().sum())
```

```
int64
0
```

2.68 Var 67 EnclosedPorch

pies cuadrados de porch cerrado, **está bien definida**.

[73]:
```
print(train['EnclosedPorch'].dtype)
print(train['EnclosedPorch'].isnull().sum())
```

```
int64
0
```

2.69 Var 68 3SsnPorch

pies cuadrados de proch de tres estaciones Q_Q, ,**está bien definida**.

[74]:
```
print(train['3SsnPorch'].dtype)
print(train['3SsnPorch'].isnull().sum())
```

```
int64
0
```

2.70 Var 69 ScreenPorch

pies cuadrados de fachada del proch, **está bien definida**.

[75]:
```
print(train['ScreenPorch'].dtype)
print(train['ScreenPorch'].isnull().sum())
```

```
int64
0
```

2.71 Var 70 PoolArea

pies cuadrados de la superficie de la alberca, **está bien definida**.

[76]:
```
print(train['PoolArea'].dtype)
print(train['PoolArea'].isnull().sum())
```

```
int64
0
```

2.72 Var 71 PoolQC

Calidad de la alberca, **está bien definida**.

[77]:
```
print(train['PoolQC'].dtype)
print(train['PoolQC'].isnull().sum())
```

```
object
0
```

2.73 Var 72 Fence

calidad de la cerca, **está bien definida**.

[78]:
```
print(train['Fence'].dtype)
print(train['Fence'].isnull().sum())
```

```
object
0
```

2.74 Var 73 MiscFeature

articulos miscelaneos, **es´ta bien definida**.

[79]:
```
print(train['MiscFeature'].dtype)
print(train['MiscFeature'].isnull().sum())
```

```
object
0
```

2.75 Var 74 MiscVal

valor de los articulos miscelaneos, **está bien definida**.

[80]:
```
print(train['MiscVal'].dtype)
print(train['MiscVal'].isnull().sum())
```

```
int64
0
```

2.76 Var 75 MoSold

Mes de venta, **está bien definida**.

[81]:
```
print(train['MoSold'].dtype)
print(train['MoSold'].isnull().sum())
```

```
int64
0
```

2.77 Var 76 YrSold

año de venta, **está bien definida**.

[82]:
```
print(train['YrSold'].dtype)
print(train['YrSold'].isnull().sum())
```

```
int64
0
```

2.78 Var 77 SaleType

tipo de venta, **está bien definida**

[83]:
```
print(train['SaleType'].dtype)
print(train['SaleType'].isnull().sum())
```

```
object
0
```

2.79 Var 78 SaleCondition

condiciones en que se dio la venta, **está bien definida**

[84]:
```
print(train['SaleCondition'].dtype)
print(train['SaleCondition'].isnull().sum())
```

```
object
0
```

2.80 Var 79 SalePrice

Precio de venta, **está bien definida**.

[85]:
```
print(train['SalePrice'].dtype)
print(train['SalePrice'].isnull().sum())
```

```
int64
0
```

2.81 Resumen y comentarios

La variable 0 "MSSubClass", originalmente identificada como entero, es en realidad categórica.

Existe inconsistencia en cuanto a la categorización pues hay variables que evalúan calidad como "ExterQual" y "ExterCond" cuyas categórias son:

```
Ex    Excellent
Gd    Good
TA    Average/Typical
Fa    Fair
Po    Poor
```

Sin embargo, "OverallCond" es:

```
10    Very Excellent
9     Excellent
8     Very Good
7     Good
6     Above Average
```

```
5     Average
4     Below Average
3     Fair
2     Poor
1     Very Poor
```

Aunque ambas "miden" o "califican" unas decidieron hacerlo catgóricamente y otras numéricamente.

Se decidió dejar "OverallCond" como numérica.

3 Breve análisis descriptivo de los datos

3.1 Variables categóricas

Primeramente, si se realiza un histograma se puede inferir acerca de la distribución de los datos, por ejemplo, a continuación se realiza para aquellas variables que son categóricas, esto porque las variables numéricas sería mejor analizarlo con una matriz de correlación.

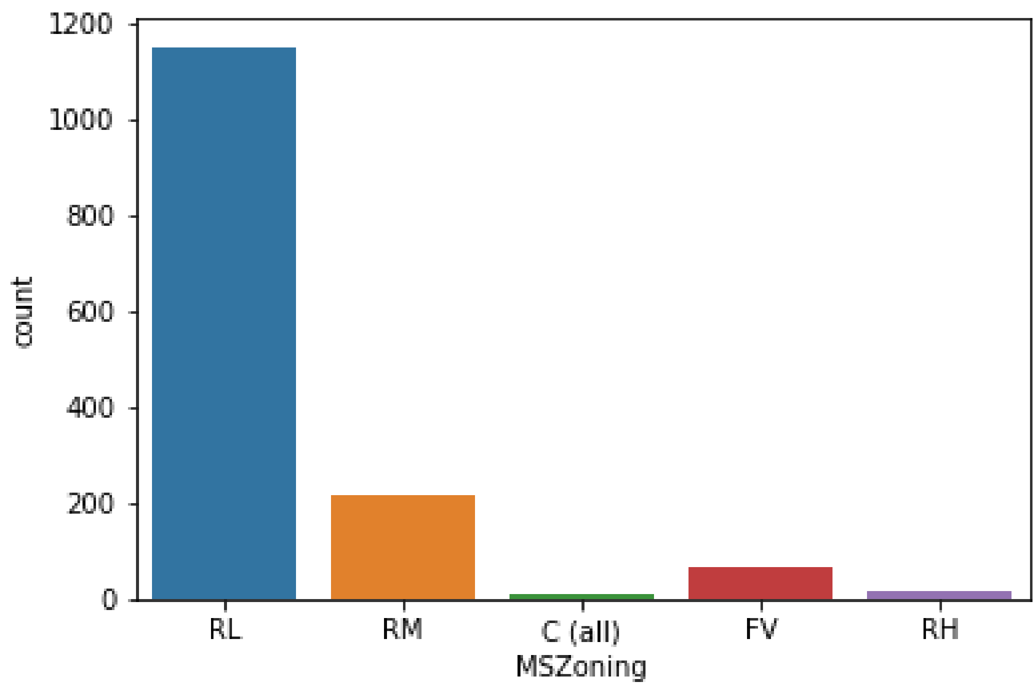
[86]:
```
#Las variables que son object
import numpy as np
import copy
trainObj = train.select_dtypes(include=['object' or 'category']).copy()
#también lo hacemos para test
testObj = test.select_dtypes(include=['object'or 'category']).copy()
trainObj.head()
```

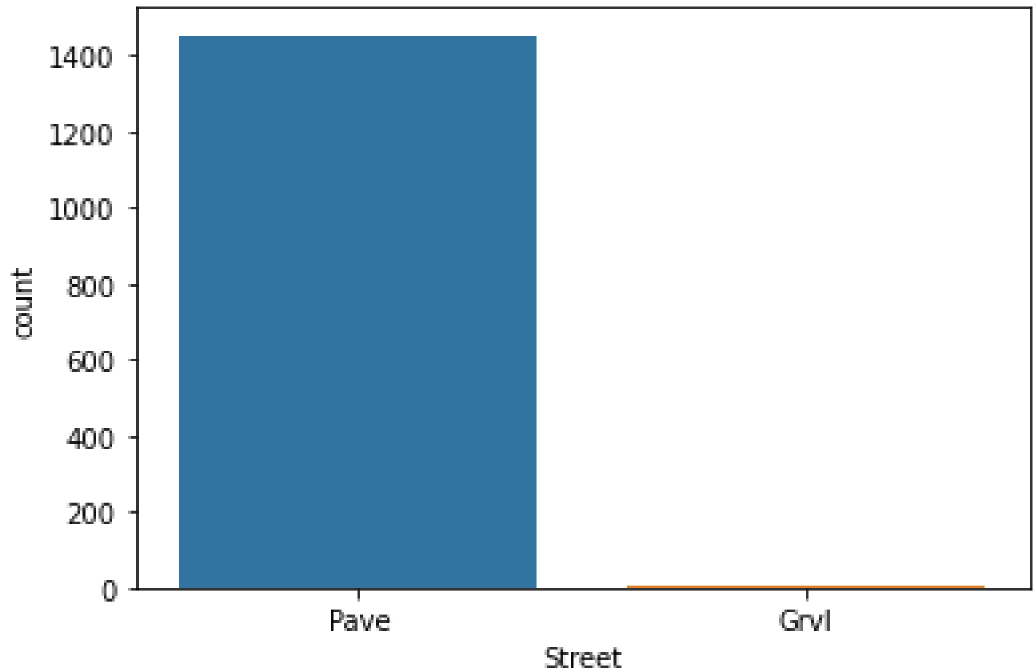
[86]:
```
   MSZoning Street Alley LotShape LandContour Utilities LotConfig LandSlope  \
Id
1        RL   Pave   NoA      Reg         Lvl    AllPub    Inside       Gtl
2        RL   Pave   NoA      Reg         Lvl    AllPub       FR2       Gtl
3        RL   Pave   NoA      IR1         Lvl    AllPub    Inside       Gtl
4        RL   Pave   NoA      IR1         Lvl    AllPub    Corner       Gtl
5        RL   Pave   NoA      IR1         Lvl    AllPub       FR2       Gtl

   Neighborhood Condition1  ... GarageType GarageFinish GarageQual GarageCond  \
Id                          ...
1       CollgCr       Norm  ...     Attchd          RFn         TA         TA
2       Veenker      Feedr  ...     Attchd          RFn         TA         TA
3       CollgCr       Norm  ...     Attchd          RFn         TA         TA
4       Crawfor       Norm  ...     Detchd          Unf         TA         TA
5       NoRidge       Norm  ...     Attchd          RFn         TA         TA

   PavedDrive PoolQC Fence MiscFeature SaleType SaleCondition
Id
1           Y    NoA   NoA         NoA       WD        Normal
2           Y    NoA   NoA         NoA       WD        Normal
```

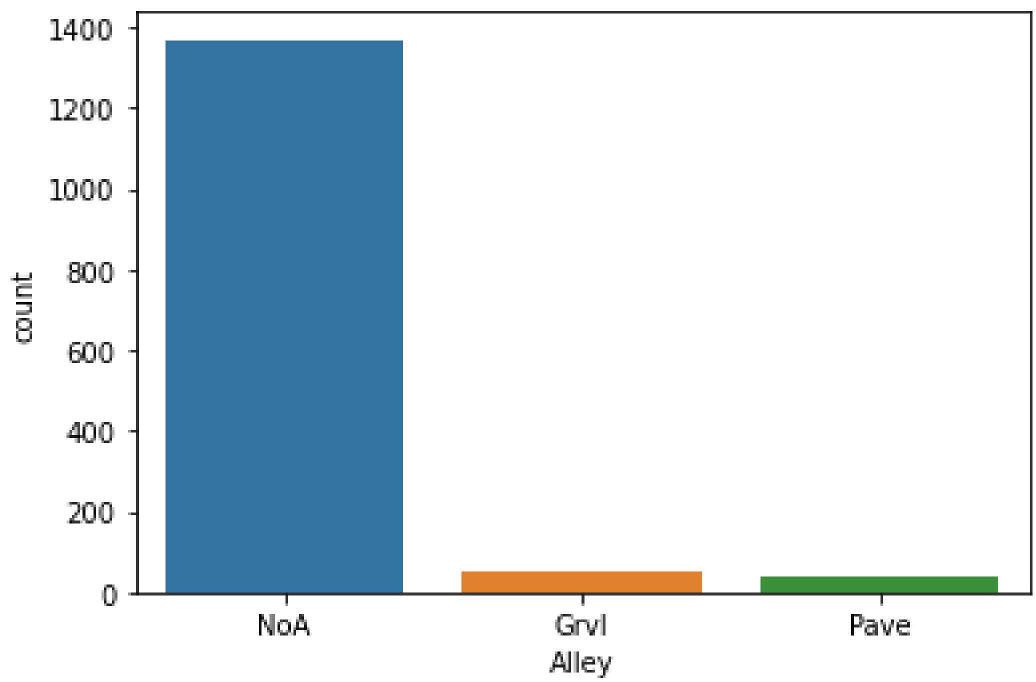
```
3              Y     NoA    NoA            NoA        WD          Normal
4              Y     NoA    NoA            NoA        WD         Abnorml
5              Y     NoA    NoA            NoA        WD          Normal

[5 rows x 43 columns]
```

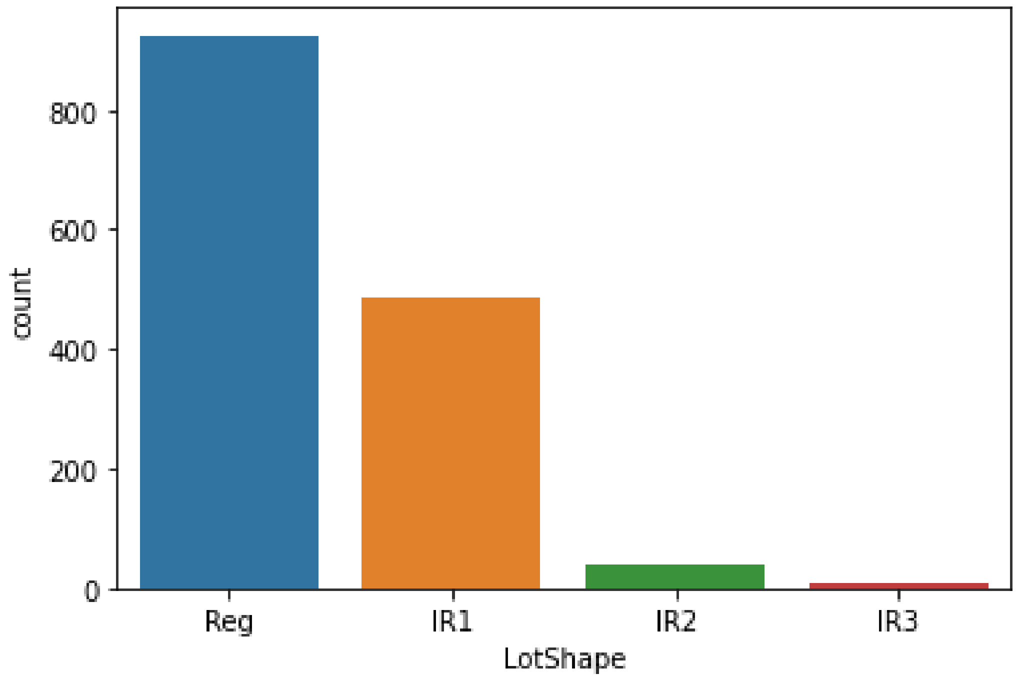
[87]:
```
%matplotlib inline
import seaborn as sns
import matplotlib.pyplot as plt
for i, col in enumerate(trainObj.columns):
    plt.figure(i)
    sns.countplot(x=col, data=trainObj)
```

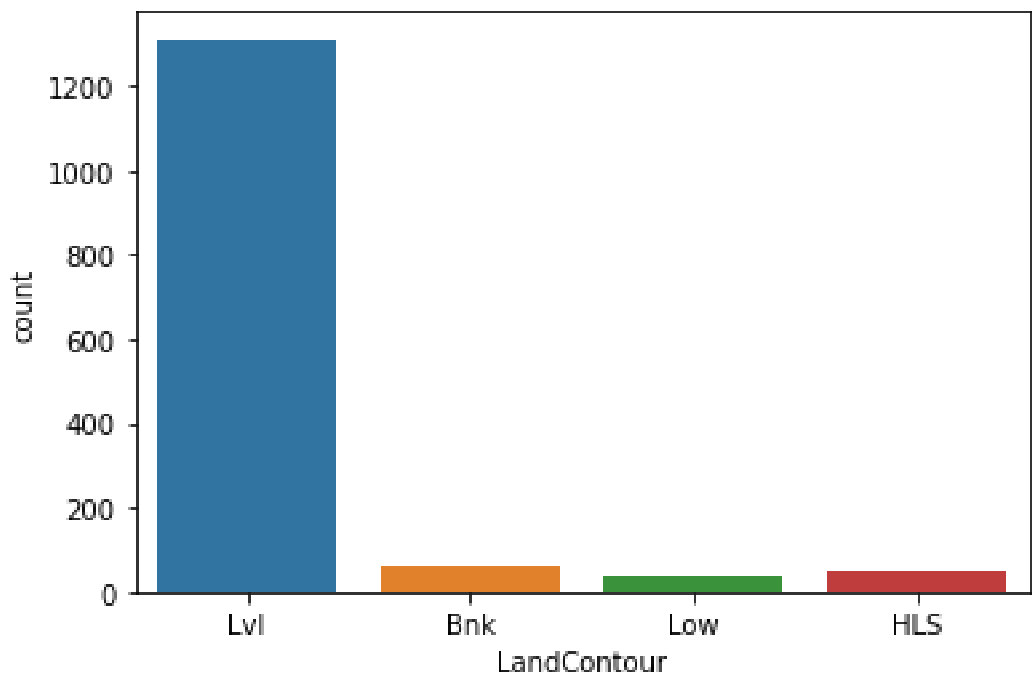
```
<ipython-input-87-d937b5dcecfe>:5: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface
(`matplotlib.pyplot.figure`) are retained until explicitly closed and may
consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  plt.figure(i)
```

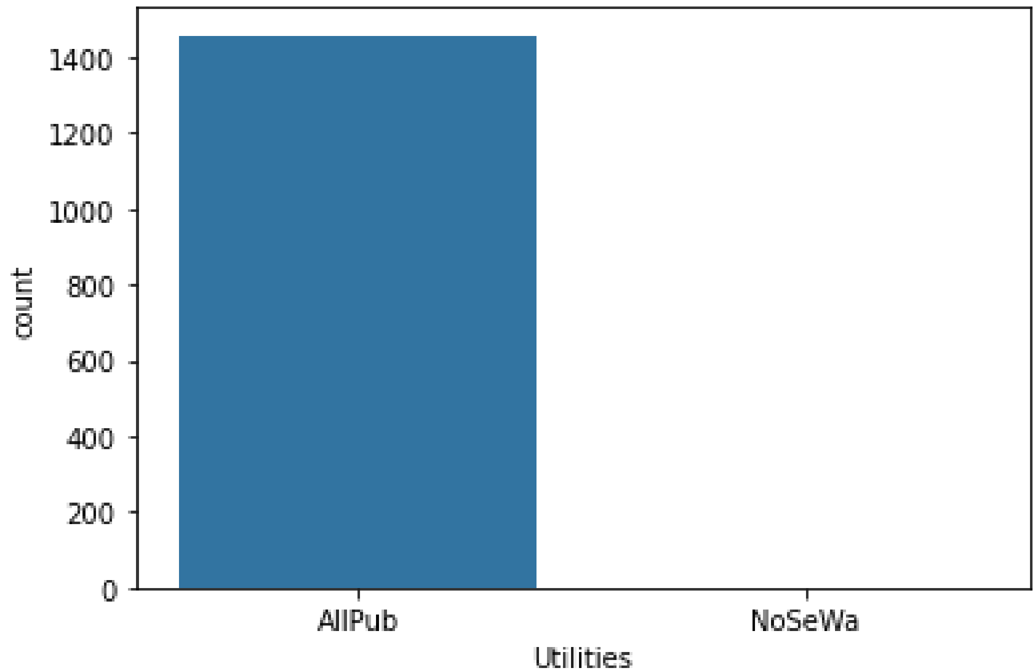


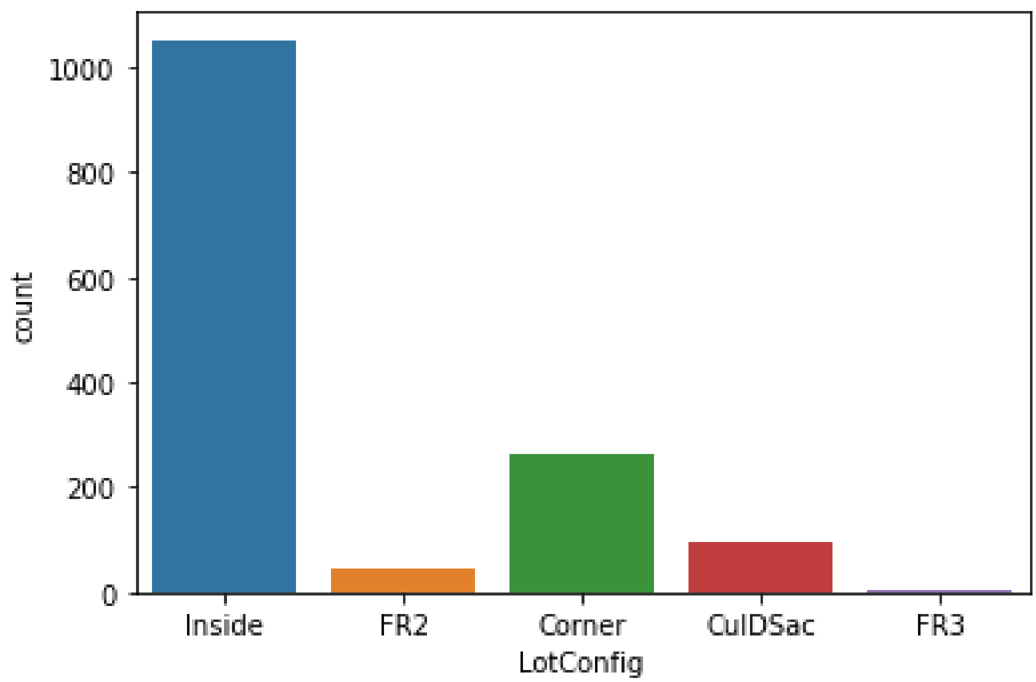

1400
1200
1000
800
600
400
200
0
count
Pave
Grvl
Street

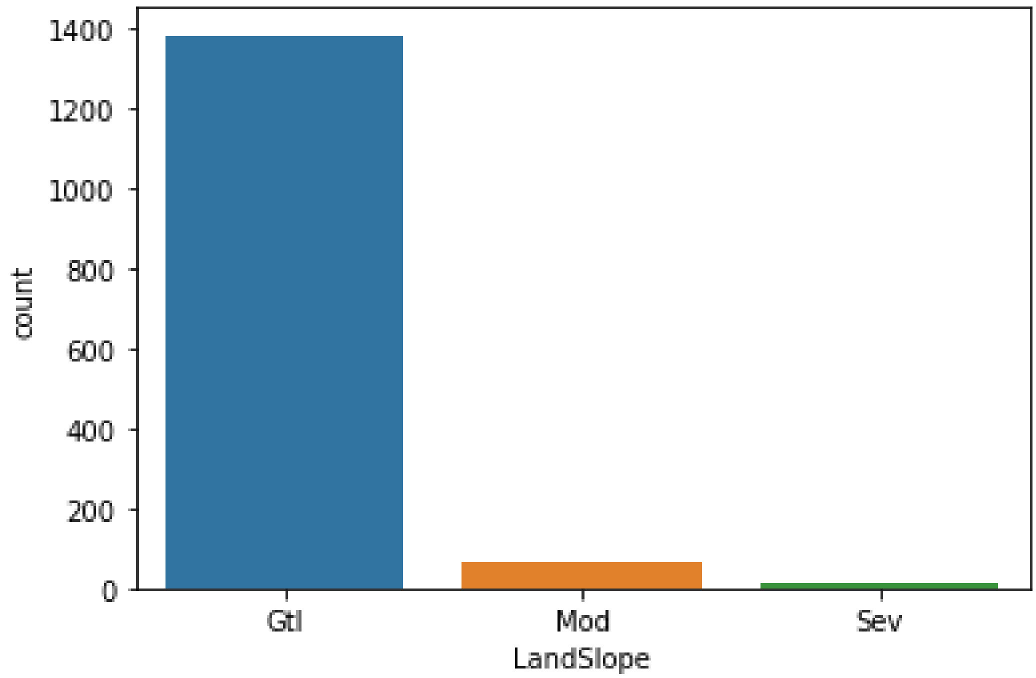

1400
1200
1000
800
600
400
200
0
count
NoA
Grvl
Pave
Alley

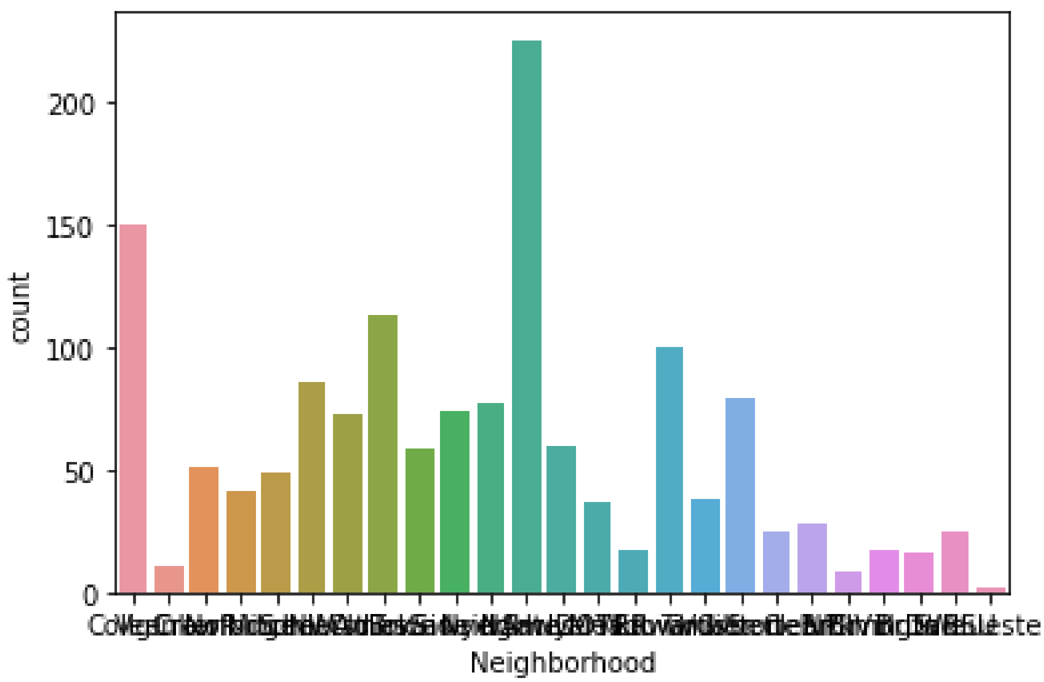

count
800
600
400
200
0
Reg
IR1
IR2
IR3
LotShape

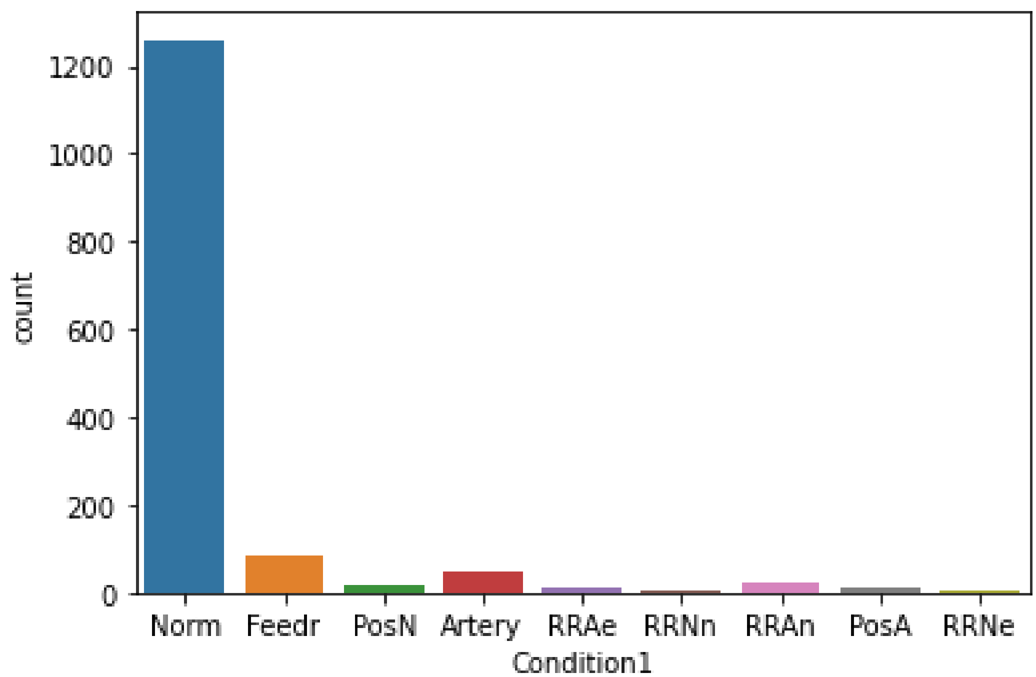

count
1200
1000
800
600
400
200
0
Lvl
Bnk
Low
HLS
LandContour

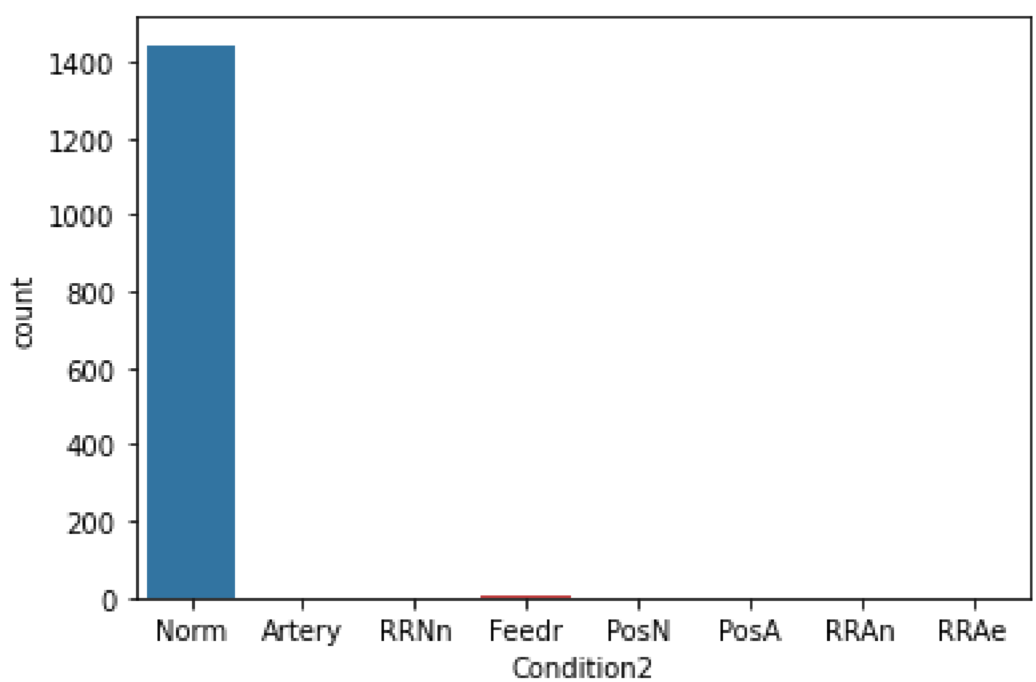

1400
1200
1000
800
600
400
200
0
count
AllPub
NoSeWa
Utilities

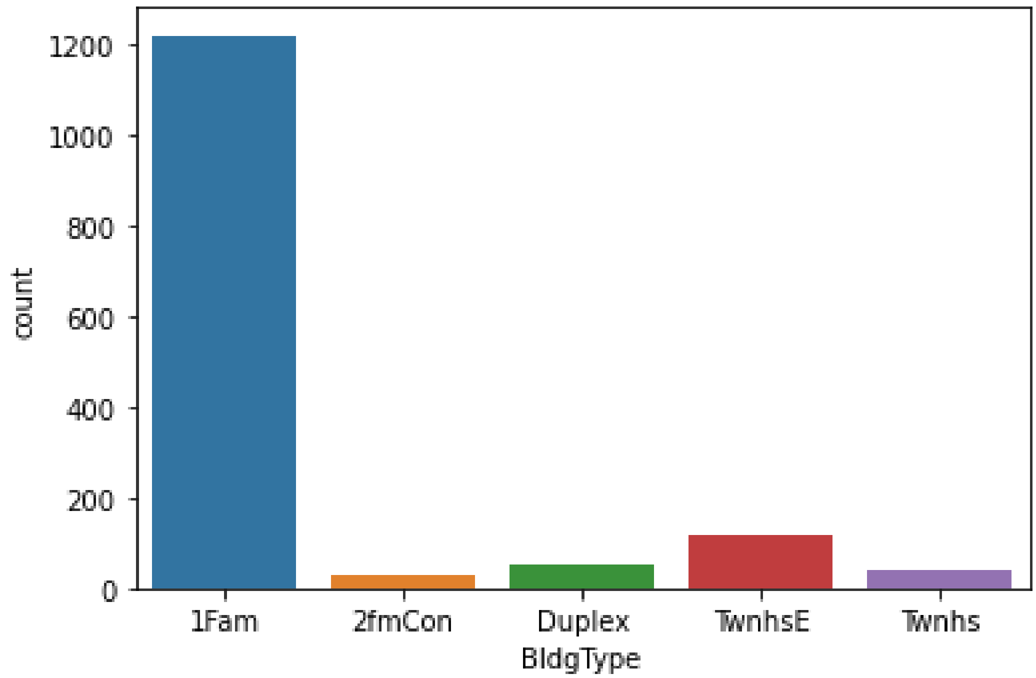

1000
800
600
400
200
0
count
Inside
FR2
Corner
CulDSac
FR3
LotConfig

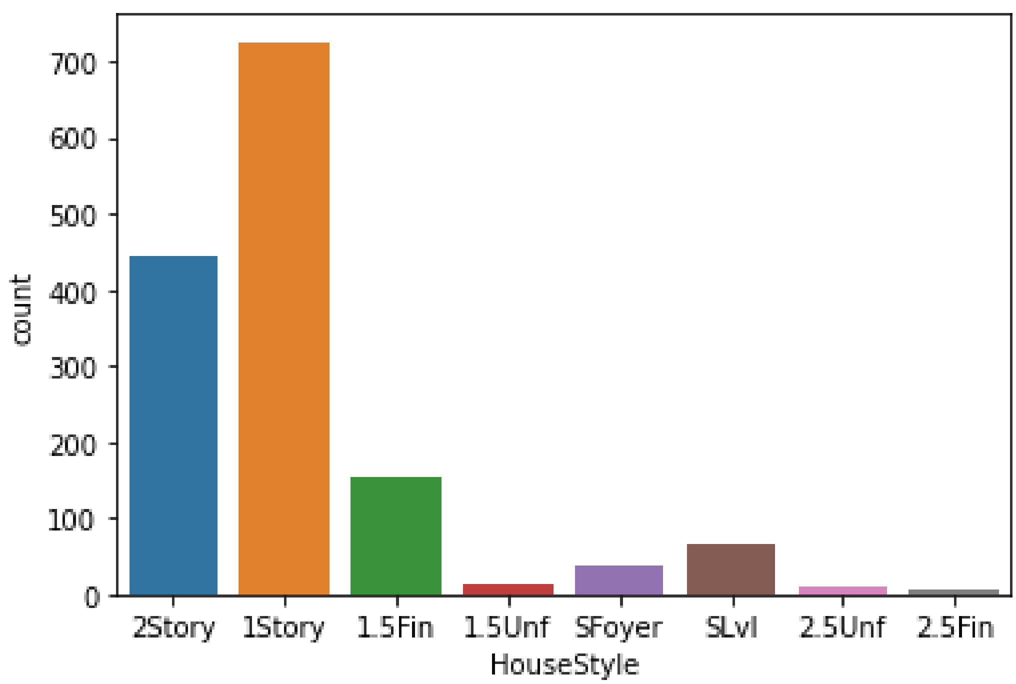

1400
1200
1000
800
600
400
200
0
count
Gtl
Mod
Sev
LandSlope

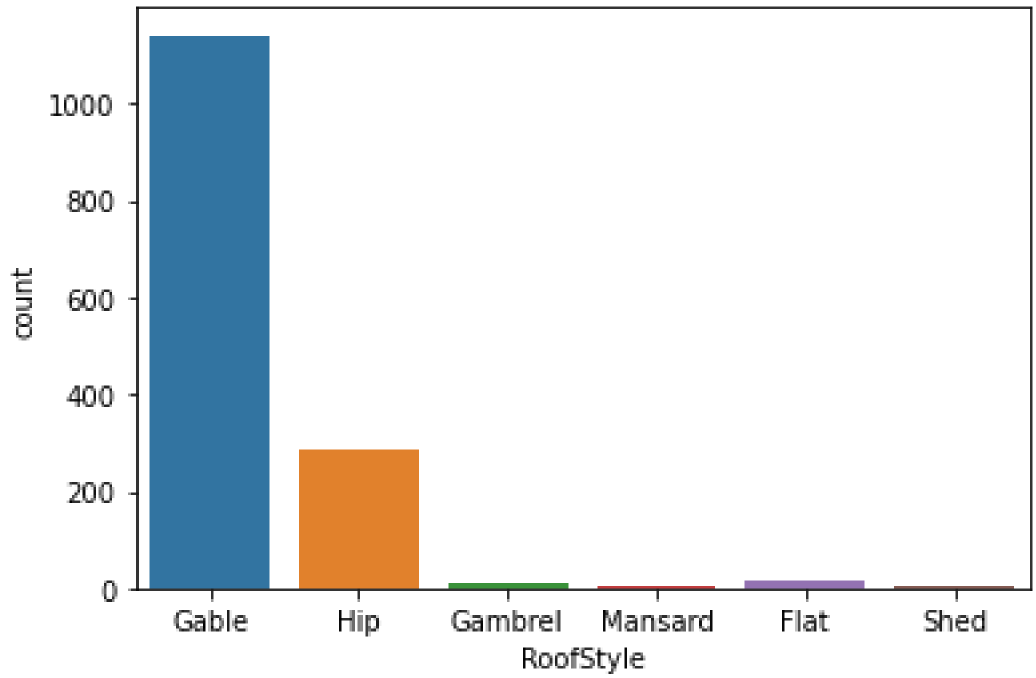

200
150
100
50
0
count
Neighborhood


1200
1000
800
600
400
200
0
count
Norm
Feedr
PosN
Artery
RRAe
RRNn
RRAn
PosA
RRNe
Condition1

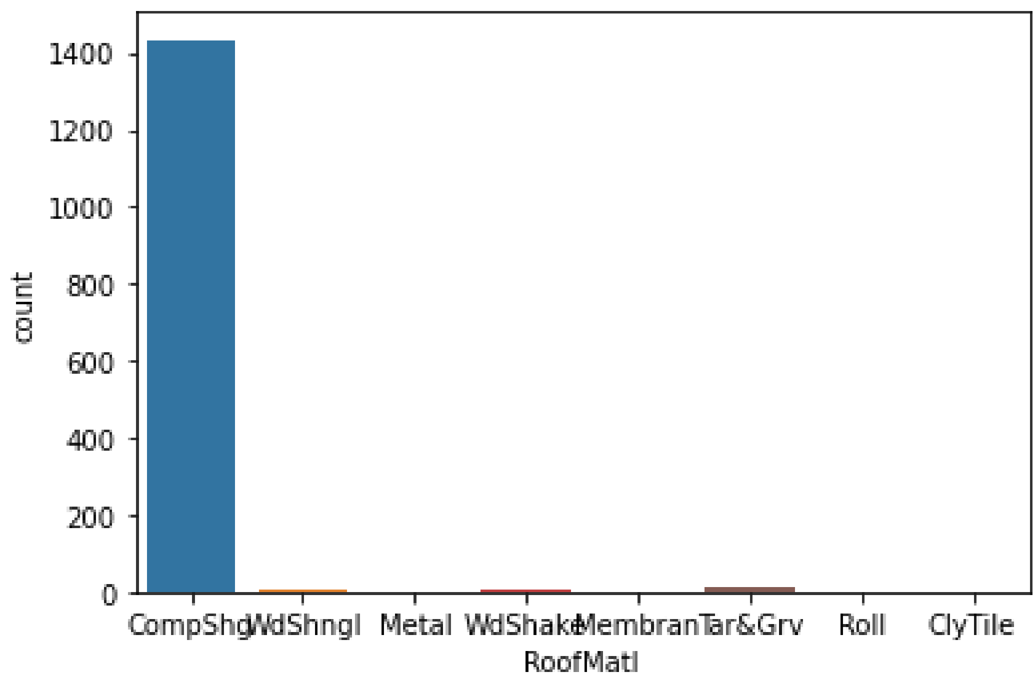

1400
1200
1000
800
600
400
200
0
count
Norm
Artery
RRNn
Feedr
PosN
PosA
RRAn
RRAe
Condition2


1200
1000
800
600
400
200
0
count
1Fam
2fmCon
Duplex
TwnhsE
Twnhs
BldgType

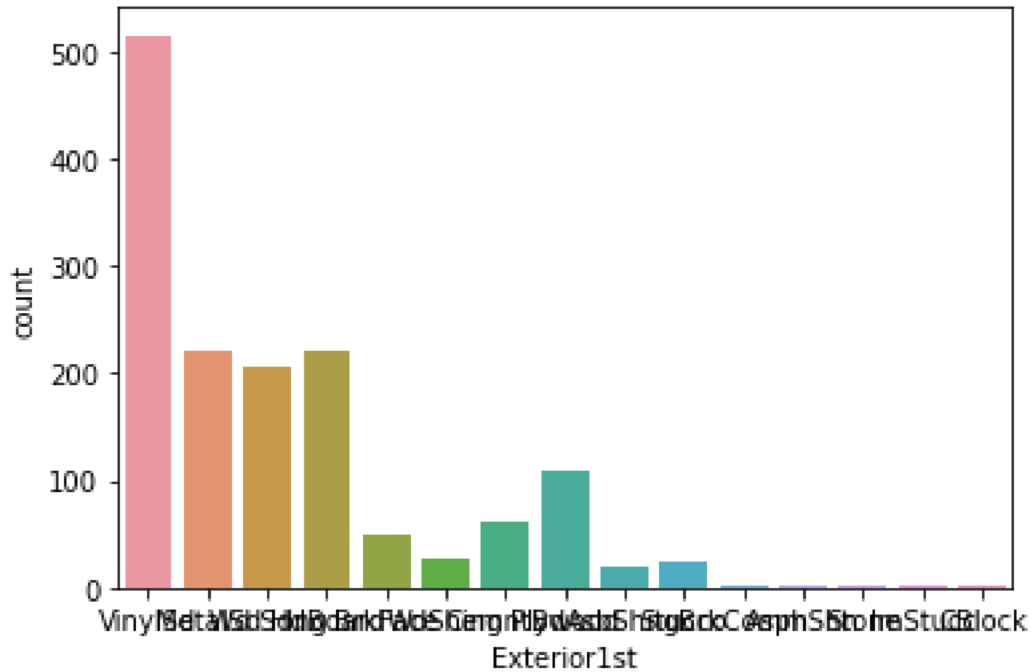

700
600
500
400
300
200
100
0
count
2Story
1Story
1.5Fin
1.5Unf
SFoyer
SLvl
2.5Unf
2.5Fin
HouseStyle

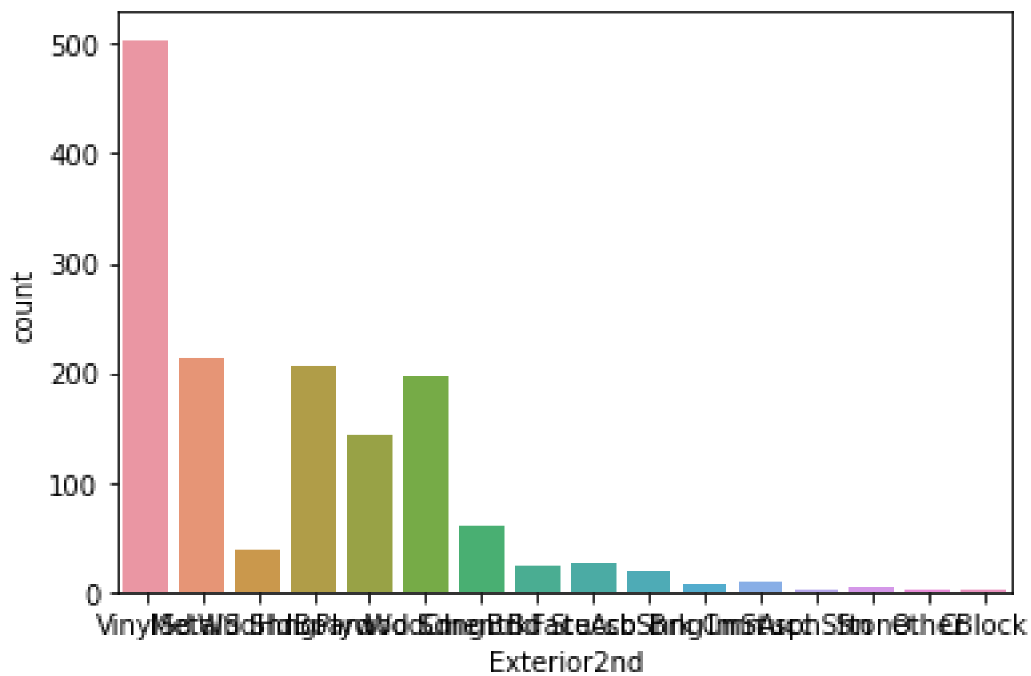


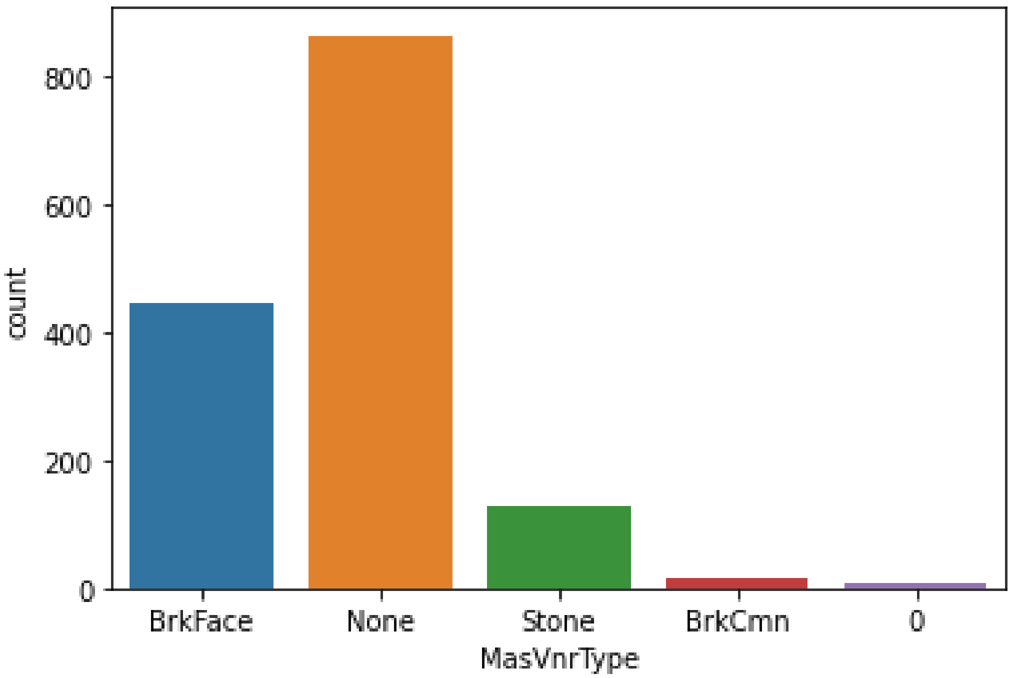
count
1000
800
600
400
200
0
Gable
Hip
Gambrel
Mansard
Flat
Shed
RoofStyle



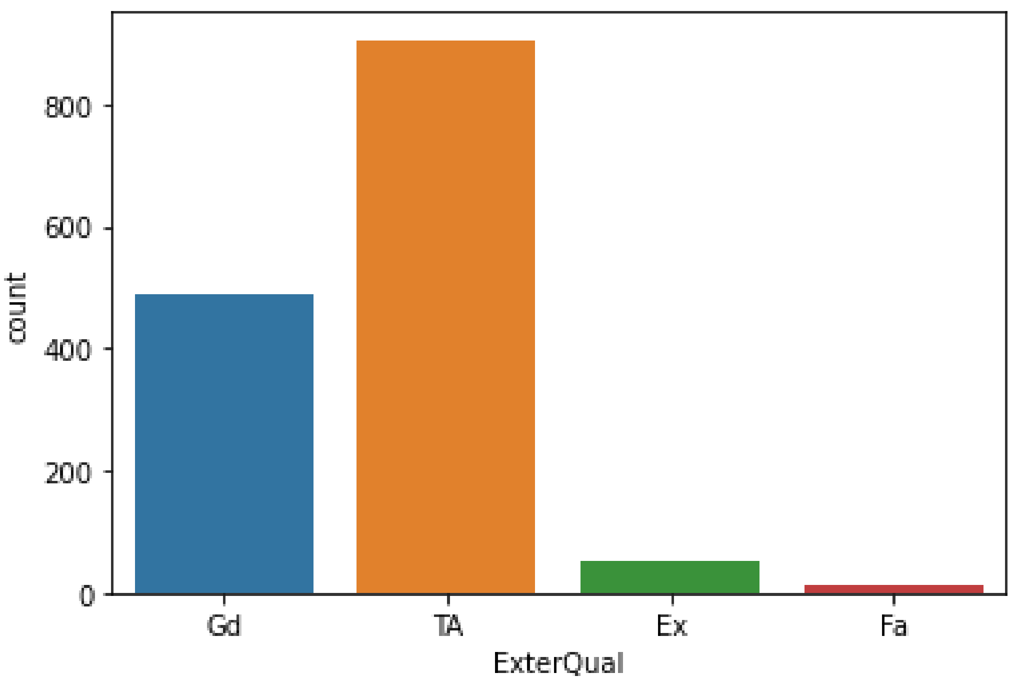
count
1400
1200
1000
800
600
400
200
0
CompShg
WdShngl
Metal
WdShake
Membran
Tar&Grv
Roll
ClyTile
RoofMatl


count
Exterior1st

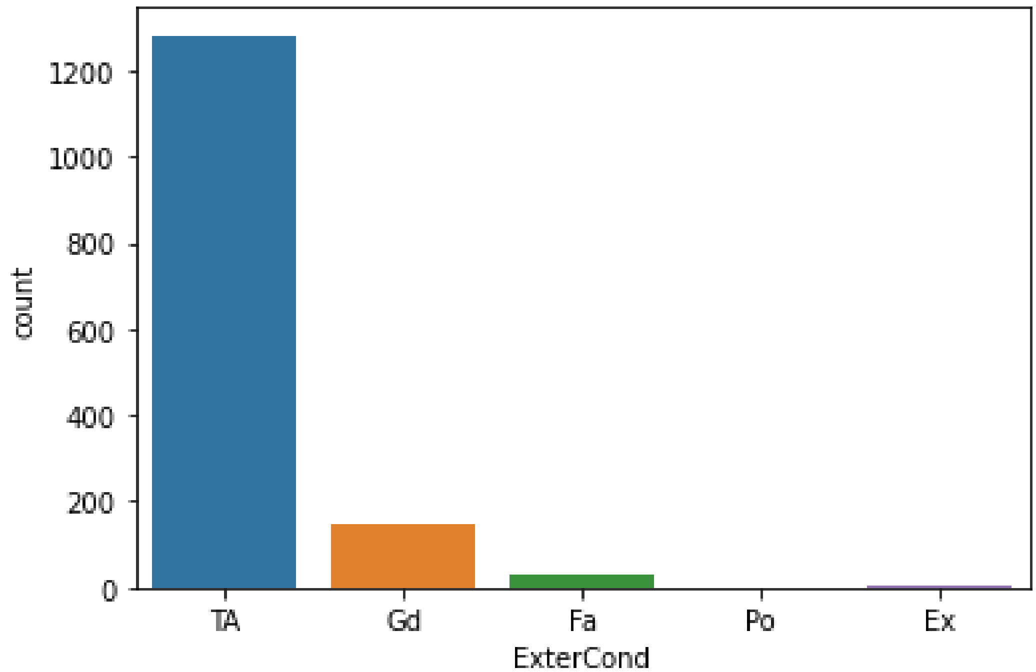

count
Exterior2nd

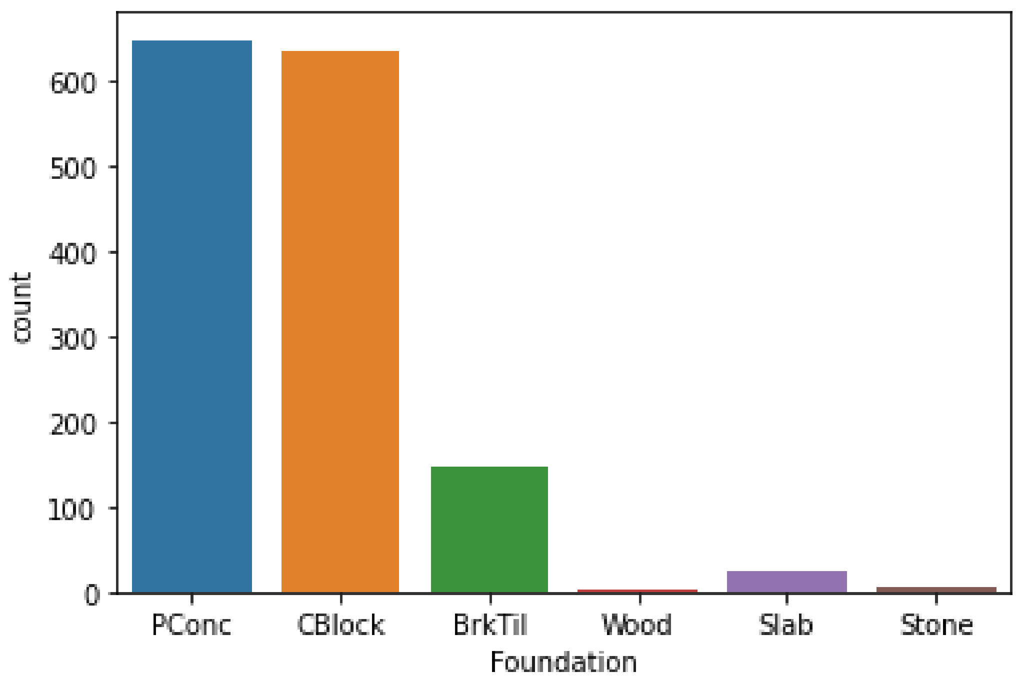


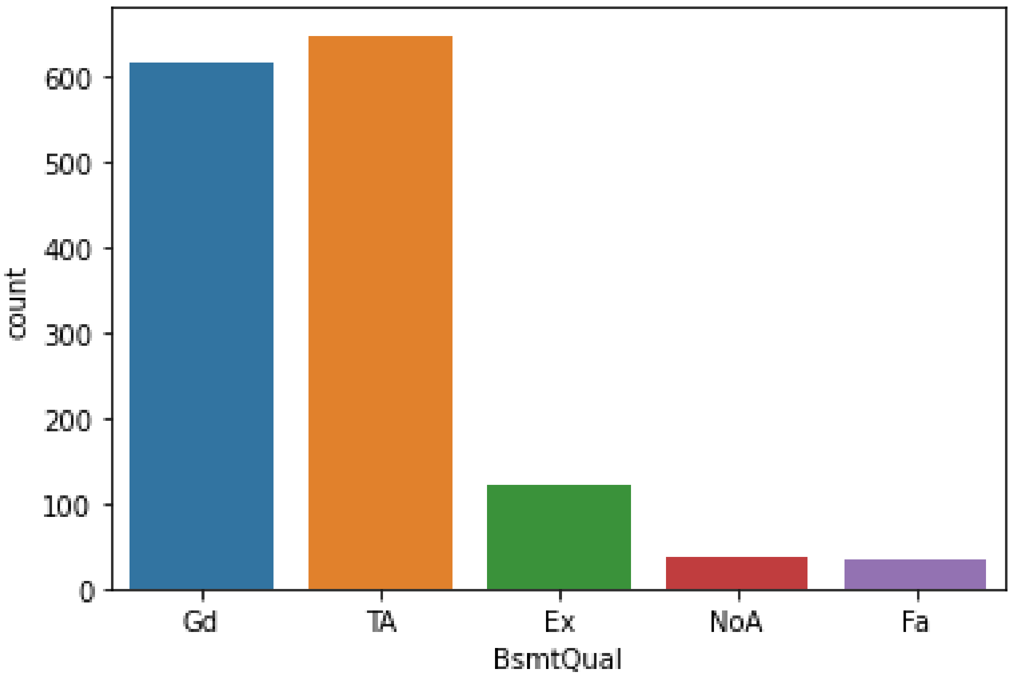
800
600
400
200
0
count
BrkFace
None
Stone
BrkCmn
0
MasVnrType

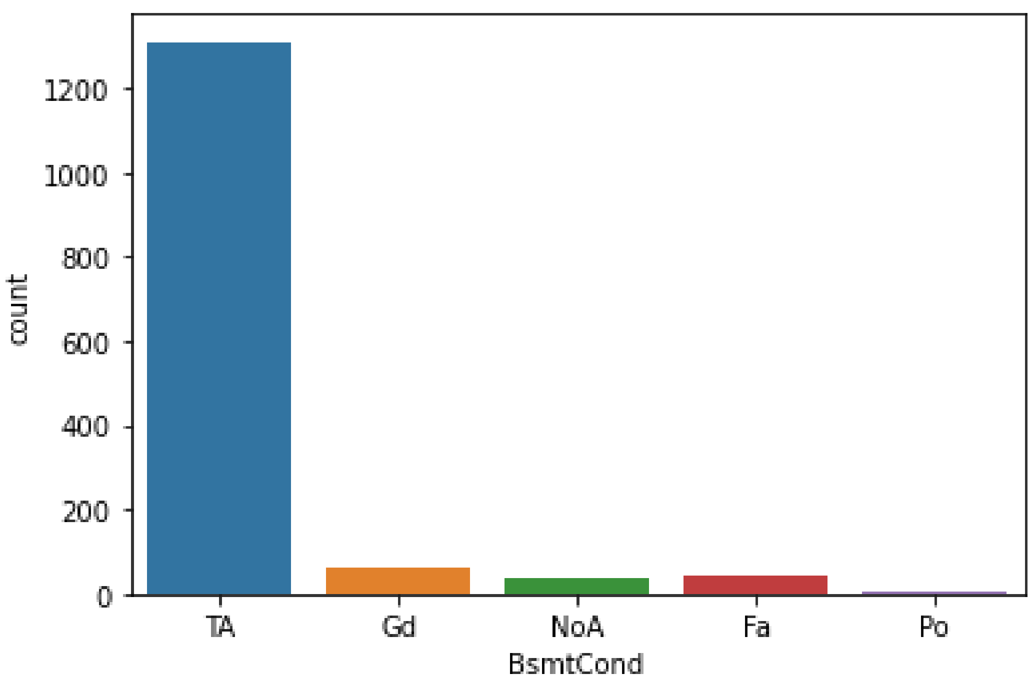


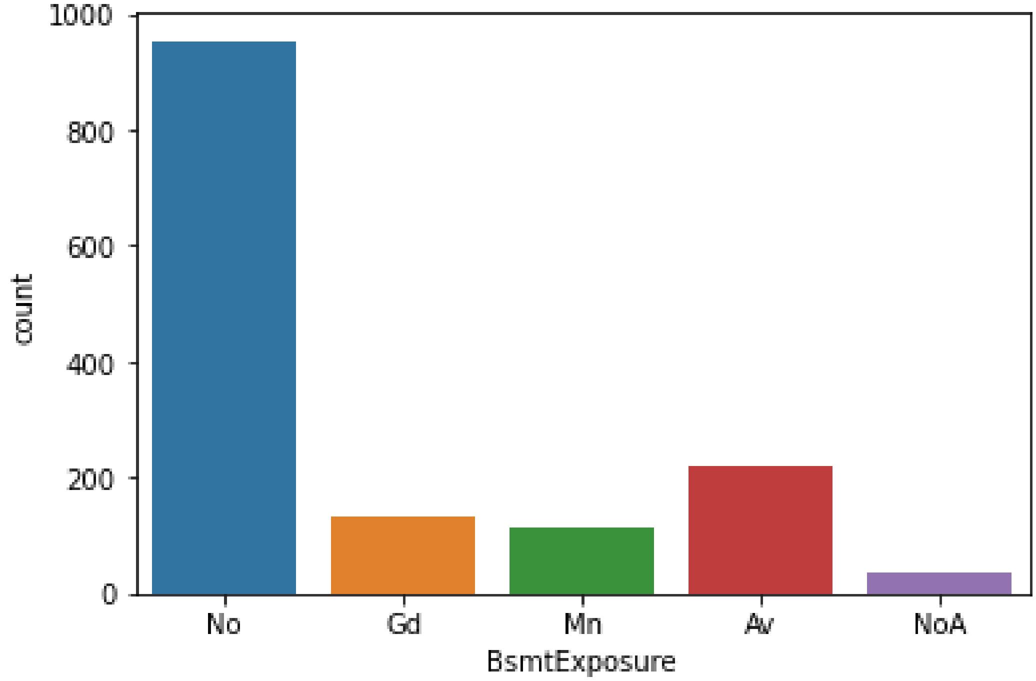
800
600
400
200
0
count
Gd
TA
Ex
Fa
ExterQual

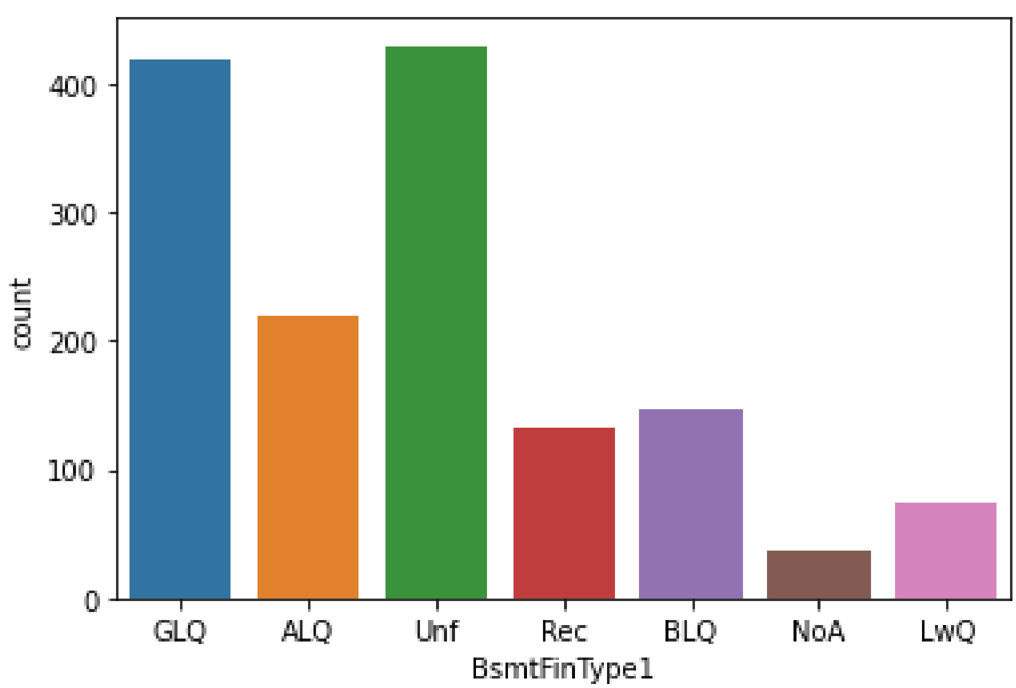

count
1200
1000
800
600
400
200
0
TA
Gd
Fa
Po
Ex
ExterCond

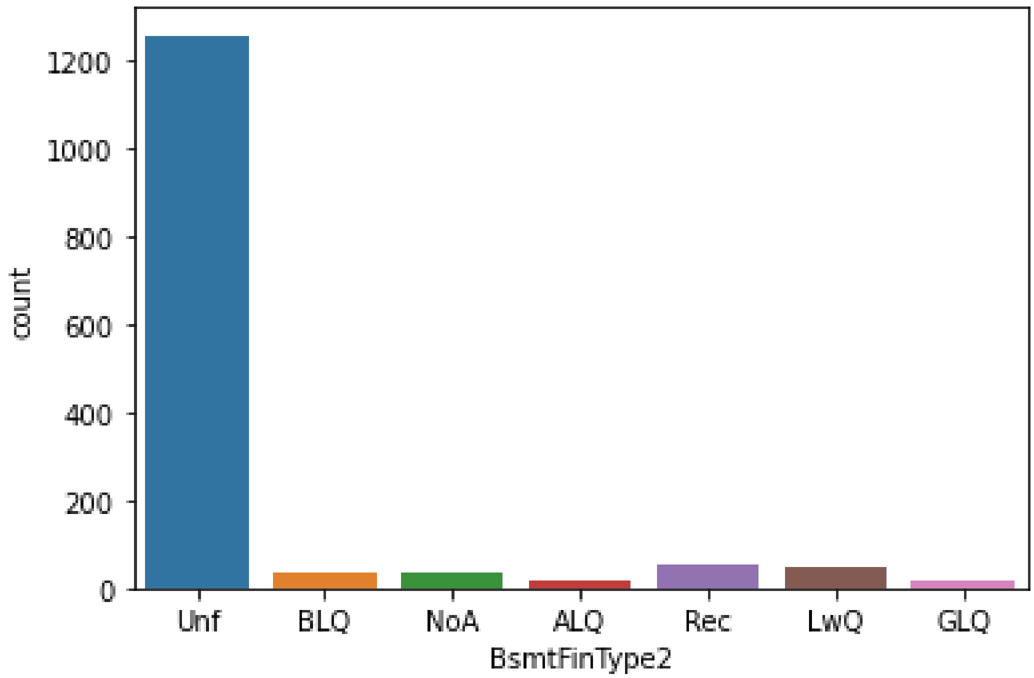

count
600
500
400
300
200
100
0
PConc
CBlock
BrkTil
Wood
Slab
Stone
Foundation

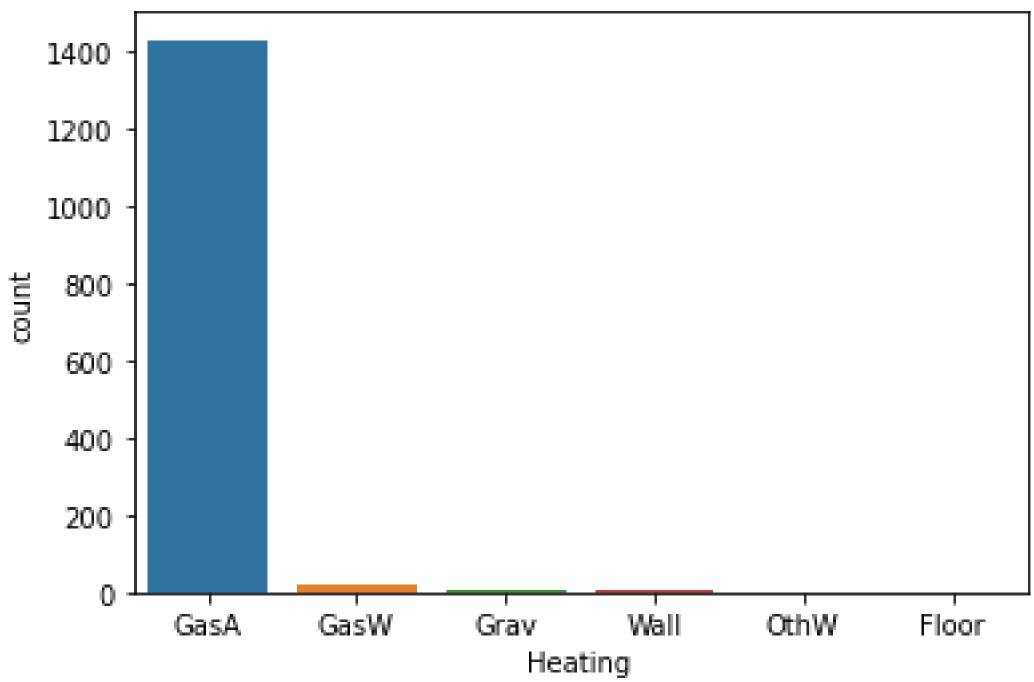

count
0
100
200
300
400
500
600
Gd
TA
Ex
NoA
Fa
BsmtQual

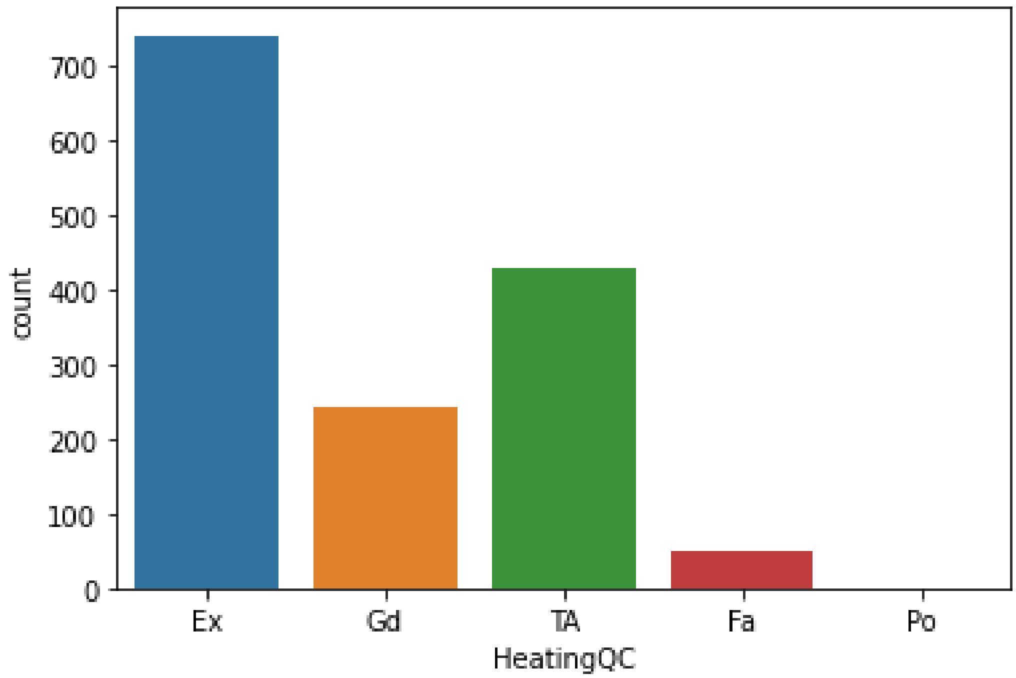

count
0
200
400
600
800
1000
1200
TA
Gd
NoA
Fa
Po
BsmtCond

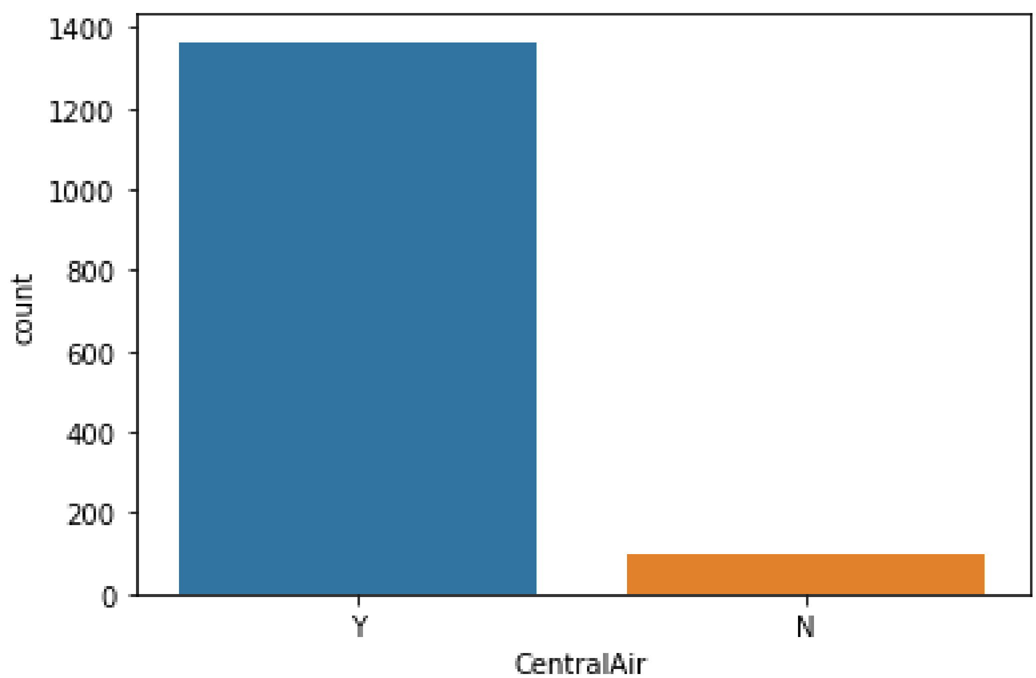

1000
800
600
400
200
0
count
No
Gd
Mn
Av
NoA
BsmtExposure

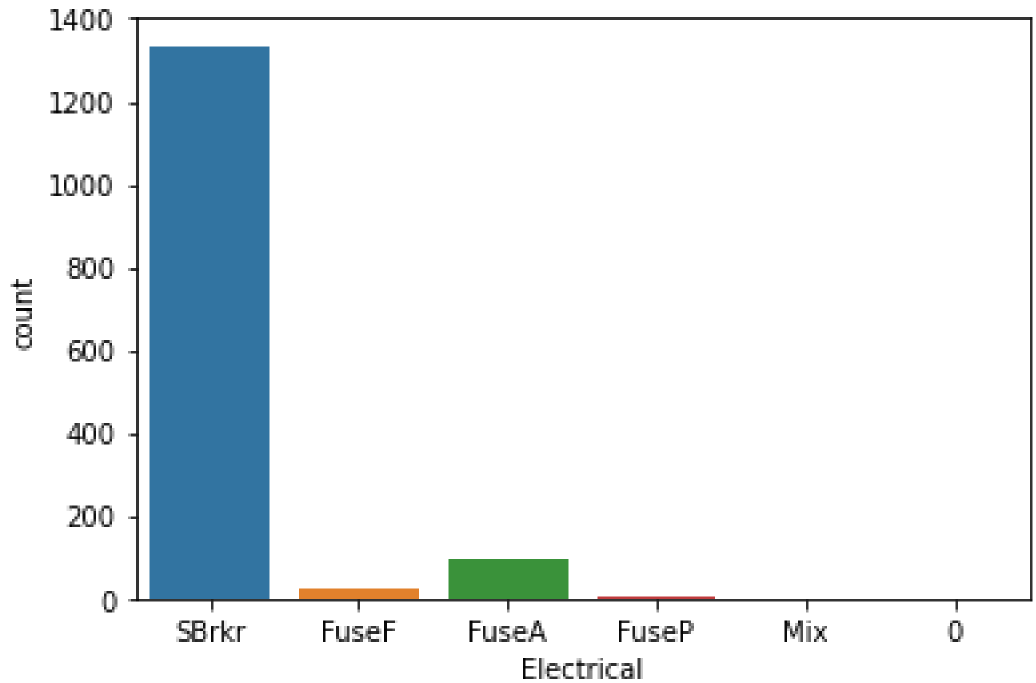

400
300
200
100
0
count
GLQ
ALQ
Unf
Rec
BLQ
NoA
LwQ
BsmtFinType1

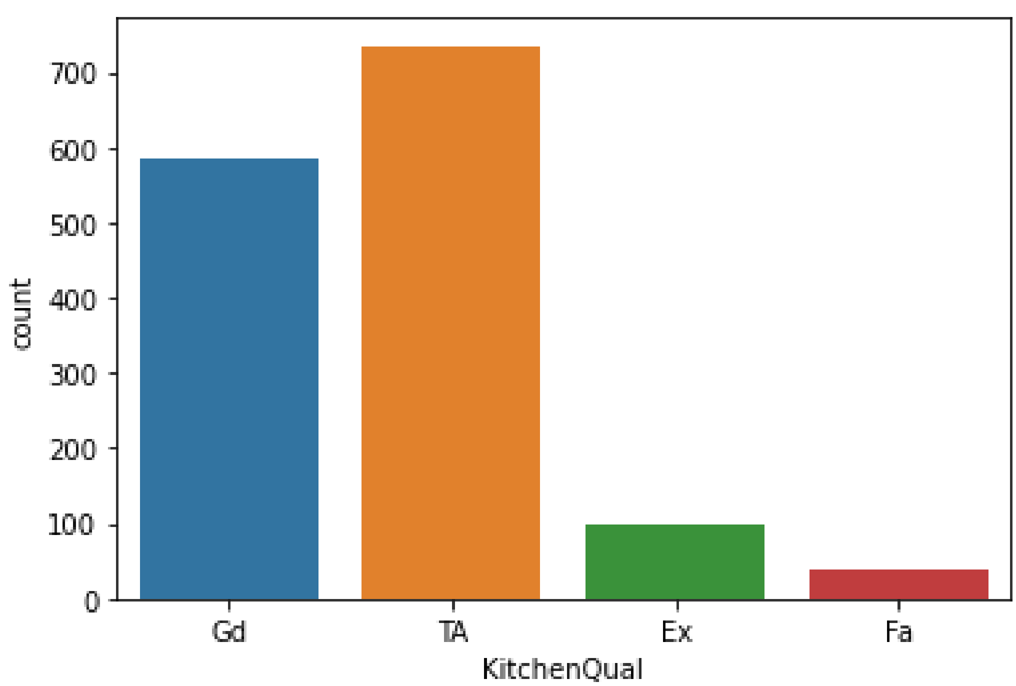

1200
1000
800
600
400
200
0
count
Unf
BLQ
NoA
ALQ
Rec
LwQ
GLQ
BsmtFinType2

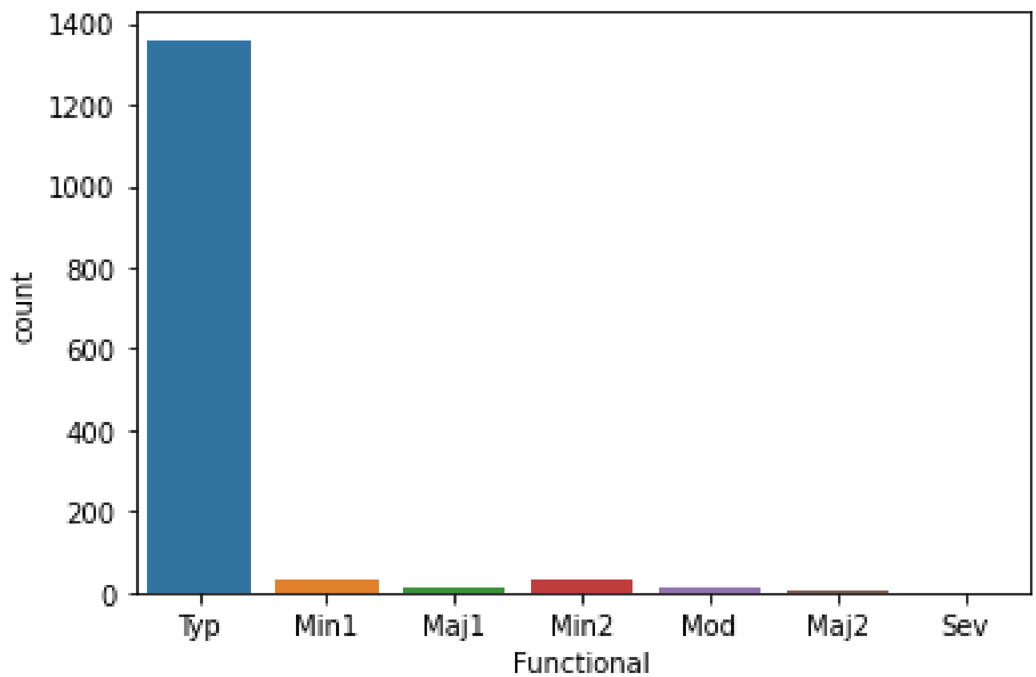

1400
1200
1000
800
600
400
200
0
count
GasA
GasW
Grav
Wall
OthW
Floor
Heating

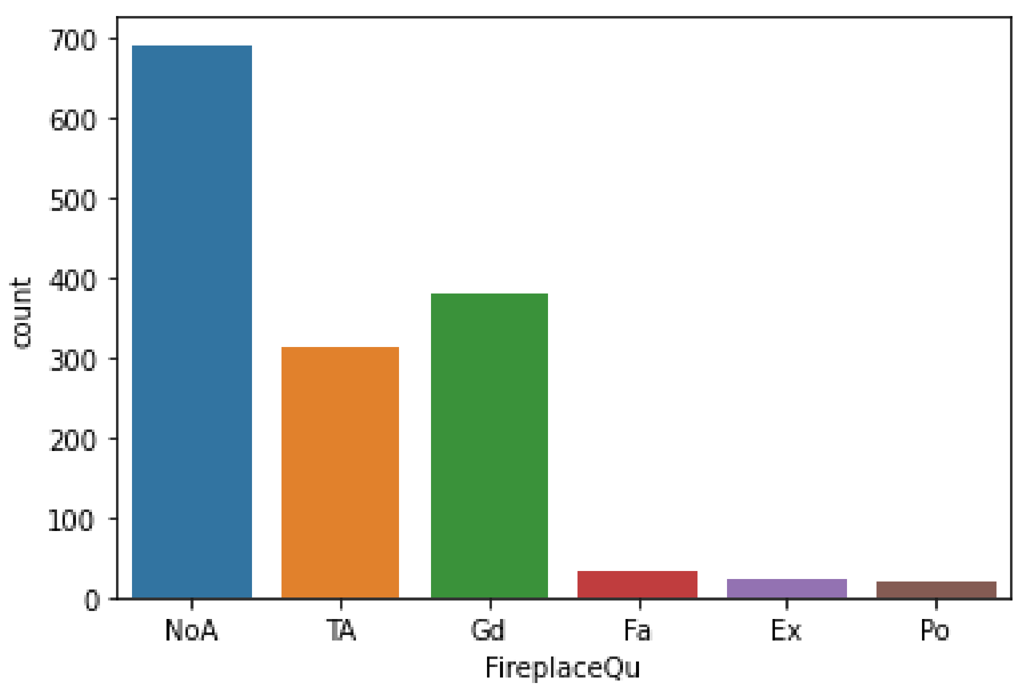

700
600
500
400
300
200
100
0
count
Ex
Gd
TA
Fa
Po
HeatingQC

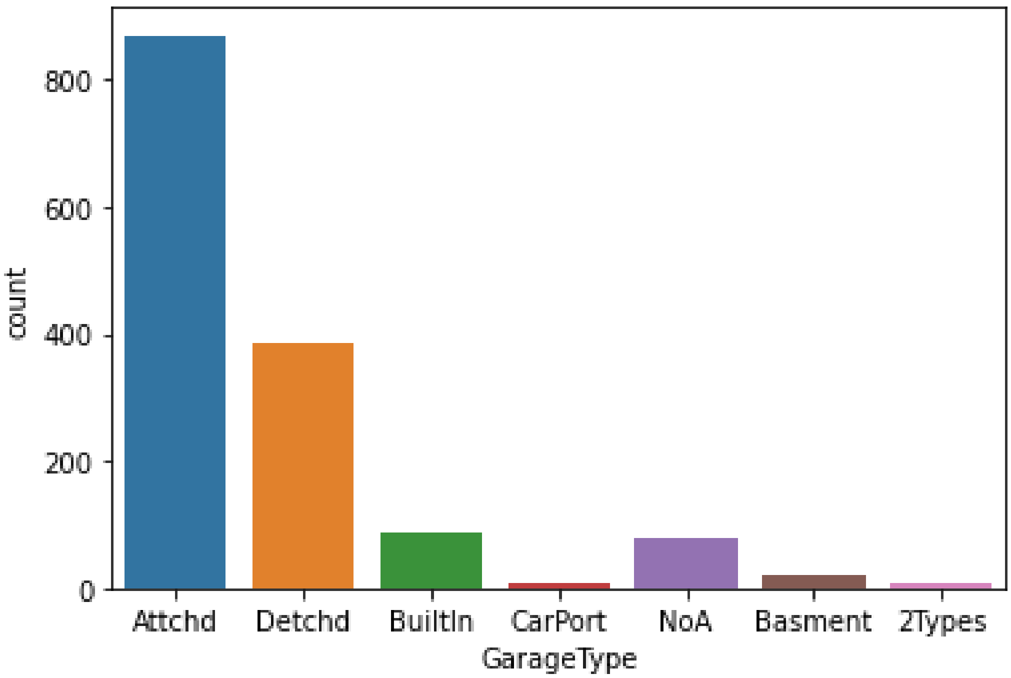

1400
1200
1000
800
600
400
200
0
count
Y
N
CentralAir

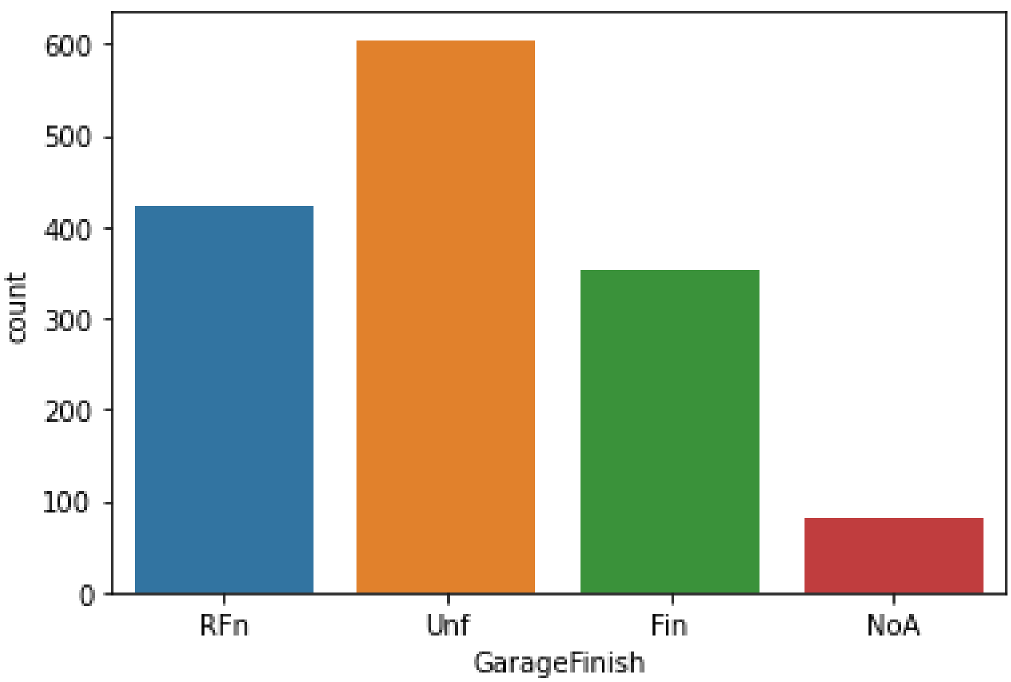

1400
1200
1000
800
600
400
200
0
count
SBrkr
FuseF
FuseA
FuseP
Mix
0
Electrical

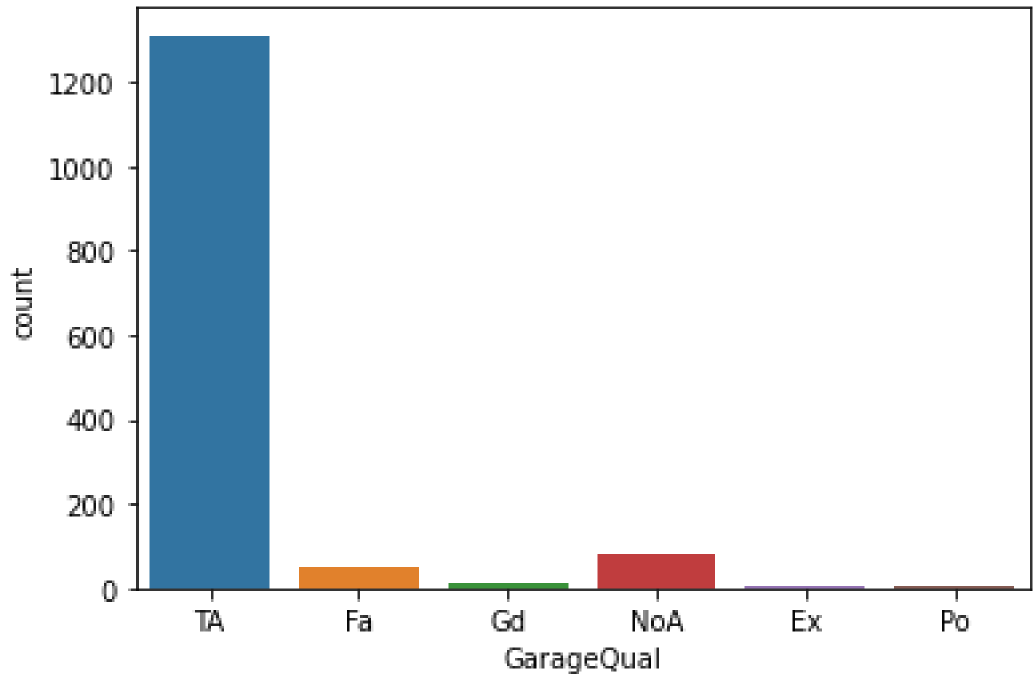

700
600
500
400
300
200
100
0
count
Gd
TA
Ex
Fa
KitchenQual

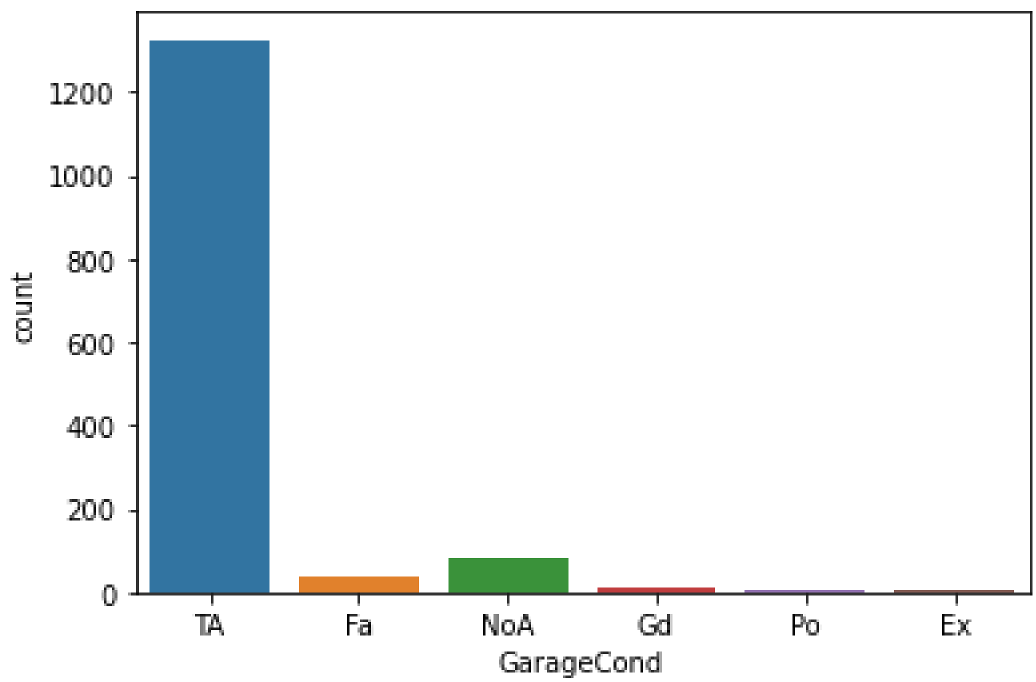

1400
1200
1000
800
600
400
200
0
count
Typ
Min1
Maj1
Min2
Mod
Maj2
Sev
Functional

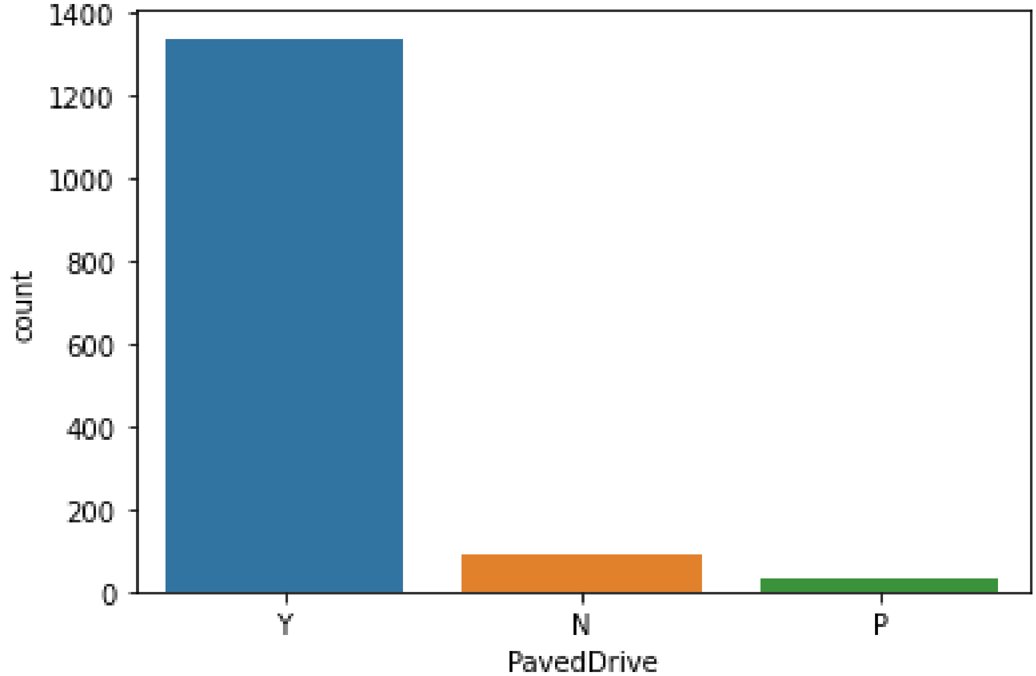

700
600
500
400
300
200
100
0
count
NoA
TA
Gd
Fa
Ex
Po
FireplaceQu

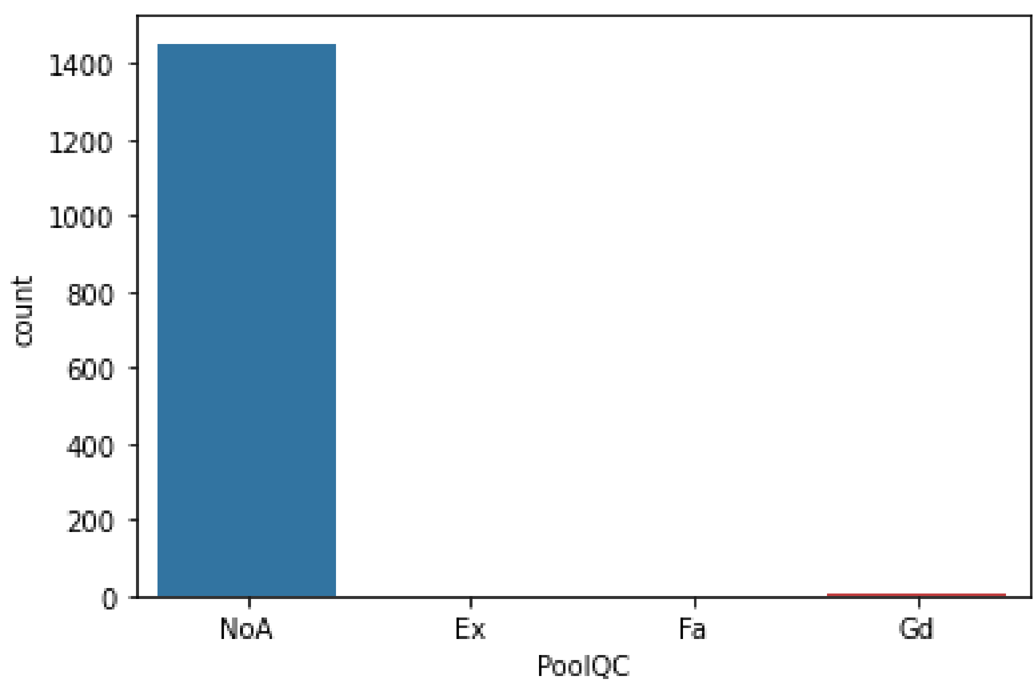

count
800
600
400
200
0
Attchd
Detchd
BuiltIn
CarPort
NoA
Basment
2Types
GarageType

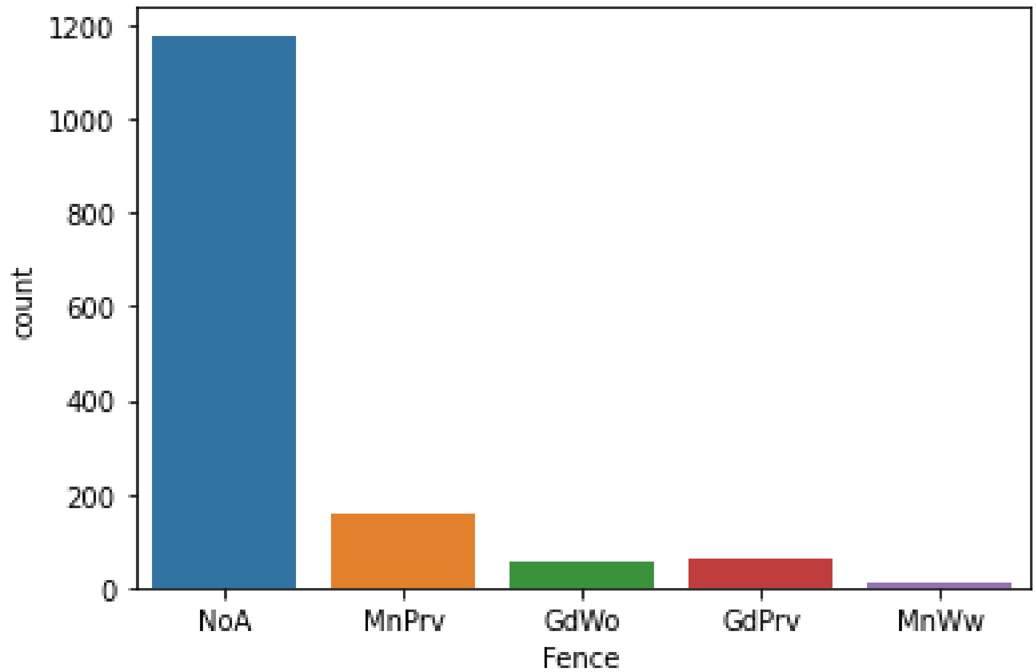

count
600
500
400
300
200
100
0
RFn
Unf
Fin
NoA
GarageFinish

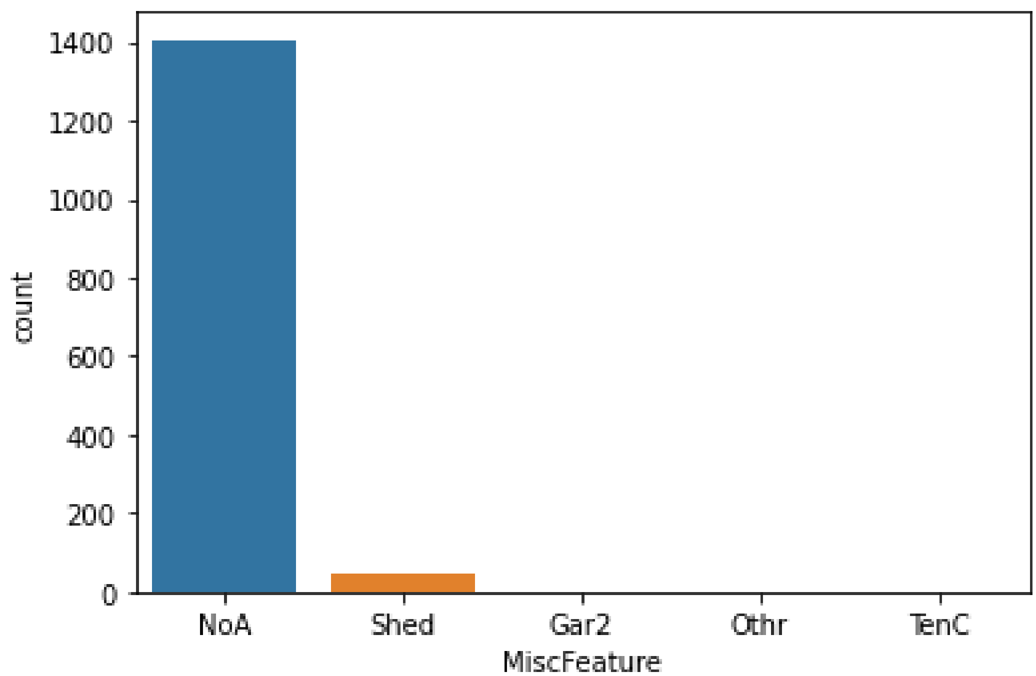

1200
1000
800
600
400
200
0
count
TA
Fa
Gd
NoA
Ex
Po
GarageQual

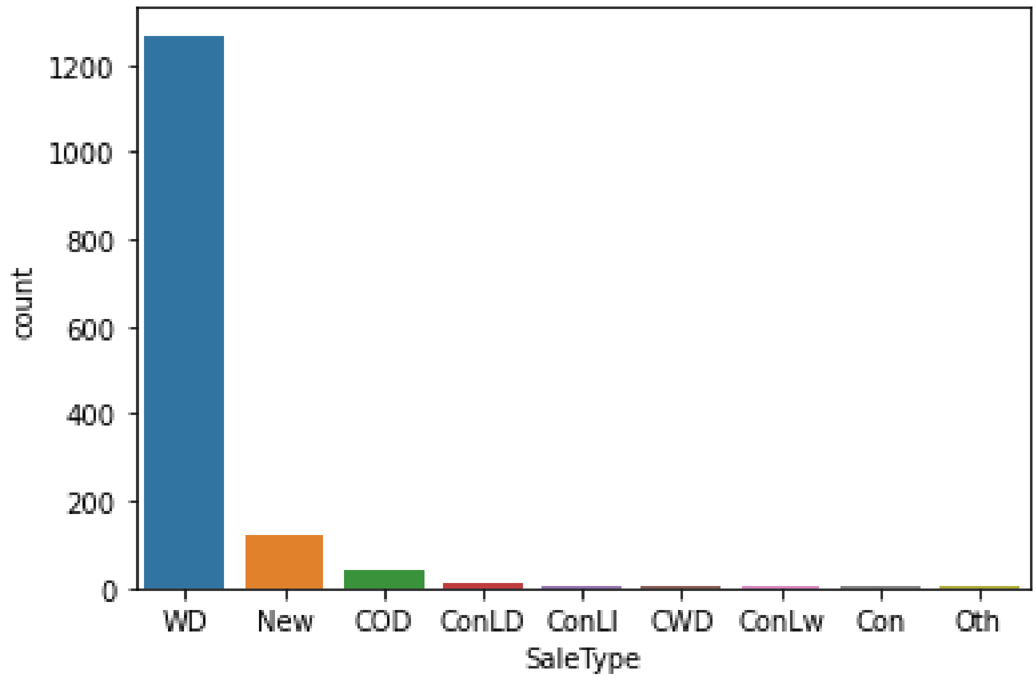

1200
1000
800
600
400
200
0
count
TA
Fa
NoA
Gd
Po
Ex
GarageCond

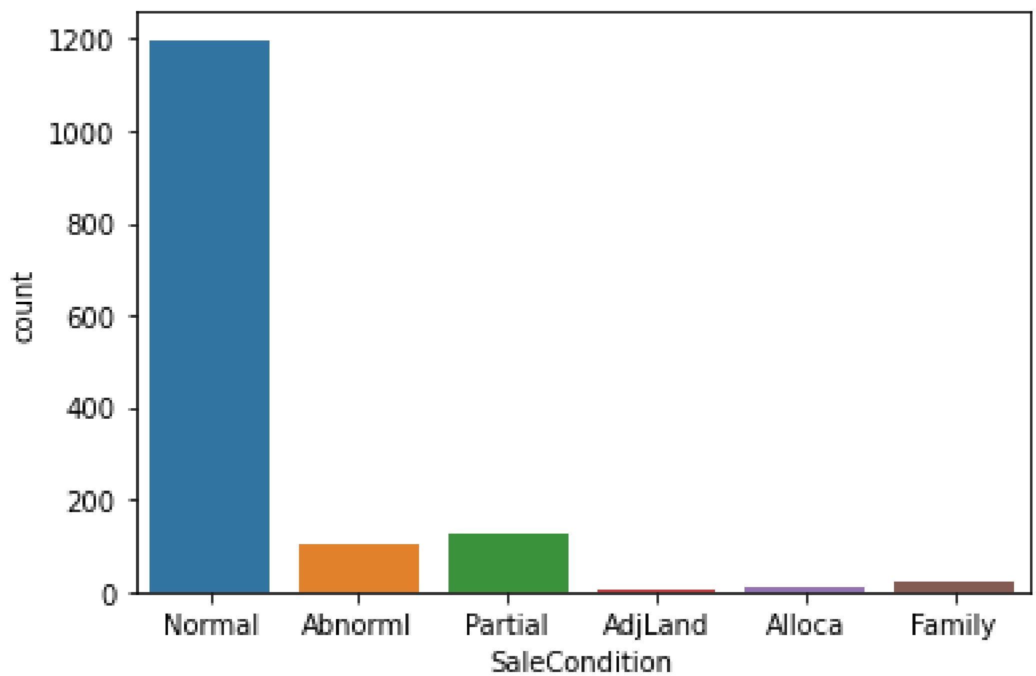

1400
1200
1000
800
600
400
200
0
count
Y
N
P
PavedDrive


1400
1200
1000
800
600
400
200
0
count
NoA
Ex
Fa
Gd
PoolQC


count
1200
1000
800
600
400
200
0
NoA
MnPrv
GdWo
GdPrv
MnWw
Fence


count
1400
1200
1000
800
600
400
200
0
NoA
Shed
Gar2
Othr
TenC
MiscFeature


1200
1000
800
600
400
200
0
count
WD
New
COD
ConLD
ConLI
CWD
ConLw
Con
Oth
SaleType


1200
1000
800
600
400
200
0
count
Normal
Abnorml
Partial
AdjLand
Alloca
Family
SaleCondition

3.1.1 Conclusión

Existen varias variables que no tienen mucho sentido, por ejemplo "street"en la que casi el total de las casas pertencen a la categoría "Pave", o la variable "Utilities" sucede lo mismo, todas las casas son "AllPub", y también con "Condition2", "RoofMatl", "Heating", "Functional" y "PoolQC". Seguramente el método LASSO hará que estas variables tengan coeficiente cero, pues realmente no describen/modelan el precio de venta.

[88]:
```
to_drop0 = ['Street', 'Utilities', 'Condition2', 'RoofMatl', 'Heating',␣
 ↪'Functional', 'PoolQC', ]
trainObj=trainObj.drop(trainObj[to_drop0], axis=1).copy()
```

[89]:
```
#Variables explicativas categóricas
XtrainCat = trainObj.copy()
#también lo hacemos para test
XtestCat = test[XtrainCat.columns]
XtrainCat
```

[89]:
```
     MSZoning Alley LotShape LandContour LotConfig LandSlope Neighborhood  \
Id
1          RL   NoA      Reg         Lvl    Inside       Gtl      CollgCr
2          RL   NoA      Reg         Lvl       FR2       Gtl      Veenker
3          RL   NoA      IR1         Lvl    Inside       Gtl      CollgCr
4          RL   NoA      IR1         Lvl    Corner       Gtl      Crawfor
5          RL   NoA      IR1         Lvl       FR2       Gtl      NoRidge
...       ...   ...      ...         ...       ...       ...          ...
1456       RL   NoA      Reg         Lvl    Inside       Gtl      Gilbert
1457       RL   NoA      Reg         Lvl    Inside       Gtl       NWAmes
1458       RL   NoA      Reg         Lvl    Inside       Gtl      Crawfor
1459       RL   NoA      Reg         Lvl    Inside       Gtl        NAmes
1460       RL   NoA      Reg         Lvl    Inside       Gtl      Edwards

     Condition1 BldgType HouseStyle  ... FireplaceQu GarageType GarageFinish  \
Id                                   ...
1          Norm     1Fam     2Story  ...         NoA     Attchd          RFn
2         Feedr     1Fam     1Story  ...          TA     Attchd          RFn
3          Norm     1Fam     2Story  ...          TA     Attchd          RFn
4          Norm     1Fam     2Story  ...          Gd     Detchd          Unf
5          Norm     1Fam     2Story  ...          TA     Attchd          RFn
...         ...      ...        ...  ...         ...        ...          ...
1456       Norm     1Fam     2Story  ...          TA     Attchd          RFn
1457       Norm     1Fam     1Story  ...          TA     Attchd          Unf
1458       Norm     1Fam     2Story  ...          Gd     Attchd          RFn
1459       Norm     1Fam     1Story  ...         NoA     Attchd          Unf
1460       Norm     1Fam     1Story  ...         NoA     Attchd          Fin

     GarageQual GarageCond PavedDrive  Fence MiscFeature SaleType  \
Id
```

```
1              TA          TA          Y    NoA          NoA        WD
2              TA          TA          Y    NoA          NoA        WD
3              TA          TA          Y    NoA          NoA        WD
4              TA          TA          Y    NoA          NoA        WD
5              TA          TA          Y    NoA          NoA        WD
...           ...         ...        ...    ...          ...       ...
1456           TA          TA          Y    NoA          NoA        WD
1457           TA          TA          Y  MnPrv          NoA        WD
1458           TA          TA          Y  GdPrv         Shed        WD
1459           TA          TA          Y    NoA          NoA        WD
1460           TA          TA          Y    NoA          NoA        WD

     SaleCondition
Id
1           Normal
2           Normal
3           Normal
4          Abnorml
5           Normal
...            ...
1456        Normal
1457        Normal
1458        Normal
1459        Normal
1460        Normal

[1460 rows x 36 columns]
```

3.2 Variables numéricas

Para ver la matríz de correlaciones primero seleccionamos las variables numéricas de la base de datos Train

[90]:
```
trainNum = train.select_dtypes(include=['int64' or 'float64']).copy()
trainNum
```

[90]:
```
      LotArea  OverallQual  OverallCond  YearBuilt  YearRemodAdd  BsmtFinSF1  \
Id
1        8450            7            5       2003          2003         706
2        9600            6            8       1976          1976         978
3       11250            7            5       2001          2002         486
4        9550            7            5       1915          1970         216
5       14260            8            5       2000          2000         655
...       ...          ...          ...        ...           ...         ...
1456     7917            6            5       1999          2000           0
1457    13175            6            6       1978          1988         790
1458     9042            7            9       1941          2006         275
```

```
1459    9717            5             6       1950          1996          49
1460    9937            5             6       1965          1965         830

      BsmtFinSF2  BsmtUnfSF  TotalBsmtSF  1stFlrSF  ...  WoodDeckSF  \
Id                                                  ...
1              0        150          856       856  ...           0
2              0        284         1262      1262  ...         298
3              0        434          920       920  ...           0
4              0        540          756       961  ...           0
5              0        490         1145      1145  ...         192
...          ...        ...          ...       ...  ...         ...
1456           0        953          953       953  ...           0
1457         163        589         1542      2073  ...         349
1458           0        877         1152      1188  ...           0
1459        1029          0         1078      1078  ...         366
1460         290        136         1256      1256  ...         736

      OpenPorchSF  EnclosedPorch  3SsnPorch  ScreenPorch  PoolArea  MiscVal  \
Id
1              61              0          0            0         0        0
2               0              0          0            0         0        0
3              42              0          0            0         0        0
4              35            272          0            0         0        0
5              84              0          0            0         0        0
...           ...            ...        ...          ...       ...      ...
1456           40              0          0            0         0        0
1457            0              0          0            0         0        0
1458           60              0          0            0         0     2500
1459            0            112          0            0         0        0
1460           68              0          0            0         0        0

      MoSold  YrSold  SalePrice
Id
1          2    2008     208500
2          5    2007     181500
3          9    2008     223500
4          2    2006     140000
5         12    2008     250000
...      ...     ...        ...
1456       8    2007     175000
1457       2    2010     210000
1458       5    2010     266500
1459       4    2010     142125
1460       6    2008     147500

[1460 rows x 33 columns]
```

Ahora busquemos variables que no estén correlacionadas con la variable respuesta.

[91]:
```
MC=trainNum.corr()
SaleCorr=abs(MC.loc[:,"SalePrice"])
aux=SaleCorr.sort_values(ascending=False)
to_drop=aux.index[aux<0.1]
to_drop
```

[91]:
```
Index(['PoolArea', 'OverallCond', 'MoSold', '3SsnPorch', 'YrSold',
       'LowQualFinSF', 'MiscVal', 'BsmtHalfBath', 'BsmtFinSF2'],
      dtype='object')
```

Las variables anteriores tienen una correlación menor a 0.1, no tiene caso que estén en el modelo, por lo tanto las quitamos de la matriz.

[92]:
```
#Generamos la matriz de X de variables numéricas quitando la var resp
XtrainNum = trainNum.iloc[:,0:32].copy()
XtrainNum=XtrainNum.drop(XtrainNum[to_drop], axis=1).copy()
XtrainNum
```

[92]:
```
      LotArea  OverallQual  YearBuilt  YearRemodAdd  BsmtFinSF1  BsmtUnfSF  \
Id
1        8450            7       2003          2003         706        150
2        9600            6       1976          1976         978        284
3       11250            7       2001          2002         486        434
4        9550            7       1915          1970         216        540
5       14260            8       2000          2000         655        490
...       ...          ...        ...           ...         ...        ...
1456     7917            6       1999          2000           0        953
1457    13175            6       1978          1988         790        589
1458     9042            7       1941          2006         275        877
1459     9717            5       1950          1996          49          0
1460     9937            5       1965          1965         830        136

      TotalBsmtSF  1stFlrSF  2ndFlrSF  GrLivArea  ...  BedroomAbvGr  \
Id                                                ...
1             856       856       854       1710  ...             3
2            1262      1262         0       1262  ...             3
3             920       920       866       1786  ...             3
4             756       961       756       1717  ...             3
5            1145      1145      1053       2198  ...             4
...           ...       ...       ...        ...  ...           ...
1456          953       953       694       1647  ...             3
1457         1542      2073         0       2073  ...             3
1458         1152      1188      1152       2340  ...             4
1459         1078      1078         0       1078  ...             2
1460         1256      1256         0       1256  ...             3
```

```
      KitchenAbvGr  TotRmsAbvGrd  Fireplaces  GarageCars  GarageArea  \
Id
1                1             8           0           2         548
2                1             6           1           2         460
3                1             6           1           2         608
4                1             7           1           3         642
5                1             9           1           3         836
...            ...           ...         ...         ...         ...
1456             1             7           1           2         460
1457             1             7           2           2         500
1458             1             9           2           1         252
1459             1             5           0           1         240
1460             1             6           0           1         276

      WoodDeckSF  OpenPorchSF  EnclosedPorch  ScreenPorch
Id
1              0           61              0            0
2            298            0              0            0
3              0           42              0            0
4              0           35            272            0
5            192           84              0            0
...          ...          ...            ...          ...
1456           0           40              0            0
1457         349            0              0            0
1458           0           60              0            0
1459         366            0            112            0
1460         736           68              0            0

[1460 rows x 23 columns]
```

[93]:
```
print(XtrainNum.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1460 entries, 1 to 1460
Data columns (total 23 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   LotArea        1460 non-null   int64
 1   OverallQual    1460 non-null   int64
 2   YearBuilt      1460 non-null   int64
 3   YearRemodAdd   1460 non-null   int64
 4   BsmtFinSF1     1460 non-null   int64
 5   BsmtUnfSF      1460 non-null   int64
 6   TotalBsmtSF    1460 non-null   int64
 7   1stFlrSF       1460 non-null   int64
 8   2ndFlrSF       1460 non-null   int64
 9   GrLivArea      1460 non-null   int64
```

```
 10  BsmtFullBath   1460 non-null   int64
 11  FullBath       1460 non-null   int64
 12  HalfBath       1460 non-null   int64
 13  BedroomAbvGr   1460 non-null   int64
 14  KitchenAbvGr   1460 non-null   int64
 15  TotRmsAbvGrd   1460 non-null   int64
 16  Fireplaces     1460 non-null   int64
 17  GarageCars     1460 non-null   int64
 18  GarageArea     1460 non-null   int64
 19  WoodDeckSF     1460 non-null   int64
 20  OpenPorchSF    1460 non-null   int64
 21  EnclosedPorch  1460 non-null   int64
 22  ScreenPorch    1460 non-null   int64
dtypes: int64(23)
memory usage: 273.8 KB
None
```

A continuación, vamos a explorar la correlación entre las variables explicativas y filtramos aquellas que tienen una correlación (en valor absoluto) mayor que 0.6

[94]:
```
import numpy as np
corr_matrix = XtrainNum.corr().abs()
#La mitad de arriba de la matriz de correlaciones en valores absolutos
upper = corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).astype(np.
 ↪bool))
#Lista de variables con correlación mayor a 0.6
to_drop2 = [column for column in upper.columns if any(upper[column] > 0.6)]
to_drop2
#
```

[94]:
```
['1stFlrSF',
 'GrLivArea',
 'BsmtFullBath',
 'FullBath',
 'HalfBath',
 'TotRmsAbvGrd',
 'GarageCars',
 'GarageArea']
```

Estas variables tienen una correlación mayor a 0.6 con alguna otra de las variables explicativas, por ejemplo, '1stFlrSF' tiene una correlación de 0.819530 con 'TotalBsmtSF', basta con dejar solo una de estas dos variables y el algoritmo anterior seleccionó a '1stFlrSF', 'GrLivArea' tiene una correlación de 0.687501 con '2ndFlrSF', basta con dejar solo una de estas dos variables y así sucesivamente.

[95]:
```
corr_matrix.iloc[:,[7,9]]
```

[95]:
```
               1stFlrSF  GrLivArea
LotArea        0.299475   0.263116
```

```
OverallQual    0.476224   0.593007
YearBuilt      0.281986   0.199010
YearRemodAdd   0.240379   0.287389
BsmtFinSF1     0.445863   0.208171
BsmtUnfSF      0.317987   0.240257
TotalBsmtSF    0.819530   0.454868
1stFlrSF       1.000000   0.566024
2ndFlrSF       0.202646   0.687501
GrLivArea      0.566024   1.000000
BsmtFullBath   0.244671   0.034836
FullBath       0.380637   0.630012
HalfBath       0.119916   0.415772
BedroomAbvGr   0.127401   0.521270
KitchenAbvGr   0.068101   0.100063
TotRmsAbvGrd   0.409516   0.825489
Fireplaces     0.410531   0.461679
GarageCars     0.439317   0.467247
GarageArea     0.489782   0.468997
WoodDeckSF     0.235459   0.247433
OpenPorchSF    0.211671   0.330224
EnclosedPorch  0.065292   0.009113
ScreenPorch    0.088758   0.101510
```

[96]:
```
XtrainNum=XtrainNum.drop(XtrainNum[to_drop2], axis=1).copy()
XtrainNum
```

[96]:
```
      LotArea  OverallQual  YearBuilt  YearRemodAdd  BsmtFinSF1  BsmtUnfSF  \
Id
1        8450            7       2003          2003         706        150
2        9600            6       1976          1976         978        284
3       11250            7       2001          2002         486        434
4        9550            7       1915          1970         216        540
5       14260            8       2000          2000         655        490
...       ...          ...        ...           ...         ...        ...
1456     7917            6       1999          2000           0        953
1457    13175            6       1978          1988         790        589
1458     9042            7       1941          2006         275        877
1459     9717            5       1950          1996          49          0
1460     9937            5       1965          1965         830        136

      TotalBsmtSF  2ndFlrSF  BedroomAbvGr  KitchenAbvGr  Fireplaces  \
Id
1             856       854             3             1           0
2            1262         0             3             1           1
3             920       866             3             1           1
4             756       756             3             1           1
5            1145      1053             4             1           1
```

```
...           ...      ...           ...           ...         ...
1456          953      694             3             1           1
1457         1542        0             3             1           2
1458         1152     1152             4             1           2
1459         1078        0             2             1           0
1460         1256        0             3             1           0

      WoodDeckSF  OpenPorchSF  EnclosedPorch  ScreenPorch
Id
1              0           61              0            0
2            298            0              0            0
3              0           42              0            0
4              0           35            272            0
5            192           84              0            0
...          ...          ...            ...          ...
1456           0           40              0            0
1457         349            0              0            0
1458           0           60              0            0
1459         366            0            112            0
1460         736           68              0            0

[1460 rows x 15 columns]
```

Y únicamente nos quedamos con 15 variables numéricas para inlcuir en el modelo.

[97]:
```
#Preparamos la base de Test
XtestNum = test[XtrainNum.columns]
```

3.3 Conclusión

La matriz de variables explicativas queda de la siguietne forma con 58 variables:

[98]:
```
Xtrain = pd.merge(XtrainNum, XtrainCat, on='Id')
#lo mismo para test
Xtest = pd.merge(XtestNum, XtestCat, on='Id')
Xtrain.head()
```

[98]:
```
    LotArea  OverallQual  YearBuilt  YearRemodAdd  BsmtFinSF1  BsmtUnfSF  \
Id
1      8450            7       2003          2003         706        150
2      9600            6       1976          1976         978        284
3     11250            7       2001          2002         486        434
4      9550            7       1915          1970         216        540
5     14260            8       2000          2000         655        490

    TotalBsmtSF  2ndFlrSF  BedroomAbvGr  KitchenAbvGr  ...  FireplaceQu  \
Id                                                     ...
```

```
1          856       854            3             1  ...          NoA
2         1262         0            3             1  ...           TA
3          920       866            3             1  ...           TA
4          756       756            3             1  ...           Gd
5         1145      1053            4             1  ...           TA

    GarageType  GarageFinish  GarageQual  GarageCond PavedDrive Fence  \
Id
1       Attchd           RFn          TA          TA          Y   NoA
2       Attchd           RFn          TA          TA          Y   NoA
3       Attchd           RFn          TA          TA          Y   NoA
4       Detchd           Unf          TA          TA          Y   NoA
5       Attchd           RFn          TA          TA          Y   NoA

   MiscFeature SaleType SaleCondition
Id
1          NoA       WD        Normal
2          NoA       WD        Normal
3          NoA       WD        Normal
4          NoA       WD       Abnorml
5          NoA       WD        Normal

[5 rows x 51 columns]
```

4 Modelo LASSO

Al momento de trabajar las variables explicativas con la que se va a modelar la variable "PriceSale" conviene trabajar juntas las bases de datos de train y de test, para generar las mismas variables doomies.

[99]:
```
Xjoin = pd.concat([Xtrain, Xtest])
Xjoin.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2919 entries, 1 to 2919
Data columns (total 51 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   LotArea        2919 non-null   int64
 1   OverallQual    2919 non-null   int64
 2   YearBuilt      2919 non-null   int64
 3   YearRemodAdd   2919 non-null   int64
 4   BsmtFinSF1     2919 non-null   float64
 5   BsmtUnfSF      2919 non-null   float64
 6   TotalBsmtSF    2919 non-null   float64
 7   2ndFlrSF       2919 non-null   int64
 8   BedroomAbvGr   2919 non-null   int64
```

```
 9   KitchenAbvGr   2919 non-null   int64
 10  Fireplaces     2919 non-null   int64
 11  WoodDeckSF     2919 non-null   int64
 12  OpenPorchSF    2919 non-null   int64
 13  EnclosedPorch  2919 non-null   int64
 14  ScreenPorch    2919 non-null   int64
 15  MSZoning       2919 non-null   object
 16  Alley          2919 non-null   object
 17  LotShape       2919 non-null   object
 18  LandContour    2919 non-null   object
 19  LotConfig      2919 non-null   object
 20  LandSlope      2919 non-null   object
 21  Neighborhood   2919 non-null   object
 22  Condition1     2919 non-null   object
 23  BldgType       2919 non-null   object
 24  HouseStyle     2919 non-null   object
 25  RoofStyle      2919 non-null   object
 26  Exterior1st    2919 non-null   object
 27  Exterior2nd    2919 non-null   object
 28  MasVnrType     2919 non-null   object
 29  ExterQual      2919 non-null   object
 30  ExterCond      2919 non-null   object
 31  Foundation     2919 non-null   object
 32  BsmtQual       2919 non-null   object
 33  BsmtCond       2919 non-null   object
 34  BsmtExposure   2919 non-null   object
 35  BsmtFinType1   2919 non-null   object
 36  BsmtFinType2   2919 non-null   object
 37  HeatingQC      2919 non-null   object
 38  CentralAir     2919 non-null   object
 39  Electrical     2919 non-null   object
 40  KitchenQual    2919 non-null   object
 41  FireplaceQu    2919 non-null   object
 42  GarageType     2919 non-null   object
 43  GarageFinish   2919 non-null   object
 44  GarageQual     2919 non-null   object
 45  GarageCond     2919 non-null   object
 46  PavedDrive     2919 non-null   object
 47  Fence          2919 non-null   object
 48  MiscFeature    2919 non-null   object
 49  SaleType       2919 non-null   object
 50  SaleCondition  2919 non-null   object
dtypes: float64(3), int64(12), object(36)
memory usage: 1.2+ MB
```

[100]:
```
dummies = []
for i in Xjoin.columns:
```

```
    if (Xjoin[i].dtype=='object'):
        dummies.append(i)
dummies
```

[100]:
```
['MSZoning',
 'Alley',
 'LotShape',
 'LandContour',
 'LotConfig',
 'LandSlope',
 'Neighborhood',
 'Condition1',
 'BldgType',
 'HouseStyle',
 'RoofStyle',
 'Exterior1st',
 'Exterior2nd',
 'MasVnrType',
 'ExterQual',
 'ExterCond',
 'Foundation',
 'BsmtQual',
 'BsmtCond',
 'BsmtExposure',
 'BsmtFinType1',
 'BsmtFinType2',
 'HeatingQC',
 'CentralAir',
 'Electrical',
 'KitchenQual',
 'FireplaceQu',
 'GarageType',
 'GarageFinish',
 'GarageQual',
 'GarageCond',
 'PavedDrive',
 'Fence',
 'MiscFeature',
 'SaleType',
 'SaleCondition']
```

[101]:
```
status =  pd.get_dummies(Xjoin[dummies],drop_first=True) ## one hot encoding on␣
 ↪all variables
Xjoin = pd.concat([Xjoin,status],axis=1)
Xjoin.drop(dummies,axis=1,inplace=True)
Xjoin.head()
```

[101]:
```
    LotArea  OverallQual  YearBuilt  YearRemodAdd  BsmtFinSF1  BsmtUnfSF  \
Id
1      8450            7       2003          2003       706.0      150.0
2      9600            6       1976          1976       978.0      284.0
3     11250            7       2001          2002       486.0      434.0
4      9550            7       1915          1970       216.0      540.0
5     14260            8       2000          2000       655.0      490.0

    TotalBsmtSF  2ndFlrSF  BedroomAbvGr  KitchenAbvGr  ...  SaleType_ConLI  \
Id                                                      ...
1         856.0       854             3             1  ...               0
2        1262.0         0             3             1  ...               0
3         920.0       866             3             1  ...               0
4         756.0       756             3             1  ...               0
5        1145.0      1053             4             1  ...               0

    SaleType_ConLw  SaleType_New  SaleType_Oth  SaleType_WD  \
Id
1                0             0             0            1
2                0             0             0            1
3                0             0             0            1
4                0             0             0            1
5                0             0             0            1

    SaleCondition_AdjLand  SaleCondition_Alloca  SaleCondition_Family  \
Id
1                       0                     0                     0
2                       0                     0                     0
3                       0                     0                     0
4                       0                     0                     0
5                       0                     0                     0

    SaleCondition_Normal  SaleCondition_Partial
Id
1                      1                      0
2                      1                      0
3                      1                      0
4                      0                      0
5                      1                      0

[5 rows x 215 columns]
```

Ahora separamos la matriz Xjoin nuevamente:

[102]:
```
Xtrain = Xjoin.iloc[0:1460,]
Xtrain
```

[102]:

```
      LotArea  OverallQual  YearBuilt  YearRemodAdd  BsmtFinSF1  BsmtUnfSF  \
Id
1        8450            7       2003          2003       706.0      150.0
2        9600            6       1976          1976       978.0      284.0
3       11250            7       2001          2002       486.0      434.0
4        9550            7       1915          1970       216.0      540.0
5       14260            8       2000          2000       655.0      490.0
...       ...          ...        ...           ...         ...        ...
1456     7917            6       1999          2000         0.0      953.0
1457    13175            6       1978          1988       790.0      589.0
1458     9042            7       1941          2006       275.0      877.0
1459     9717            5       1950          1996        49.0        0.0
1460     9937            5       1965          1965       830.0      136.0

      TotalBsmtSF  2ndFlrSF  BedroomAbvGr  KitchenAbvGr  ...  SaleType_ConLI  \
Id                                                       ...
1           856.0       854             3             1  ...               0
2          1262.0         0             3             1  ...               0
3           920.0       866             3             1  ...               0
4           756.0       756             3             1  ...               0
5          1145.0      1053             4             1  ...               0
...           ...       ...           ...           ...  ...             ...
1456        953.0       694             3             1  ...               0
1457       1542.0         0             3             1  ...               0
1458       1152.0      1152             4             1  ...               0
1459       1078.0         0             2             1  ...               0
1460       1256.0         0             3             1  ...               0

      SaleType_ConLw  SaleType_New  SaleType_Oth  SaleType_WD  \
Id
1                  0             0             0            1
2                  0             0             0            1
3                  0             0             0            1
4                  0             0             0            1
5                  0             0             0            1
...              ...           ...           ...          ...
1456               0             0             0            1
1457               0             0             0            1
1458               0             0             0            1
1459               0             0             0            1
1460               0             0             0            1

      SaleCondition_AdjLand  SaleCondition_Alloca  SaleCondition_Family  \
Id
1                         0                     0                     0
2                         0                     0                     0
3                         0                     0                     0
```

```
4                         0                      0                      0
5                         0                      0                      0
...                     ...                    ...                    ...
1456                      0                      0                      0
1457                      0                      0                      0
1458                      0                      0                      0
1459                      0                      0                      0
1460                      0                      0                      0

      SaleCondition_Normal  SaleCondition_Partial
Id
1                        1                      0
2                        1                      0
3                        1                      0
4                        0                      0
5                        1                      0
...                    ...                    ...
1456                     1                      0
1457                     1                      0
1458                     1                      0
1459                     1                      0
1460                     1                      0

[1460 rows x 215 columns]
```

[103]:
```python
Xtest = Xjoin.iloc[1460:2919,]
Xtest
```

[103]:
```
      LotArea  OverallQual  YearBuilt  YearRemodAdd  BsmtFinSF1  BsmtUnfSF  \
Id
1461    11622            5       1961          1961       468.0      270.0
1462    14267            6       1958          1958       923.0      406.0
1463    13830            5       1997          1998       791.0      137.0
1464     9978            6       1998          1998       602.0      324.0
1465     5005            8       1992          1992       263.0     1017.0
...       ...          ...        ...           ...         ...        ...
2915     1936            4       1970          1970         0.0      546.0
2916     1894            4       1970          1970       252.0      294.0
2917    20000            5       1960          1996      1224.0        0.0
2918    10441            5       1992          1992       337.0      575.0
2919     9627            7       1993          1994       758.0      238.0

      TotalBsmtSF  2ndFlrSF  BedroomAbvGr  KitchenAbvGr  ...  SaleType_ConLI  \
Id                                                       ...
1461        882.0         0             2             1  ...               0
1462       1329.0         0             3             1  ...               0
1463        928.0       701             3             1  ...               0
```

```
1464     926.0     678           3           1  ...              0
1465    1280.0       0           2           1  ...              0
...        ...     ...         ...         ...  ...            ...
2915     546.0     546           3           1  ...              0
2916     546.0     546           3           1  ...              0
2917    1224.0       0           4           1  ...              0
2918     912.0       0           3           1  ...              0
2919     996.0    1004           3           1  ...              0

      SaleType_ConLw  SaleType_New  SaleType_Oth  SaleType_WD  \
Id
1461               0             0             0            1
1462               0             0             0            1
1463               0             0             0            1
1464               0             0             0            1
1465               0             0             0            1
...              ...           ...           ...          ...
2915               0             0             0            1
2916               0             0             0            1
2917               0             0             0            1
2918               0             0             0            1
2919               0             0             0            1

      SaleCondition_AdjLand  SaleCondition_Alloca  SaleCondition_Family  \
Id
1461                      0                     0                     0
1462                      0                     0                     0
1463                      0                     0                     0
1464                      0                     0                     0
1465                      0                     0                     0
...                     ...                   ...                   ...
2915                      0                     0                     0
2916                      0                     0                     0
2917                      0                     0                     0
2918                      0                     0                     0
2919                      0                     0                     0

      SaleCondition_Normal  SaleCondition_Partial
Id
1461                     1                      0
1462                     1                      0
1463                     1                      0
1464                     1                      0
1465                     1                      0
...                    ...                    ...
2915                     1                      0
2916                     0                      0
```

```
2917                        0                        0
2918                        1                        0
2919                        1                        0

[1459 rows x 215 columns]
```

[104]:
```
ytrain=train.iloc[:,79]
ytrain.head()
```

[104]:
```
Id
1     208500
2     181500
3     223500
4     140000
5     250000
Name: SalePrice, dtype: int64
```

[105]:
```
from sklearn.linear_model import Lasso
alpha = 0.0002

lasso = Lasso(alpha=alpha)

lasso.fit(Xtrain, ytrain)
#lasso.coef_
```

```
C:\Users\Brendis\anaconda3\lib\site-
packages\sklearn\linear_model\_coordinate_descent.py:529: ConvergenceWarning:
Objective did not converge. You might want to increase the number of iterations.
Duality gap: 539010529782.86694, tolerance: 920791133.4609977
  model = cd_fast.enet_coordinate_descent(
```

[105]:
```
Lasso(alpha=0.0002)
```

[106]:
```
lasso_df=pd.DataFrame()
lasso_df['Features'] = Xtrain.columns
lasso_df['Coefficients']=lasso.coef_
lasso_df['ABS Coefficients'] = abs(lasso.coef_)
lasso_df.sort_values(by=['ABS Coefficients'],ascending=False,inplace = True)
lasso_df
```

[106]:
```
             Features  Coefficients  ABS Coefficients
184     GarageQual_Po -1.950691e+05      1.950691e+05
181     GarageQual_Fa -1.701431e+05      1.701431e+05
190     GarageCond_TA  1.582615e+05      1.582615e+05
187     GarageCond_Gd  1.575977e+05      1.575977e+05
185     GarageQual_TA -1.569200e+05      1.569200e+05
..                ...           ...               ...
```

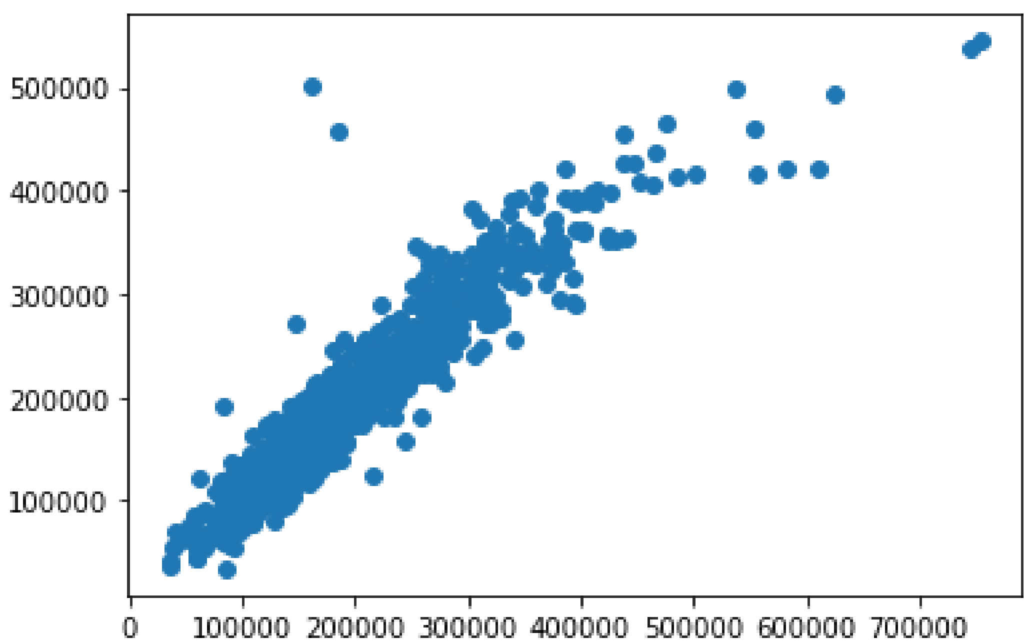
```
4          BsmtFinSF1  3.454576e+00      3.454576e+00
192       PavedDrive_Y -3.393544e+00      3.393544e+00
0              LotArea  6.964409e-01      6.964409e-01
5            BsmtUnfSF -2.356912e-01      2.356912e-01
178  GarageFinish_NoA  1.439774e-08      1.439774e-08

[215 rows x 3 columns]
```

[107]:
```
ygorro = lasso.predict(Xtrain)
ygorro
```

[107]:
```
array([202008.24150912, 207726.4447252 , 218608.58357301, ...,
       285253.41807998, 158414.88687291, 137232.71578127])
```

[108]:
```
plt.scatter(ytrain,ygorro)
```

[108]: <matplotlib.collections.PathCollection at 0x277200eb670>



5 Probando con la base test

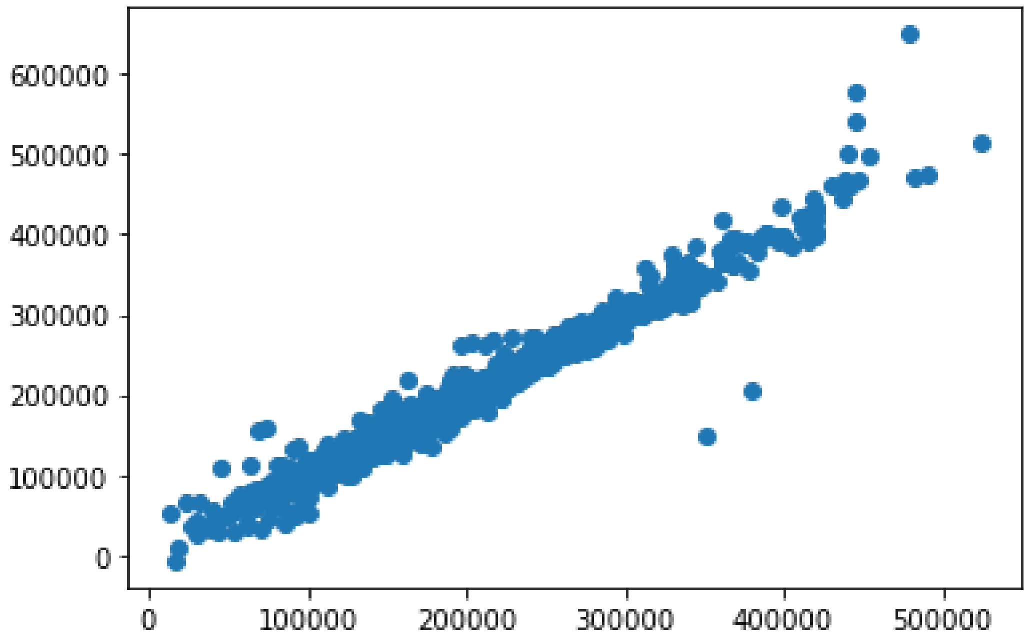
[109]:
```
ygorrotest = lasso.predict(Xtest)
ygorrotest
```

[109]:
```
array([117985.19656361, 132780.29437485, 182562.19910007, ...,
       168991.9958716 , 103551.60634709, 209912.42505859])
```

Acontinuación se grafica la estimación del precio de venta de la base de datos test usando el modelo lasso contra el precio estimado de la base de datos

[110]:
```
plt.scatter(ygorrotest,sample_submission['SalePrice'])
```

[110]:
```
<matplotlib.collections.PathCollection at 0x2771ff881f0>
```



5.1 Conclusión

Existe demasiada dispersión entre los precios estimados del modelo lasso obtenido en el presente análisis comparado con la estimación proporcionada en la base de datos ample_submission.

[111]:
```
##Guardar las predicciones en un csv , quoting=csv.QUOTE_ALL
import csv
i=1461
with open('My_sample_submission.csv', 'w', newline="") as myfile:
    wr = csv.writer(myfile)
    titulos = ['Id', 'SalePrice']
    wr.writerow(titulos)
    for word in ygorrotest:
        list=[i,ygorrotest[i-1461]]
        wr.writerow(list)
        i=i+1
```