Homework 2: Turkish Inflation vs Unemployment, Linear Regression and 95% Prediction Intervals

In this homework we were asked to find and download any data of our choice that we expected to be correlated. I was not very creative and ended up choosing Turkey's inflation and unemployment data that I retrieved from the Worldbank API. My expectation was that inflation and unemployment should be negatively correlated as when employment increases, wages increase which drives inflation.

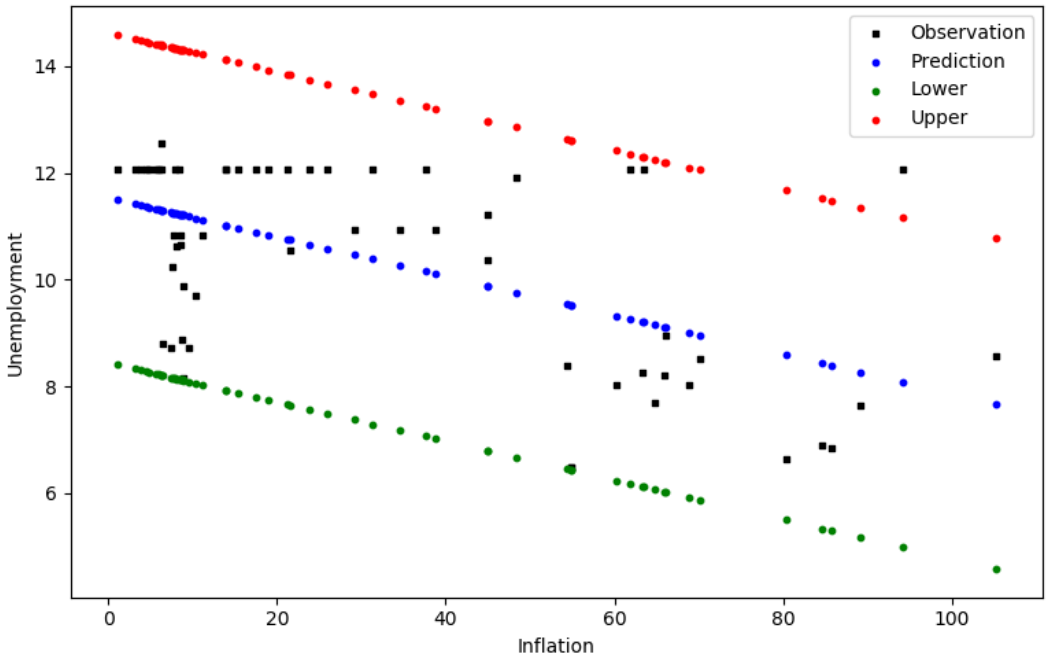
After accesing the data, we were asked to clean it from NaNs, for which I implemented the CleanData class. This class not only fills NaNs, but first orders the data ascendingly as to reduce errors caused by the filling process of the NaNs. The class also checks for other basics such as input type and data lengths being equal.

Once the data is accessed and cleaned, we were asked to apply a regression formula on it to make predictions and find confidence intervals. I am not coming from a statistics, mathematics or engineering background, so I choose the only regression formula I learnt in my Business classes where; b0 = ybar – (xbar*b1) and b1 = sum ((xi-xbar)*(yi-ybar)) / sum ((xi-xbar)^2). For the 95% prediction intervals I used the formula yhat +- RMSE*2, where yhat is the prediction and RMSE is root mean squared error.

Using the betas from the regression, I predicted what values (unemployment) would result from my independent variable (inflation). So, unemployment = b0 + b1*inflation. On each prediction value I also applied the 95% prediction interval to find an upper and lower limit. I also masked the results to show the outliers, the data that are outside the 95% prediction interval. **The betas were as follows:**

b0	b1
11.5327	-0.0367

The resulting scatter plot with predictions and 95% confidence interval:



Numerical Results and the final resulting pandas.DataFrame:

	inflation	unemployment	upper	predictions	lower	outlier	error
0	1.119638	12.0600	14.581885	11.491696	8.401508	False	-0.568304
1	3.172857	12.0600	14.506631	11.416442	8.326253	False	-0.643558
2	3.888320	12.0600	14.480408	11.390219	8.300030	False	-0.669781
3	4.555534	12.0600	14.455953	11.365764	8.275576	False	-0.694236
4	4.924194	12.0600	14.442441	11.352252	8.262064	False	-0.707748
5	5.664740	12.0600	14.415299	11.325110	8.234921	False	-0.734891
6	6.046240	12.0600	14.401316	11.311127	8.220938	False	-0.748873
7	6.250977	12.5522	14.393812	11.303623	8.213434	False	-1.248577
.							
.							
.							
49	65.978568	8.2100	12.204684	9.114495	6.024306	False	0.904495
50	66.093843	8.9620	12.200459	9.110270	6.020081	False	0.148270
51	68.809643	8.0400	12.100920	9.010731	5.920542	False	0.970731
52	70.076104	8.5090	12.054502	8.964313	5.874124	False	0.455313
53	80.412151	6.6290	11.675666	8.585477	5.495289	False	1.956477
54	84.641343	6.8880	11.520658	8.430470	5.340281	False	1.542470
55	85.669362	6.8410	11.482980	8.392791	5.302602	False	1.551791
56	89.113317	7.6440	11.356752	8.266563	5.176375	False	0.622563
57	94.260860	12.0600	11.168085	8.077896	4.987708	True	-3.982104
58	105.214986	8.5770	10.766596	7.676407	4.586218	False	-0.900593

Conclusion:

The data was negatively correlated as I expected, the numpy.corrcoef resulted in a -0.56 which is fairly strong. My linear regression estimates look fine on the plot, however I am not very happy with the prediction intervals as they are linear, I would have wanted a method with curved intervals near the middle. Overall there was only one outlier and I am ok with this analysis.