In [2]:
```
#import libraries
import numpy as np
import pandas as pd
from sklearn import model_selection
import matplotlib.pyplot as plt
import seaborn as sb
data=pd.read_csv(r"D:/AI_Machinlearning/datasets/ENERGY/data_energy.csv")
data.head(10)
```

Out[2]:

	X0	X1	X2	X3	X4	X5	X6	Y
0	1	200	169.20	3.0	0.06	-7.5	0	25.4697
1	1	200	169.20	3.0	0.06	-7.5	1	23.3763
2	1	200	169.20	3.0	0.06	-7.5	2	21.0503
3	1	200	165.60	3.0	0.08	-7.5	0	24.8882
4	1	200	165.60	3.0	0.08	-7.5	1	22.7948
5	1	200	165.60	3.0	0.08	-7.5	2	20.5851
6	1	200	162.00	3.0	0.10	-7.5	0	24.4230
7	1	200	162.00	3.0	0.10	-7.5	1	22.3296
8	1	200	162.00	3.0	0.10	-7.5	2	20.1199
9	1	200	157.92	2.8	0.06	-7.5	0	22.2133

In [3]:
```
data.describe()
```

Out[3]:

	X0	X1	X2	X3	X4	X5	X6	Y
count	648.0	648.000000	648.00000	648.000000	648.000000	648.000000	648.000000	648.000000
mean	1.0	131.500000	127.60463	2.966667	0.080000	-6.027778	1.000000	16.502037
std	0.0	56.736681	27.84246	0.124818	0.016343	1.500901	0.817127	4.088046
min	1.0	70.000000	83.88000	2.800000	0.060000	-7.500000	0.000000	8.722500
25%	1.0	83.000000	102.94750	2.800000	0.060000	-7.500000	0.000000	13.374500
50%	1.0	116.000000	122.63500	3.000000	0.080000	-7.500000	1.000000	15.816800
75%	1.0	171.500000	152.71000	3.100000	0.100000	-4.500000	2.000000	19.305800
max	1.0	240.000000	186.49000	3.100000	0.100000	-4.500000	2.000000	28.958700

In [5]:
```
#correlation
d=data.drop([ "X0"] , axis= 1)

corr_matrix=d.corr()
corr_matrix
```

Out[5]:

	X1	X2	X3	X4	X5	X6	Y
X1	1.000000e+00	9.627679e-01	1.760304e-15	-3.494011e-16	-0.012360	-5.779923e-17	0.841644
X2	9.627679e-01	1.000000e+00	1.940204e-01	-8.882735e-02	-0.012028	-5.213296e-17	0.905232
X3	1.760304e-15	1.940204e-01	1.000000e+00	-7.426370e-17	-0.002475	2.060982e-17	0.329717
X4	-3.494011e-16	-8.882735e-02	-7.426370e-17	1.000000e+00	0.011342	5.621789e-18	-0.072698
X5	-1.236024e-02	-1.202806e-02	-2.475066e-03	1.134218e-02	1.000000	7.561450e-03	-0.227335
X6	-5.779923e-17	-5.213296e-17	2.060982e-17	5.621789e-18	0.007561	1.000000e+00	-0.297251
Y	8.416439e-01	9.052316e-01	3.297175e-01	-7.269848e-02	-0.227335	-2.972512e-01	1.000000

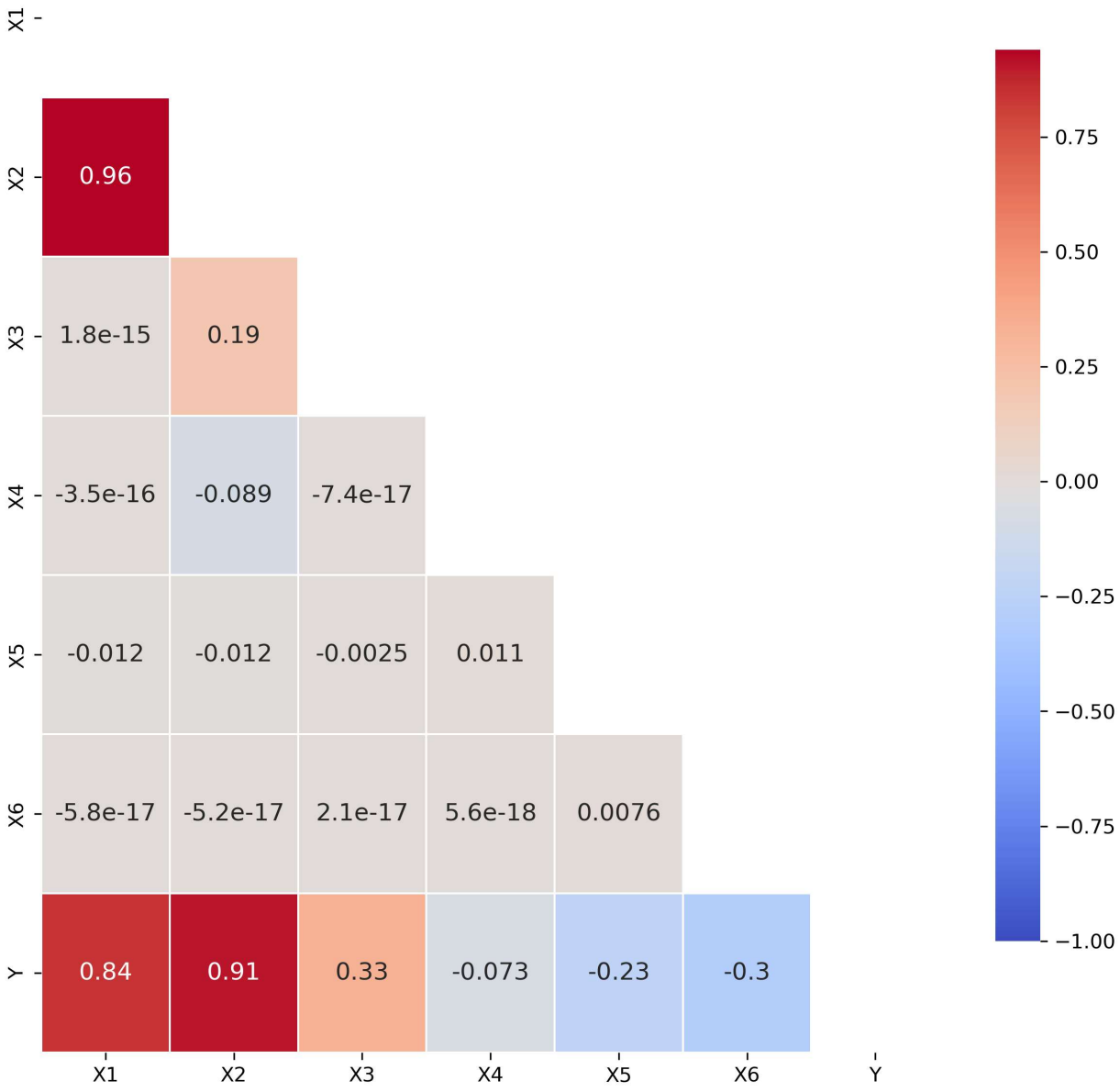
In [6]:
```
mask=np.zeros_like(corr_matrix , dtype=bool)
mask[np.triu_indices_from(mask)] = True

print(mask.shape)
print(corr_matrix.corr().shape)
```

```
(7, 7)
(7, 7)
```

In [7]:
```
#correlation graph
plt.figure(figsize=(10,10) , dpi=300)

sb.heatmap(corr_matrix , vmin=-1 ,cmap='coolwarm', annot=True , robust = True , cbar=
           cbar_kws={"shrink":0.8}, annot_kws={"size":12} ,  linewidth=0.5)
```

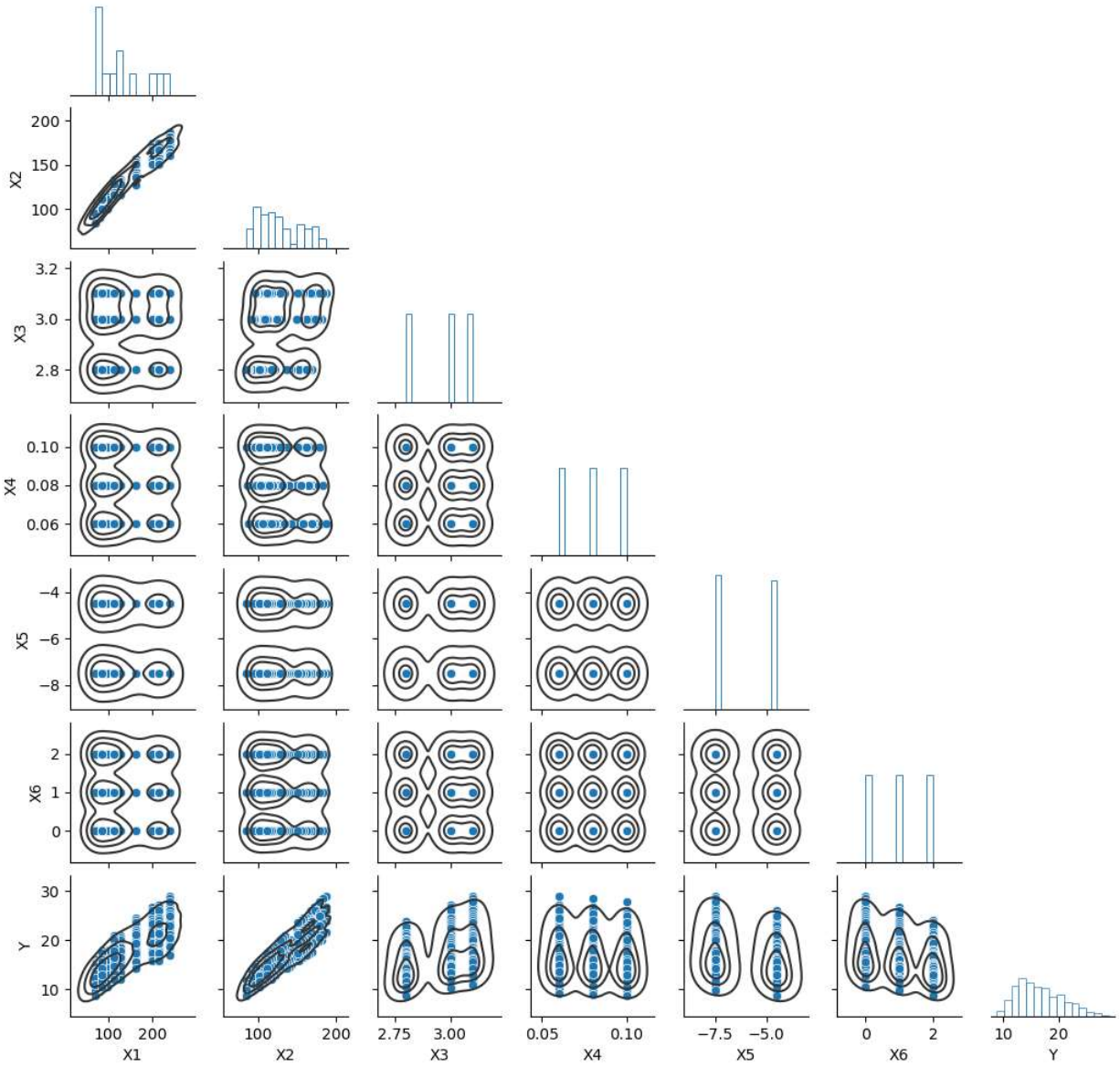
Out[7]:

```
<Axes: >
```



In [12]:
```
plt.figure(figsize=(2,2) , dpi=300 )


g = sb.pairplot(d, diag_kind="auto", corner= True  ,aspect=1 , height=1.5, diag_kws=di
g.map_lower(sb.kdeplot, levels=4, color=".2" )
```

Out[12]:
```
<seaborn.axisgrid.PairGrid at 0x21d36e24dc0>

<Figure size 600x600 with 0 Axes>
```



In [19]:
```
X= np.array(data.drop(["Y" , "X0"] , axis= 1))
y=np.array(data["Y"])
```

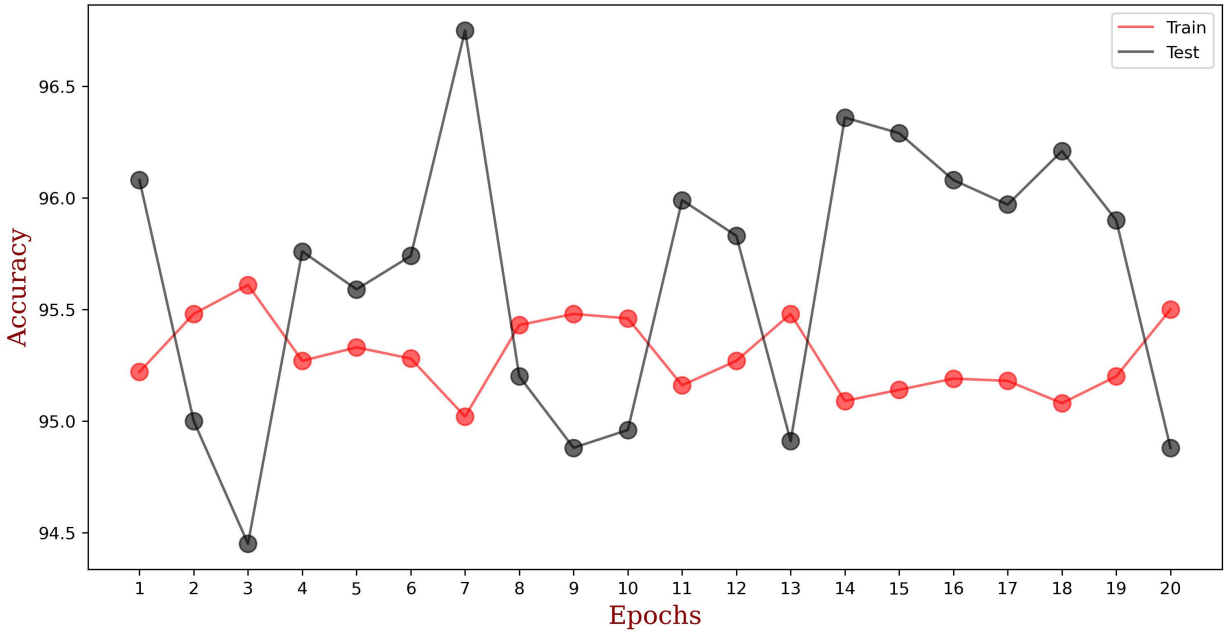
In [36]:
```
#train and accuracy
acctrain=[]
acctest=[]
for i in range(1,21):
    X_train, X_test, y_train, y_test = model_selection.train_test_split(X, y, test_siz
    if i==1:
        print('X_train dimension= ', X_train.shape)
        print('X_test dimension= ', X_test.shape)
        print('y_train dimension= ', y_train.shape)
        print('y_test dimension= ', y_test.shape)
        print("---------------------------------")
        theta= np.linalg.pinv(X_train)@y_train
    y_hat= X_train@theta
    numerator=sum ((y_train - y_hat)**2)
    Denominator=sum ((y_train - y_train.mean())**2)
    rse= numerator / Denominator
    R_squared= 1 - rse
```

```
    acctrain.append(round(float(R_squared*100), 2))
    y_hattest= X_test@theta
    numerator2=sum ((y_test - y_hattest)**2)
    Denominator2=sum ((y_test - y_test.mean())**2)
    rsetest= numerator2 / Denominator2
    R_squared2= 1 - rsetest
    acctest.append(round(float(R_squared2*100), 2))




print(acctrain)
print(acctest)
print("----------------------------------")
print(np.array(acctrain).mean())
print(np.array(acctest).mean())
```

```
X_train dimension=  (518, 6)
X_test dimension=  (130, 6)
y_train dimension=  (518,)
y_test dimension=  (130,)
----------------------------------
[95.22, 95.46, 95.58, 95.26, 95.32, 95.26, 95.0, 95.42, 95.47, 95.43, 95.12, 95.26, 9
5.47, 95.08, 95.13, 95.18, 95.15, 95.05, 95.19, 95.47]
[96.08, 95.08, 94.59, 95.81, 95.63, 95.79, 96.88, 95.23, 94.89, 95.07, 96.2, 95.85, 9
4.97, 96.42, 96.31, 96.1, 96.16, 96.35, 95.97, 95.01]
----------------------------------
95.276
95.71950000000001
```

In [21]:
```
np.set_printoptions(precision=2)
```

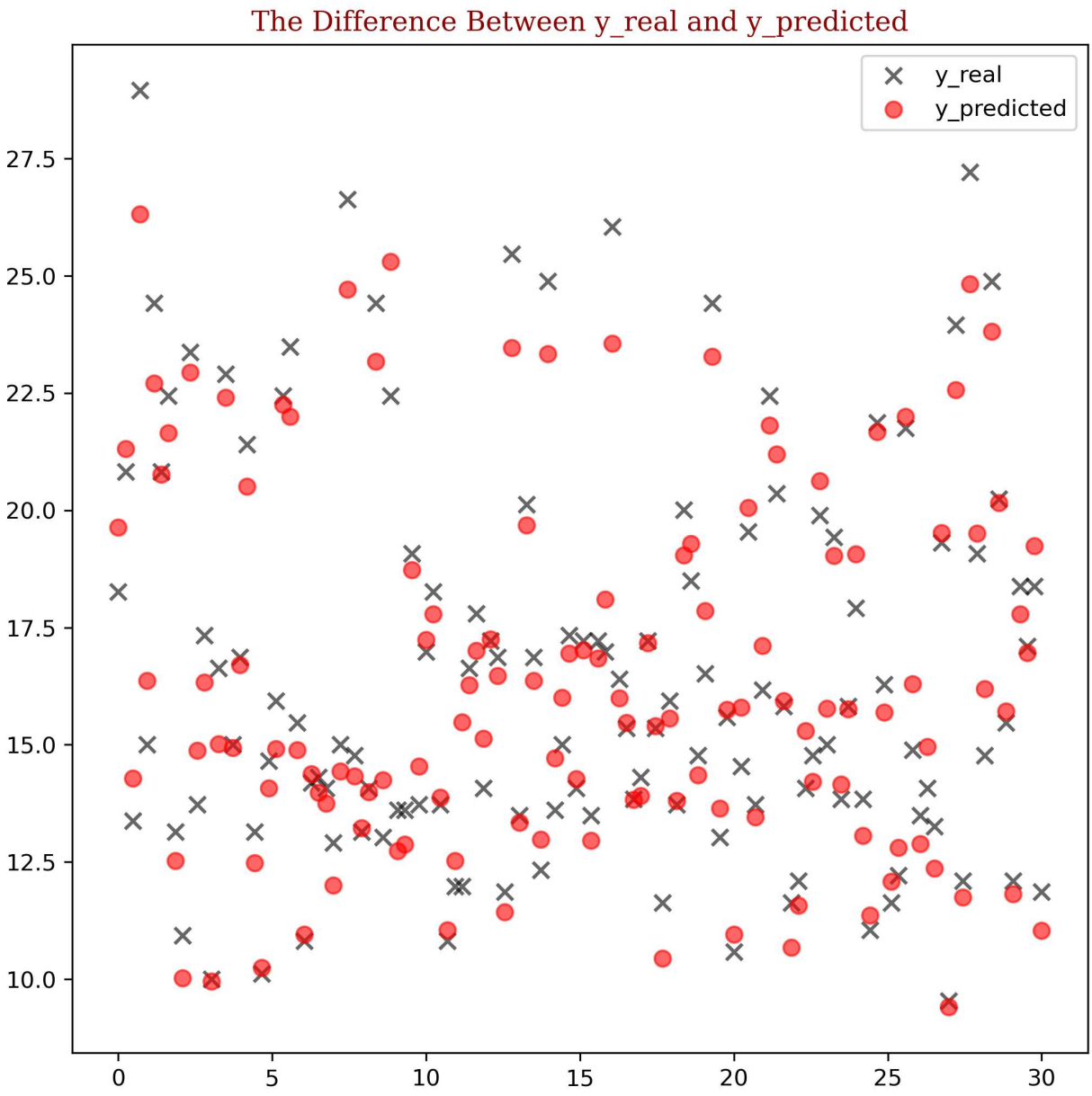
In [22]:
```
fig, ax = plt.subplots(figsize=(12,6) , dpi=300)
#fig.suptitle('test title', fontsize=120)

font1 = {'family':'serif','color':'darkred','size':12}
font2 = {'family':'serif','color':'darkred','size':15}

x4= [i for i in range (1,21)]

plt.xticks(np.arange(min(x4), max(x4)+1, 1.0))


ax.set_xlabel('Epochs', fontsize = 15,fontdict = font2)
ax.set_ylabel('Accuracy', fontsize =15, fontdict = font2)


#plt.legend(["Train" , "Test"], loc="upper left" , fontsize="40")


plt.scatter(x4 , acctrain, alpha=0.6 , c="red",s = 100)
plt.scatter(x4 , acctest, alpha=0.6 , c="black" , s=100)
plt.plot(x4 , acctrain, alpha=0.6 , color="red", label='Train')
plt.plot(x4 , acctest, alpha=0.6 , color="black", label='Test')
plt.legend()

plt.show()
```



In [23]:
```
import pandas as pd

d={"train_accuracy":acctrain   , "test_accuracy": acctest  }

dataframe=pd.DataFrame(d , index=range(1,21))
display(dataframe)
```

	train_accuracy	test_accuracy
1	95.22	96.08
2	95.48	95.00
3	95.61	94.45
4	95.27	95.76
5	95.33	95.59
6	95.28	95.74
7	95.02	96.75
8	95.43	95.20
9	95.48	94.88
10	95.46	94.96
11	95.16	95.99
12	95.27	95.83
13	95.48	94.91
14	95.09	96.36
15	95.14	96.29
16	95.19	96.08
17	95.18	95.97
18	95.08	96.21
19	95.20	95.90
20	95.50	94.88

In [30]:
```
fig = plt.figure(figsize=(8, 8) , dpi=300)
ax = fig.add_subplot(111)

x1=np.linspace(0,30,num=130)

plt.scatter(x1 , y_test, alpha=0.6 , c="black" , marker="x" , s=50  , label='y_real')
plt.scatter(x1 , y_hattest, alpha=0.6 , c="red" , s=50,label= "y_predicted" )
plt.legend()
plt.title("The Difference Between y_real and y_predicted" , font1)
#plt.scatter(y_hattest, y_test , c="red" , alpha=0.6)
plt.plot()
plt.show()
```


The Difference Between y_real and y_predicted
y_real
y_predicted
27.5
25.0
22.5
20.0
17.5
15.0
12.5
10.0
0
5
10
15
20
25
30