In [103]:
```
import pandas as pd
from sklearn import preprocessing

df=pd.read_csv(r"C:\Users\91809\Desktop\train.csv")
```

In [104]:
```
df.head()
```

Out[104]:

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

5 rows × 81 columns

In [105]:
```
df=df.dropna(axis=1)
```

In [106]:
```
df.head()
```

Out[106]:

	Id	MSSubClass	MSZoning	LotArea	Street	LotShape	LandContour	Utilities	LotConfig	LandSlope	...	EnclosedPorch	3SsnPorch
0	1	60	RL	8450	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	0	0
1	2	20	RL	9600	Pave	Reg	Lvl	AllPub	FR2	Gtl	...	0	0
2	3	60	RL	11250	Pave	IR1	Lvl	AllPub	Inside	Gtl	...	0	0
3	4	70	RL	9550	Pave	IR1	Lvl	AllPub	Corner	Gtl	...	272	0
4	5	60	RL	14260	Pave	IR1	Lvl	AllPub	FR2	Gtl	...	0	0

5 rows × 62 columns

In [107]:
```
from sklearn.preprocessing import LabelEncoder
```

In [108]:
```
number=LabelEncoder()
df.keys()
df['LandContour']=number.fit_transform(df['LandContour'].astype('str'))
```

In [109]:
```
df.head()
```

Out[109]:

	Id	MSSubClass	MSZoning	LotArea	Street	LotShape	LandContour	Utilities	LotConfig	LandSlope	...	EnclosedPorch	3SsnPorch
0	1	60	RL	8450	Pave	Reg	3	AllPub	Inside	Gtl	...	0	0
1	2	20	RL	9600	Pave	Reg	3	AllPub	FR2	Gtl	...	0	0
2	3	60	RL	11250	Pave	IR1	3	AllPub	Inside	Gtl	...	0	0
3	4	70	RL	9550	Pave	IR1	3	AllPub	Corner	Gtl	...	272	0
4	5	60	RL	14260	Pave	IR1	3	AllPub	FR2	Gtl	...	0	0

5 rows × 62 columns

In [110]:
```
df['Utilities']=number.fit_transform(df['Utilities'].astype('str'))
df['LotConfig']=number.fit_transform(df['LotConfig'].astype('str'))
df['LandSlope']=number.fit_transform(df['LandSlope'].astype('str'))
df['Neighborhood']=number.fit_transform(df['Neighborhood'].astype('str'))
df['Condition1']=number.fit_transform(df['Condition1'].astype('str'))
df['Condition2']=number.fit_transform(df['Condition2'].astype('str'))
df['SaleType']=number.fit_transform(df['SaleType'].astype('str'))
df['SaleCondition']=number.fit_transform(df['SaleCondition'].astype('str'))
df['Street']=number.fit_transform(df['Street'].astype('str'))
```

In [111]:
```
df.head()
```

Out[111]:

	Id	MSSubClass	MSZoning	LotArea	Street	LotShape	LandContour	Utilities	LotConfig	LandSlope	...	EnclosedPorch	3SsnPorch
0	1	60	RL	8450	1	Reg	3	0	4	0	...	0	0
1	2	20	RL	9600	1	Reg	3	0	2	0	...	0	0
2	3	60	RL	11250	1	IR1	3	0	4	0	...	0	0
3	4	70	RL	9550	1	IR1	3	0	0	0	...	272	0
4	5	60	RL	14260	1	IR1	3	0	2	0	...	0	0

5 rows × 62 columns

In [112]:
```
df.shape
```

Out[112]:
```
(1460, 62)
```

In [113]:
```
df.dtypes
```

Out[113]:
```
Id              int64
MSSubClass      int64
MSZoning       object
LotArea         int64
Street          int32
                ...
MoSold          int64
YrSold          int64
SaleType        int32
SaleCondition   int32
SalePrice       int64
Length: 62, dtype: object
```

In [114]:
```
x=df.columns
```

In [115]:
```
x
```

Out[115]:
```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotArea', 'Street', 'LotShape',
       'LandContour', 'Utilities', 'LotConfig', 'LandSlope', 'Neighborhood',
       'Condition1', 'Condition2', 'BldgType', 'HouseStyle', 'OverallQual',
       'OverallCond', 'YearBuilt', 'YearRemodAdd', 'RoofStyle', 'RoofMatl',
       'Exterior1st', 'Exterior2nd', 'ExterQual', 'ExterCond', 'Foundation',
       'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF',
       'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath',
       'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual', 'TotRmsAbvGrd',
       'Functional', 'Fireplaces', 'GarageCars', 'GarageArea', 'PavedDrive',
       'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', '3SsnPorch',
       'ScreenPorch', 'PoolArea', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition', 'SalePrice'],
      dtype='object')
```

In [116]:
```
for y in x:
    df[y]=number.fit_transform(df[y].astype('str'))
```

In [117]:
```
df.head()
```

Out[117]:

	Id	MSSubClass	MSZoning	LotArea	Street	LotShape	LandContour	Utilities	LotConfig	LandSlope	...	EnclosedPorch	3SsnPorch
0	0	9	3	859	1	3	3	0	4	0	...	0	0
1	572	4	3	1030	1	3	3	0	2	0	...	0	0
2	653	9	3	161	1	0	3	0	4	0	...	0	0
3	794	10	3	1021	1	0	3	0	0	0	...	78	0
4	905	9	3	386	1	0	3	0	2	0	...	0	0

5 rows × 62 columns

In [118]:
```
df.dtypes
```

Out[118]:
```
Id              int32
MSSubClass      int32
MSZoning        int32
LotArea         int32
Street          int32
                ...
MoSold          int32
YrSold          int32
SaleType        int32
SaleCondition   int32
SalePrice       int32
Length: 62, dtype: object
```

In [119]:
```
df.describe()
```

Out[119]:

	Id	MSSubClass	MSZoning	LotArea	Street	LotShape	LandContour	Utilities	LotConfig	LandSlo
count	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000
mean	729.500000	6.214384	3.028767	565.210959	0.995890	1.942466	2.777397	0.000685	3.019178	0.062
std	421.610009	3.543318	0.632017	314.364525	0.063996	1.409156	0.707666	0.026171	1.622634	0.276
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
25%	364.750000	4.000000	3.000000	289.750000	1.000000	0.000000	3.000000	0.000000	2.000000	0.000
50%	729.500000	5.000000	3.000000	603.000000	1.000000	3.000000	3.000000	0.000000	4.000000	0.000
75%	1094.250000	9.000000	3.000000	839.000000	1.000000	3.000000	3.000000	0.000000	4.000000	0.000
max	1459.000000	14.000000	4.000000	1072.000000	1.000000	3.000000	3.000000	1.000000	4.000000	2.000

8 rows × 62 columns

In [120]:
```
from sklearn.linear_model import LinearRegression
```

In [121]:
```
from sklearn.model_selection import train_test_split
```

In [ ]:
```

```

In [126]:
```
y=df['SalePrice']
x= df.drop('SalePrice', 1)
```

In [127]:
```
y.head()
```

Out[127]:
```
0    343
1    270
2    373
3    125
4    425
Name: SalePrice, dtype: int32
```

In [128]:
```
y.shape
```

Out[128]:
```
(1460,)
```

In [129]:
```
x.shape
```

Out[129]:
```
(1460, 61)
```

In [130]:
```
df.shape
```

Out[130]:
```
(1460, 62)
```

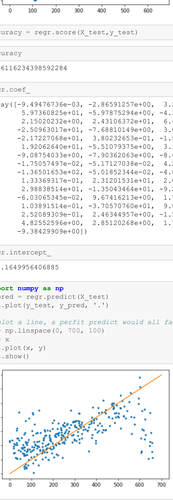
In [173]:
```
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.21)
```

In [174]:
```
regr = LinearRegression()
regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

```
0.36116234398592284
```

In [175]:
```
y_pred = regr.predict(X_test)
import matplotlib.pyplot as plt
```

In [176]:
```
plt.scatter(y_test,y_pred)
plt.show()
```



In [177]:
```
accuracy = regr.score(X_test,y_test)
```

In [178]:
```
accuracy
```

Out[178]:
```
0.36116234398592284
```

In [179]:
```
regr.coef_
```

Out[179]:
```
array([-9.49476736e-03, -2.86591257e+00,  3.24391608e+00, -4.13878929e-02,
        5.97360825e+01, -5.97875294e+00, -4.35526710e+00, -1.12455749e+02,
        2.15029232e+00,  2.43106372e+01,  6.44526756e-01, -7.72787856e+00,
       -2.50963017e+01, -7.68810149e+00,  3.03380397e+00,  7.93638514e+00,
       -2.17227068e+01,  3.80232653e-01, -1.50105375e-02, -1.70624621e+00,
        1.82062640e+01, -5.51079375e+00,  3.30063499e+00, -3.99952528e+01,
       -9.08754033e+00, -7.90362063e+00, -8.69064766e-02, -1.22411001e-01,
       -1.75057497e-02, -5.17127038e-02,  4.20144140e+01, -2.93858261e+00,
       -1.36501653e-02, -5.01852344e-02, -4.81146576e-02, -5.80277429e+00,
        1.33369317e-01,  2.31201531e+01,  2.06756999e+01,  6.29307821e+01,
        2.98838514e-01, -1.35043464e-01, -9.23393301e+01, -2.71317139e+01,
       -6.03065345e-02,  9.67416213e+00,  1.71734451e+01, -1.64718692e+01,
        1.03891514e-01, -3.70570760e-01,  8.05944086e-02,  8.93164393e-02,
        2.52089309e-01,  2.46344957e+00, -1.52769675e-01,  7.70522200e+00,
        4.82552596e+00,  2.85120268e+00,  1.76237111e+00,  4.93719644e-01,
       -9.38429093e+00])
```

In [180]:
```
regr.intercept_
```

Out[180]:
```
690.1649956406885
```

In [182]:
```
import numpy as np
y_pred = regr.predict(X_test)
plt.plot(y_test, y_pred, '.')

# plot a line, a perfit predict would all fall on this line
x = np.linspace(0, 700, 100)
y = x
plt.plot(x, y)
plt.show()
```


In [ ]:
```

```