Assignment 1A - Question 1

Regression

In [1]:

```
import pandas
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm
import scipy.stats as stats
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.preprocessing import PolynomialFeatures
```

import data

Data imported using pandas.read_csv

Check data using .head() and .shape

In [2]:

```
data = pandas.read_csv('CAB420_Assessment_1A_Data\Data\Q1\communities.csv')
data.head()
```

Out[2]:

	state	county	community	communityname string	fold	population	householdsize	racepctblac
0	8	?	?	Lakewoodcity	1	0.19	0.33	0.(
1	53	?	?	Tukwilacity	1	0.00	0.16	0.1
2	24	?	?	Aberdeentown	1	0.00	0.42	0.4
3	34	5	81440	Willingborotownship	1	0.04	0.77	1.(
4	42	95	6096	Bethlehemtownship	1	0.01	0.55	0.(

5 rows × 128 columns

In [3]:

```
data.shape
```

Out[3]:

```
(1994, 128)
```

remove first 5 columns from dataset and remove all non-values

Change "?" to NaN and use dropna() function

In [4]:

```
data_to_use = data.iloc[:,5:]
data_to_use[data_to_use == "?"] = np.nan # change all "?" to NaN
data_to_use = data_to_use.dropna()
```

Check values as above

In [5]:

```
data_to_use.shape
```

Out[5]:

```
(319, 123)
```

In [6]:

```
data_to_use.head()
```

Out[6]:

	population	householdsize	racepctblack	racePctWhite	racePctAsian	racePctHisp	agePc
0	0.19	0.33	0.02	0.90	0.12	0.17	
16	0.15	0.31	0.40	0.63	0.14	0.06	
20	0.25	0.54	0.05	0.71	0.48	0.30	
21	1.00	0.42	0.47	0.59	0.12	0.05	
23	0.11	0.43	0.04	0.89	0.09	0.06	

5 rows × 123 columns

In [7]:

```
columns = data_to_use.columns.tolist()
print(columns)
```

```
[' population ', ' householdsize ', ' racepctblack ', ' racePctWhite ', '
racePctAsian ', ' racePctHisp ', ' agePct12t21 ', ' agePct12t29 ', ' agePc
t16t24 ', ' agePct65up ', ' numbUrban ', ' pctUrban ', ' medIncome ', ' pc
tWWage ', ' pctWFarmSelf ', ' pctWInvInc ', ' pctWSocSec ', ' pctWPubAsst
', ' pctWRetire ', ' medFamInc ', ' perCapInc ', ' whitePerCap ', ' blackP
erCap ', ' indianPerCap ', ' AsianPerCap ', ' OtherPerCap ', ' HispPerCap
', ' NumUnderPov ', ' PctPopUnderPov ', ' PctLess9thGrade ', ' PctNotHSGra
d ', ' PctBSorMore ', ' PctUnemployed ', ' PctEmploy ', ' PctEmplManu ', '
PctEmplProfServ ', ' PctOccupManu ', ' PctOccupMgmtProf ', ' MalePctDivorc
e ', ' MalePctNevMarr ', ' FemalePctDiv ', ' TotalPctDiv ', ' PersPerFam
', ' PctFam2Par ', ' PctKids2Par ', ' PctYoungKids2Par ', ' PctTeen2Par ',
' PctWorkMomYoungKids ', ' PctWorkMom ', ' NumIlleg ', ' PctIlleg ', ' Num
Immig ', ' PctImmigRecent ', ' PctImmigRec5 ', ' PctImmigRec8 ', ' PctImmi
gRec10 ', ' PctRecentImmig ', ' PctRecImmig5 ', ' PctRecImmig8 ', ' PctRec
Immig10 ', ' PctSpeakEnglOnly ', ' PctNotSpeakEnglWell ', ' PctLargHouseFa
m ', ' PctLargHouseOccup ', ' PersPerOccupHous ', ' PersPerOwnOccHous ', '
PersPerRentOccHous ', ' PctPersOwnOccup ', ' PctPersDenseHous ', ' PctHous
Less3BR ', ' MedNumBR ', ' HousVacant ', ' PctHousOccup ', ' PctHousOwnOcc
', ' PctVacantBoarded ', ' PctVacMore6Mos ', ' MedYrHousBuilt ', ' PctHous
NoPhone ', ' PctWOFullPlumb ', ' OwnOccLowQuart ', ' OwnOccMedVal ', ' Own
OccHiQuart ', ' RentLowQ ', ' RentMedian ', ' RentHighQ ', ' MedRent ', '
MedRentPctHousInc ', ' MedOwnCostPctInc ', ' MedOwnCostPctIncNoMtg ', ' Nu
mInShelters ', ' NumStreet ', ' PctForeignBorn ', ' PctBornSameState ', '
PctSameHouse85 ', ' PctSameCity85 ', ' PctSameState85 ', ' LemasSwornFT ',
' LemasSwFTPerPop ', ' LemasSwFTFieldOps ', ' LemasSwFTFieldPerPop ', ' Le
masTotalReq ', ' LemasTotReqPerPop ', ' PolicReqPerOffic ', ' PolicPerPop
', ' RacialMatchCommPol ', ' PctPolicWhite ', ' PctPolicBlack ', ' PctPoli
cHisp ', ' PctPolicAsian ', ' PctPolicMinor ', ' OfficAssgnDrugUnits ', '
NumKindsDrugsSeiz ', ' PolicAveOTWorked ', ' LandArea ', ' PopDens ', ' Pc
tUsePubTrans ', ' PolicCars ', ' PolicOperBudg ', ' LemasPctPolicOnPatr ',
' LemasGangUnitDeploy ', ' LemasPctOfficDrugUn ', ' PolicBudgPerPop ', ' V
iolentCrimesPerPop ']
```

plot data_to_use

Code taken from Week 2 Example 1

In [8]:

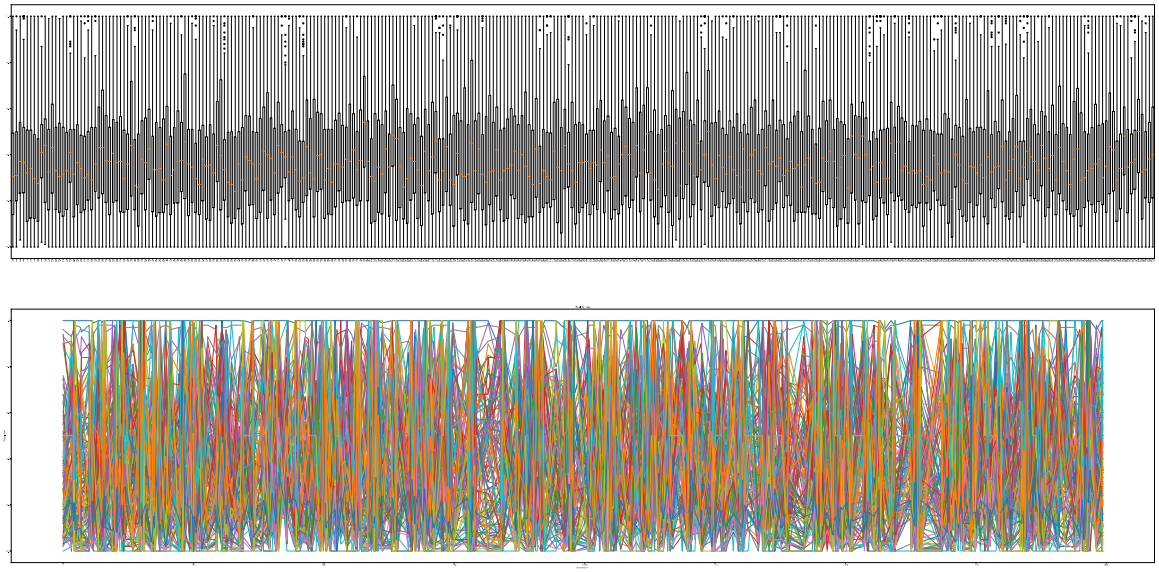
```
for column in data_to_use.iloc[:,1:]:
    data_to_use[column] = pandas.to_numeric(data_to_use[column], downcast="float")
```

In [9]:

```
fig = plt.figure()
fig.set_figheight(50)
fig.set_figwidth(100)
ax = fig.add_subplot(2, 1, 1)
ax.boxplot(data_to_use.iloc[:,1:].transpose())
ax = fig.add_subplot(2, 1, 2)
ax.plot(data_to_use.iloc[:,1:])
ax.set_xlabel('Observation')
ax.set_ylabel('Population')
ax.set_title('Total Crimes')
```

Out[9]:

```
Text(0.5, 1.0, 'Total Crimes')
```



Linear Regression

Covert Data to Categorical

Code taken from Week 2 Example 1

In [10]:

```
print(data_to_use[" population "].astype("category"))
```

```
0       0.19
16      0.15
20      0.25
21      1.00
23      0.11
        ...
1969    1.00
1981    0.07
1991    0.16
1992    0.08
1993    0.20
Name:  population , Length: 319, dtype: category
Categories (65, float64): [0.00, 0.01, 0.02, 0.03, ..., 0.91, 0.96, 0.99,
1.00]
C:\Python39\lib\site-packages\pandas\io\formats\format.py:1403: FutureWarn
ing: Index.ravel returning ndarray is deprecated; in a future version this
will return a view on self.
  for val, m in zip(values.ravel(), mask.ravel())
C:\Python39\lib\site-packages\pandas\io\formats\format.py:1403: FutureWarn
ing: Index.ravel returning ndarray is deprecated; in a future version this
will return a view on self.
  for val, m in zip(values.ravel(), mask.ravel())
```

In [11]:

```
dummies = pandas.get_dummies(data_to_use[" population "].astype("category"), drop_first
=True)
X = pandas.concat([data_to_use.iloc[:,1:-1], dummies], axis=1)
X = sm.add_constant(X)
Y = data_to_use.iloc[:,-1]
print(X)
print(Y)
```

```
      const   householdsize    racepctblack    racePctWhite    racePctAsia
n   \
0       1.0            0.33            0.02            0.90            0.
12
16      1.0            0.31            0.40            0.63            0.
14
20      1.0            0.54            0.05            0.71            0.
48
21      1.0            0.42            0.47            0.59            0.
12
23      1.0            0.43            0.04            0.89            0.
09
...     ...             ...             ...             ...
...
1969    1.0            0.29            0.21            0.29            1.
00
1981    1.0            0.38            0.17            0.84            0.
11
1991    1.0            0.37            0.25            0.69            0.
04
1992    1.0            0.51            0.06            0.87            0.
22
1993    1.0            0.78            0.14            0.46            0.
24

      racePctHisp    agePct12t21    agePct12t29    agePct16t24   \
0            0.17           0.34           0.47           0.29
16           0.06           0.58           0.72           0.65
20           0.30           0.42           0.48           0.28
21           0.05           0.41           0.53           0.34
23           0.06           0.45           0.48           0.31
...           ...            ...            ...            ...
1969         0.26           0.24           0.47           0.28
1981         0.04           0.35           0.41           0.30
1991         0.25           0.35           0.50           0.31
1992         0.10           0.58           0.74           0.63
1993         0.77           0.50           0.62           0.40

      agePct65up   ...  0.7  0.73  0.74  0.78  0.8  0.81  0.91  0.96  0.9
9  \
0           0.32  ...    0     0     0     0    0     0     0     0
0
16          0.47  ...    0     0     0     0    0     0     0     0
0
20          0.32  ...    0     0     0     0    0     0     0     0
0
21          0.33  ...    0     0     0     0    0     0     0     0
0
23          0.46  ...    0     0     0     0    0     0     0     0
0
...          ...  ...  ...   ...   ...   ...  ...   ...   ...   ...
...
1969        0.46  ...    0     0     0     0    0     0     0     0
0
1981        0.64  ...    0     0     0     0    0     0     0     0
0
1991        0.54  ...    0     0     0     0    0     0     0     0
0
1992        0.41  ...    0     0     0     0    0     0     0     0
0
1993        0.17  ...    0     0     0     0    0     0     0     0
```

```
0

      1.0
0       0
16      0
20      0
21      1
23      0
...   ...
1969    1
1981    0
1991    0
1992    0
1993    0

[319 rows x 186 columns]
0       0.20
16      0.49
20      0.34
21      0.69
23      0.63
        ...
1969    0.75
1981    0.07
1991    0.23
1992    0.19
1993    0.48
Name:  ViolentCrimesPerPop , Length: 319, dtype: float32
```

Split data into training, test and validation sets

Code taken from Week 2 Example 1 train - 70% test - 15% validation - 15%

In [12]:

```
num_samples = data_to_use.shape[0]
training_samples = int(num_samples*0.7)
validation_samples = int(num_samples*0.15)
X_train = X.iloc[0:training_samples, :]
Y_train = Y.iloc[0:training_samples]
X_val = X.iloc[training_samples:(training_samples + validation_samples), :]
Y_val = Y.iloc[training_samples:(training_samples + validation_samples)]
X_test = X.iloc[(training_samples + validation_samples):, :]
Y_test = Y.iloc[(training_samples + validation_samples):]
```

Check values

In [13]:

```
print(X_train.shape)
print(Y_train.shape)
print(X_val.shape)
print(Y_val.shape)
print(X_test.shape)
print(Y_test.shape)
```

```
(223, 186)
(223,)
(47, 186)
(47,)
(49, 186)
(49,)
```

Model using sm.OLS

Code taken from Week 2 Example 1

In [14]:

```
model = sm.OLS(Y_train, X_train)
trained_model = model.fit()
print(trained_model.summary())
```

OLS Regression Results

Dep. Variable:	ViolentCrimesPerPop	R-squared:	0.942
Model:	OLS	Adj. R-squared:	0.720
Method:	Least Squares	F-statistic:	4.237
Date:	Sun, 11 Apr 2021	Prob (F-statistic):	7.72e-08
Time:	17:06:19	Log-Likelihood:	288.85
No. Observations:	223	AIC:	-223.7
Df Residuals:	46	BIC:	379.4
Df Model:	176		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.3247	4.698	0.069	0.945	-9.132	9.781
householdsize	2.0198	1.437	1.405	0.167	-0.873	4.913
racepctblack	-0.0693	0.432	-0.160	0.873	-0.940	0.801
racePctWhite	-0.1774	0.538	-0.330	0.743	-1.260	0.905
racePctAsian	0.1226	0.280	0.438	0.663	-0.441	0.686
racePctHisp	-0.3997	0.383	-1.045	0.302	-1.170	0.370
agePct12t21	2.3624	1.244	1.900	0.064	-0.141	4.866
agePct12t29	0.4263	1.409	0.303	0.763	-2.409	3.262
agePct16t24	-2.8353	1.697	-1.670	0.102	-6.252	0.582
agePct65up	1.1263	1.304	0.864	0.392	-1.498	3.750
numbUrban	6.0855	4.525	1.345	0.185	-3.023	15.194
pctUrban	-0.3357	0.435	-0.772	0.444	-1.211	0.539
medIncome	-0.5971	1.627	-0.367	0.715	-3.871	2.677
pctWWage	0.1532	1.038	0.148	0.883	-1.936	2.243
pctWFarmSelf	-0.1349	0.277	-0.487	0.628	-0.692	0.422
pctWInvInc	-1.2739	0.759	-1.678	0.100	-2.802	0.254
pctWSocSec	-1.1297	1.162	-0.972	0.336	-3.469	1.210
pctWPubAsst	-0.5122	0.522	-0.982	0.331	-1.562	0.538

```
 pctWRetire               -0.0487      0.368     -0.132      0.895       -
0.789       0.692
 medFamInc                 0.7721      1.146      0.674      0.504       -
1.534       3.078
 perCapInc                 0.1420      1.273      0.112      0.912       -
2.421       2.704
 whitePerCap              -0.4106      0.928     -0.442      0.660       -
2.279       1.458
 blackPerCap              -0.4534      0.446     -1.017      0.314       -
1.351       0.444
 indianPerCap             -0.0345      0.207     -0.166      0.869       -
0.451       0.382
 AsianPerCap               0.2041      0.230      0.889      0.379       -
0.258       0.666
 OtherPerCap               0.3906      0.265      1.476      0.147       -
0.142       0.923
 HispPerCap               -0.1625      0.323     -0.503      0.617       -
0.812       0.487
 NumUnderPov              -0.7978      0.725     -1.101      0.277       -
2.256       0.661
 PctPopUnderPov            0.6835      0.771      0.886      0.380       -
0.869       2.236
 PctLess9thGrade           0.1856      0.587      0.316      0.753       -
0.995       1.366
 PctNotHSGrad             -0.7643      0.796     -0.960      0.342       -
2.367       0.838
 PctBSorMore              -1.0607      0.701     -1.513      0.137       -
2.472       0.351
 PctUnemployed             0.3793      0.484      0.784      0.437       -
0.595       1.353
 PctEmploy                 0.2727      1.005      0.271      0.787       -
1.749       2.295
 PctEmplManu              -0.2784      0.253     -1.099      0.278       -
0.788       0.232
 PctEmplProfServ           0.3656      0.404      0.905      0.370       -
0.448       1.179
 PctOccupManu              0.5983      0.500      1.196      0.238       -
0.409       1.605
 PctOccupMgmtProf          0.9597      0.937      1.025      0.311       -
0.926       2.845
 MalePctDivorce            4.2759      2.696      1.586      0.120       -
1.151       9.703
 MalePctNevMarr            0.2602      0.842      0.309      0.759       -
1.434       1.954
 FemalePctDiv              4.3283      3.671      1.179      0.244       -
3.061      11.718
 TotalPctDiv              -9.2185      6.049     -1.524      0.134      -2
1.394       2.957
 PersPerFam               -1.2028      1.791     -0.672      0.505       -
4.807       2.401
 PctFam2Par               -1.0183      1.381     -0.737      0.465       -
3.798       1.762
 PctKids2Par               0.4302      1.363      0.316      0.754       -
2.314       3.174
 PctYoungKids2Par         -0.6026      0.571     -1.055      0.297       -
1.752       0.547
 PctTeen2Par              -0.4324      0.458     -0.943      0.350       -
1.355       0.490
 PctWorkMomYoungKids       0.2400      0.610      0.393      0.696       -
0.989       1.469
 PctWorkMom               -0.3964      0.664     -0.597      0.554       -
```

```
1.733       0.941
 NumIlleg                0.3183      0.840      0.379      0.706       -
1.372       2.009
 PctIlleg               -0.0930      0.393     -0.236      0.814       -
0.885       0.699
 NumImmig               -0.5356      0.492     -1.088      0.282       -
1.526       0.455
 PctImmigRecent         -0.7717      0.632     -1.222      0.228       -
2.043       0.500
 PctImmigRec5            0.6756      0.912      0.741      0.463       -
1.161       2.512
 PctImmigRec8           -0.5938      1.101     -0.539      0.592       -
2.810       1.622
 PctImmigRec10           0.7340      0.848      0.866      0.391       -
0.973       2.441
 PctRecentImmig          0.1372      1.016      0.135      0.893       -
1.907       2.181
 PctRecImmig5            0.4263      1.967      0.217      0.829       -
3.533       4.385
 PctRecImmig8            0.3483      2.537      0.137      0.891       -
4.759       5.456
 PctRecImmig10          -1.1269      1.960     -0.575      0.568       -
5.073       2.819
 PctSpeakEnglOnly        0.4526      0.571      0.793      0.432       -
0.696       1.602
 PctNotSpeakEnglWell     0.7768      0.532      1.459      0.151       -
0.295       1.849
 PctLargHouseFam         0.6379      1.800      0.354      0.725       -
2.986       4.262
 PctLargHouseOccup      -0.2976      2.133     -0.140      0.890       -
4.591       3.996
 PersPerOccupHous       -1.8658      2.842     -0.657      0.515       -
7.586       3.854
 PersPerOwnOccHous       0.4295      1.094      0.393      0.696       -
1.773       2.632
 PersPerRentOccHous     -1.2593      1.013     -1.243      0.220       -
3.299       0.780
 PctPersOwnOccup        -2.3551      2.245     -1.049      0.300       -
6.875       2.164
 PctPersDenseHous        0.5635      0.667      0.845      0.402       -
0.779       1.906
 PctHousLess3BR         -0.5988      0.622     -0.963      0.341       -
1.851       0.653
 MedNumBR               -0.0508      0.124     -0.411      0.683       -
0.300       0.198
 HousVacant             -0.3007      0.579     -0.519      0.606       -
1.467       0.865
 PctHousOccup           -0.0193      0.420     -0.046      0.964       -
0.864       0.825
 PctHousOwnOcc           2.7869      2.456      1.135      0.262       -
2.157       7.731
 PctVacantBoarded       -0.1159      0.168     -0.692      0.493       -
0.453       0.221
 PctVacMore6Mos         -0.3255      0.329     -0.989      0.328       -
0.988       0.337
 MedYrHousBuilt         -0.1362      0.248     -0.549      0.586       -
0.636       0.364
 PctHousNoPhone          0.2202      0.361      0.609      0.545       -
0.507       0.948
 PctWOFullPlumb         -0.4697      0.290     -1.619      0.112       -
1.054       0.114
```

```
 OwnOccLowQuart            -0.0727      1.521     -0.048      0.962      -
3.134       2.989
 OwnOccMedVal               0.0509      2.205      0.023      0.982      -
4.387       4.489
 OwnOccHiQuart             -0.3108      1.026     -0.303      0.763      -
2.377       1.755
 RentLowQ                  -0.5200      0.830     -0.627      0.534      -
2.190       1.150
 RentMedian                -0.5716      1.996     -0.286      0.776      -
4.589       3.446
 RentHighQ                  0.9920      0.956      1.038      0.305      -
0.932       2.916
 MedRent                    0.7688      1.414      0.544      0.589      -
2.077       3.614
 MedRentPctHousInc         -0.6105      0.366     -1.670      0.102      -
1.346       0.125
 MedOwnCostPctInc           0.2961      0.348      0.852      0.399      -
0.404       0.996
 MedOwnCostPctIncNoMtg      0.1847      0.274      0.675      0.503      -
0.366       0.736
 NumInShelters             -0.3761      0.206     -1.830      0.074      -
0.790       0.038
 NumStreet                  0.2323      0.177      1.310      0.197      -
0.125       0.589
 PctForeignBorn             0.2541      0.705      0.360      0.720      -
1.166       1.674
 PctBornSameState          -0.0838      0.519     -0.161      0.872      -
1.129       0.962
 PctSameHouse85            -0.3772      0.625     -0.604      0.549      -
1.635       0.880
 PctSameCity85              0.5889      0.476      1.236      0.223      -
0.370       1.548
 PctSameState85             0.0340      0.458      0.074      0.941      -
0.888       0.956
 LemasSwornFT               1.9384      3.237      0.599      0.552      -
4.578       8.455
 LemasSwFTPerPop            0.0400      0.553      0.072      0.943      -
1.073       1.153
 LemasSwFTFieldOps          0.4604      3.191      0.144      0.886      -
5.964       6.885
 LemasSwFTFieldPerPop      -0.3490      1.467     -0.238      0.813      -
3.301       2.603
 LemasTotalReq             -0.1287      0.763     -0.169      0.867      -
1.664       1.406
 LemasTotReqPerPop         -0.0325      0.418     -0.078      0.938      -
0.873       0.808
 PolicReqPerOffic           0.1844      0.259      0.713      0.480      -
0.336       0.705
 PolicPerPop                0.0400      0.553      0.072      0.943      -
1.073       1.153
 RacialMatchCommPol         0.0562      0.253      0.223      0.825      -
0.452       0.565
 PctPolicWhite             -0.3437      0.347     -0.991      0.327      -
1.042       0.354
 PctPolicBlack              0.2782      0.550      0.506      0.615      -
0.828       1.385
 PctPolicHisp               0.7756      0.638      1.215      0.230      -
0.509       2.060
 PctPolicAsian              0.3360      0.212      1.586      0.120      -
0.090       0.762
 PctPolicMinor             -1.0237      0.833     -1.228      0.226      -
```

```
2.701       0.654
 OfficAssgnDrugUnits        0.5258      0.938      0.560      0.578      -
1.363       2.414
 NumKindsDrugsSeiz         -0.0327      0.108     -0.302      0.764      -
0.251       0.186
 PolicAveOTWorked          -0.0268      0.111     -0.241      0.810      -
0.250       0.196
 LandArea                   0.2391      0.308      0.777      0.441      -
0.380       0.858
 PopDens                   -0.0718      0.215     -0.334      0.740      -
0.504       0.361
 PctUsePubTrans             0.1117      0.178      0.626      0.534      -
0.247       0.470
 PolicCars                  0.4876      0.389      1.253      0.216      -
0.295       1.271
 PolicOperBudg             -1.8154      1.476     -1.230      0.225      -
4.787       1.156
 LemasPctPolicOnPatr        0.0041      0.218      0.019      0.985      -
0.436       0.444
 LemasGangUnitDeploy        0.0331      0.056      0.595      0.555      -
0.079       0.145
 LemasPctOfficDrugUn       -0.1940      0.112     -1.727      0.091      -
0.420       0.032
 PolicBudgPerPop            0.6354      0.578      1.100      0.277      -
0.527       1.798
0.01                        1.3396      1.645      0.814      0.420      -
1.971       4.650
0.02                        1.9928      1.541      1.293      0.202      -
1.109       5.095
0.03                        1.8944      1.574      1.203      0.235      -
1.275       5.063
0.04                        1.5138      1.440      1.051      0.299      -
1.385       4.413
0.05                        1.8644      1.403      1.329      0.190      -
0.960       4.689
0.06                        1.5518      1.369      1.134      0.263      -
1.204       4.307
0.07                        1.5002      1.341      1.118      0.269      -
1.200       4.200
0.08                        1.4518      1.318      1.101      0.277      -
1.202       4.106
0.09                        1.4686      1.257      1.168      0.249      -
1.062       3.999
0.1                         1.4755      1.218      1.211      0.232      -
0.977       3.928
0.11                        1.3353      1.153      1.158      0.253      -
0.985       3.656
0.12                        1.3621      1.124      1.212      0.232      -
0.900       3.624
0.13                        1.1898      1.053      1.130      0.264      -
0.930       3.310
0.14                        1.1953      1.029      1.161      0.252      -
0.877       3.268
0.15                        1.0254      0.967      1.060      0.295      -
0.922       2.973
0.16                        1.0137      0.932      1.087      0.283      -
0.863       2.891
0.17                        1.2332      0.930      1.326      0.191      -
0.638       3.105
0.18                        1.1779      0.847      1.392      0.171      -
0.526       2.882
```

```
0.19                    0.9902      0.819      1.208      0.233      -
0.659      2.639
0.2                     0.8357      0.774      1.080      0.286      -
0.722      2.394
0.21                    0.7899      0.781      1.012      0.317      -
0.782      2.361
0.22                    0.7904      0.743      1.064      0.293      -
0.705      2.286
0.23                    0.7619      0.601      1.267      0.212      -
0.448      1.972
0.24                    0.5888      0.591      0.997      0.324      -
0.600      1.777
0.25                    0.8198      0.571      1.435      0.158      -
0.330      1.970
0.26                    0.3929      0.501      0.784      0.437      -
0.616      1.402
0.27                    0.4189      0.498      0.841      0.405      -
0.584      1.422
0.28                    0.4416      0.383      1.152      0.255      -
0.330      1.213
0.29                    0.4825      0.402      1.199      0.237      -
0.328      1.293
0.3                     0.3608      0.384      0.939      0.353      -
0.413      1.135
0.31                   -0.0608      0.357     -0.171      0.865      -
0.779      0.657
0.32                    0.4159      0.303      1.372      0.177      -
0.194      1.026
0.34                    0.6154      0.285      2.157      0.036
0.041      1.190
0.35                    0.1074      0.218      0.493      0.625      -
0.331      0.546
0.36                  3.66e-15   3.32e-15      1.103      0.276   -3.0
2e-15    1.03e-14
0.4                    -0.2144      0.286     -0.750      0.457      -
0.790      0.361
0.41                    0.0310      0.361      0.086      0.932      -
0.696      0.758
0.42                 -2.023e-15   1.73e-15     -1.173      0.247    -5.
5e-15    1.45e-15
0.43                  5.575e-15   3.32e-15      1.681      0.099    -1.
1e-15    1.22e-14
0.45                 -6.832e-15   4.41e-15     -1.548      0.128   -1.5
7e-14    2.05e-15
0.46                   -0.2163      0.533     -0.405      0.687      -
1.290      0.857
0.47                   -0.5754      0.524     -1.099      0.278      -
1.630      0.479
0.51                 -3.958e-15   3.39e-15     -1.168      0.249   -1.0
8e-14    2.86e-15
0.52                   -0.8164      0.717     -1.139      0.260      -
2.259      0.626
0.55                   -0.4481      0.822     -0.545      0.588      -
2.103      1.207
0.56                   -0.5799      1.088     -0.533      0.597      -
2.771      1.611
0.57                   -0.9859      0.981     -1.006      0.320      -
2.960      0.988
0.58                   -0.9278      0.948     -0.978      0.333      -
2.837      0.981
0.6                    -1.1152      1.121     -0.995      0.325      -
```

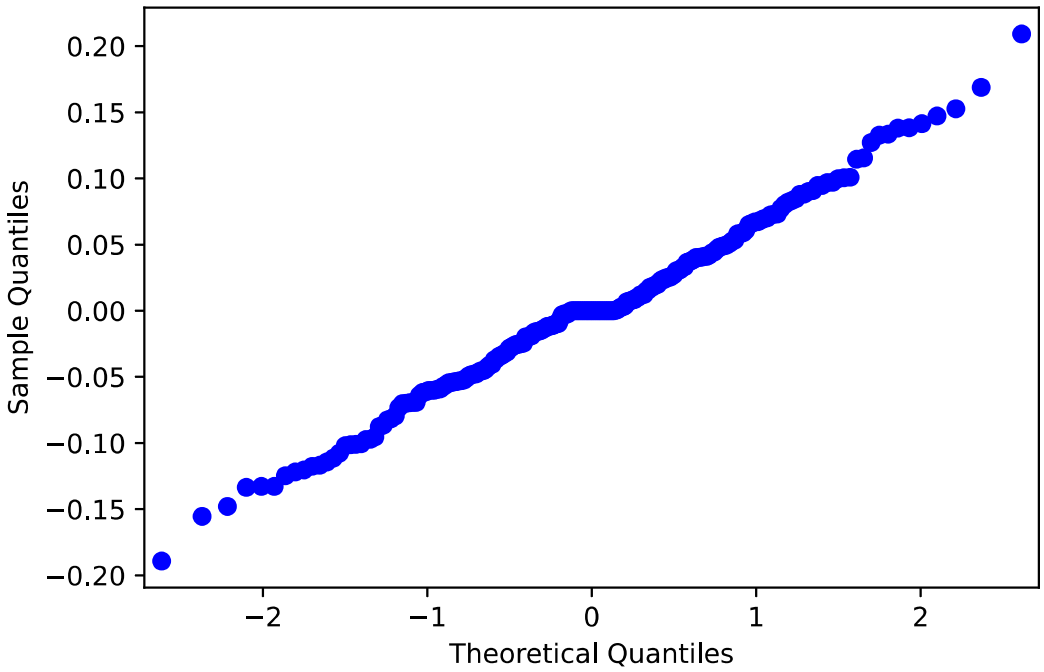
```
3.372       1.141
0.62                       -0.9201      1.157     -0.795      0.430      -
3.248       1.408
0.64                       -1.3526      1.219     -1.110      0.273      -
3.806       1.101
0.67                       -1.1888      1.342     -0.886      0.380      -
3.890       1.512
0.68                       -1.2988      1.475     -0.881      0.383      -
4.268       1.670
0.69                       -1.3201      1.424     -0.927      0.359      -
4.187       1.547
0.7                     -3.993e-17   4.45e-17     -0.897      0.374    -1.
3e-16    4.97e-17
0.73                       -1.8591      1.688     -1.101      0.277      -
5.257       1.539
0.74                             0          0        nan        nan
0           0
0.78                       -2.6496      1.883     -1.407      0.166      -
6.441       1.141
0.8                        -2.3629      1.972     -1.198      0.237      -
6.333       1.607
0.81                       -2.2868      1.998     -1.144      0.258      -
6.309       1.735
0.91                       -2.7686      2.439     -1.135      0.262      -
7.678       2.141
0.96                       -3.2226      2.730     -1.181      0.244      -
8.717       2.272
0.99                       -3.6422      2.764     -1.318      0.194      -
9.205       1.921
1.0                        -3.2915      2.837     -1.160      0.252      -
9.002       2.419
==============================================================================
====
Omnibus:                        1.054   Durbin-Watson:
2.042
Prob(Omnibus):                  0.590   Jarque-Bera (JB):
0.747
Skew:                           0.103   Prob(JB):
0.688
Kurtosis:                       3.195   Cond. No.                     1.03
e+16
==============================================================================
====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 4.71e-29. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
```

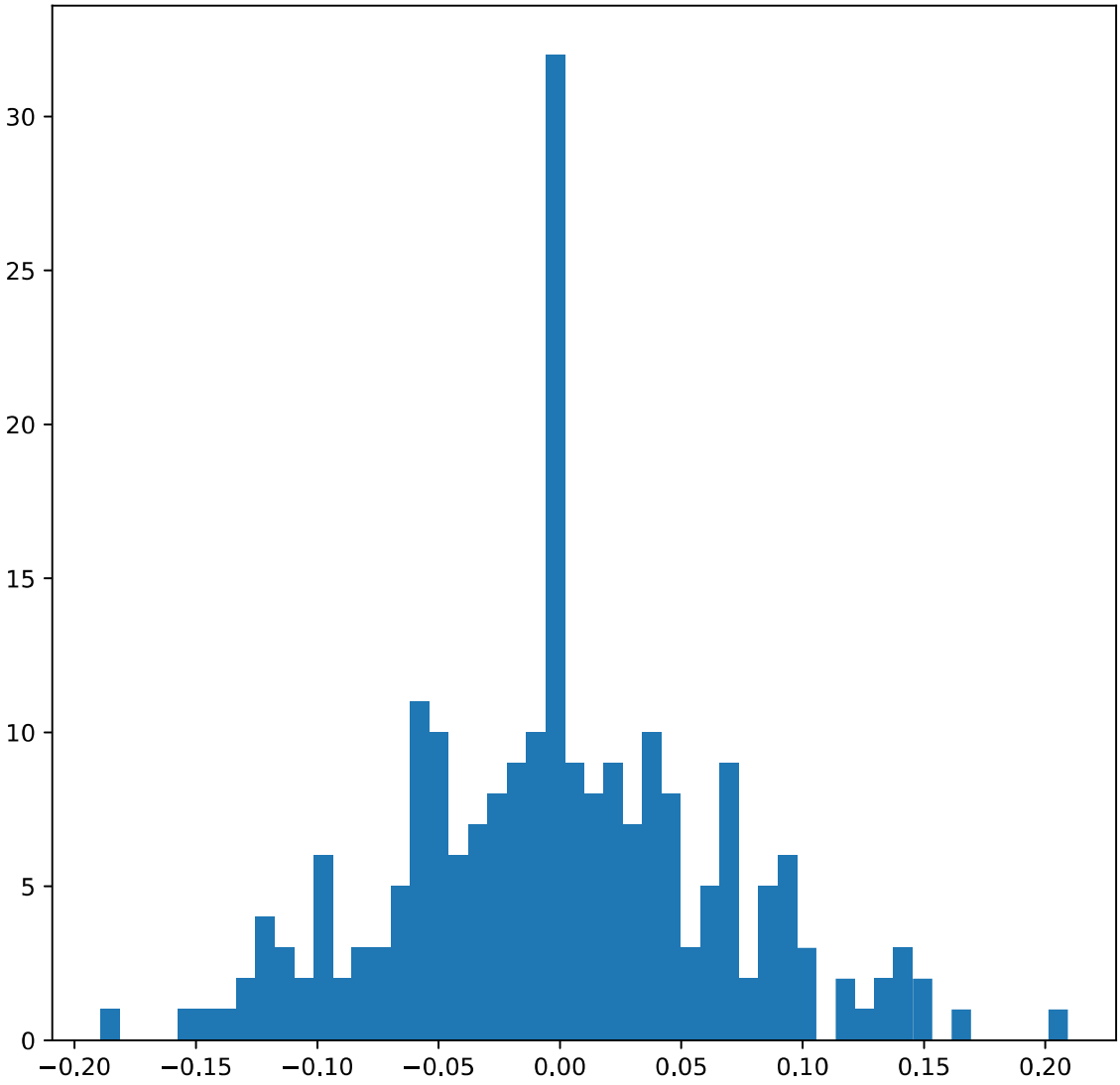
Plot model using qqPlot, Histogram and Standard Plot

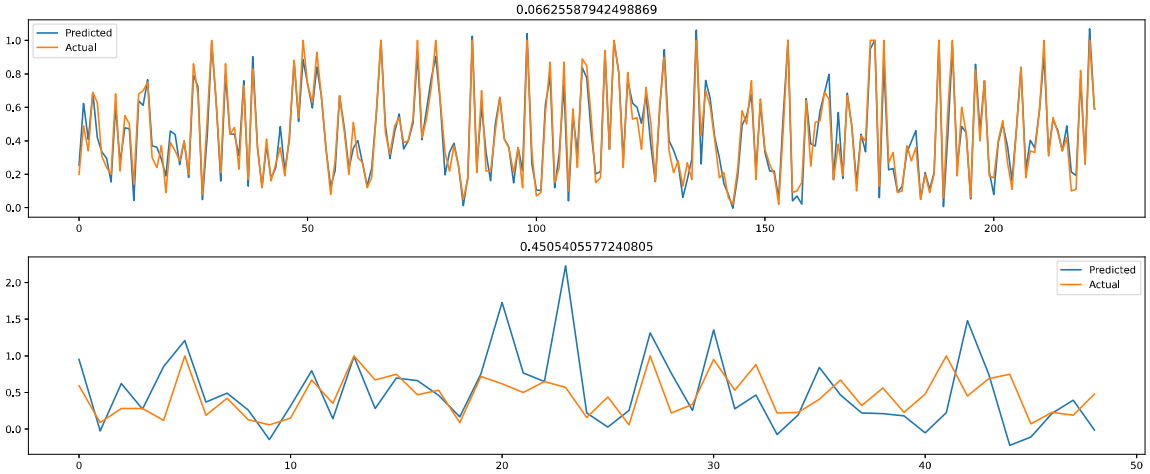
Code taken from Week 2 Example 1

In [15]:

```
f = sm.qqplot(trained_model.resid)
fig = plt.figure(figsize=[8, 8])
ax = fig.add_subplot(1, 1, 1)
ax.hist(trained_model.resid, 50)

Y_train_pred = trained_model.predict(X_train)
Y_test_pred = trained_model.predict(X_test)
rmse_train = np.sqrt(np.mean((Y_train_pred - Y_train)**2))
rmse_test = np.sqrt(np.mean((Y_test_pred - Y_test)**2))

fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(2, 1, 1)
ax.plot(np.arange(len(Y_train_pred)), Y_train_pred, label='Predicted')
ax.plot(np.arange(len(Y_train_pred)), Y_train, label='Actual')
ax.set_title(rmse_train)
ax.legend()
ax = fig.add_subplot(2, 1, 2)
ax.plot(np.arange(len(Y_test_pred)), Y_test_pred, label='Predicted')
ax.plot(np.arange(len(Y_test_pred)), Y_test, label='Actual')
ax.set_title(rmse_test)
ax.legend();
```


0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
Sample Quantiles
−2
−1
0
1
2
Theoretical Quantiles


30
25
20
15
10
5
0
−0.20
−0.15
−0.10
−0.05
0.00
0.05
0.10
0.15
0.20



Data displays large amount of overfitting

Therefore, we need to remove columns that are unnecessary

Used https://www.nap.edu/read/4422/chapter/5#228 (https://www.nap.edu/read/4422/chapter/5#228) to decide with columns to keep

In [16]:

```
data_smaller = data_to_use.drop(columns=[' householdsize ', ' numbUrban ', ' pctUrban '
, ' whitePerCap ', ' blackPerCap ', ' indianPerCap ', ' AsianPerCap ', ' OtherPerCap ',
' HispPerCap ', ' PctLess9thGrade ', ' PctNotHSGrad ', ' PctBSorMore ', ' MalePctDivorc
e ', ' MalePctNevMarr ', ' FemalePctDiv ', ' TotalPctDiv ', ' PersPerFam ', ' PctFam2Pa
r ', ' PctKids2Par ', ' PctYoungKids2Par ', ' PctTeen2Par ', ' PctWorkMomYoungKids ', '
PctWorkMom ', ' NumIlleg ', ' PctIlleg ', ' NumImmig ', ' PctImmigRecent ', ' PctImmigR
ec5 ', ' PctImmigRec8 ', ' PctImmigRec10 ', ' PctRecentImmig ', ' PctRecImmig5 ', ' Pct
RecImmig8 ', ' PctRecImmig10 ', ' PctSpeakEnglOnly ', ' PctNotSpeakEnglWell ', ' PctLar
gHouseFam ', ' PctLargHouseOccup ', ' PersPerOccupHous ', ' PersPerOwnOccHous ', ' Pers
PerRentOccHous ', ' PctPersOwnOccup ', ' PctPersDenseHous ', ' PctHousLess3BR ', ' MedN
umBR ', ' HousVacant ', ' PctHousOccup ', ' PctHousOwnOcc ', ' PctVacantBoarded ', ' Pc
tVacMore6Mos ', ' MedYrHousBuilt ', ' PctHousNoPhone ', ' PctWOFullPlumb ', ' OwnOccLow
Quart ', ' OwnOccMedVal ', ' OwnOccHiQuart ', ' RentLowQ ', ' RentMedian ', ' RentHighQ
', ' MedRent ', ' MedRentPctHousInc ', ' MedOwnCostPctInc ', ' MedOwnCostPctIncNoMtg ',
' PctForeignBorn ', ' PctBornSameState ', ' PctSameHouse85 ', ' PctSameCity85 ', ' PctS
ameState85 ', ' LemasSwornFT ', ' LemasSwFTPerPop ', ' LemasSwFTFieldOps ', ' LemasSwFT
FieldPerPop ', ' LemasTotalReq ', ' LemasTotReqPerPop ', ' PolicReqPerOffic ', ' PolicP
erPop ', ' RacialMatchCommPol ', ' PctPolicWhite ', ' PctPolicBlack ', ' PctPolicHisp '
, ' PctPolicAsian ', ' PctPolicMinor ', ' OfficAssgnDrugUnits ', ' NumKindsDrugsSeiz ',
' PolicAveOTWorked ', ' LandArea ', ' PopDens ', ' PctUsePubTrans ', ' PolicCars ', ' P
olicOperBudg ', ' LemasPctPolicOnPatr ', ' LemasGangUnitDeploy ', ' LemasPctOfficDrugUn
', ' PolicBudgPerPop '])
print(data_smaller)
```

```
      population    racepctblack    racePctWhite    racePctAsian   \
0             0.19            0.02            0.90            0.12
16            0.15            0.40            0.63            0.14
20            0.25            0.05            0.71            0.48
21            1.00            0.47            0.59            0.12
23            0.11            0.04            0.89            0.09
...            ...             ...             ...             ...
1969          1.00            0.21            0.29            1.00
1981          0.07            0.17            0.84            0.11
1991          0.16            0.25            0.69            0.04
1992          0.08            0.06            0.87            0.22
1993          0.20            0.14            0.46            0.24

      racePctHisp    agePct12t21    agePct12t29    agePct16t24   \
0             0.17            0.34            0.47            0.29
16            0.06            0.58            0.72            0.65
20            0.30            0.42            0.48            0.28
21            0.05            0.41            0.53            0.34
23            0.06            0.45            0.48            0.31
...            ...             ...             ...             ...
1969          0.26            0.24            0.47            0.28
1981          0.04            0.35            0.41            0.30
1991          0.25            0.35            0.50            0.31
1992          0.10            0.58            0.74            0.63
1993          0.77            0.50            0.62            0.40

      agePct65up    medIncome   ...   PctPopUnderPov    PctUnemployed   \
0             0.32         0.37  ...             0.19             0.27
16            0.47         0.22  ...             0.48             0.37
20            0.32         0.33  ...             0.34             0.59
21            0.33         0.28  ...             0.33             0.30
23            0.46         0.22  ...             0.36             0.33
...            ...          ...  ...              ...              ...
1969          0.46         0.36  ...             0.33             0.38
1981          0.64         0.50  ...             0.10             0.20
1991          0.54         0.31  ...             0.31             0.50
1992          0.41         0.44  ...             0.16             0.37
1993          0.17         0.40  ...             0.35             0.47

      PctEmploy    PctEmplManu    PctEmplProfServ    PctOccupManu   \
0            0.68           0.23               0.41            0.25
16           0.44           0.08               0.73            0.21
20           0.43           0.38               0.41            0.43
21           0.59           0.18               0.33            0.35
23           0.53           0.43               0.47            0.44
...           ...            ...                ...             ...
1969         0.53           0.18               0.52            0.24
1981         0.56           0.30               0.69            0.20
1991         0.46           0.63               0.40            0.54
1992         0.57           0.44               0.57            0.27
1993         0.58           0.53               0.20            0.60

      PctOccupMgmtProf    NumInShelters    NumStreet    ViolentCrimesPerP
op
0                 0.52             0.04         0.00
0.20
16                0.53             0.23         0.02
0.49
20                0.36             0.12         0.09
0.34
21                0.38             0.39         0.36
```

```
0.69
23                  0.33            0.05         0.01
0.63
...                  ...             ...          ...
...
1969                0.57            1.00         1.00
0.75
1981                0.58            0.02         0.00
0.07
1991                0.32            0.06         0.02
0.23
1992                0.48            0.04         0.01
0.19
1993                0.24            0.08         0.08
0.48

[319 rows x 29 columns]
```

In [17]:

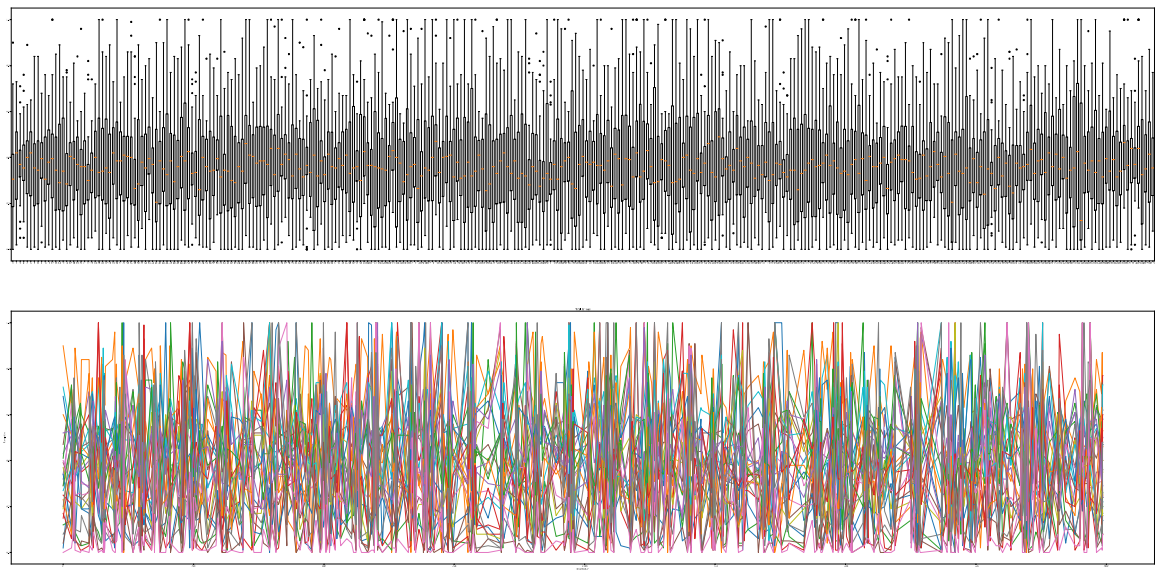
```
for column in data_smaller.iloc[:,1:]:
    data_smaller[column] = pandas.to_numeric(data_smaller[column], downcast="float")
```

In [18]:

```
fig = plt.figure()
fig.set_figheight(50)
fig.set_figwidth(100)
ax = fig.add_subplot(2, 1, 1)
ax.boxplot(data_smaller.iloc[:,1:].transpose())
ax = fig.add_subplot(2, 1, 2)
ax.plot(data_smaller.iloc[:,1:])
ax.set_xlabel('Observation')
ax.set_ylabel('Population')
ax.set_title('Total Crimes')
```

Out[18]:

```
Text(0.5, 1.0, 'Total Crimes')
```



In [19]:

```
print(data_smaller[" population "].astype("category"))
```

```
0       0.19
16      0.15
20      0.25
21      1.00
23      0.11
        ...
1969    1.00
1981    0.07
1991    0.16
1992    0.08
1993    0.20
Name:  population , Length: 319, dtype: category
Categories (65, float64): [0.00, 0.01, 0.02, 0.03, ..., 0.91, 0.96, 0.99,
1.00]
C:\Python39\lib\site-packages\pandas\io\formats\format.py:1403: FutureWarn
ing: Index.ravel returning ndarray is deprecated; in a future version this
will return a view on self.
  for val, m in zip(values.ravel(), mask.ravel())
C:\Python39\lib\site-packages\pandas\io\formats\format.py:1403: FutureWarn
ing: Index.ravel returning ndarray is deprecated; in a future version this
will return a view on self.
  for val, m in zip(values.ravel(), mask.ravel())
```

In [20]:

```
dummies_small = pandas.get_dummies(data_smaller[" population "].astype("category"), dro
p_first=True)
X_small = pandas.concat([data_smaller.iloc[:,1:-1], dummies_small], axis=1)
X_small = sm.add_constant(X_small)
Y_small = data_smaller.iloc[:,-1]
print(X_small)
print(Y_small)
```

```
      const   racepctblack    racePctWhite    racePctAsian    racePctHisp
\
0       1.0            0.02            0.90            0.12           0.17
16      1.0            0.40            0.63            0.14           0.06
20      1.0            0.05            0.71            0.48           0.30
21      1.0            0.47            0.59            0.12           0.05
23      1.0            0.04            0.89            0.09           0.06
...     ...             ...             ...             ...            ...
1969    1.0            0.21            0.29            1.00           0.26
1981    1.0            0.17            0.84            0.11           0.04
1991    1.0            0.25            0.69            0.04           0.25
1992    1.0            0.06            0.87            0.22           0.10
1993    1.0            0.14            0.46            0.24           0.77

       agePct12t21    agePct12t29    agePct16t24    agePct65up    medIncom
e   \
0             0.34           0.47           0.29          0.32         0.
37
16            0.58           0.72           0.65          0.47         0.
22
20            0.42           0.48           0.28          0.32         0.
33
21            0.41           0.53           0.34          0.33         0.
28
23            0.45           0.48           0.31          0.46         0.
22
...            ...            ...            ...           ...
...
1969          0.24           0.47           0.28          0.46         0.
36
1981          0.35           0.41           0.30          0.64         0.
50
1991          0.35           0.50           0.31          0.54         0.
31
1992          0.58           0.74           0.63          0.41         0.
44
1993          0.50           0.62           0.40          0.17         0.
40

      ...  0.7  0.73  0.74  0.78  0.8  0.81  0.91  0.96  0.99  1.0
0     ...    0     0     0     0    0     0     0     0     0    0
16    ...    0     0     0     0    0     0     0     0     0    0
20    ...    0     0     0     0    0     0     0     0     0    0
21    ...    0     0     0     0    0     0     0     0     0    1
23    ...    0     0     0     0    0     0     0     0     0    0
...   ...  ...   ...   ...   ...  ...   ...   ...   ...   ...  ...
1969  ...    0     0     0     0    0     0     0     0     0    1
1981  ...    0     0     0     0    0     0     0     0     0    0
1991  ...    0     0     0     0    0     0     0     0     0    0
1992  ...    0     0     0     0    0     0     0     0     0    0
1993  ...    0     0     0     0    0     0     0     0     0    0

[319 rows x 92 columns]
0       0.20
16      0.49
20      0.34
21      0.69
23      0.63
        ...
1969    0.75
1981    0.07
```

```
1991    0.23
1992    0.19
1993    0.48
Name:  ViolentCrimesPerPop , Length: 319, dtype: float32
```

In [21]:

```
num_samples_small = data_smaller.shape[0]
training_samples_small = int(num_samples_small*0.7)
validation_samples_small = int(num_samples_small*0.15)
X_train_small = X_small.iloc[0:training_samples_small, :]
Y_train_small = Y_small.iloc[0:training_samples_small]
X_val_small = X_small.iloc[training_samples_small:(training_samples_small + validation_
samples_small), :]
Y_val_small = Y_small.iloc[training_samples_small:(training_samples_small + validation_
samples_small)]
X_test_small = X_small.iloc[(training_samples_small + validation_samples_small):, :]
Y_test_small = Y_small.iloc[(training_samples_small + validation_samples_small):]
```

In [22]:

```
print(X_train_small.shape)
print(Y_train_small.shape)
print(X_val_small.shape)
print(Y_val_small.shape)
print(X_test_small.shape)
print(Y_test_small.shape)
```

```
(223, 92)
(223,)
(47, 92)
(47,)
(49, 92)
(49,)
```

In [23]:

```
model = sm.OLS(Y_train_small, X_train_small)
trained_model_small = model.fit()
print(trained_model_small.summary())
```

OLS Regression Results

Dep. Variable:	ViolentCrimesPerPop	R-squared:	0.798
Model:	OLS	Adj. R-squared:	0.678
Method:	Least Squares	F-statistic:	6.621
Date:	Sun, 11 Apr 2021	Prob (F-statistic):	1.36e-22
Time:	17:07:11	Log-Likelihood:	149.98
No. Observations:	223	AIC:	-132.0
Df Residuals:	139	BIC:	154.2
Df Model:	83		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.3358	0.652	2.048	0.042	0.046	2.626
racepctblack	0.2673	0.194	1.375	0.171	-0.117	0.652
racePctWhite	-0.1789	0.225	-0.794	0.429	-0.625	0.267
racePctAsian	-0.0177	0.113	-0.157	0.876	-0.242	0.206
racePctHisp	0.1218	0.106	1.148	0.253	-0.088	0.332
agePct12t21	0.0437	0.458	0.096	0.924	-0.861	0.948
agePct12t29	-1.0048	0.786	-1.278	0.203	-2.559	0.550
agePct16t24	0.8956	0.788	1.136	0.258	-0.663	2.454
agePct65up	0.3629	0.584	0.622	0.535	-0.791	1.517
medIncome	-0.2719	0.632	-0.430	0.668	-1.521	0.977
pctWWage	-0.6706	0.475	-1.413	0.160	-1.609	0.268
pctWFarmSelf	-0.1284	0.137	-0.938	0.350	-0.399	0.142
pctWInvInc	-0.4169	0.311	-1.341	0.182	-1.031	0.198
pctWSocSec	-0.7359	0.511	-1.441	0.152	-1.746	0.274
pctWPubAsst	0.2427	0.173	1.404	0.163	-0.099	0.585
pctWRetire	-0.0494	0.145	-0.340	0.734	-0.337	0.238
medFamInc	0.1409	0.577	0.244	0.807	-1.000	1.282
perCapInc	0.1209	0.397	0.304	0.761	-0.665	0.906

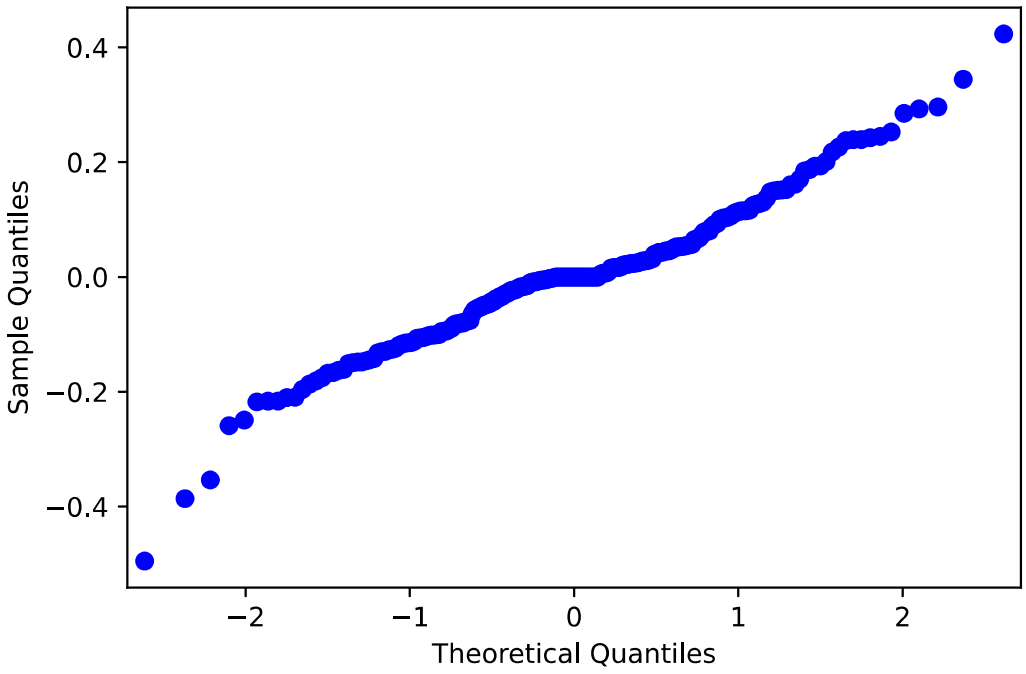
```
 NumUnderPov          -0.2088      0.301     -0.694      0.489      -0.803
0.386
 PctPopUnderPov       -0.1335      0.303     -0.441      0.660      -0.732
0.465
 PctUnemployed         0.0891      0.176      0.506      0.614      -0.259
0.437
 PctEmploy             0.3719      0.434      0.857      0.393      -0.487
1.230
 PctEmplManu          -0.0438      0.135     -0.324      0.747      -0.312
0.224
 PctEmplProfServ       0.0338      0.183      0.184      0.854      -0.329
0.397
 PctOccupManu         -0.1757      0.201     -0.873      0.384      -0.574
0.222
 PctOccupMgmtProf     -0.2176      0.309     -0.704      0.483      -0.829
0.394
 NumInShelters         0.0377      0.123      0.306      0.760      -0.206
0.281
 NumStreet             0.1972      0.097      2.026      0.045       0.005
0.390
0.01                  -0.0552      0.127     -0.436      0.663      -0.305
0.195
0.02               -7.201e-05      0.215     -0.000      1.000      -0.426
0.426
0.03                  0.0433      0.197      0.220      0.826      -0.346
0.432
0.04                  -0.0402      0.126     -0.319      0.750      -0.289
0.209
0.05                  0.0015      0.125      0.012      0.991      -0.245
0.248
0.06                  -0.1137      0.107     -1.066      0.288      -0.324
0.097
0.07                  -0.0650      0.101     -0.641      0.522      -0.265
0.135
0.08                  -0.1080      0.104     -1.038      0.301      -0.314
0.098
0.09                  -0.0924      0.098     -0.948      0.345      -0.285
0.100
0.1                   0.0593      0.095      0.624      0.533      -0.128
0.247
0.11                  -0.0420      0.093     -0.452      0.652      -0.226
0.142
0.12                  -0.0382      0.097     -0.393      0.695      -0.230
0.154
0.13                  -0.0715      0.090     -0.794      0.428      -0.250
0.106
0.14                  -0.0514      0.089     -0.581      0.562      -0.227
0.124
0.15                  -0.0218      0.092     -0.237      0.813      -0.204
0.160
0.16                  -0.1125      0.084     -1.346      0.181      -0.278
0.053
0.17                  -0.0830      0.104     -0.799      0.426      -0.288
0.122
0.18                  0.1001      0.115      0.873      0.384      -0.127
0.327
0.19                  -0.0455      0.095     -0.481      0.631      -0.233
0.142
0.2                   -0.2482      0.108     -2.302      0.023      -0.461
-0.035
0.21                  -0.0254      0.178     -0.142      0.887      -0.378
```

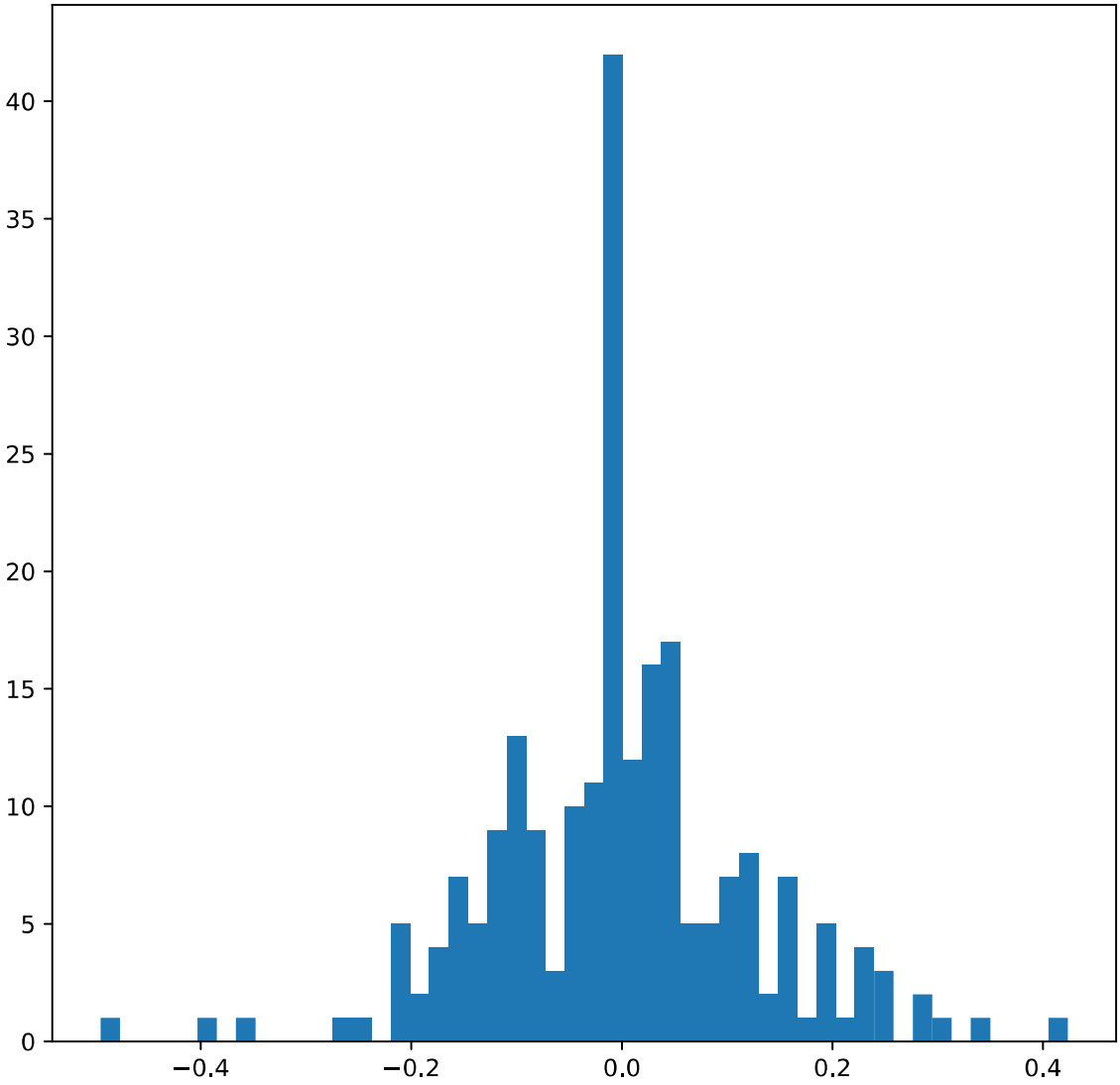
0.327					
0.22	-0.0102	0.171	-0.060	0.953	-0.348
0.327					
0.23	0.0471	0.123	0.382	0.703	-0.197
0.291					
0.24	-0.0420	0.092	-0.456	0.649	-0.224
0.140					
0.25	0.1233	0.104	1.188	0.237	-0.082
0.329					
0.26	-0.0641	0.124	-0.516	0.606	-0.309
0.181					
0.27	-0.2144	0.117	-1.840	0.068	-0.445
0.016					
0.28	-0.1559	0.098	-1.592	0.114	-0.350
0.038					
0.29	-0.1106	0.124	-0.895	0.373	-0.355
0.134					
0.3	-0.0688	0.164	-0.418	0.676	-0.394
0.256					
0.31	-0.0866	0.162	-0.536	0.593	-0.406
0.233					
0.32	-0.0180	0.168	-0.107	0.915	-0.351
0.315					
0.34	0.3770	0.118	3.207	0.002	0.145
0.609					
0.35	0.1727	0.117	1.480	0.141	-0.058
0.403					
0.36	7.402e-17	1.35e-16	0.549	0.584	-1.92e-16
3.41e-16					
0.4	-0.1010	0.120	-0.839	0.403	-0.339
0.137					
0.41	0.1761	0.175	1.006	0.316	-0.170
0.522					
0.42	-9.407e-17	1.02e-16	-0.920	0.359	-2.96e-16
1.08e-16					
0.43	5.671e-17	1.22e-16	0.467	0.641	-1.84e-16
2.97e-16					
0.45	6.106e-16	3.6e-16	1.696	0.092	-1.01e-16
1.32e-15					
0.46	0.0485	0.184	0.264	0.792	-0.314
0.411					
0.47	0.0795	0.161	0.493	0.623	-0.240
0.399					
0.51	2.758e-17	7.01e-17	0.393	0.695	-1.11e-16
1.66e-16					
0.52	-0.0280	0.166	-0.168	0.866	-0.357
0.301					
0.55	0.0083	0.199	0.042	0.967	-0.384
0.401					
0.56	0.0930	0.253	0.367	0.714	-0.408
0.594					
0.57	0.0997	0.132	0.753	0.452	-0.162
0.361					
0.58	-0.0656	0.131	-0.499	0.618	-0.325
0.194					
0.6	0.3113	0.168	1.848	0.067	-0.022
0.644					
0.62	0.2383	0.144	1.651	0.101	-0.047
0.524					
0.64	0.0058	0.184	0.031	0.975	-0.358
0.370					

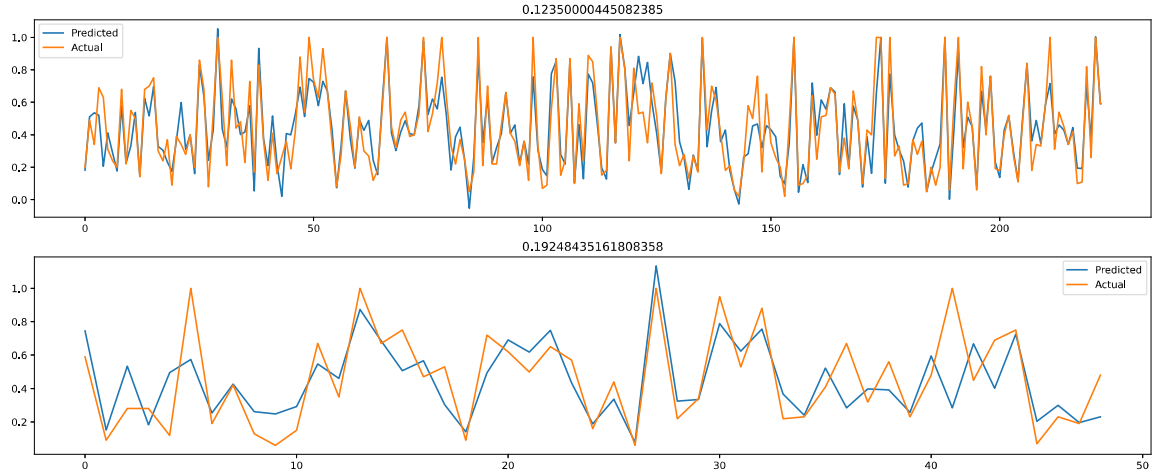
```
0.67                  0.1788      0.182       0.981      0.328      -0.181
0.539
0.68                  0.5333      0.173       3.083      0.002       0.191
0.875
0.69                  0.3947      0.186       2.124      0.035       0.027
0.762
0.7                4.092e-17   3.52e-17       1.162      0.247   -2.87e-17
1.11e-16
0.73                 -0.0747      0.190      -0.393      0.695      -0.451
0.301
0.74               -1.287e-17   1.79e-17      -0.719      0.473   -4.83e-17
2.25e-17
0.78                  0.0654      0.216       0.302      0.763      -0.362
0.493
0.8                  -0.1136      0.227      -0.502      0.617      -0.561
0.334
0.81                  0.1247      0.169       0.738      0.462      -0.210
0.459
0.91                  0.1847      0.224       0.823      0.412      -0.259
0.629
0.96                  0.1927      0.226       0.851      0.396      -0.255
0.640
0.99                 -0.0468      0.236      -0.198      0.843      -0.513
0.419
1.0                   0.0911      0.173       0.528      0.599      -0.250
0.432
==============================================================================
====
Omnibus:                       11.139   Durbin-Watson:
1.994
Prob(Omnibus):                  0.004   Jarque-Bera (JB):                2
5.230
Skew:                          -0.035   Prob(JB):                     3.32
e-06
Kurtosis:                       4.646   Cond. No.                     1.33
e+16
==============================================================================
====

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 6.5e-30. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

In [25]:

```
fig = plt.figure(figsize=[20, 12])
ax = fig.add_subplot(3, 1, 1)
ax.plot(trained_model_small.predict(X_train_small), label='Predicted')
ax.plot(Y_train_small, label='Actual')
ax.legend()
ax.set_title('Training Data')
ax = fig.add_subplot(3, 1, 2)
ax.plot(trained_model_small.predict(X_val_small), label='Predicted')
ax.plot(Y_val_small, label='Actual')
ax.legend()
ax.set_title('Validation Data')
ax = fig.add_subplot(3, 1, 3)
ax.plot(trained_model_small.predict(X_test_small), label='Predicted')
ax.plot(Y_test_small, label='Actual')
ax.legend()
ax.set_title('Testing Data');
```


Sample Quantiles
Theoretical Quantiles
0.4
0.2
0.0
−0.2
−0.4
−2
−1
0
1
2

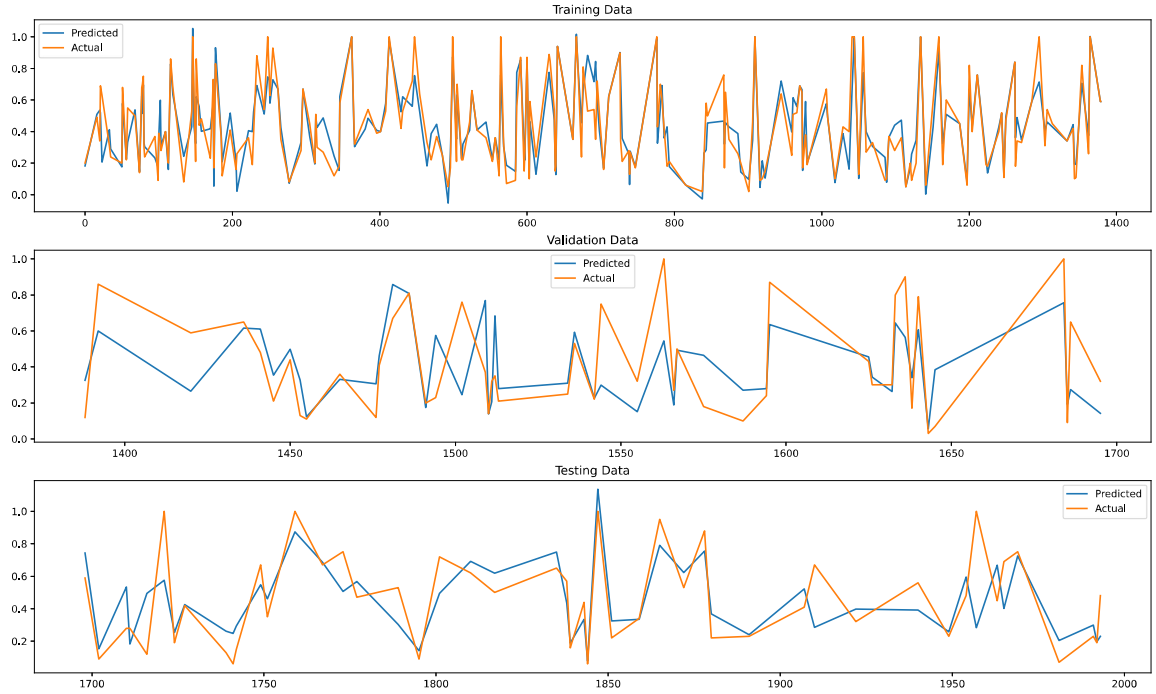

40
35
30
25
20
15
10
5
0
−0.4
−0.2
0.0
0.2
0.4


0.12350000445082385
Predicted
Actual
1.0
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200

0.19248435161808358
Predicted
Actual
1.0
0.8
0.6
0.4
0.2
0
10
20
30
40
50

In [24]:

```
f = sm.qqplot(trained_model_small.resid)
fig = plt.figure(figsize=[8, 8])
ax = fig.add_subplot(1, 1, 1)
ax.hist(trained_model_small.resid, 50)

Y_train_pred_small = trained_model_small.predict(X_train_small)
Y_test_pred_small = trained_model_small.predict(X_test_small)
rmse_train_small = np.sqrt(np.mean((Y_train_pred_small - Y_train_small)**2))
rmse_test_small = np.sqrt(np.mean((Y_test_pred_small - Y_test_small)**2))

fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(2, 1, 1)
ax.plot(np.arange(len(Y_train_pred_small)), Y_train_pred_small, label='Predicted')
ax.plot(np.arange(len(Y_train_pred_small)), Y_train_small, label='Actual')
ax.set_title(rmse_train_small)
ax.legend()
ax = fig.add_subplot(2, 1, 2)
ax.plot(np.arange(len(Y_test_pred_small)), Y_test_pred_small, label='Predicted')
ax.plot(np.arange(len(Y_test_pred_small)), Y_test_small, label='Actual')
ax.set_title(rmse_test_small)
ax.legend();
```



LASSO Regression

Calculate Y_μ, Y_σ

Code taken from Week 2 Example 1

In [26]:

```
Y_mu = np.mean(Y)
Y_sigma = np.std(Y)

Y_std = (Y - Y_mu) / Y_sigma
```

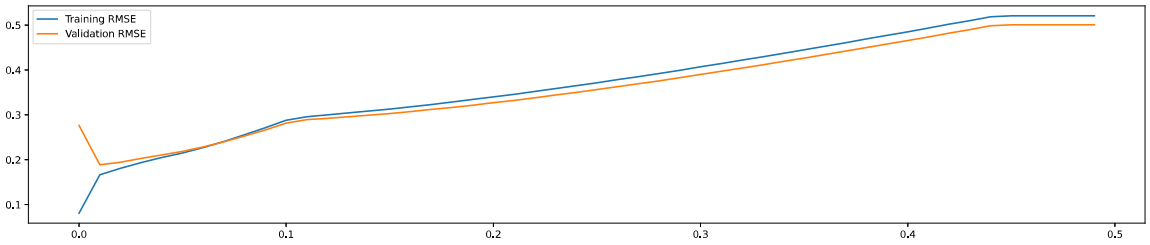
Plot λ and Coefficients

Code taken from Week 2 Example 1

In [27]:

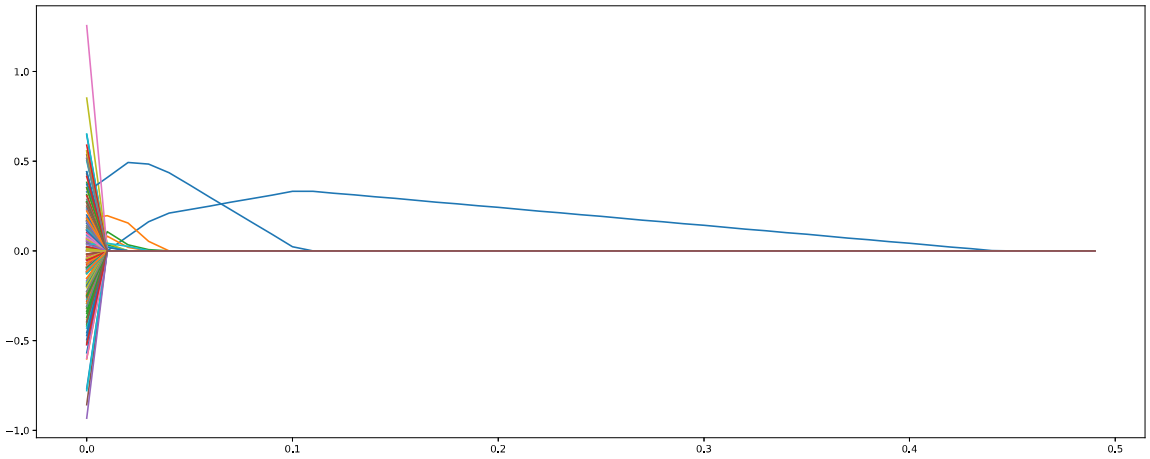
```
lambdas = np.arange(0.0, 0.5, 0.01)
rmse_train = []
rmse_validation = []
coeffs = []
for l in lambdas:
    trained_model_lasso = Lasso(fit_intercept=False, alpha=l).fit(X_train, Y_train)
    coeffs.append(trained_model_lasso.coef_)
    rmse_train.append(np.sqrt(np.mean((trained_model_lasso.predict(X_train) - Y_train)*
*2)))
    rmse_validation.append(np.sqrt(np.mean((trained_model_lasso.predict(X_val) - Y_val)
**2)))

fig = plt.figure(figsize=[20, 4])
ax = fig.add_subplot(1, 1, 1)
ax.plot(lambdas, rmse_train, label='Training RMSE')
ax.plot(lambdas, rmse_validation, label='Validation RMSE')
ax.legend();
```

```
<ipython-input-27-d90910a7a831>:6: UserWarning: With alpha=0, this algorit
hm does not converge well. You are advised to use the LinearRegression est
imator
  trained_model_lasso = Lasso(fit_intercept=False, alpha=l).fit(X_train, Y
_train)
C:\Python39\lib\site-packages\sklearn\linear_model\_coordinate_descent.py:
530: UserWarning: Coordinate descent with no regularization may lead to un
expected results and is discouraged.
  model = cd_fast.enet_coordinate_descent(
C:\Python39\lib\site-packages\sklearn\linear_model\_coordinate_descent.py:
530: ConvergenceWarning: Objective did not converge. You might want to inc
rease the number of iterations. Duality gap: 0.7249298855334498, toleranc
e: 0.006049739990407825
  model = cd_fast.enet_coordinate_descent(
```



In [28]:

```
fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(1, 1, 1)
ax.plot(lambdas, coeffs);
```



Find best λ value

Code taken from Week 2 Example 1

In [29]:

```
best_lambda = lambdas[np.argmin(rmse_validation)]
print(best_lambda)
```

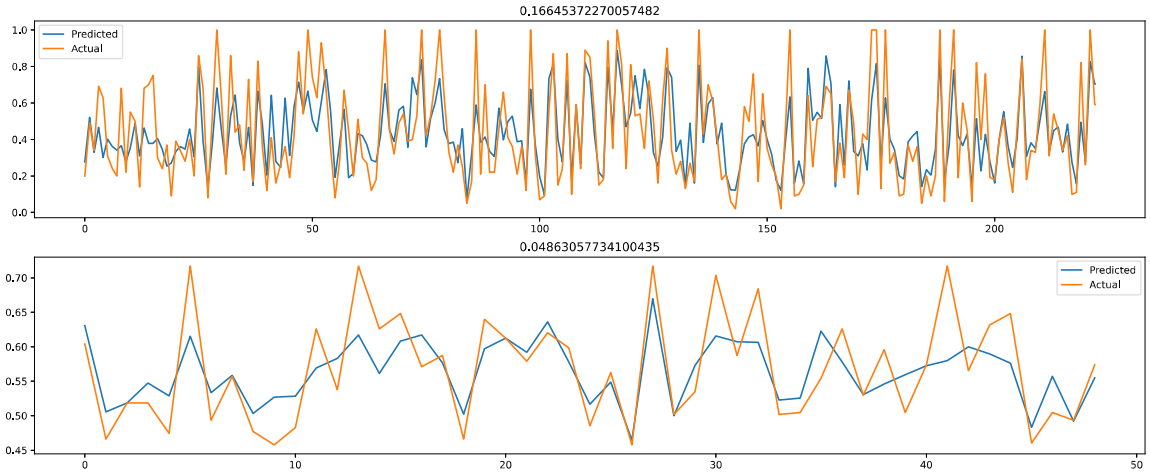
```
0.01
```

Plot trained model

Code taken from Week 2 Example 1

In [30]:

```
trained_model_lasso = Lasso(fit_intercept=False, alpha=best_lambda).fit(X_train, Y_trai
n)

Y_train_pred = trained_model_lasso.predict(X_train)
Y_test_pred = trained_model_lasso.predict(X_test)
rmse_train = np.sqrt(np.mean((Y_train_pred - Y_train)**2))
rmse_test = np.sqrt(np.mean(((Y_test_pred*Y_sigma + Y_mu) - (Y_test*Y_sigma + Y_mu))**2
))

fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(2, 1, 1)
ax.plot(np.arange(len(Y_train_pred)), Y_train_pred, label='Predicted')
ax.plot(np.arange(len(Y_train_pred)), Y_train, label='Actual')
ax.set_title(rmse_train)
ax.legend()
ax = fig.add_subplot(2, 1, 2)
ax.plot(np.arange(len(Y_test_pred)), Y_test_pred*Y_sigma + Y_mu, label='Predicted')
ax.plot(np.arange(len(Y_test_pred)), Y_test*Y_sigma + Y_mu, label='Actual')
ax.set_title(rmse_test)
ax.legend();
```



In [31]:

```
sum(trained_model_lasso.coef_ != 0)
```

Out[31]:

11

Use smaller model

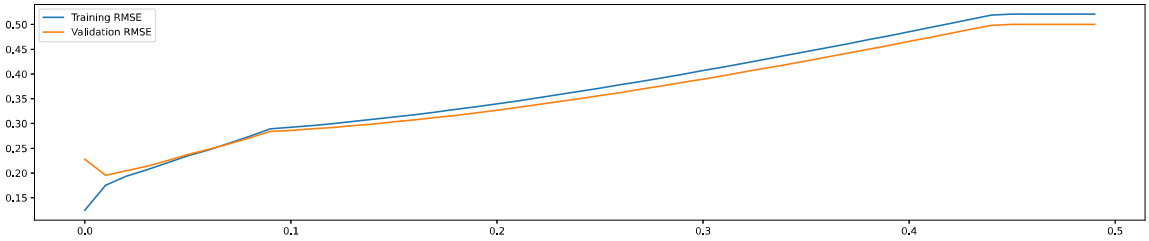
In [32]:

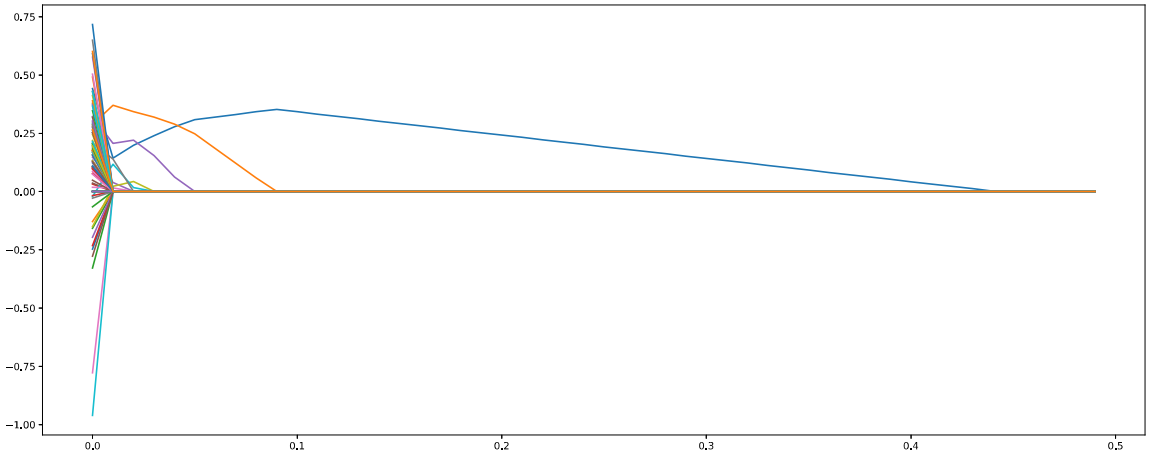
```
Y_mu_small = np.mean(Y_small)
Y_sigma_small = np.std(Y_small)

Y_std_small = (Y_small - Y_mu_small) / Y_sigma_small
```

In [33]:

```
lambdas_small = np.arange(0.0, 0.5, 0.01)
rmse_train_small = []
rmse_validation_small = []
coeffs_small = []
for l in lambdas_small:
    trained_model_lasso_small = Lasso(fit_intercept=False, alpha=l).fit(X_train_small,
Y_train_small)
    coeffs_small.append(trained_model_lasso_small.coef_)
    rmse_train_small.append(np.sqrt(np.mean((trained_model_lasso_small.predict(X_train_
small) - Y_train_small)**2)))
    rmse_validation_small.append(np.sqrt(np.mean((trained_model_lasso_small.predict(X_v
al_small) - Y_val_small)**2)))

fig = plt.figure(figsize=[20, 4])
ax = fig.add_subplot(1, 1, 1)
ax.plot(lambdas_small, rmse_train_small, label='Training RMSE')
ax.plot(lambdas_small, rmse_validation_small, label='Validation RMSE')
ax.legend();

fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(1, 1, 1)
ax.plot(lambdas_small, coeffs_small);
```

```
<ipython-input-33-c30ef1d5228b>:6: UserWarning: With alpha=0, this algorit
hm does not converge well. You are advised to use the LinearRegression est
imator
  trained_model_lasso_small = Lasso(fit_intercept=False, alpha=l).fit(X_tr
ain_small, Y_train_small)
C:\Python39\lib\site-packages\sklearn\linear_model\_coordinate_descent.py:
530: UserWarning: Coordinate descent with no regularization may lead to un
expected results and is discouraged.
  model = cd_fast.enet_coordinate_descent(
C:\Python39\lib\site-packages\sklearn\linear_model\_coordinate_descent.py:
530: ConvergenceWarning: Objective did not converge. You might want to inc
rease the number of iterations. Duality gap: 1.7406594664803237, toleranc
e: 0.006049739990407825
  model = cd_fast.enet_coordinate_descent(
```





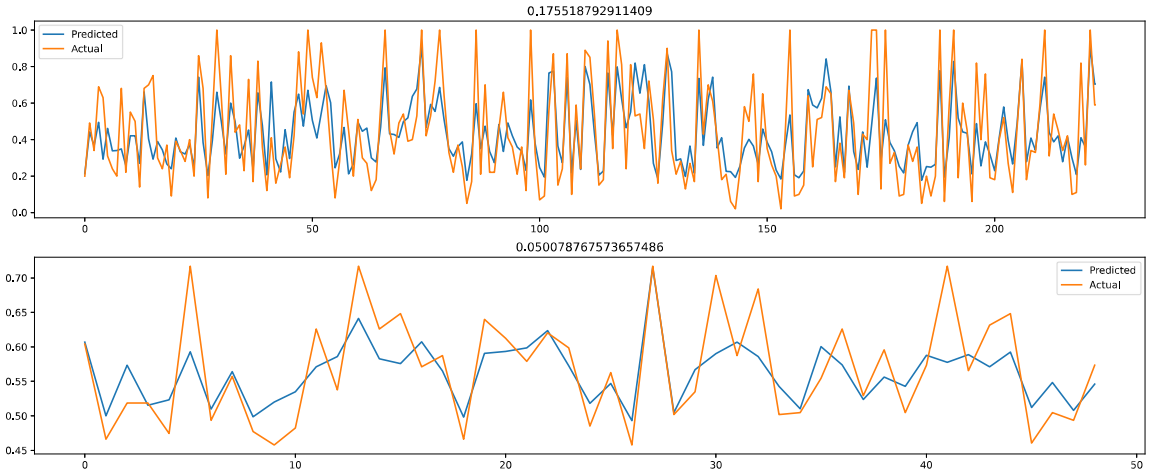
In [34]:

```
best_lambda_small = lambdas_small[np.argmin(rmse_validation)]
print(best_lambda_small)
```

```
0.01
```

In [35]:

```
trained_model_lasso_small = Lasso(fit_intercept=False, alpha=best_lambda_small).fit(X_t
rain_small, Y_train_small)

Y_train_pred_small = trained_model_lasso_small.predict(X_train_small)
Y_test_pred_small = trained_model_lasso_small.predict(X_test_small)
rmse_train_small = np.sqrt(np.mean((Y_train_pred_small - Y_train_small)**2))
rmse_test_small = np.sqrt(np.mean(((Y_test_pred_small*Y_sigma_small + Y_mu_small) - (Y_
test_small*Y_sigma_small + Y_mu_small))**2))

fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(2, 1, 1)
ax.plot(np.arange(len(Y_train_pred_small)), Y_train_pred_small, label='Predicted')
ax.plot(np.arange(len(Y_train_pred_small)), Y_train_small, label='Actual')
ax.set_title(rmse_train_small)
ax.legend()
ax = fig.add_subplot(2, 1, 2)
ax.plot(np.arange(len(Y_test_pred_small)), Y_test_pred_small*Y_sigma_small + Y_mu_small
, label='Predicted')
ax.plot(np.arange(len(Y_test_pred_small)), Y_test_small*Y_sigma_small + Y_mu_small, lab
el='Actual')
ax.set_title(rmse_test_small)
ax.legend();
```



Ridge Regression

Train and plot Ridge model

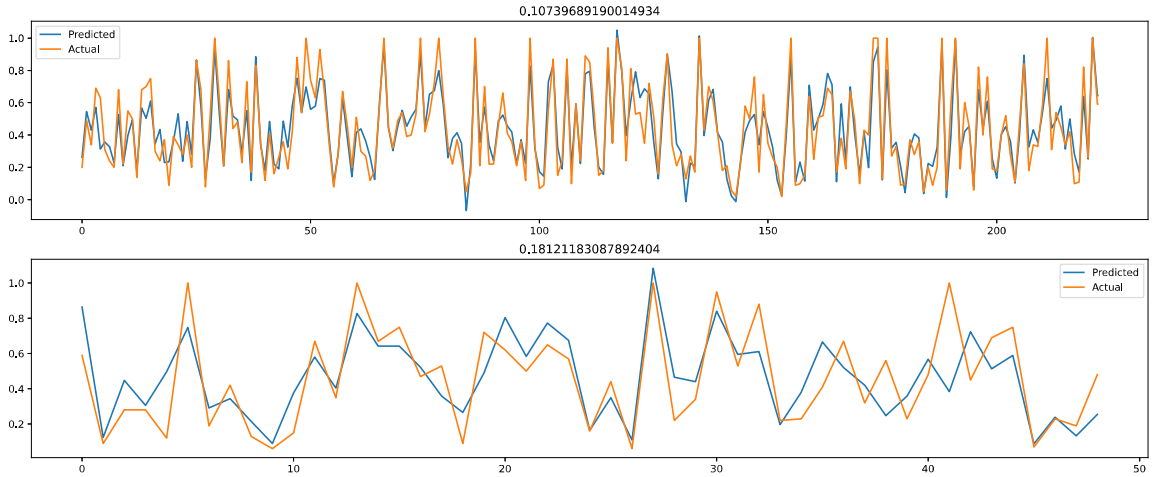
Code taken from Week 2 Example 1

In [36]:

```
trained_model_ridge = Ridge(fit_intercept=False, alpha=1).fit(X = X_train, y = Y_train)

Y_train_pred = trained_model_ridge.predict(X_train)
Y_test_pred = trained_model_ridge.predict(X_test)
rmse_train = np.sqrt(np.mean((Y_train_pred - Y_train)**2))
rmse_test = np.sqrt(np.mean((Y_test_pred - Y_test)**2))

fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(2, 1, 1)
ax.plot(np.arange(len(Y_train_pred)), Y_train_pred, label='Predicted')
ax.plot(np.arange(len(Y_train_pred)), Y_train, label='Actual')
ax.set_title(rmse_train)
ax.legend()
ax = fig.add_subplot(2, 1, 2)
ax.plot(np.arange(len(Y_test_pred)), Y_test_pred, label='Predicted')
ax.plot(np.arange(len(Y_test_pred)), Y_test, label='Actual')
ax.set_title(rmse_test)
ax.legend()
```

Out[36]:

```
<matplotlib.legend.Legend at 0x223b1e42fd0>
```



Calculate λ

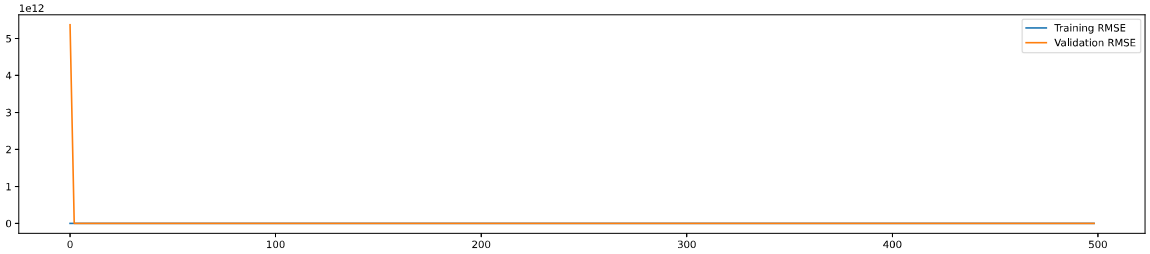
Code taken from Week 2 Example 1

In [37]:

```
lambdas = np.arange(0, 500, 2)
rmse_train = []
rmse_validation = []
for l in lambdas:
    trained_model_ridge = Ridge(fit_intercept=False, alpha=l).fit(X = X_train, y = Y_tr
ain)
    rmse_train.append(np.sqrt(np.mean((trained_model_ridge.predict(X_train) - Y_train)*
*2)))
    rmse_validation.append(np.sqrt(np.mean((trained_model_ridge.predict(X_val) - Y_val)
**2)))
```

In [38]:

```
fig = plt.figure(figsize=[20, 4])
ax = fig.add_subplot(1, 1, 1)
ax.plot(lambdas, rmse_train, label='Training RMSE')
ax.plot(lambdas, rmse_validation, label='Validation RMSE')
ax.legend();
```

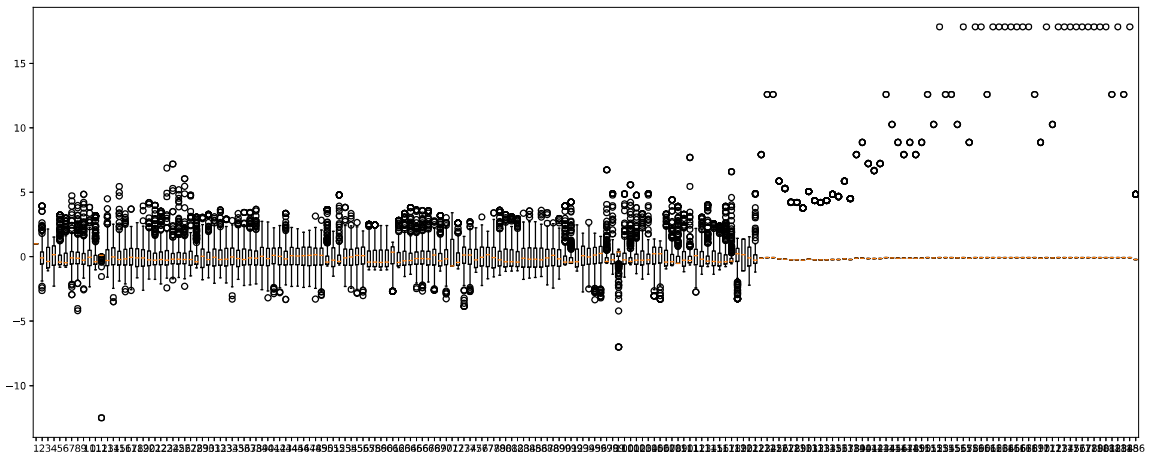


Calculate μ and σ

Code taken from Week 2 Example 1

In [39]:

```
mu = np.mean(X.iloc[:,1:],0)
sigma = np.std(X.iloc[:,1:],0)
X.iloc[:,1:] = (X.iloc[:,1:] - mu) / sigma
fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(1, 1, 1)
ax.boxplot(X);
```



In [40]:

```
X = np.concatenate([X, dummies], axis=1)

X_train = X[0:training_samples, :]
Y_train = Y_std[0:training_samples]
X_val = X[training_samples:(training_samples + validation_samples), :]
Y_val = Y_std[training_samples:(training_samples + validation_samples)]
X_test = X[(training_samples + validation_samples):, :]
Y_test = Y_std[(training_samples + validation_samples):]
```

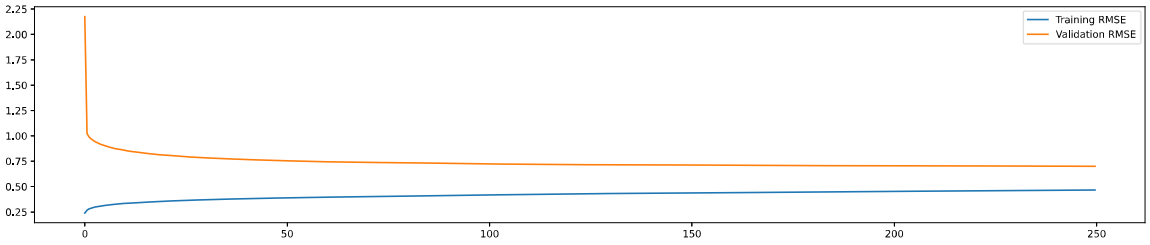
Plot RMSE vs λ

Code taken from Week 2 Example 1

In [41]:

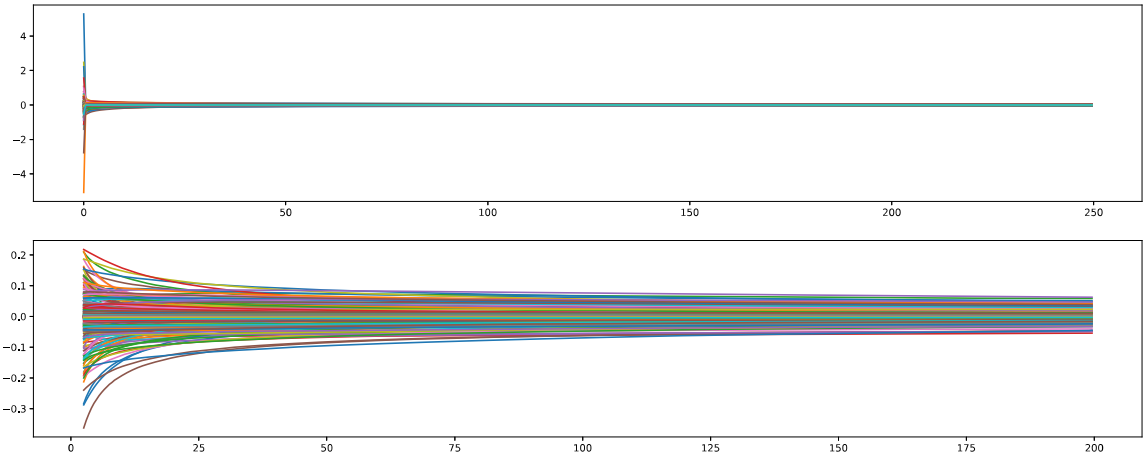
```
lambdas = np.arange(0, 250, 0.5)
rmse_train = []
rmse_validation = []
coeffs = []
for l in lambdas:
    trained_model_ridge = Ridge(fit_intercept=False, alpha=l).fit(X_train, Y_train)
    coeffs.append(trained_model_ridge.coef_)
    rmse_train.append(np.sqrt(np.mean((trained_model_ridge.predict(X_train) - Y_train)*
*2)))
    rmse_validation.append(np.sqrt(np.mean((trained_model_ridge.predict(X_val) - Y_val)
**2)))

fig = plt.figure(figsize=[20, 4])
ax = fig.add_subplot(1, 1, 1)
ax.plot(lambdas, rmse_train, label='Training RMSE')
ax.plot(lambdas, rmse_validation, label='Validation RMSE')
ax.legend();
```

```
C:\Python39\lib\site-packages\sklearn\linear_model\_ridge.py:190: UserWarn
ing: Singular matrix in solving dual problem. Using least-squares solution
instead.
  warnings.warn("Singular matrix in solving dual problem. Using "
```



In [42]:

```
fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(2, 1, 1)
ax.plot(lambdas, coeffs);
coeffs = np.array(coeffs)
ax = fig.add_subplot(2, 1, 2)
ax.plot(lambdas[5:400], coeffs[5:400,:]);
```



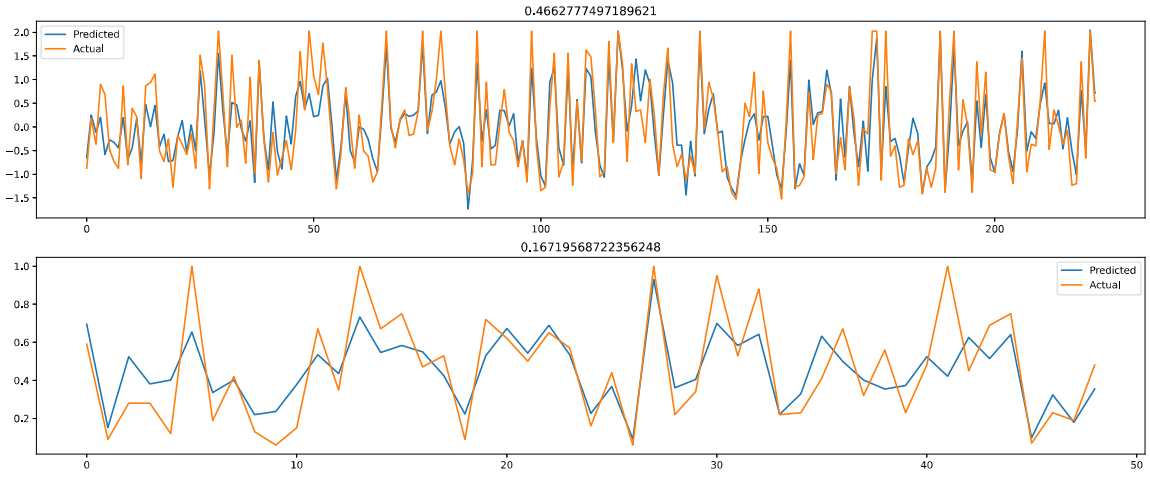
Find best λ and plot new trained Ridge model

Code taken from Week 2 Example 1

In [43]:

```
best_lambda = lambdas[np.argmin(rmse_validation)]
print(best_lambda)
trained_model_ridge = Ridge(fit_intercept=False, alpha=best_lambda).fit(X_train, Y_trai
n)

Y_train_pred = trained_model_ridge.predict(X_train)
Y_test_pred = trained_model_ridge.predict(X_test)
rmse_train = np.sqrt(np.mean((Y_train_pred - Y_train)**2))
rmse_test = np.sqrt(np.mean(((Y_test_pred*Y_sigma + Y_mu) - (Y_test*Y_sigma + Y_mu))**2
))

fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(2, 1, 1)
ax.plot(np.arange(len(Y_train_pred)), Y_train_pred, label='Predicted')
ax.plot(np.arange(len(Y_train_pred)), Y_train, label='Actual')
ax.set_title(rmse_train)
ax.legend()
ax = fig.add_subplot(2, 1, 2)
ax.plot(np.arange(len(Y_test_pred)), Y_test_pred*Y_sigma + Y_mu, label='Predicted')
ax.plot(np.arange(len(Y_test_pred)), Y_test*Y_sigma + Y_mu, label='Actual')
ax.set_title(rmse_test)
ax.legend();
```

249.5



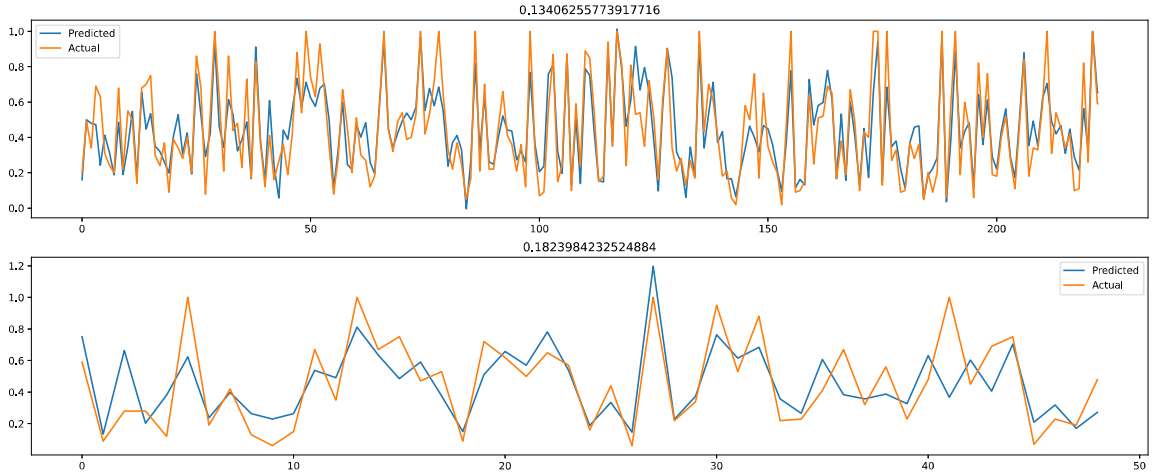
Test smaller model

In [44]:

```
trained_model_ridge_small = Ridge(fit_intercept=False, alpha=1).fit(X = X_train_small,
y = Y_train_small)

Y_train_pred_small = trained_model_ridge_small.predict(X_train_small)
Y_test_pred_small = trained_model_ridge_small.predict(X_test_small)
rmse_train_small = np.sqrt(np.mean((Y_train_pred_small - Y_train_small)**2))
rmse_test_small = np.sqrt(np.mean((Y_test_pred_small - Y_test_small)**2))

fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(2, 1, 1)
ax.plot(np.arange(len(Y_train_pred_small)), Y_train_pred_small, label='Predicted')
ax.plot(np.arange(len(Y_train_pred_small)), Y_train_small, label='Actual')
ax.set_title(rmse_train_small)
ax.legend()
ax = fig.add_subplot(2, 1, 2)
ax.plot(np.arange(len(Y_test_pred_small)), Y_test_pred_small, label='Predicted')
ax.plot(np.arange(len(Y_test_pred_small)), Y_test_small, label='Actual')
ax.set_title(rmse_test_small)
ax.legend()
```

Out[44]:

```
<matplotlib.legend.Legend at 0x223b2d3c910>
```

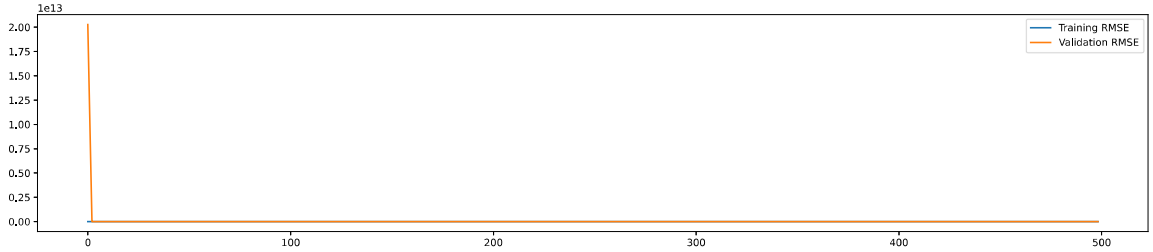


In [45]:

```
lambdas_small = np.arange(0, 500, 2)
rmse_train_small = []
rmse_validation_small = []
for l in lambdas_small:
    trained_model_ridge_small = Ridge(fit_intercept=False, alpha=l).fit(X = X_train_sma
ll, y = Y_train_small)
    rmse_train_small.append(np.sqrt(np.mean((trained_model_ridge_small.predict(X_train_
small) - Y_train_small)**2)))
    rmse_validation_small.append(np.sqrt(np.mean((trained_model_ridge_small.predict(X_v
al_small) - Y_val_small)**2)))
```

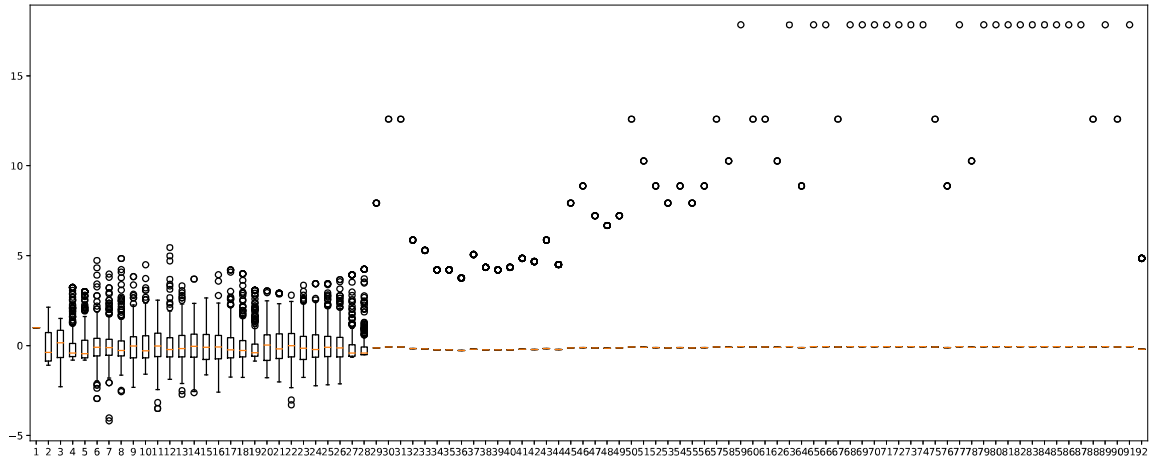
In [46]:

```
fig = plt.figure(figsize=[20, 4])
ax = fig.add_subplot(1, 1, 1)
ax.plot(lambdas_small, rmse_train_small, label='Training RMSE')
ax.plot(lambdas_small, rmse_validation_small, label='Validation RMSE')
ax.legend();
```



In [47]:

```
mu_small = np.mean(X_small.iloc[:,1:],0)
sigma_small = np.std(X_small.iloc[:,1:],0)
X_small.iloc[:,1:] = (X_small.iloc[:,1:] - mu_small) / sigma_small
fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(1, 1, 1)
ax.boxplot(X_small);
```

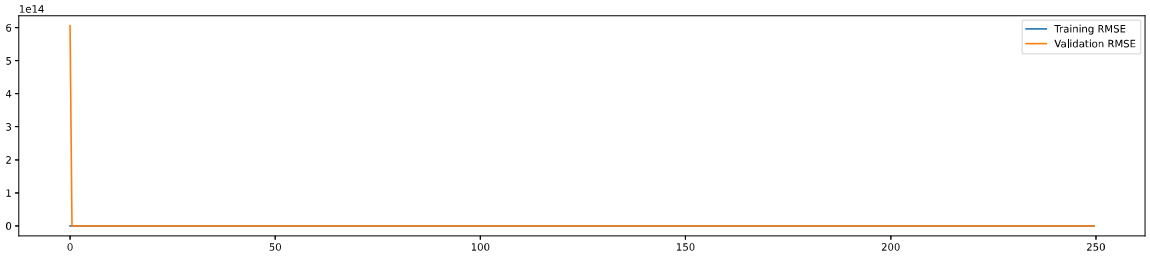


In [48]:

```
X_small = np.concatenate([X_small, dummies_small], axis=1)

X_train_small = X_small[0:training_samples_small, :]
Y_train_small = Y_std_small[0:training_samples_small]
X_val_small = X_small[training_samples_small:(training_samples_small + validation_sampl
es_small), :]
Y_val_small = Y_std_small[training_samples_small:(training_samples_small + validation_s
amples_small)]
X_test_small = X_small[(training_samples_small + validation_samples_small):, :]
Y_test_small = Y_std_small[(training_samples_small + validation_samples_small):]
```

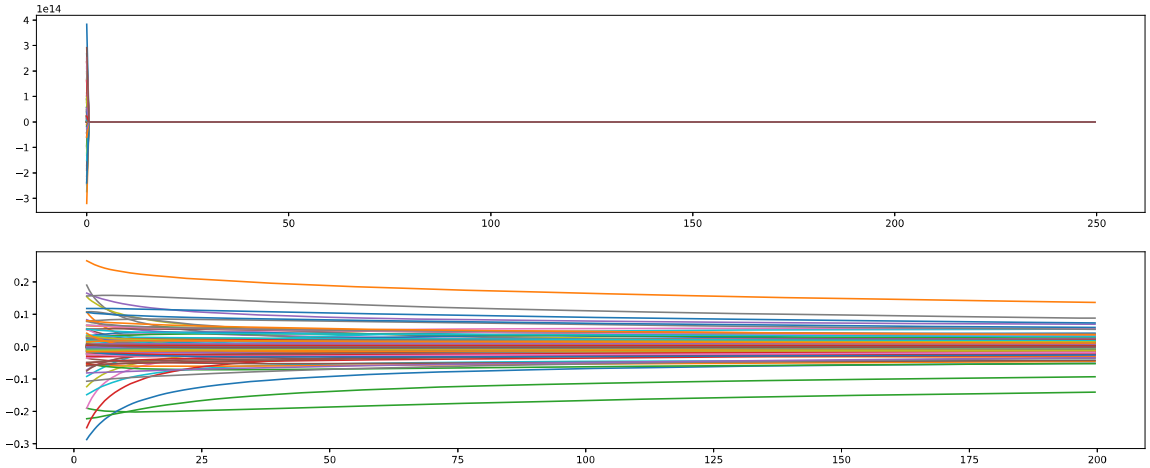
In [49]:

```
lambdas_small = np.arange(0, 250, 0.5)
rmse_train_small = []
rmse_validation_small = []
coeffs_small = []
for l in lambdas_small:
    trained_model_ridge_small = Ridge(fit_intercept=False, alpha=l).fit(X_train_small,
Y_train_small)
    coeffs_small.append(trained_model_ridge_small.coef_)
    rmse_train_small.append(np.sqrt(np.mean((trained_model_ridge_small.predict(X_train_
small) - Y_train_small)**2)))
    rmse_validation_small.append(np.sqrt(np.mean((trained_model_ridge_small.predict(X_v
al_small) - Y_val_small)**2)))

fig = plt.figure(figsize=[20, 4])
ax = fig.add_subplot(1, 1, 1)
ax.plot(lambdas_small, rmse_train_small, label='Training RMSE')
ax.plot(lambdas_small, rmse_validation_small, label='Validation RMSE')
ax.legend();
```



In [50]:

```
fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(2, 1, 1)
ax.plot(lambdas_small, coeffs_small);
coeffs_small = np.array(coeffs_small)
ax = fig.add_subplot(2, 1, 2)
ax.plot(lambdas_small[5:400], coeffs_small[5:400,:]);
```



In [ ]:

```
best_lambda_small = lambdas_small[np.argmin(rmse_validation_small)]
print(best_lambda_small)
trained_model_ridge_small = Ridge(fit_intercept=False, alpha=best_lambda_small).fit(X_t
rain_small, Y_train_small)

Y_train_pred_small = trained_model_ridge_small.predict(X_train_small)
Y_test_pred_small = trained_model_ridge_small.predict(X_test_small)
rmse_train_small = np.sqrt(np.mean((Y_train_pred_small - Y_train_small)**2))
rmse_test_small = np.sqrt(np.mean(((Y_test_pred_small*Y_sigma_small + Y_mu_small) - (Y_
test_small*Y_sigma_small + Y_mu_small))**2))

fig = plt.figure(figsize=[20, 8])
ax = fig.add_subplot(2, 1, 1)
ax.plot(np.arange(len(Y_train_pred_small)), Y_train_pred_small, label='Predicted')
ax.plot(np.arange(len(Y_train_pred_small)), Y_train_small, label='Actual')
ax.set_title(rmse_train_small)
ax.legend()
ax = fig.add_subplot(2, 1, 2)
ax.plot(np.arange(len(Y_test_pred_small)), Y_test_pred_small*Y_sigma_small + Y_mu_small
, label='Predicted')
ax.plot(np.arange(len(Y_test_pred_small)), Y_test_small*Y_sigma_small + Y_mu_small, lab
el='Actual')
ax.set_title(rmse_test_small)
ax.legend();
```

In [ ]:

```

```