

# HW\_1

September 29, 2022

```
[504]: # ===== 1a =====
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import sklearn.datasets as dt
from sklearn.model_selection import train_test_split
df = pandas.read_csv('Housing.csv', header = 0)

list_bm = ['mainroad', 'guestroom', 'basement', 'hotwaterheating',
           ↪ 'airconditioning', 'prefarea']

def binary_map(x):
    return x.map({'yes': 1, 'no': 0})

df[list_bm] = df[list_bm].apply(binary_map)

price = df['price']
area = df['area']
bedroom = df['bedrooms']
bathroom = df['bathrooms']
stories = df['stories']
mainroad = df['mainroad']
guestroom = df['guestroom']
basement = df['basement']
waterheating = df['hotwaterheating']
ac = df['airconditioning']
parking = df['parking']
prefarea = df['prefarea']
furnished = df['furnishingstatus']

theta_0 = 0
theta_1 = 1
a = 0.1
X = []
Y = df.values[:,0]
theta_old = []
theta_new = []
m = len(price)
cost_history = []
loss = []
h_pred = []
h_pred2 = []
```

```

theta = []

for i in range(m):
    x_list = area[i] + bedroom[i] + bathroom[i] + stories[i] + parking[i]
    X.append(x_list)

for i in range(m):
    h = (X[i] * theta_1) + theta_0
    h_pred.append(h)
    errors = h_pred[i] - Y[i]
    sqrErrors = np.square(errors)
    j = np.sum(sqrErrors)
    loss.append(j/(m))

for j in range(m):
    pred = theta_0 + (theta_1 * h_pred[j])
    h_pred2.append(pred)
    errors = h_pred2[j] - (h_pred[j] + Y[j])
    sum_delta = (pred - errors)
    theta_new = theta_old - sum_delta
    theta.append(theta_new)
    cost = a * (np.square(errors))
    cost_history.append(cost/m)

plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.scatter(X,Y)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

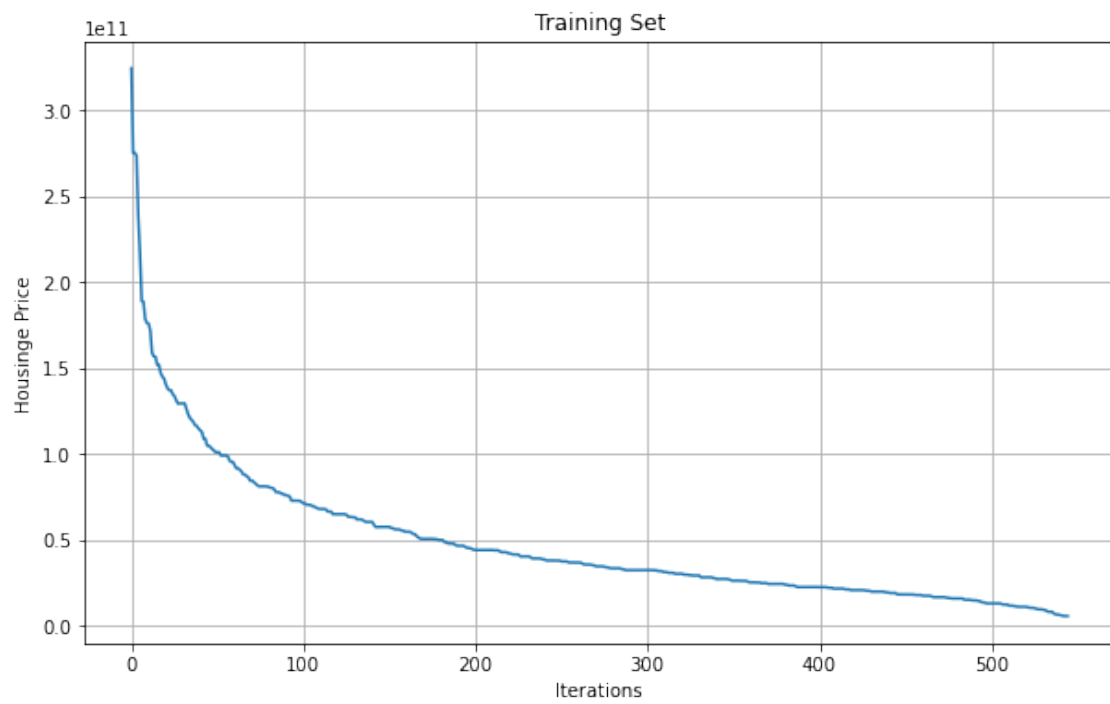
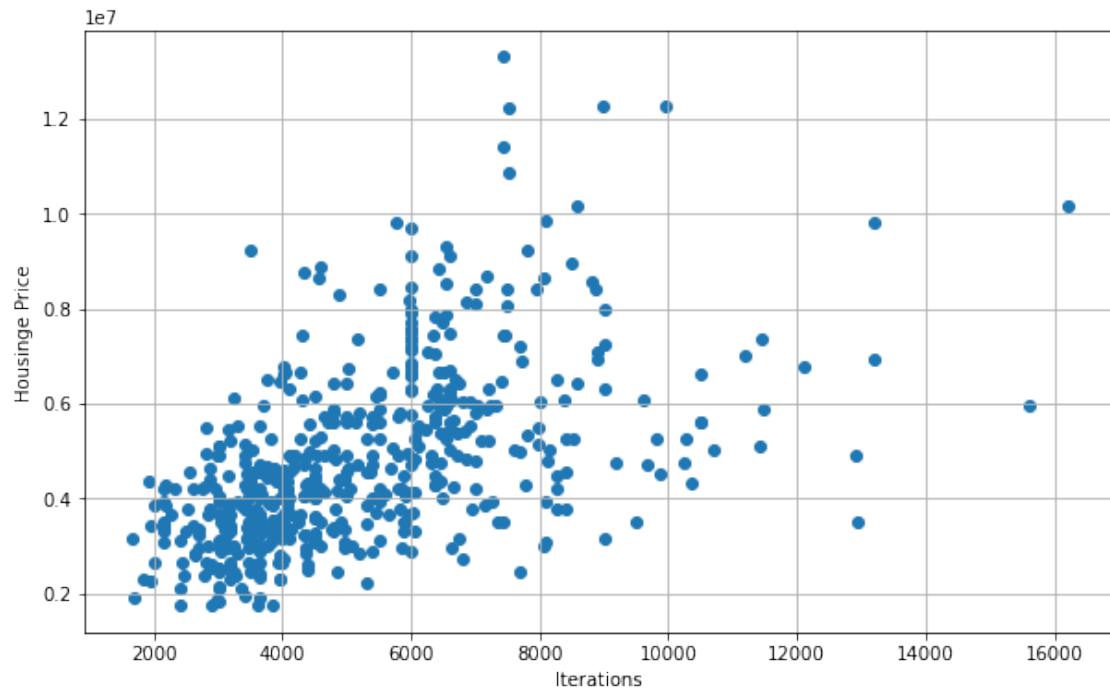
plt.grid()
plt.plot(loss)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.title('Training Set')
plt.show()

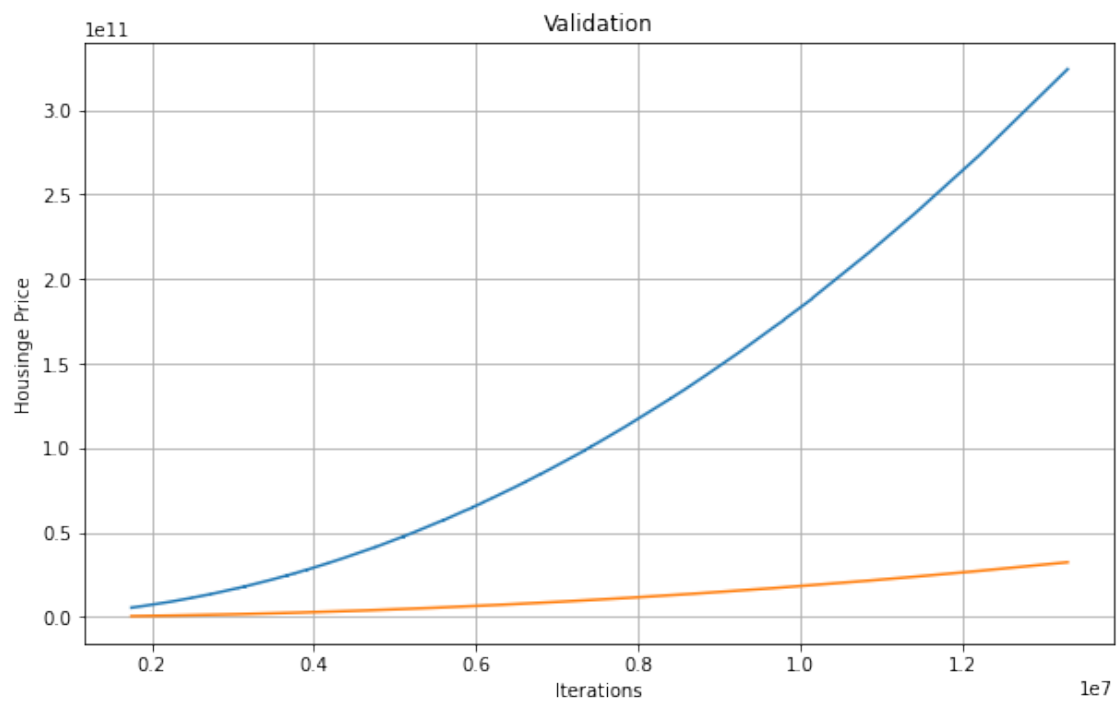
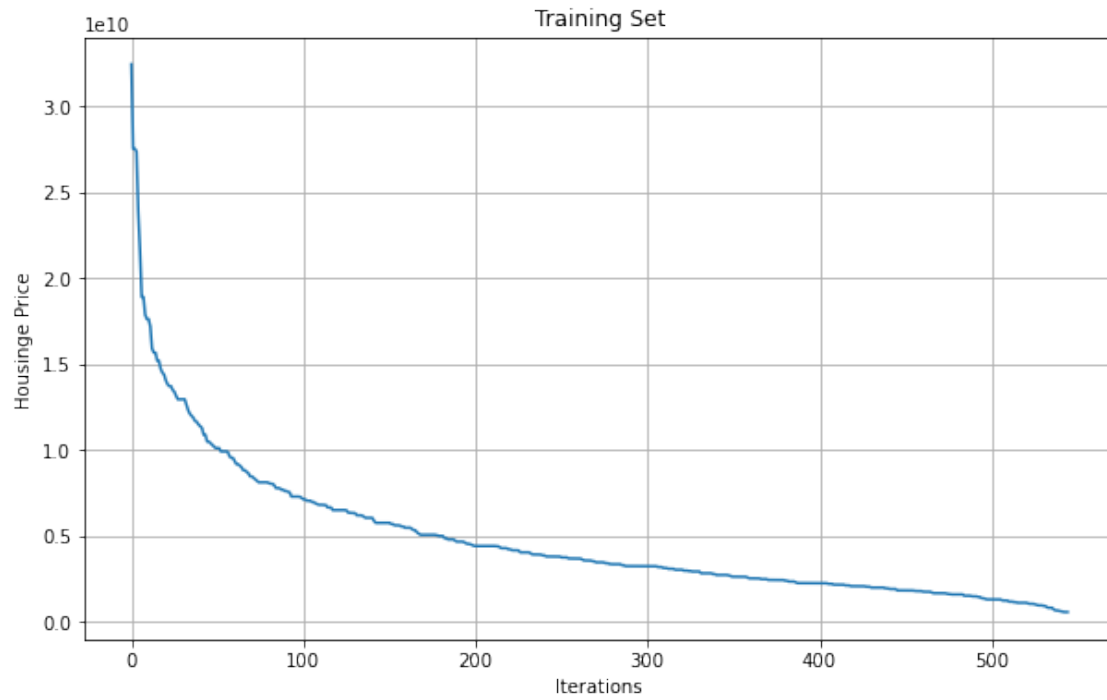
plt.grid()
plt.plot(cost_history)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.title('Training Set')
plt.show()

plt.grid()
plt.plot(price,loss)
plt.plot(price,cost_history)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.title('Validation')

```

```
plt.show()
```





```
[505]: # ===== 1b =====
import pandas as pd
```

```

import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import sklearn.datasets as dt
from sklearn.model_selection import train_test_split
df = pandas.read_csv('Housing.csv', header = 0)

list_bm = ['mainroad', 'guestroom', 'basement', 'hotwaterheating',
↳ 'airconditioning', 'prefarea']

def binary_map(x):
    return x.map({'yes': 1, 'no': 0})

df[list_bm] = df[list_bm].apply(binary_map)

price = df['price']
area = df['area']
bedroom = df['bedrooms']
bathroom = df['bathrooms']
stories = df['stories']
mainroad = df['mainroad']
guestroom = df['guestroom']
basement = df['basement']
waterheating = df['hotwaterheating']
ac = df['airconditioning']
parking = df['parking']
prefarea = df['prefarea']
furnished = df['furnishingstatus']

theta_0 = 0
theta_1 = 1
a = 0.1
X = []
Y = df.values[:,0]
theta_old = 0
theta_new = 0
m = len(price)
cost_history = []
loss = []
h_pred = []
h_pred2 = []
theta = []

for i in range(m):
    x_list = ac[i] + prefarea[i] + waterheating[i] + area[i] + bedroom[i] +
↳ bathroom[i] + stories[i] + parking[i] + mainroad[i] + guestroom[i] + basement[i]
    X.append(x_list)

for i in range(m):
    h = (X[i] * theta_1) + theta_0
    h_pred.append(h)
    errors = h_pred[i] - Y[i]

```

```

sqrErrors = np.square(errors)
j = np.sum(sqrErrors)
loss.append(j/(m))

for j in range(m):
    pred = theta_0 + (theta_1 * h_pred[j])
    h_pred2.append(pred)
    errors = h_pred2[j] - (h_pred[j] + Y[j])
    sum_delta = (pred - errors)
    theta_new = theta_old - sum_delta
    theta.append(theta_new)
    cost = a * (np.square(errors))
    cost_history.append(cost/m)

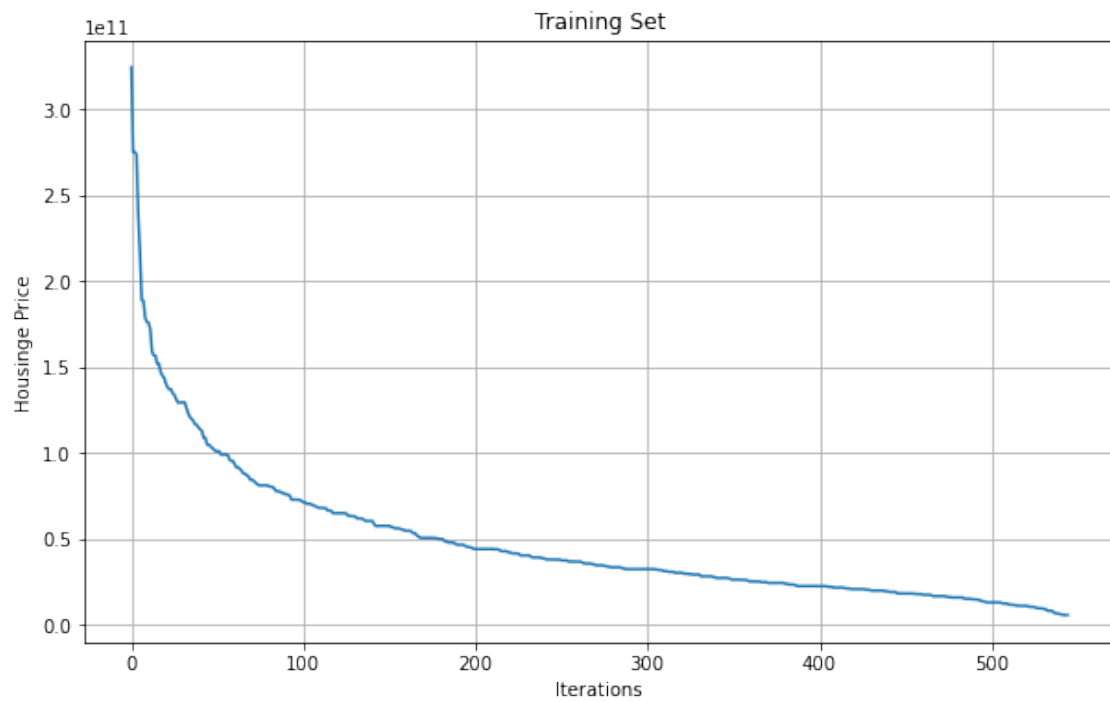
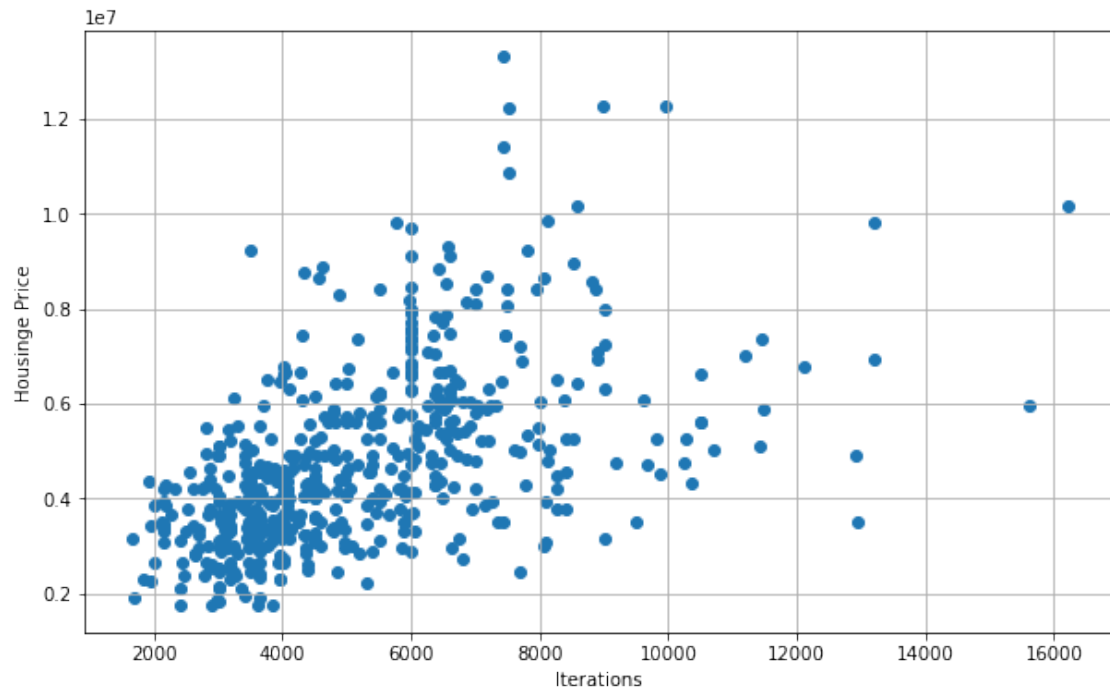
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.scatter(X,Y)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

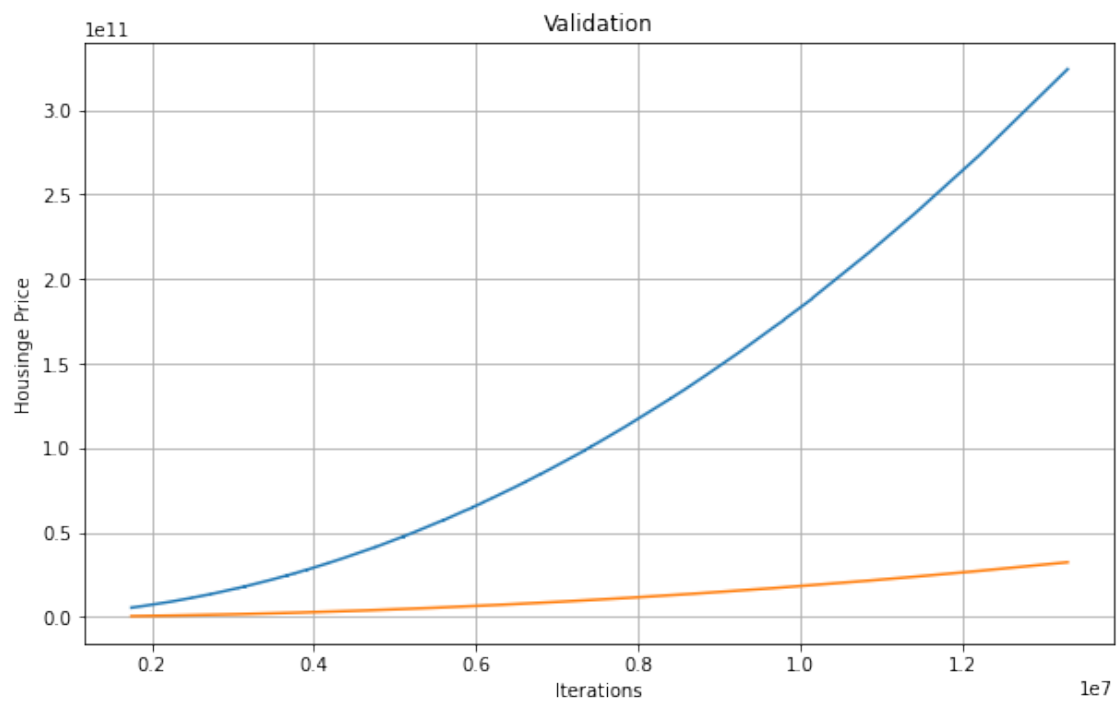
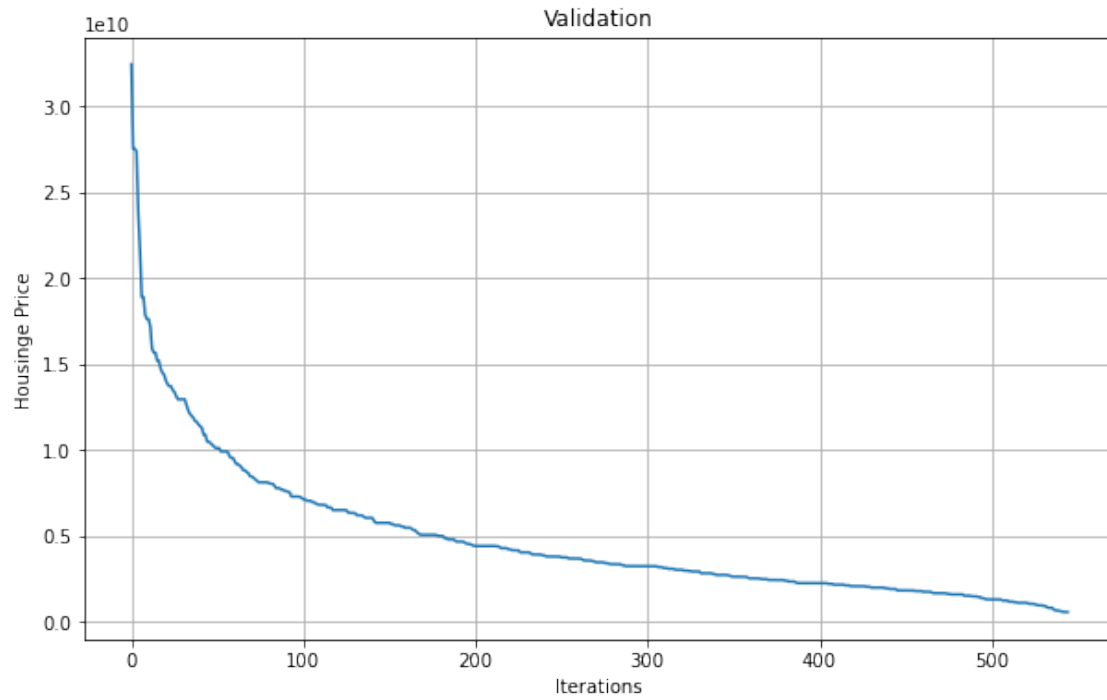
plt.grid()
plt.plot(loss)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.title('Training Set')
plt.show()

plt.grid()
plt.plot(cost_history)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.title('Validation')
plt.show()

plt.grid()
plt.plot(price,loss)
plt.plot(price,cost_history)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.title('Validation')
plt.show()

```





```
[511]: # ===== 2a Normalization =====
import pandas as pd
```



```

import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import sklearn.datasets as dt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler, StandardScaler
import warnings
warnings.filterwarnings('ignore')

df = pandas.read_csv('Housing.csv', header = 0)

list_bm = ['mainroad', 'guestroom', 'basement', 'hotwaterheating',
    ↪ 'airconditioning', 'prefarea']

def binary_map(x):
    return x.map({'yes': 1, 'no': 0})

df[list_bm] = df[list_bm].apply(binary_map)

price = df['price']
area = df['area']
bedroom = df['bedrooms']
bathroom = df['bathrooms']
stories = df['stories']
mainroad = df['mainroad']
guestroom = df['guestroom']
basement = df['basement']
waterheating = df['hotwaterheating']
ac = df['airconditioning']
parking = df['parking']
prefarea = df['prefarea']
furnished = df['furnishingstatus']

np.random.seed(0)
df_train, df_test = train_test_split(df, train_size = 0.8, test_size = 0.2,
    ↪ random_state = 42)

num_vars = ['area', 'bedrooms', 'bathrooms', 'stories', 'parking', 'price']
df_newTrain = df_train[num_vars]
df_newTest = df_test[num_vars]

theta_0 = 0
theta_1 = 1
a = 0.1
theta_old = 0
theta_new = 0
cost_history = []
loss = []
h_pred = []
h_pred2 = []
theta = []
X = []

```

```

# define standard scaler
scaler = MinMaxScaler()
df_newTrain[num_vars]= scaler.fit_transform(df_newTrain[num_vars])
y_newTrain = df_newTrain.pop('price')
x_newTrain = df_newTrain
Y = y_newTrain.values

X_minmax = scaler.fit_transform(x_newTrain)

X0 = df_newTrain.values[:,0]
X1 = df_newTrain.values[:,1]
X2 = df_newTrain.values[:,2]
X3 = df_newTrain.values[:,3]
X4 = df_newTrain.values[:,4]

for i in range(len(X_minmax)):
    x_list = X0[i] + X1[i] + X2[i] + X3[i] + X4[i]
    X.append(x_list)

for i in range(len(X_minmax)):
    h = (X[i] * theta_1) + theta_0
    h_pred.append(h)
    errors = h_pred[i] - Y[i]
    sqrErrors = np.square(errors)
    j = np.sum(sqrErrors)
    loss.append(j/(m))

loss = sorted(loss)

for j in range(len(X_minmax)):
    pred = theta_0 + (theta_1 * loss[j])
    h_pred2.append(pred)
    errors = h_pred2[j] - (h_pred[j] + Y[j])
    sum_delta = (pred - errors)
    theta_new = theta_old - sum_delta
    theta.append(theta_new)
    cost = a * (np.square(errors))
    cost_history.append(cost/m)

cost_history = sorted(cost_history)

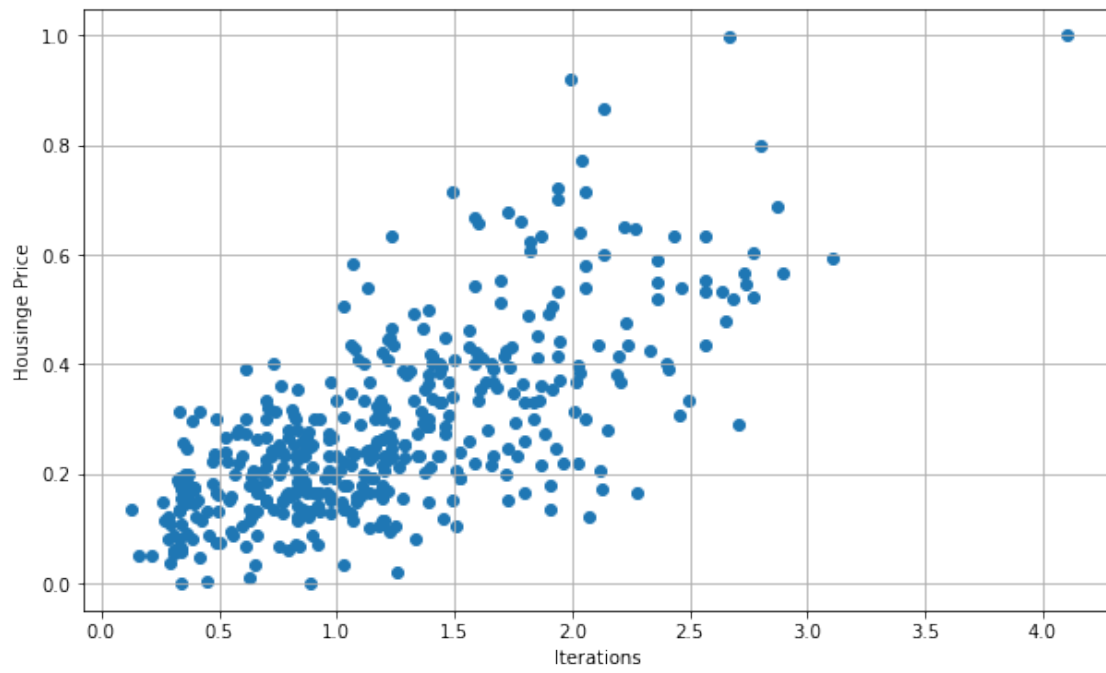
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.scatter(X,Y)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

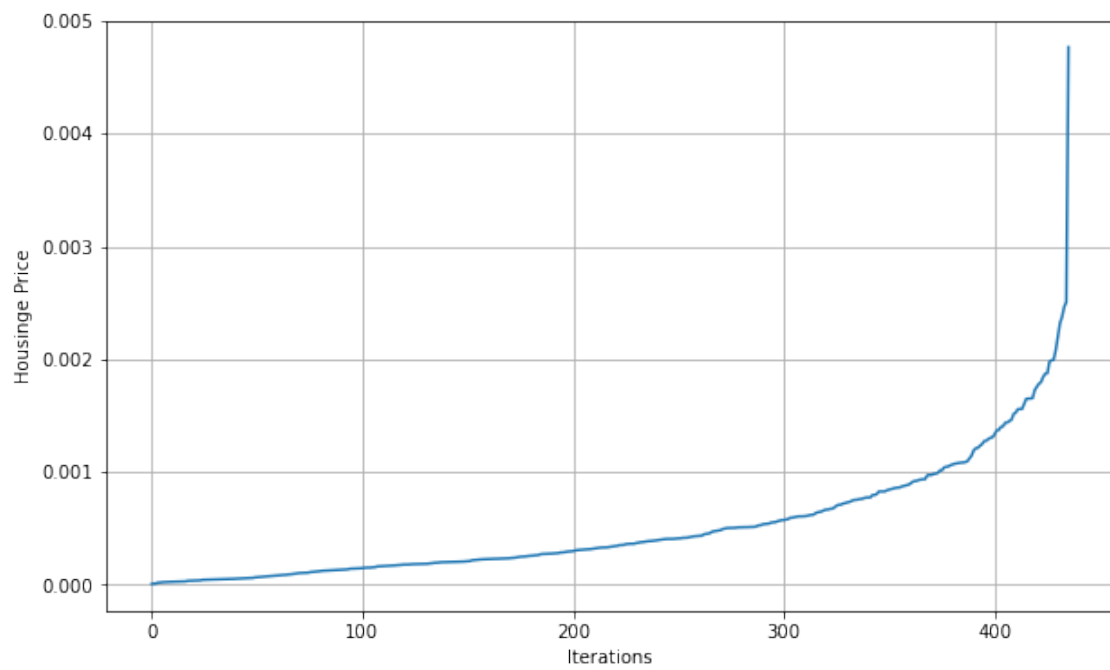
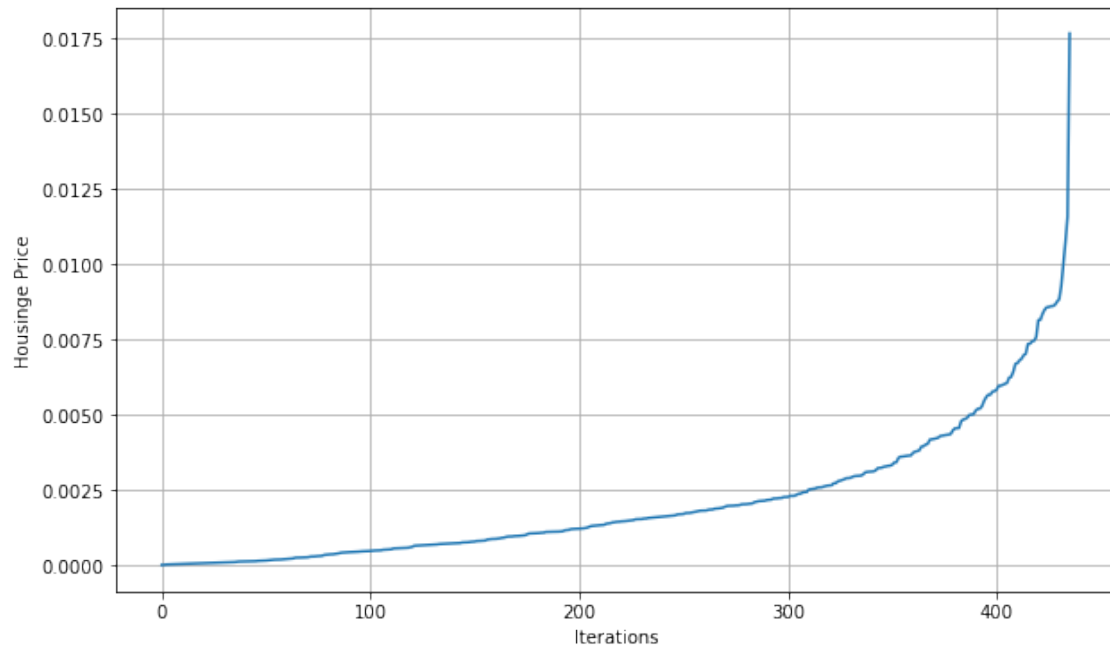
plt.grid()
plt.plot(loss)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

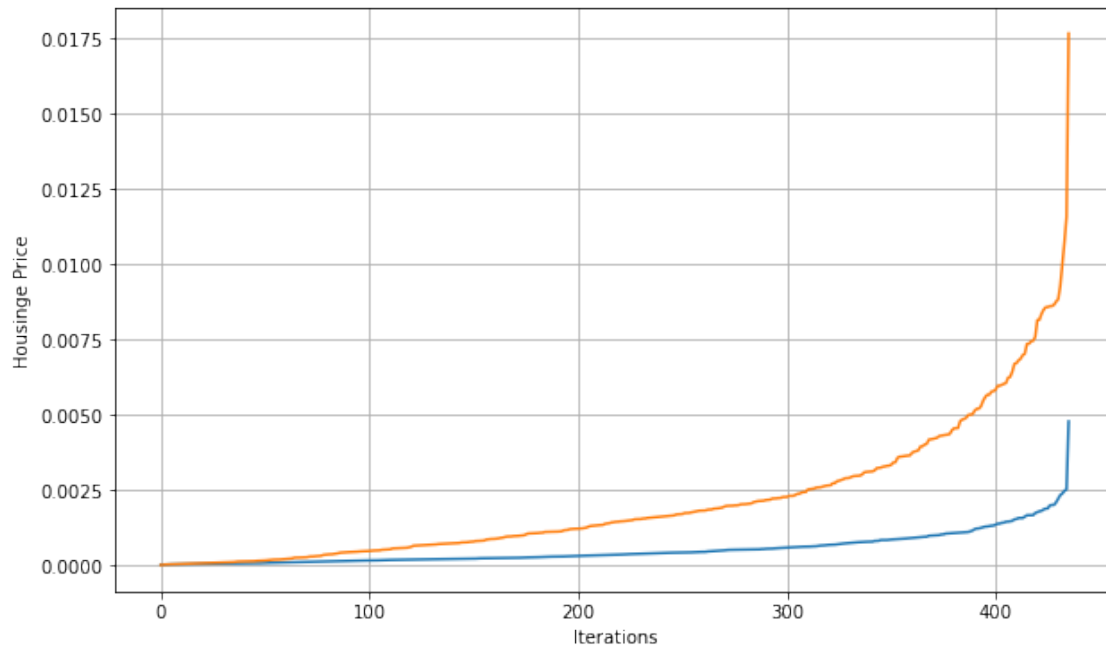
```

```
plt.grid()
plt.plot(cost_history)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()
```

```
plt.grid()
plt.plot(cost_history)
plt.plot(loss)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()
```







```
[551]: # ===== 2a Standardization =====
import pandas as pd
import numpy as np
import math
import statistics
import matplotlib
import matplotlib.pyplot as plt
import sklearn.datasets as dt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler, StandardScaler
import warnings
warnings.filterwarnings('ignore')

df = pandas.read_csv('Housing.csv', header = 0)

list_bm = ['mainroad', 'guestroom', 'basement', 'hotwaterheating', 'airconditioning', 'prefarea']

def binary_map(x):
    return x.map({'yes': 1, 'no': 0})

def standard_deviation(x,mean,n):
    a = np.square(abs(x - mean))
    sd = math.sqrt(sum(a)/n)
    return sd

def standard_scale(x,mean,sd):
    std = (x - mean)/sd
    return std
```

```

df[list_bm] = df[list_bm].apply(binary_map)

price = df['price']
area = df['area']
bedroom = df['bedrooms']
bathroom = df['bathrooms']
stories = df['stories']
mainroad = df['mainroad']
guestroom = df['guestroom']
basement = df['basement']
waterheating = df['hotwaterheating']
ac = df['airconditioning']
parking = df['parking']
prefarea = df['prefarea']
furnished = df['furnishingstatus']

np.random.seed(0)
df_train, df_test = train_test_split(df, train_size = 0.8, test_size = 0.2,
    random_state = 42)

num_vars = ['area', 'bedrooms', 'bathrooms', 'stories', 'parking', 'price']
df_newTrain = df_train[num_vars]
df_newTest = df_test[num_vars]

theta_0 = 0
theta_1 = 1
a = 0.1
theta_old = 0
theta_new = 0
cost_history = []
loss = []
h_pred = []
h_pred2 = []
theta = []
X = []

df_newTrain[num_vars] = scaler.fit_transform(df_newTrain[num_vars])
y_newTrain = df_newTrain.pop('price')
x_newTrain = df_newTrain
Y = y_newTrain.values

X0 = df_newTrain.values[:,0]
X1 = df_newTrain.values[:,1]
X2 = df_newTrain.values[:,2]
X3 = df_newTrain.values[:,3]
X4 = df_newTrain.values[:,4]

for i in range(len(x_newTrain)):
    x_list = X0[i] + X1[i] + X2[i] + X3[i] + X4[i]
    X.append(x_list)

```

```

x_mean = statistics.mean(X)
sd = standard_deviation(X,x_mean,len(X))
std = standard_scale(X,x_mean,sd)

for i in range(len(X)):
    h = (std[i] * theta_1) + theta_0
    h_pred.append(h)
    errors = h_pred[i] - Y[i]
    sqrErrors = np.square(errors)
    j = np.sum(sqrErrors)
    loss.append(j/(m))

loss = sorted(loss)

for j in range(len(X)):
    pred = theta_0 + (theta_1 * loss[j])
    h_pred2.append(pred)
    errors = h_pred2[j] - (h_pred[j] + Y[j])
    sum_delta = (pred - errors)
    theta_new = theta_old - sum_delta
    theta.append(theta_new)
    cost = a * (np.square(errors))
    cost_history.append(cost/m)

cost_history = sorted(cost_history)

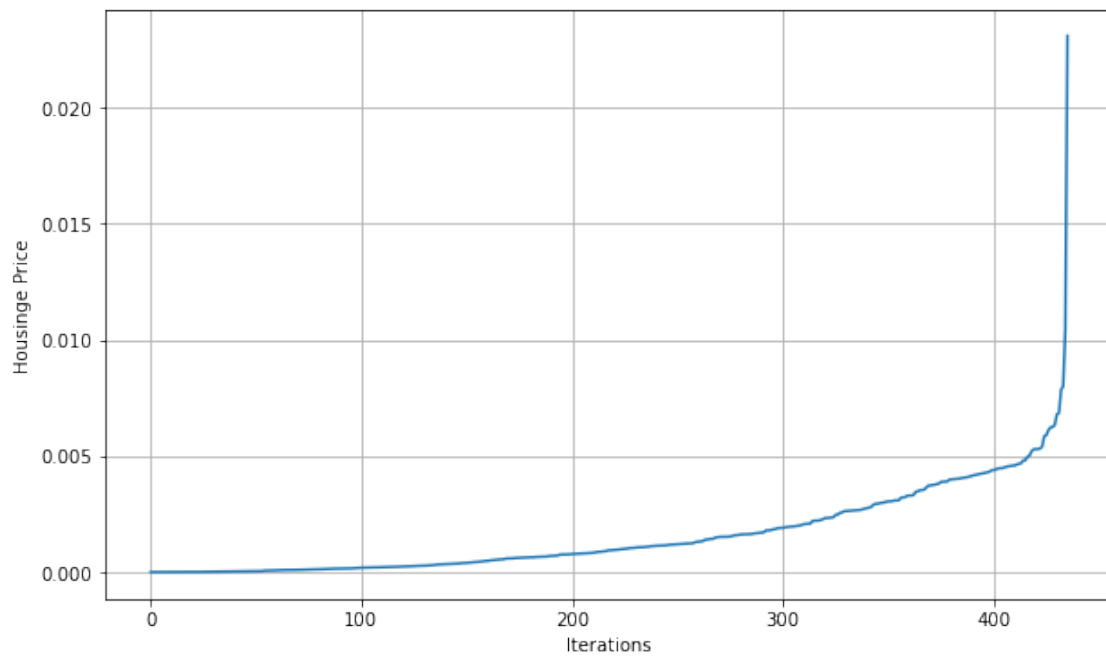
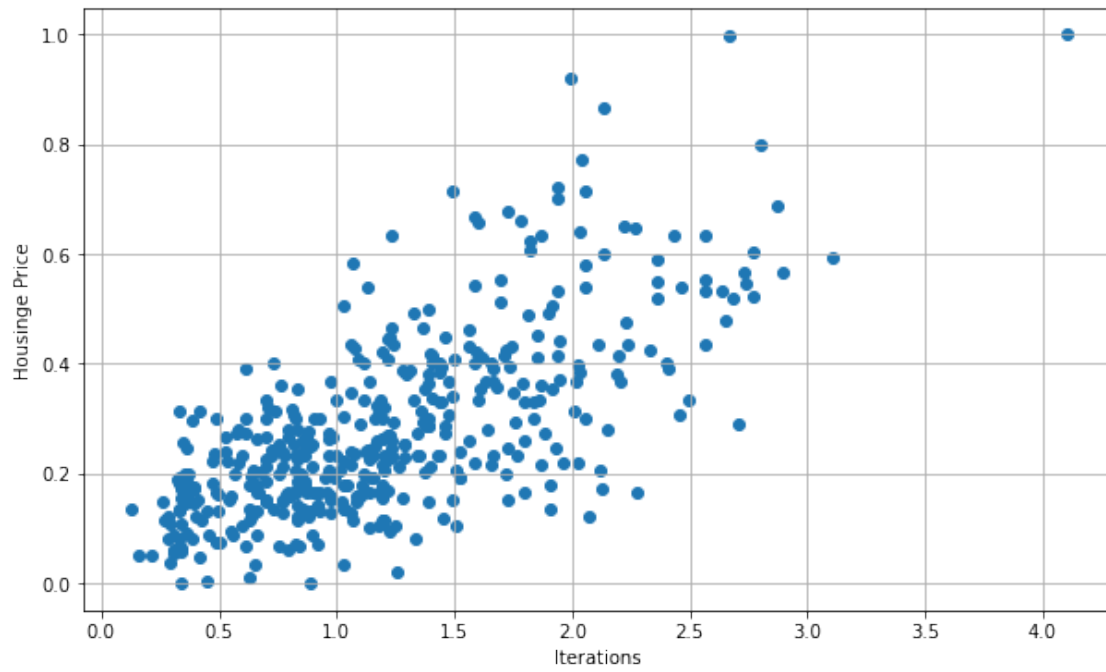
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.scatter(X,Y)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

plt.grid()
plt.plot(loss)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

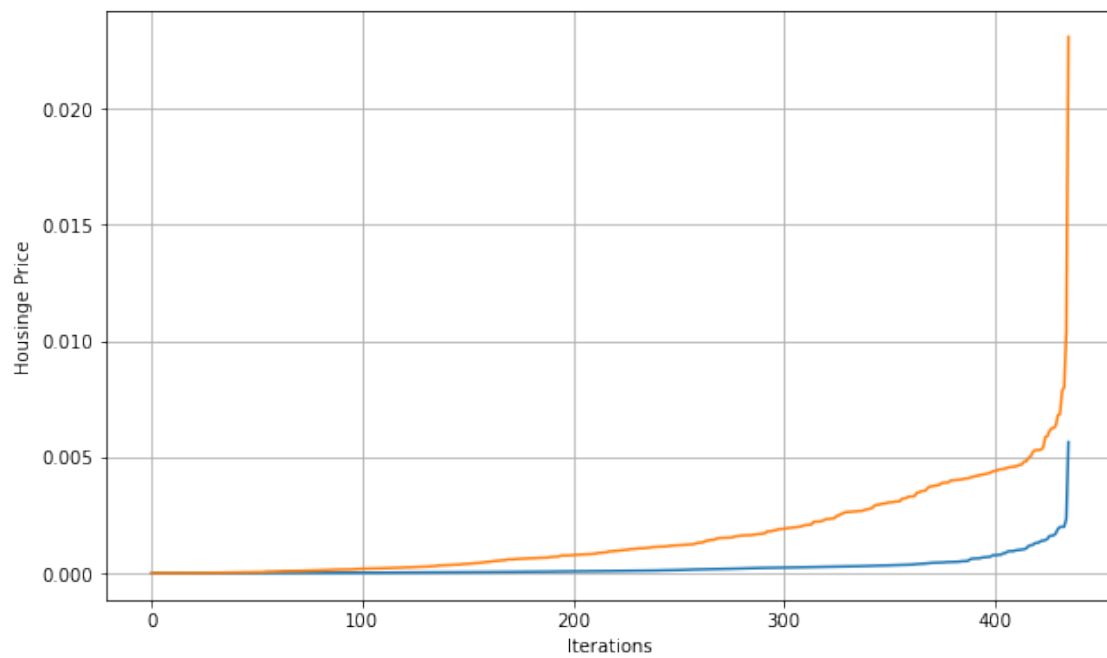
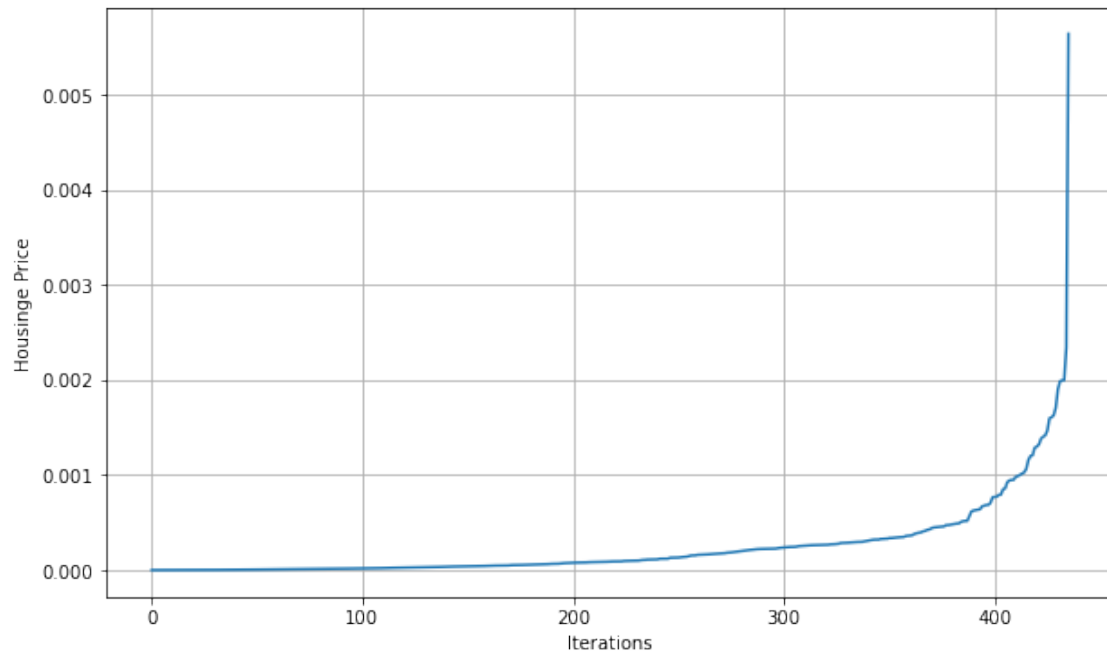
plt.grid()
plt.plot(cost_history)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

plt.grid()
plt.plot(cost_history)
plt.plot(loss)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

```







```
[668]: # ===== 2b Normalization =====
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
```

```

import sklearn.datasets as dt
from sklearn.model_selection import train_test_split
df = pandas.read_csv('Housing.csv', header = 0)

list_bm = ['mainroad', 'guestroom', 'basement', 'hotwaterheating',
↳ 'airconditioning', 'prefarea']

def binary_map(x):
    return x.map({'yes': 1, 'no': 0})

df[list_bm] = df[list_bm].apply(binary_map)

price = df['price']
area = df['area']
bedroom = df['bedrooms']
bathroom = df['bathrooms']
stories = df['stories']
mainroad = df['mainroad']
guestroom = df['guestroom']
basement = df['basement']
waterheating = df['hotwaterheating']
ac = df['airconditioning']
parking = df['parking']
prefarea = df['prefarea']
furnished = df['furnishingstatus']

np.random.seed(0)
df_train, df_test = train_test_split(df, train_size = 0.8, test_size = 0.2,
↳ random_state = 42)

num_vars = ['airconditioning', 'bedrooms', 'bathrooms', 'stories', 'parking',
↳ 'mainroad', 'guestroom', 'basement', 'prefarea', 'hotwaterheating', 'area', 'price']
df_newTrain = df_train[num_vars]
df_newTest = df_test[num_vars]

theta_0 = 0
theta_1 = 0
a = 0.01
X = []
theta_old = 0
theta_new = 0
cost_history = []
loss = []
h_pred = []
h_pred2 = []
theta = []

scaler = MinMaxScaler()
df_newTrain[num_vars] = scaler.fit_transform(df_newTrain[num_vars])
y_newTrain = df_newTrain.pop('price')
x_newTrain = df_newTrain
Y = y_newTrain.values

```

```

X0 = df_newTrain.values[:,0]
X1 = df_newTrain.values[:,1]
X2 = df_newTrain.values[:,2]
X3 = df_newTrain.values[:,3]
X4 = df_newTrain.values[:,4]
X5 = df_newTrain.values[:,5]
X6 = df_newTrain.values[:,6]
X7 = df_newTrain.values[:,7]
X8 = df_newTrain.values[:,8]
X9 = df_newTrain.values[:,9]
X10 = df_newTrain.values[:,10]

for i in range(len(x_newTrain)):
    x_list = X0[i] + X1[i] + X2[i] + X3[i] + X4[i] + X5[i] + X6[i] + X7[i] + X8[i] +
    ↪X9[i] + X10[i]
    X.append(x_list)

X = sorted(X)

for i in range(len(X)):
    h = (X[i] * theta_1) + theta_0
    h_pred.append(h)
    errors = h_pred[i] - Y[i]
    sqrErrors = np.square(errors)
    j = np.sum(sqrErrors)
    loss.append(j/(m))

loss = sorted(loss)

for j in range(len(X)):
    pred = theta_0 + (theta_1 * loss[j])
    h_pred2.append(pred)
    errors = h_pred2[j] - (h_pred[j] + Y[j])
    sum_delta = (h_pred2[j] - errors)
    theta_new = theta_old - sum_delta
    theta.append(theta_new)
    cost = a * (np.square(errors))
    cost_history.append(cost/m)

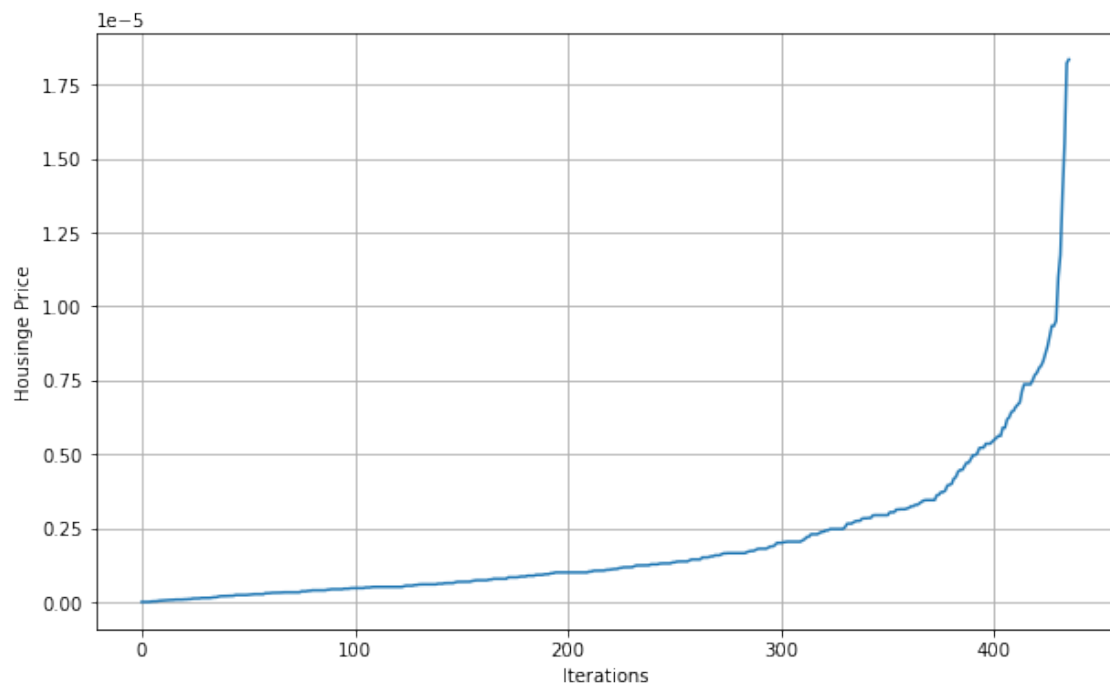
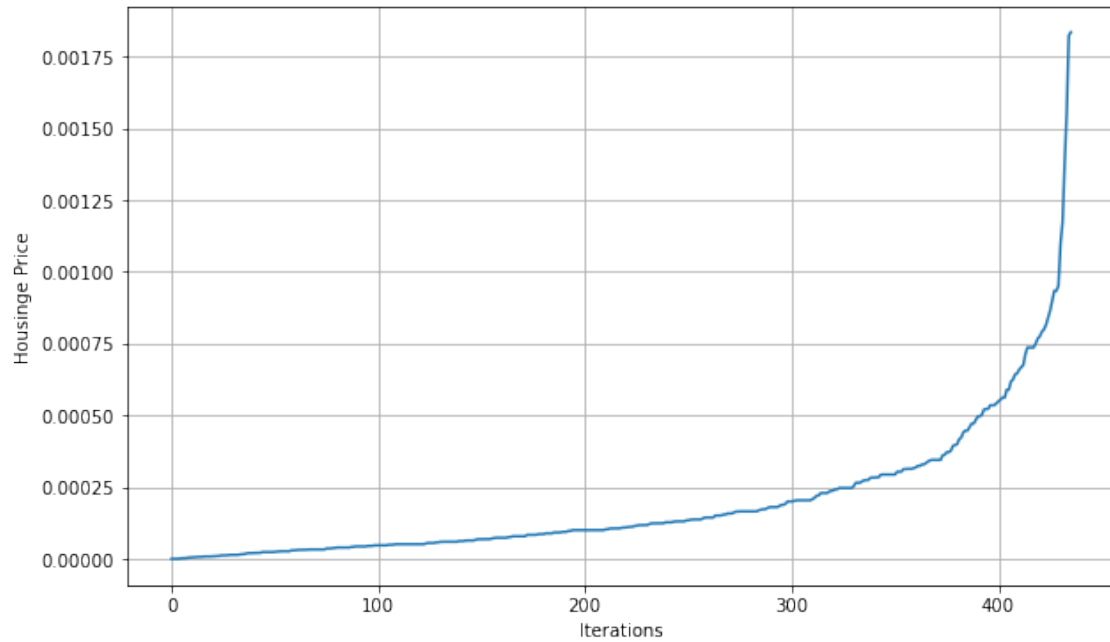
cost_history = sorted(cost_history)

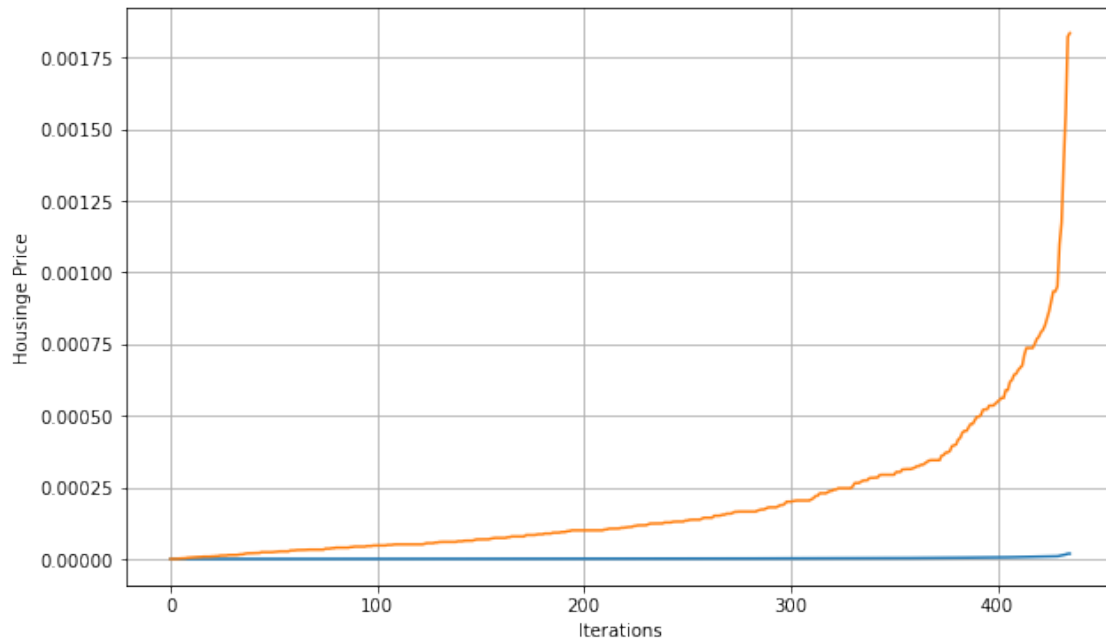
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.plot(loss)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

plt.grid()
plt.plot(cost_history)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

```

```
plt.grid()
plt.plot(cost_history)
plt.plot(loss)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()
```





```
[687]: # ===== 3a Standardization =====
import pandas as pd
import numpy as np
import math
import statistics
import matplotlib
import matplotlib.pyplot as plt
import sklearn.datasets as dt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler, StandardScaler
import warnings
warnings.filterwarnings('ignore')

df = pandas.read_csv('Housing.csv', header = 0)

list_bm = ['mainroad', 'guestroom', 'basement', 'hotwaterheating', 'airconditioning', 'prefarea']

def binary_map(x):
    return x.map({'yes': 1, 'no': 0})

def standard_deviation(x, mean, n):
    a = np.square(abs(x - mean))
    sd = math.sqrt(sum(a)/n)
    return sd

def standard_scale(x, mean, sd):
```

```

    std = (x - mean)/sd
    return std

df[list_bm] = df[list_bm].apply(binary_map)

price = df['price']
area = df['area']
bedroom = df['bedrooms']
bathroom = df['bathrooms']
stories = df['stories']
mainroad = df['mainroad']
guestroom = df['guestroom']
basement = df['basement']
waterheating = df['hotwaterheating']
ac = df['airconditioning']
parking = df['parking']
prefarea = df['prefarea']
furnished = df['furnishingstatus']

np.random.seed(0)
df_train, df_test = train_test_split(df, train_size = 0.8, test_size = 0.2,
    random_state = 42)

num_vars = ['area', 'bedrooms', 'bathrooms', 'stories', 'parking', 'price']
df_newTrain = df_train[num_vars]
df_newTest = df_test[num_vars]

theta_0 = 0
theta_1 = 1
a = 0.1
theta_old = 0
theta_new = 0
cost_history = []
loss = []
h_pred = []
h_pred2 = []
theta = []
X = []
lamda = 50

df_newTrain[num_vars] = scaler.fit_transform(df_newTrain[num_vars])
y_newTrain = df_newTrain.pop('price')
x_newTrain = df_newTrain
Y = y_newTrain.values

X0 = df_newTrain.values[:,0]
X1 = df_newTrain.values[:,1]
X2 = df_newTrain.values[:,2]
X3 = df_newTrain.values[:,3]
X4 = df_newTrain.values[:,4]

for i in range(len(x_newTrain)):

```

```

x_list = X0[i] + X1[i] + X2[i] + X3[i] + X4[i]
X.append(x_list)

x_sum = sum(X)
x_mean = statistics.mean(X)
sd = standard_deviation(X,x_mean,len(X))
std = standard_scale(X,x_mean,sd)

for i in range(len(X)):
    h = (std[i] * theta_1) + theta_0
    h_pred.append(h)
    reg = lamda * (h_pred[i])**2
    errors = h_pred[i] - Y[i] + reg
    sqrErrors = np.square(errors)
    j = np.sum(sqrErrors)
    loss.append(j/(m))

loss = sorted(loss)

for j in range(len(X)):
    pred = theta_0 + (theta_1 * loss[j])
    h_pred2.append(pred)
    errors = h_pred2[j] - (h_pred[j] + Y[j])
    sum_delta = (pred - errors)
    theta_new = theta_old - sum_delta
    theta.append(theta_new)
    cost = a * (np.square(errors))
    cost_history.append(cost/m)

plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.scatter(X,Y)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

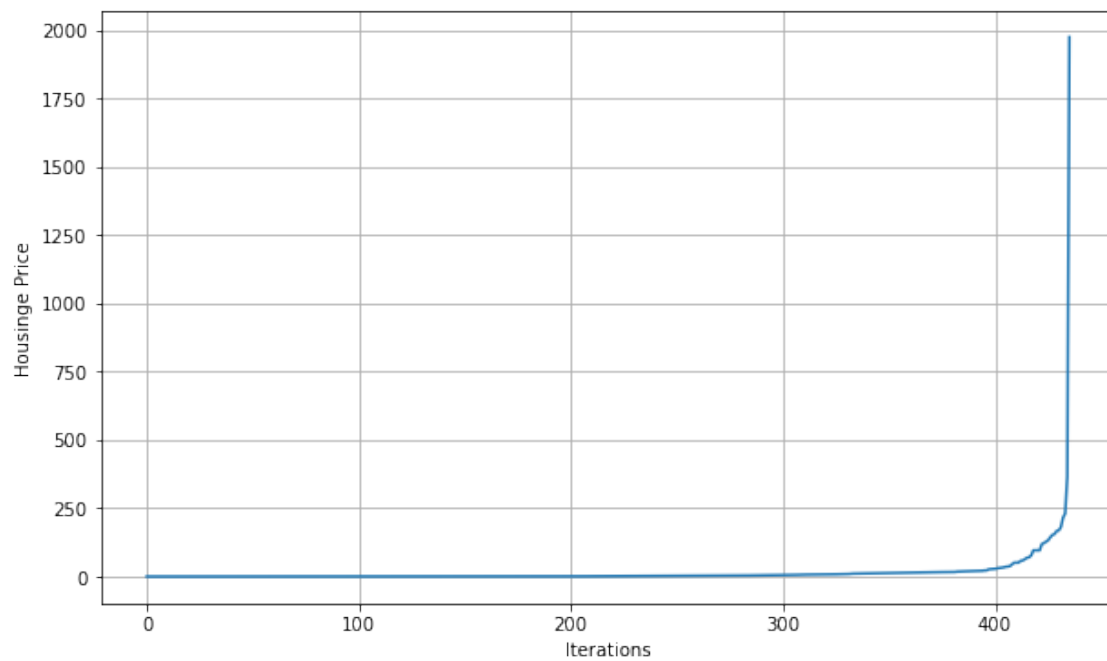
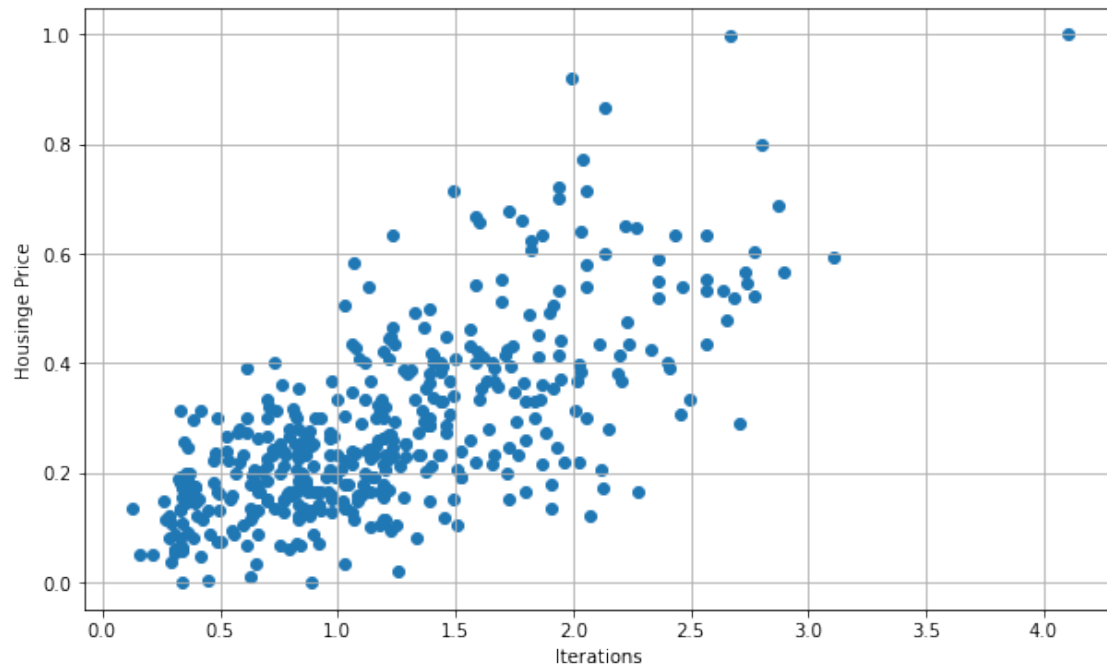
plt.grid()
plt.plot(loss)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

plt.grid()
plt.plot(cost_history)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

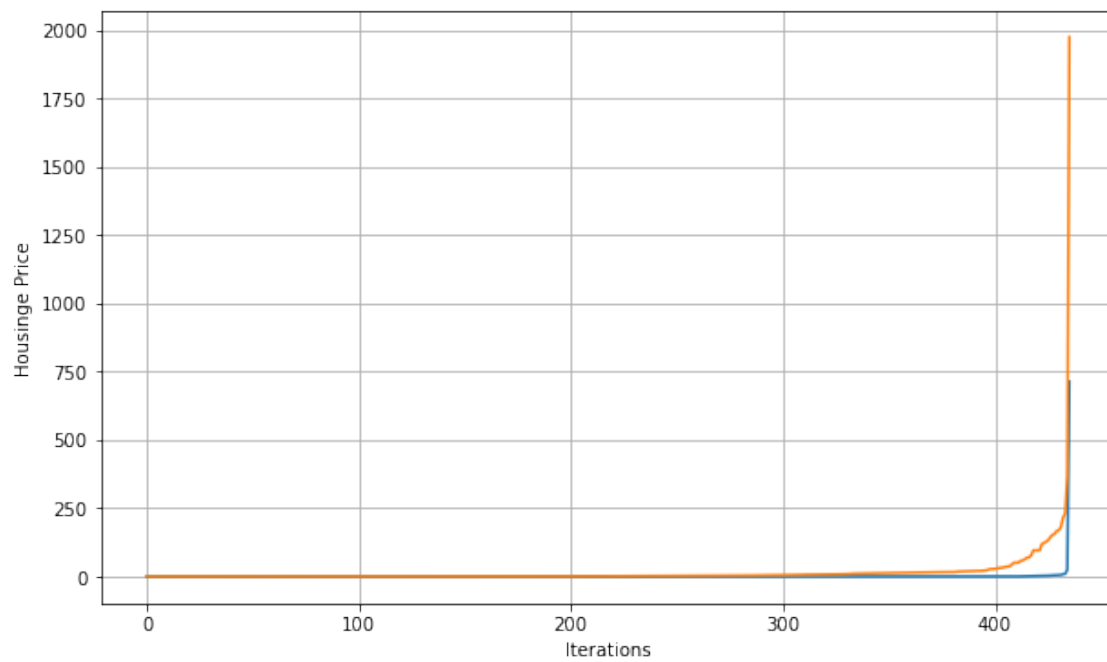
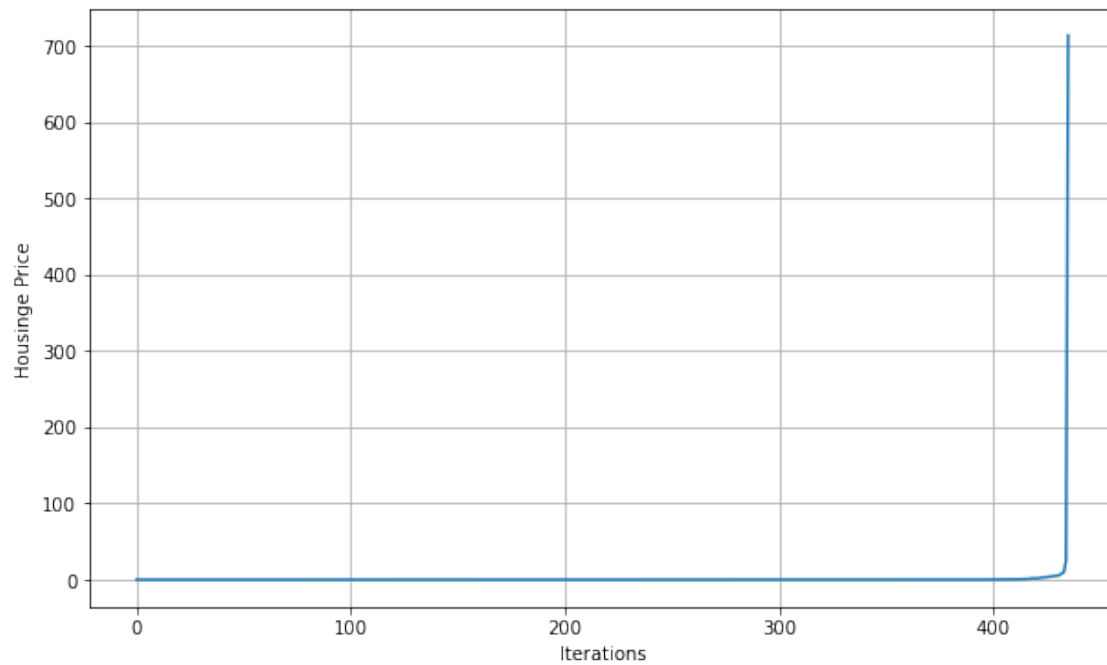
plt.grid()
plt.plot(cost_history)
plt.plot(loss)
plt.xlabel('Iterations')

```

```
plt.ylabel('Housing Price')  
plt.show()
```







```
[738]: # ===== 3b Standardization =====
import pandas as pd
import numpy as np
import matplotlib
```

```

import matplotlib.pyplot as plt
import sklearn.datasets as dt
from sklearn.model_selection import train_test_split
df = pandas.read_csv('Housing.csv', header = 0)

list_bm = ['mainroad', 'guestroom', 'basement', 'hotwaterheating',
↳ 'airconditioning', 'prefarea']

def binary_map(x):
    return x.map({'yes': 1, 'no': 0})

df[list_bm] = df[list_bm].apply(binary_map)

price = df['price']
area = df['area']
bedroom = df['bedrooms']
bathroom = df['bathrooms']
stories = df['stories']
mainroad = df['mainroad']
guestroom = df['guestroom']
basement = df['basement']
waterheating = df['hotwaterheating']
ac = df['airconditioning']
parking = df['parking']
prefarea = df['prefarea']
furnished = df['furnishingstatus']

np.random.seed(0)
df_train, df_test = train_test_split(df, train_size = 0.8, test_size = 0.2,
↳ random_state = 42)

num_vars = ['airconditioning', 'bedrooms', 'bathrooms', 'stories', 'parking',
↳ 'mainroad', 'guestroom', 'basement', 'prefarea', 'hotwaterheating', 'area', 'price']
df_newTrain = df_train[num_vars]
df_newTest = df_test[num_vars]

theta_0 = 0
theta_1 = 1
a = 0.01
X = []
theta_old = []
theta_new = []
cost_history = []
loss = []
h_pred = []
h_pred2 = []
theta = []
lamda = 100

scaler = MinMaxScaler()
df_newTrain[num_vars] = scaler.fit_transform(df_newTrain[num_vars])
y_newTrain = df_newTrain.pop('price')
x_newTrain = df_newTrain

```

```

Y = y_newTrain.values

X0 = df_newTrain.values[:,0]
X1 = df_newTrain.values[:,1]
X2 = df_newTrain.values[:,2]
X3 = df_newTrain.values[:,3]
X4 = df_newTrain.values[:,4]
X5 = df_newTrain.values[:,5]
X6 = df_newTrain.values[:,6]
X7 = df_newTrain.values[:,7]
X8 = df_newTrain.values[:,8]
X9 = df_newTrain.values[:,9]
X10 = df_newTrain.values[:,10]

for i in range(len(x_newTrain)):
    x_list = X0[i] + X1[i] + X2[i] + X3[i] + X4[i] + X5[i] + X6[i] + X7[i] + X8[i] +
    ↪X9[i] + X10[i]
    X.append(x_list)

X = sorted(X)

for i in range(len(X)):
    h = (X[i] * theta_1) + theta_0
    h_pred.append(h)
    errors = (h_pred[i] - Y[i])
    sqrErrors = np.square(errors)
    j = np.sum(sqrErrors)
    loss.append(j/(2*m))

loss = sorted(loss)

for j in range(len(X)):
    pred = theta_0 + (theta_1 * loss[j])
    h_pred2.append(pred)
    theta_old.append(pred)
    r = sum(h_pred2)
    errors = h_pred2[j] - (h_pred[j] + Y[j]) + (lamda * (r)**2)
    sum_delta = (theta_old[j] - errors)
    theta_new.append(theta_old[j])
    j_theta = theta_new[j] - sum_delta
    cost = a * (np.square(errors))
    cost_history.append(cost/m)

cost_history = sorted(cost_history)

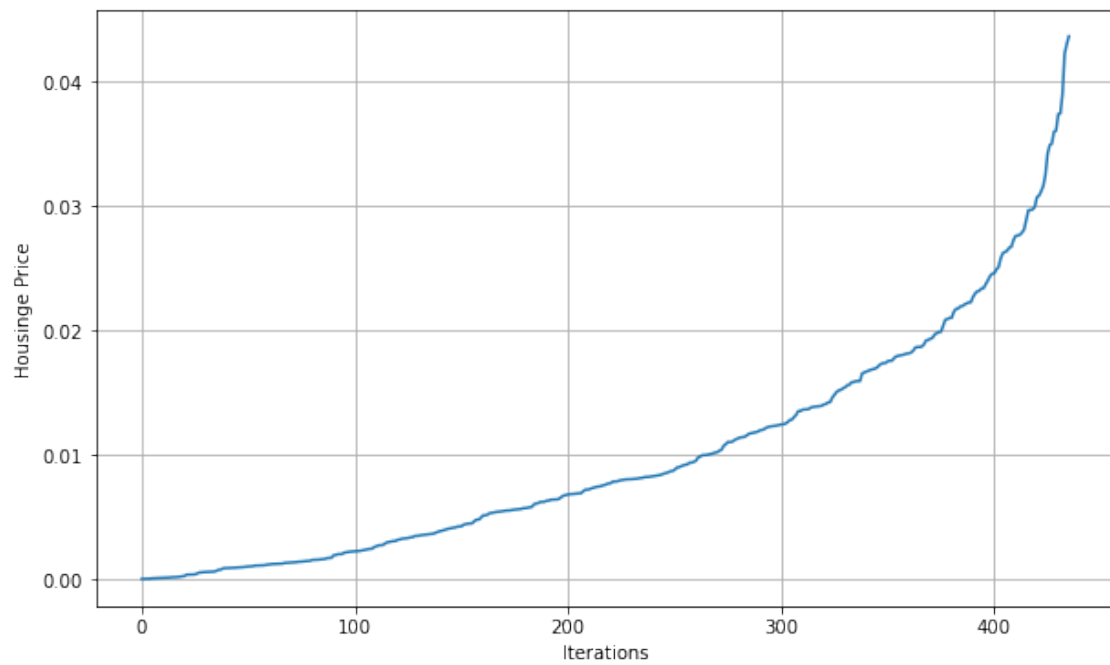
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.plot(loss)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

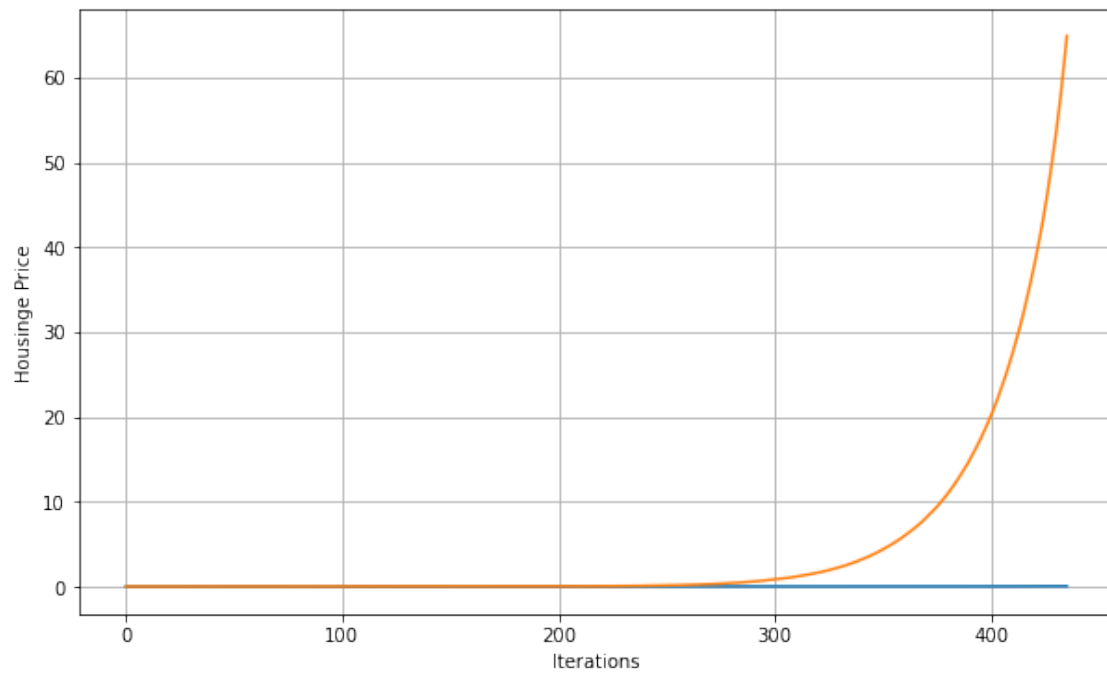
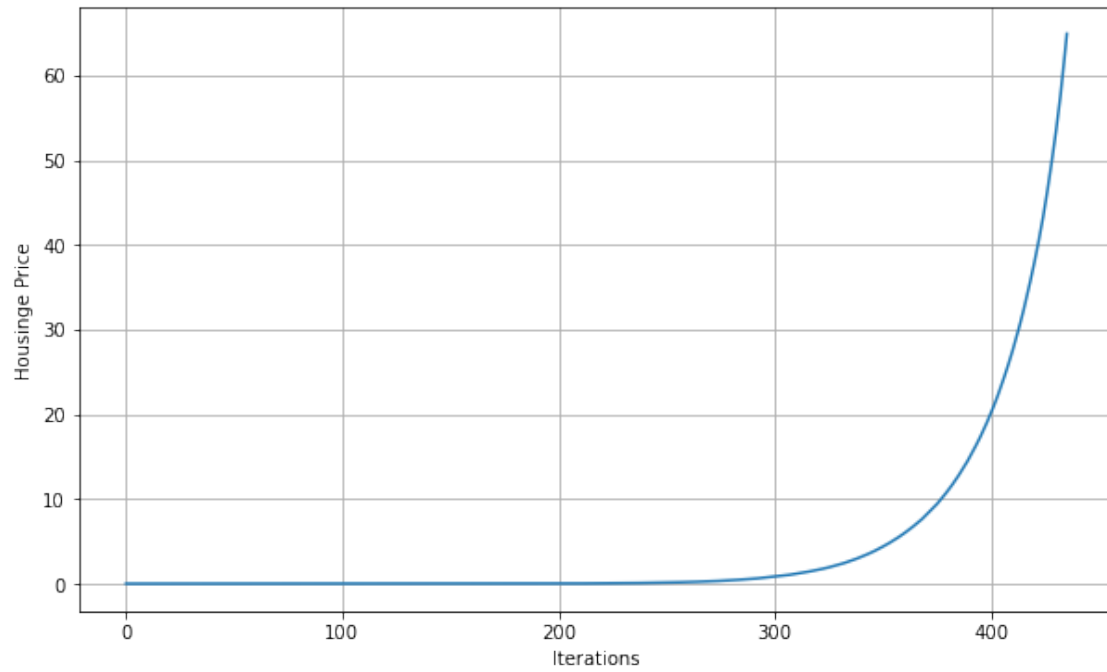
plt.grid()

```

```
plt.plot(cost_history)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

plt.grid()
plt.plot(loss)
plt.plot(cost_history)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()
```





```
[699]: # ===== 3b Normalization =====  
import pandas as pd  
import numpy as np  
import matplotlib
```

```

import matplotlib.pyplot as plt
import sklearn.datasets as dt
from sklearn.model_selection import train_test_split
df = pandas.read_csv('Housing.csv', header = 0)

list_bm = ['mainroad', 'guestroom', 'basement', 'hotwaterheating',
↪ 'airconditioning', 'prefarea']

def binary_map(x):
    return x.map({'yes': 1, 'no': 0})

df[list_bm] = df[list_bm].apply(binary_map)

price = df['price']
area = df['area']
bedroom = df['bedrooms']
bathroom = df['bathrooms']
stories = df['stories']
mainroad = df['mainroad']
guestroom = df['guestroom']
basement = df['basement']
waterheating = df['hotwaterheating']
ac = df['airconditioning']
parking = df['parking']
prefarea = df['prefarea']
furnished = df['furnishingstatus']

np.random.seed(0)
df_train, df_test = train_test_split(df, train_size = 0.8, test_size = 0.2,
↪ random_state = 42)

num_vars = ['airconditioning', 'bedrooms', 'bathrooms', 'stories', 'parking',
↪ 'mainroad', 'guestroom', 'basement', 'prefarea', 'hotwaterheating', 'area', 'price']
df_newTrain = df_train[num_vars]
df_newTest = df_test[num_vars]

theta_0 = 0
theta_1 = 0
a = 0.01
X = []
theta_old = 0
theta_new = 0
cost_history = []
loss = []
h_pred = []
h_pred2 = []
theta = []
lamda = 200

scaler = MinMaxScaler()
df_newTrain[num_vars] = scaler.fit_transform(df_newTrain[num_vars])
y_newTrain = df_newTrain.pop('price')
x_newTrain = df_newTrain

```

```

Y = y_newTrain.values

X0 = df_newTrain.values[:,0]
X1 = df_newTrain.values[:,1]
X2 = df_newTrain.values[:,2]
X3 = df_newTrain.values[:,3]
X4 = df_newTrain.values[:,4]
X5 = df_newTrain.values[:,5]
X6 = df_newTrain.values[:,6]
X7 = df_newTrain.values[:,7]
X8 = df_newTrain.values[:,8]
X9 = df_newTrain.values[:,9]
X10 = df_newTrain.values[:,10]

for i in range(len(x_newTrain)):
    x_list = X0[i] + X1[i] + X2[i] + X3[i] + X4[i] + X5[i] + X6[i] + X7[i] + X8[i] +
    ↪X9[i] + X10[i]
    X.append(x_list)

X = sorted(X)

for i in range(len(X)):
    h = (X[i] * theta_1) + theta_0
    h_pred.append(h)
    errors = (h_pred[i] - Y[i])
    sqrErrors = np.square(errors)
    j = np.sum(sqrErrors)
    loss.append(j/(m))

loss = sorted(loss)

for j in range(len(X)):
    pred = theta_0 + (theta_1 * loss[j])
    h_pred2.append(pred)
    r = sum(h_pred2)
    reg = r**2
    errors = (h_pred2[j] - (h_pred[j] + Y[j]))
    sum_delta = (h_pred2[j] - errors)
    theta_new = theta_old - sum_delta
    theta.append(theta_new)
    cost = a * (np.square(errors) + lamda * reg)
    cost_history.append(cost/m)

cost_history = sorted(cost_history)

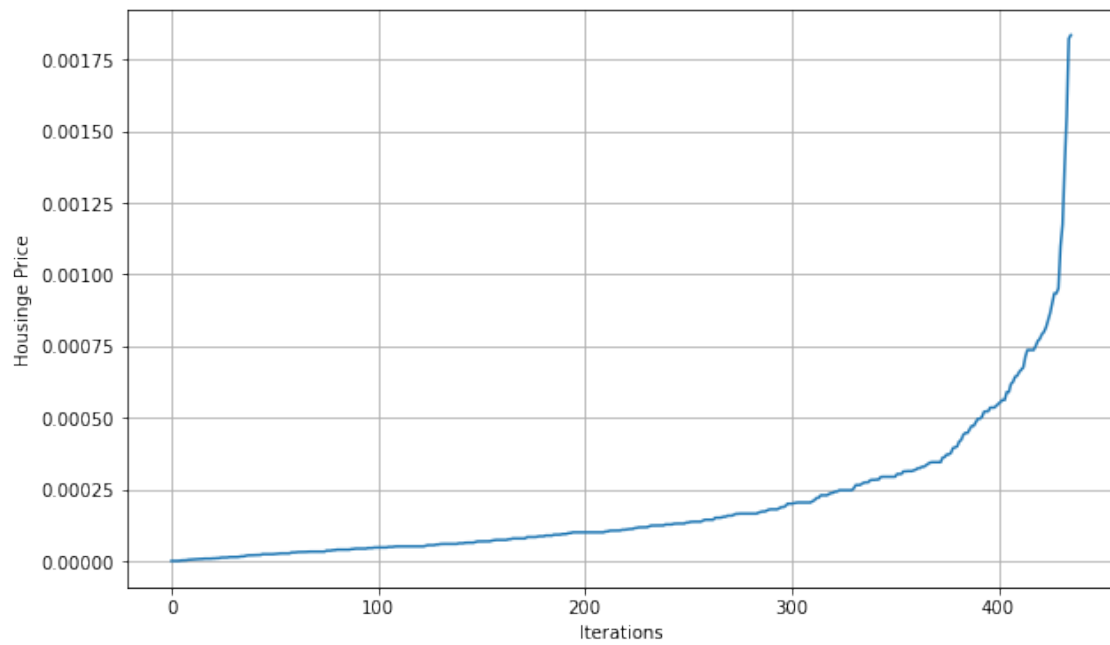
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.plot(loss)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()

plt.grid()

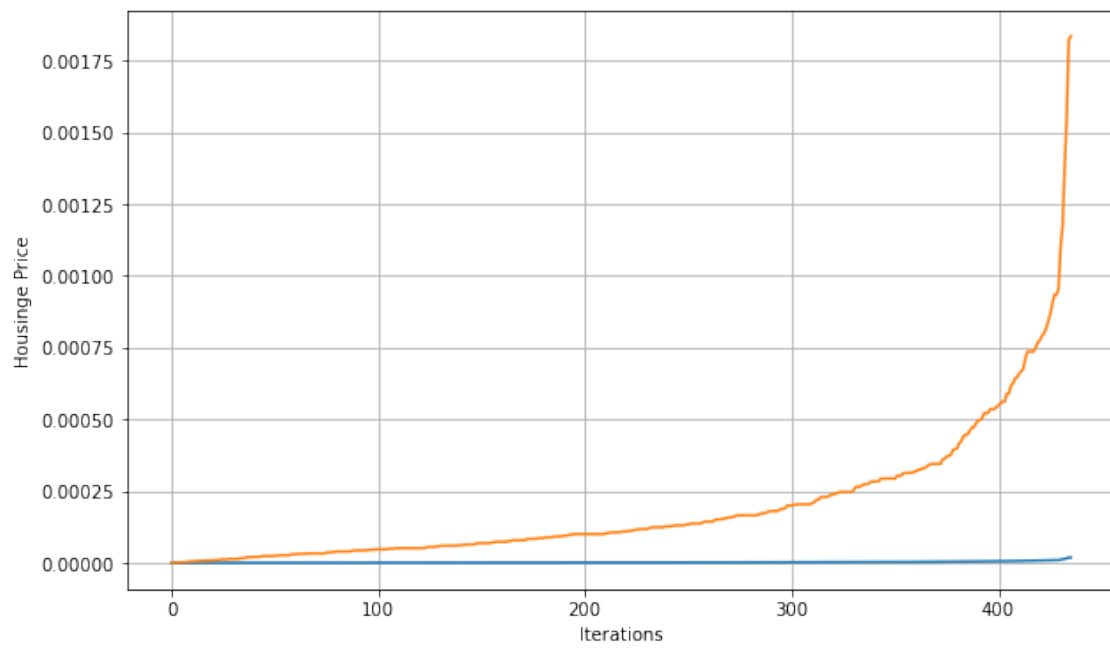
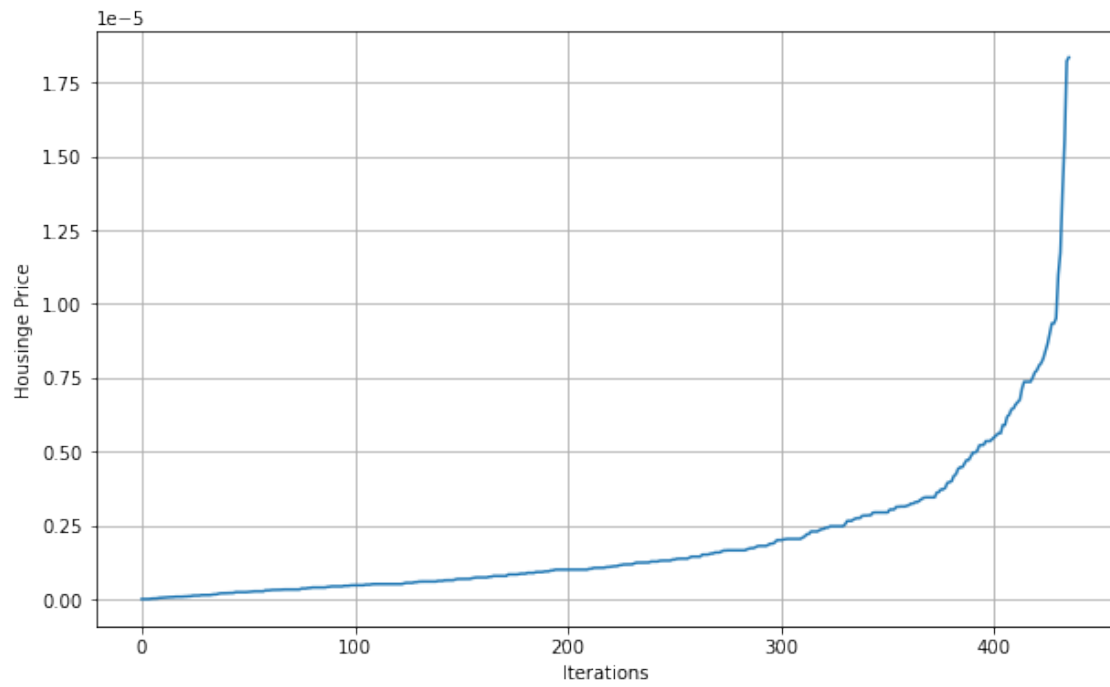
```

```
plt.plot(cost_history)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()
```

```
plt.grid()
plt.plot(cost_history)
plt.plot(loss)
plt.xlabel('Iterations')
plt.ylabel('Housing Price')
plt.show()
```







[ ]: