به نام خدا

دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده برق و کامپیوتر

هوش مصنوعی پاییز ۹۸

پروژه صفر

پیش بینی قیمت خانه

نام و نام خانوادگی

علیرضا زارع نژاد اشکذری

شماره دانشجویی

۸۱۰۱۹۶۴۷۴

شرح مختصر پروژه

هدف از انجام این پروژه طراحی یک مدل برای پیش بینی قیمت خانه می باشد. این مدل به عنوان ورودی تعدادی از مشخصه های مربوط به یک خانه را دریافت کرده، و به عنوان خروجی قیمت تخمین زده شده برای خانه را برمی گرداند . همچنین آشنایی با کتابخانه های ، numpy ، pandas matplotlib نیز در پایتون از دیگر اهداف این پروژه می باشد.

داده های ورودی و بخش اول پروژه

مجموعه داده ها در یک فایل اکسل به نام houses.csv داده شده است.مطابق زیر ابتدا با استفاده از panda فایل ورودی را می‌خوانیم و خروجی آن را به عنوان یک دیتا فریم در data می ریزیم. فایل داده شده شامل ۱۱۳۴ سطر و ۱۱ ستون می‌باشد. هر کدام از ستون‌ها مختص یکی از خصیصه های خانه می‌باشد. با توجه به آنکه داده‌های می‌توانند طبقه ای و یا داده‌ای باشند و در این پروژه نیار به کار با داده‌های طبقه ای نیستیم لذا ستون مرتبط به آن را حذف می‌کنیم . همچنین مقدار بعضی از خانه‌های جدول NaN می باشد. راه برخورد با آن جایگزین کردن میانگین ستون مورد نظر به جای آن هاست. کد مرتبط در زیر آورده شده است.

```
# here we use head function to show by default the five first rows of data.as we see there is some NaN values
#and also columns LotConfig and Neighborhood are categorical so we omit them.


#dataset is given in houses.csv file.
def read_csv_file():
    return pd.read_csv("houses.csv");

#categorical data is not used in this project. so we just keep numerical one.
def delete_categorial_data(df):
    return df.select_dtypes(include='number')

# some of the value of columns are NaN.we replace these missing data with mean of columns
def fill_nan_data_with_mean_column(df):
    return df.fillna(df.mean()).dropna(axis=1, how='all')

data = read_csv_file()
data.head(15)
```

در زیر با استفاده از تابع head چند سطر ابتدایی ورودی را بدون تغییر خواهیم دید.

```
data = read_csv_file()
data.head(15)
```

Out[3]:

	Id	MSSubClass	LotArea	LotConfig	OverallQual	LotFrontage	Neighborhood	OverallCond	BedroomAbvGr	TotRmsAbvGrd	TotalBsmtSF	YearBuilt	S
0	1	60	8450	Inside	7	65.0	CollgCr	5	3	8	856	2003	
1	2	20	9600	FR2	6	80.0	Veenker	8	3	6	1262	1976	
2	3	60	11250	Inside	7	68.0	CollgCr	5	3	6	920	2001	
3	4	70	9550	Corner	7	60.0	Crawfor	5	3	7	756	1915	
4	5	60	14260	FR2	8	84.0	NoRidge	5	4	9	1145	2000	
5	6	50	14115	Inside	5	85.0	Mitchel	5	1	5	796	1993	
6	7	20	10084	Inside	8	75.0	Somerst	5	3	7	1686	2004	
7	8	60	10382	Corner	7	NaN	NWAmes	6	3	7	1107	1973	
8	9	50	6120	Inside	7	51.0	OldTown	5	2	8	952	1931	
9	11	20	11200	Inside	5	70.0	Sawyer	5	3	5	1040	1965	
10	12	60	11924	Inside	9	85.0	NridgHt	5	4	11	1175	2005	
11	13	20	12968	Inside	5	NaN	Sawyer	6	2	4	912	1962	
12	14	20	10652	Inside	7	91.0	CollgCr	5	3	7	1494	2006	
13	15	20	10920	Corner	6	NaN	NAmes	5	2	5	1253	1960	
14	16	45	6120	Corner	7	51.0	BrkSide	8	2	5	832	1929	

همچنین پس از صدا کردن توابع بالا موارد اصلاح شده در سطر های زیر قابل مشاهده است.

```
data = delete_categorial_data(data)
data = fill_nan_data_with_mean_column(data)
data.head(15)
```

Out[4]:

	Id	MSSubClass	LotArea	OverallQual	LotFrontage	OverallCond	BedroomAbvGr	TotRmsAbvGrd	TotalBsmtSF	YearBuilt	SalePrice
0	1	60	8450	7	65.00000	5	3	8	856	2003	208.5
1	2	20	9600	6	80.00000	8	3	6	1262	1976	181.5
2	3	60	11250	7	68.00000	5	3	6	920	2001	223.5
3	4	70	9550	7	60.00000	5	3	7	756	1915	140.0
4	5	60	14260	8	84.00000	5	4	9	1145	2000	250.0
5	6	50	14115	5	85.00000	5	1	5	796	1993	143.0
6	7	20	10084	8	75.00000	5	3	7	1686	2004	307.0
7	8	60	10382	7	68.40555	6	3	7	1107	1973	200.0
8	9	50	6120	7	51.00000	5	2	8	952	1931	129.9
9	11	20	11200	5	70.00000	5	3	5	1040	1965	129.5
10	12	60	11924	9	85.00000	5	4	11	1175	2005	345.0
11	13	20	12968	5	68.40555	6	2	4	912	1962	144.0
12	14	20	10652	7	91.00000	5	3	7	1494	2006	279.5
13	15	20	10920	6	68.40555	5	2	5	1253	1960	157.0
14	16	45	6120	7	51.00000	8	2	5	832	1929	132.0

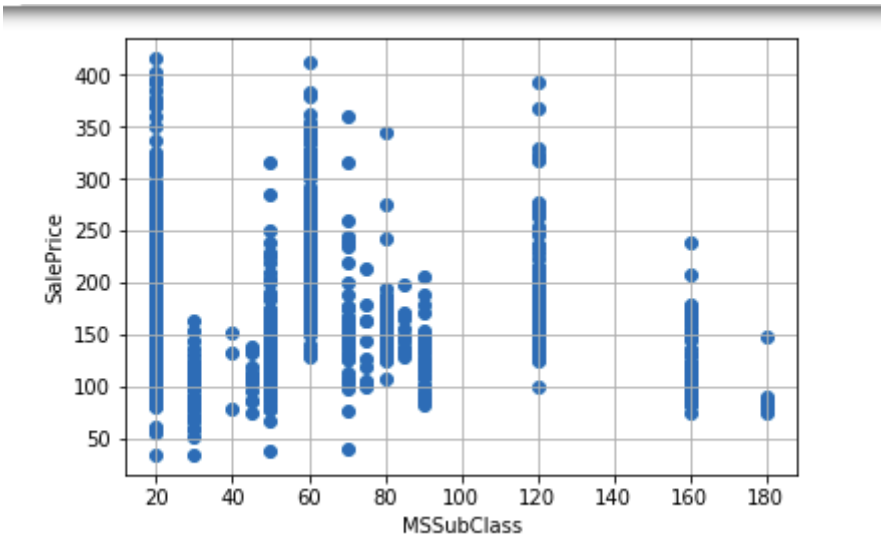
In [5]:

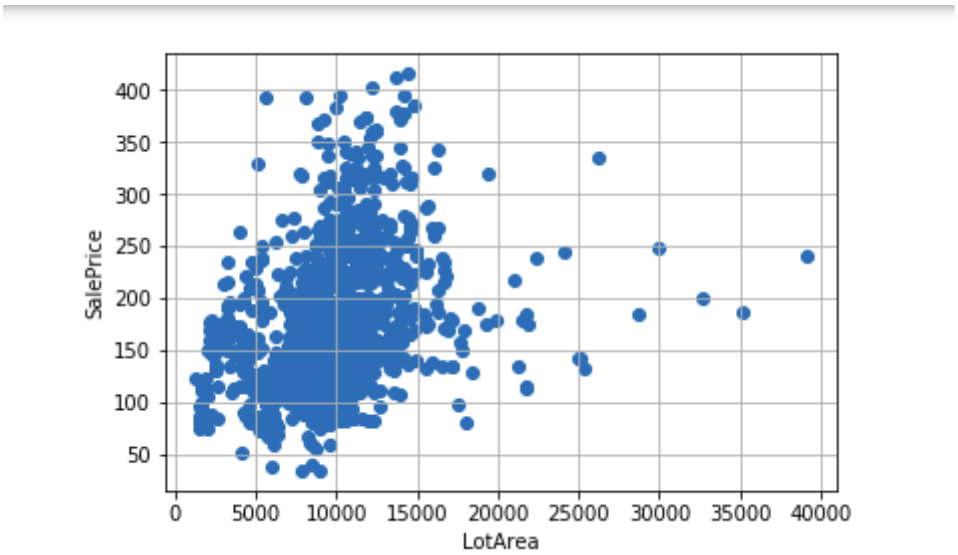
برای راحتی کار داده‌های قیمت خانه و ویژگی‌های مورد نظر را در price و X_train ذخیره می کنیم.

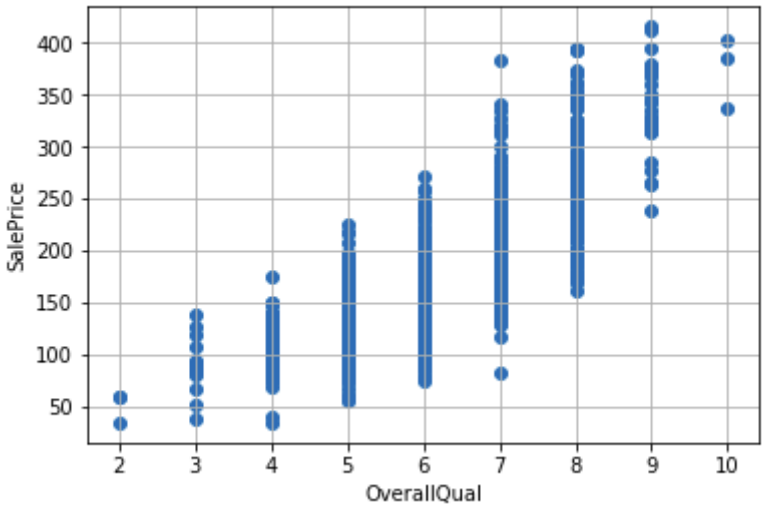
In [7]:

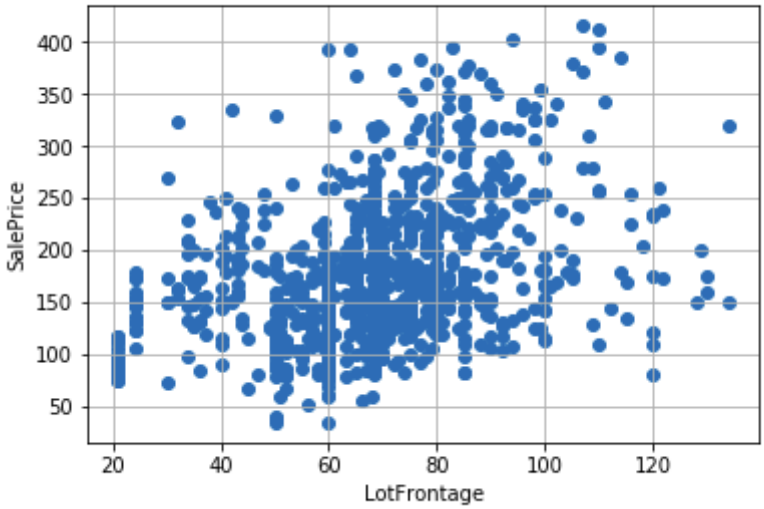
```
#now we seperate price(y) and features(X)
price = data['SalePrice']
X_train = data.drop(['Id', 'SalePrice'],1)

#X_tarin now has 1134 rows but 9 columns as features.
```

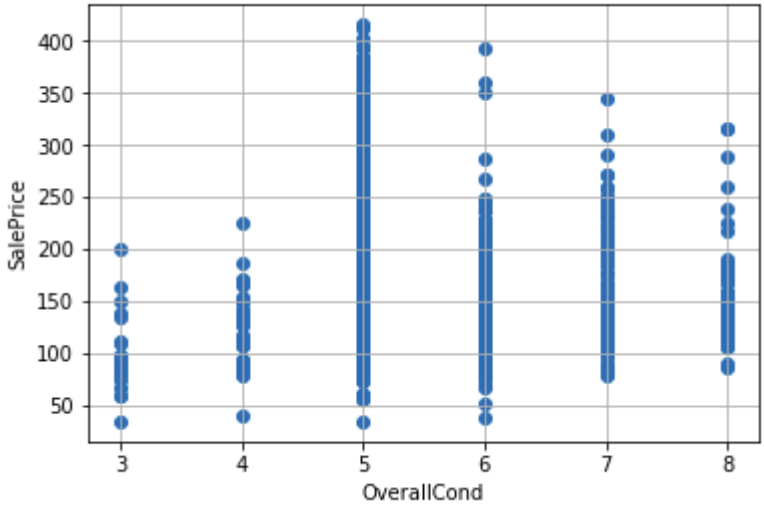
در انتها نمودار قیمت واقعی با توجه به هر کدام از ۹ ویژگی یعنی ستون‌های مرتبط را آورده‌ایم که با استفاده از scatter آورده شده است.

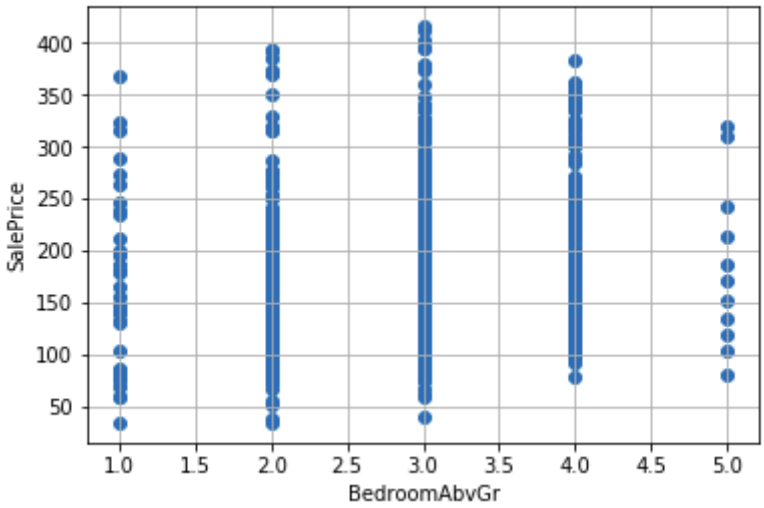


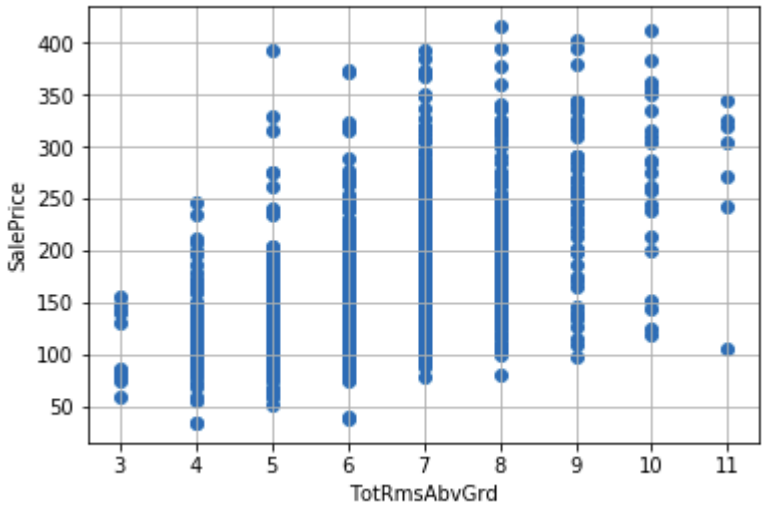


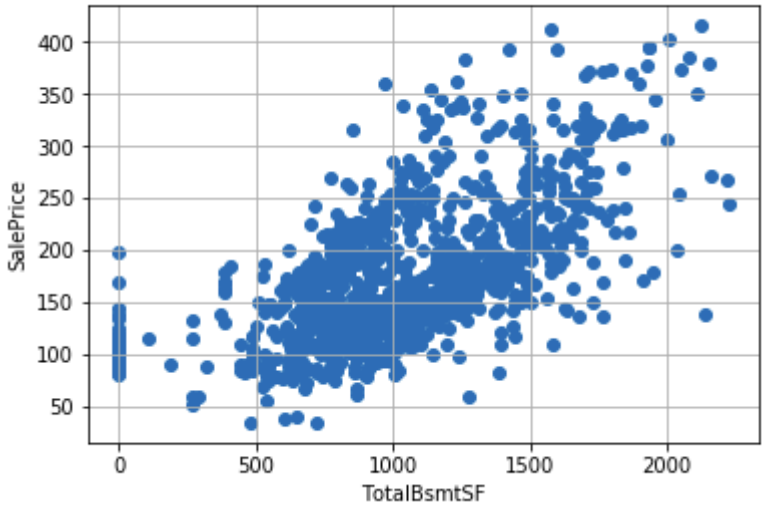

SalePrice
OverallQual
400
350
300
250
200
150
100
50
2
3
4
5
6
7
8
9
10

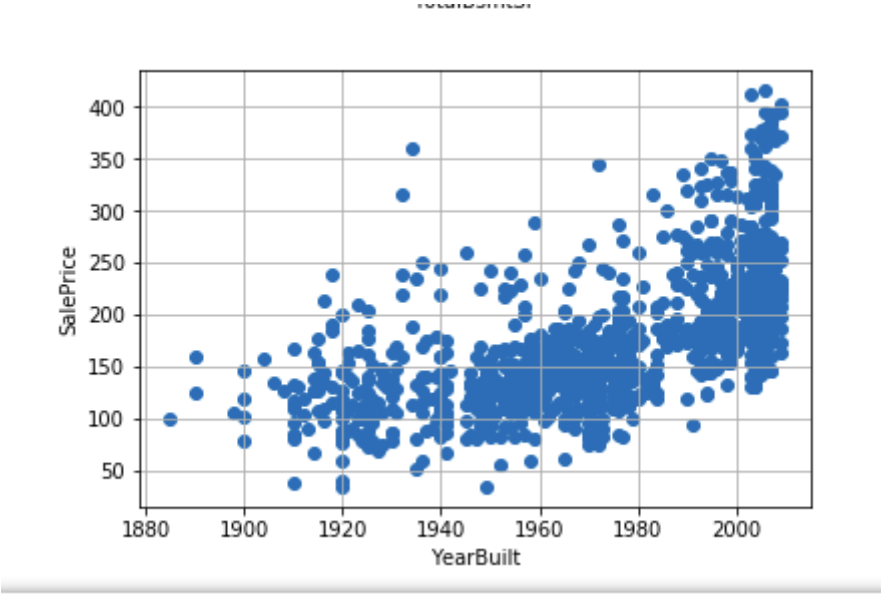

SalePrice
LotFrontage
400
350
300
250
200
150
100
50
20
40
60
80
100
120


SalePrice
OverallCond
400
350
300
250
200
150
100
50
3
4
5
6
7
8


SalePrice
BedroomAbvGr
400
350
300
250
200
150
100
50
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0


SalePrice
TotRmsAbvGrd
400
350
300
250
200
150
100
50
3
4
5
6
7
8
9
10
11


SalePrice
TotalBsmtSF
400
350
300
250
200
150
100
50
0
500
1000
1500
2000



بخش دوم تقریب گر خطی

شما در این مرحله باید به منظور تخمین قیمت خانه‌ها یک تخمینگر خطی طراحی کنید. فرمول توصیف کننده‌ی ساختار یک تخمینگر خطی در حالت کلی به شکل زیر است:

$$\hat{y} = wx + b$$

که در آن $\hat{y}$ خروجی مدل، w وزن ورودی (شیب خط)، و b مقدار bias (عرض از مبدأ) است. در حالت کلی ورودی مدل می‌تواند بیش‌از یک عدد باشد و در واقع یک بردار باشد، که در این حالت مقادیر w و b نیز برداری خواهند بود، اما در این پروژه به منظور سادگی فرض میکنیم که ورودی مدل صرفا یک عدد باشد. چیزی که دقت یک تخمینگر خطی را تعیین می‌کند، پارامترهای آن یعنی مقادیر w و b هستند. شما در این بخش باید ابتدا با نگاه کردن به نمودارهای تولید شده در بخش قبل تشخیص بدهید که رابطه‌ی کدامیک از خصیصه ها با قیمت خانه شباهت نسبتا خوبی را به یک رابطه‌ی خطی دارد. سپس سعی کنید مقادیر w و b را طوری حدس بزنید که خط رسم شده توسط آن‌ها به داده‌های واقعی نزدیک باشد. سپس به ازای همه‌ی داده‌های ورودی، نمودار قیمت واقعی در کنار قیمت تخمین زده شده را بر حسب مقدار خصیصه‌ی انتخاب شده رسم کنید.

برای اینکه عملکرد مدل خود را اندازه گیری بکنید، از معیار RMSE[۳] استفاده بکنید. تعریف این معیار به شکل زیر است:

$$L_{RMSE}(\hat{y}, y) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}$$

که در آن N تعداد داده‌ها، $\hat{y}_i$ پیش‌بینی مدل برای داده‌ی i ام، و y_i مقدار واقعی قیمت برای داده‌ی i ام است. شما

مقادیر b , w به صورت زیر حدس زده شده است و برای تعیین عملکرد مدل rmse با توجه به فرمول بالا محاسبه شد. هم چنین خروجی تقریبا ۳۸ بدست آمد که کمتر از ۹۰ می باشد.

In [2]:

```
def linear_calc(x,w = 40.16235654 , b=-68.35748451752235  ):
    return x*w + b
```

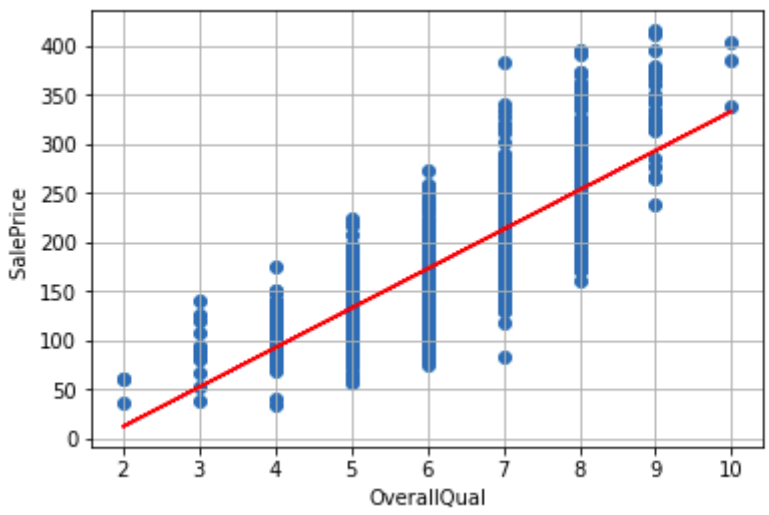
```
def RMSE_calc_partb(y,feature_x):
    y_pred = linear_calc(feature_x)
    N = len(y)
    square_sum = 0
    for i in range(N):
        temp = (y_pred[i] - y[i])**2
        square_sum += temp
    square_sum /= N
    return float(square_sum**0.5)

y = price
feature_x = data['OverallQual']
y = np.array(y)
feature_x  = np.array(feature_x )
```

In [11]:

```
rmse =  RMSE_calc_partb(y,feature_x)
print("RMSE part b:",rmse)
```

```
('RMSE part b:', 38.55949518377027)
```

حال نمودار قیمت واقعی را در کنار قیمت تخمین زده شده به صورت زیر خواهیم داشت.



بخش سوم آشنایی با کتابخانه‌ها

Numpy اصلی ترین و مهم ترین کتابخانه محاسبه عددی در پایتون است.
مهم ترین ویژگی این کتابخانه یک کلاس قدرتند نماینگر آرایه های N بعدی و مجموعه بسیار بزرگی از توابع پیچیده از جمله توابع broadcasting است. همچنین در این بخش با محیط jupyter نیزآشنا شدیم و در انتها یک تمرین مطق زیر از Numpy حل گردید.

In [5]:
```
v = np.random.rand(10)
u = np.empty(10)


# non-vectorized implementation:
for i in range(10):
    u[i] = v[i] ** 2
print('non-vectorized:', u)

# vectorized implementation:
# WRITE YOUR CODE HERE
u = np.power(v,2)

print('vectorized:', u)
```

```
('non-vectorized:', array([4.89733711e-01, 4.11056340e-01, 3.03158891e-07, 7.50296847e-02,
       2.33257642e-02, 9.26333160e-01, 3.90154320e-01, 2.68588733e-03,
       8.58836105e-01, 2.00016479e-01]))
('vectorized:', array([4.89733711e-01, 4.11056340e-01, 3.03158891e-07, 7.50296847e-02,
       2.33257642e-02, 9.26333160e-01, 3.90154320e-01, 2.68588733e-03,
       8.58836105e-01, 2.00016479e-01]))
```

In [7]:
```
a = np.random.rand(10)
b = np.random.rand(10)

# non-vectorized implementation:
u = 0
for i in range(10):
    u += abs(a[i] - b[i])
u /= N

print('non-vectorized:', u)

# vectorized implementation:
# WRITE YOUR CODE HERE

u = np.sum(abs(a-b)) / N

print('vectorized:', u)
```

```
('non-vectorized:', 3.2840330560714784e-06)
('vectorized:', 3.2840330560714784e-06)
```

In [7]:
```
a = np.random.rand(10)
b = np.random.rand(10)

# non-vectorized implementation:
u = 0
for i in range(10):
    u += abs(a[i] - b[i])
u /= N

print('non-vectorized:', u)

# vectorized implementation:
# WRITE YOUR CODE HERE

u = np.sum(abs(a-b)) / N

print('vectorized:', u)
```

```
('non-vectorized:', 3.2840330560714784e-06)
('vectorized:', 3.2840330560714784e-06)
```

بخش چهارم اصلاح کد قسمت دوم بدون حلقه

با استفاده از Numpy کد قسمت دوم بدون هیچ گونه for بازنویسی شد.

In [ ]:
```
#In this part according to what we learn from numpy and panda library we
#change code in part b that no for, while use

def RMSE_calc_partd(y,y_pred):
    y_pred = linear_calc(y_pred)
    return ((np.sum((y_pred - y)**2))/len(y))**.5

rmse =  RMSE_calc_partd(y,feature_x)
print("RMSE part d:",rmse)
```

بخش پنجم آشنایی با k-nearest-neighbors

در این مدل پارامتری وجود ندارد که مقدار بهینه ی آن را یاد گرفته و برای خانه های جدید با استفاده از آن ها تخمین را محاسبه کنیم بلکه تخمین قیمت هر خانه جدید به صورت مستقیم از روی ده هایی که داریم محاسبه می شود.

به این صورت که ابتدا نزدیک ترین K نقطه را به خانه مورد نظر می یابیم سپس میانگین قیمت این خانه ها را به عنوان قیمت خانه جدید گزارش می کنیم. ابتدا مقدار همه خصیصه ها را استاندارد سازی می کنیم تا خصیصه هایی که مقادیر بزرگتری دارند اثر سایر خصیصه ها را از بین نبرند.

In [*]:

```
#KNN function is a nonparametric model
#First standardalize dataset according to min and max of columns
#Then calculate distance
#select k smallest distancce
#return mean of them as estimate price

def KNN(input_df,train_data = X_train, k=10):
    maxx = train_data.max(axis=0)
    minn = train_data.min(axis=0)
    train_data = (train_data - minn)/(maxx-minn)
    input_df = (input_df - minn) / (maxx-minn)
    train_data['distance'] = train_data[['MSSubClass', 'LotArea', 'OverallQual', 'LotFrontage', 'OverallCond',
                        'BedroomAbvGr', 'TotRmsAbvGrd', 'TotalBsmtSF', 'YearBuilt' ]].sub(np.array(input_df)).pow(
    train_data['SalePrice'] = data['SalePrice']
    df = train_data.nsmallest(k, 'distance')
    return df['SalePrice'].mean()

#test data with sample row from datas

test_df = data.sample()
test_df_price = test_df['SalePrice']
print(test_df_price)
test_df = test_df.drop(['SalePrice','Id'],1)
print("estimated price KNN for a sample row in dataset:",KNN(test_df))
```

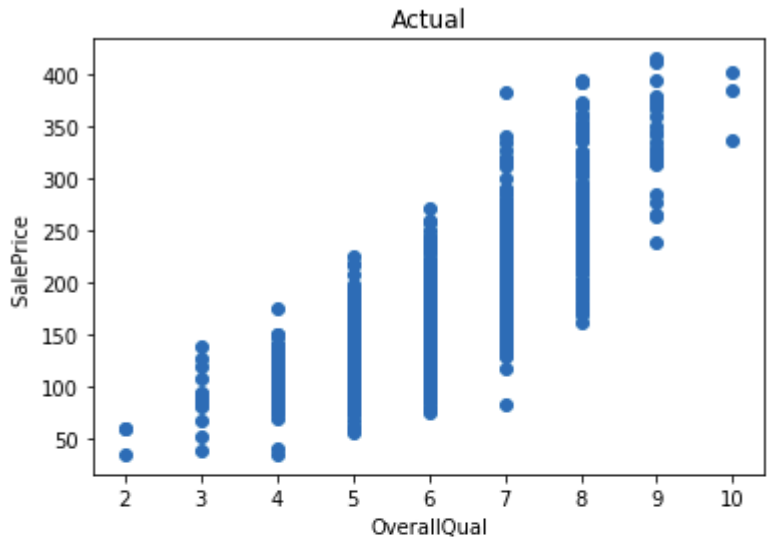
برای تست تابع knn ابتدا آمدم تمام داده ها را به عنوان ورودی به تابع دادم. خروجی این تابع قیمت پیش بینی شده به ازای هر سطر می باشد.سپس مقدار rmse را به ازای y قیمت واقعی و y_pred قیمت پیش بینی شده به دست آوردم که این مقدار حدود ۲۵ شد.

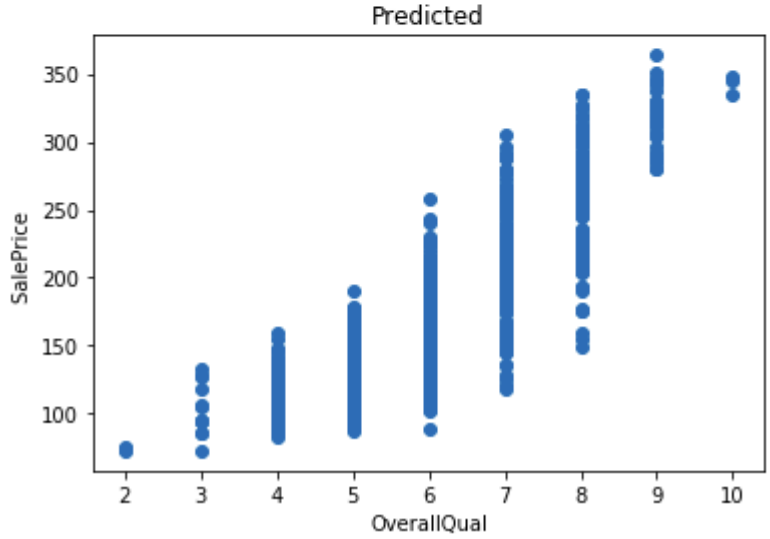
In [15]:

```
#here calculate rmse with knn
input_df = data.head(len(data))
df = input_df['SalePrice']
input_df = input_df.drop(['SalePrice','Id'],1)
input_df.head()

y_preds = []
for i in range(len(input_df)):
    y_preds.append(KNN(pd.DataFrame(data
                                    =[input_df.iloc[i].values], columns=input_df.columns)))

def RMSE_calc_parte(y,y_pred):
    return ((np.sum((y_pred - y)**2))/len(y))**.5

RMSE_calc_parte(df.values,y_preds)
```

Out[15]: 25.19223810272189

همچنین نمودار جهت مقایسه قیمت واقعی و پیش‌بینی شده بر حسب مشخصه overalqual به صورت ز خواهد بود.


Actual
SalePrice
OverallQual
400
350
300
250
200
150
100
50
2
3
4
5
6
7
8
9
10


Predicted
SalePrice
OverallQual
350
300
250
200
150
100
2
3
4
5
6
7
8
9
10

پایان