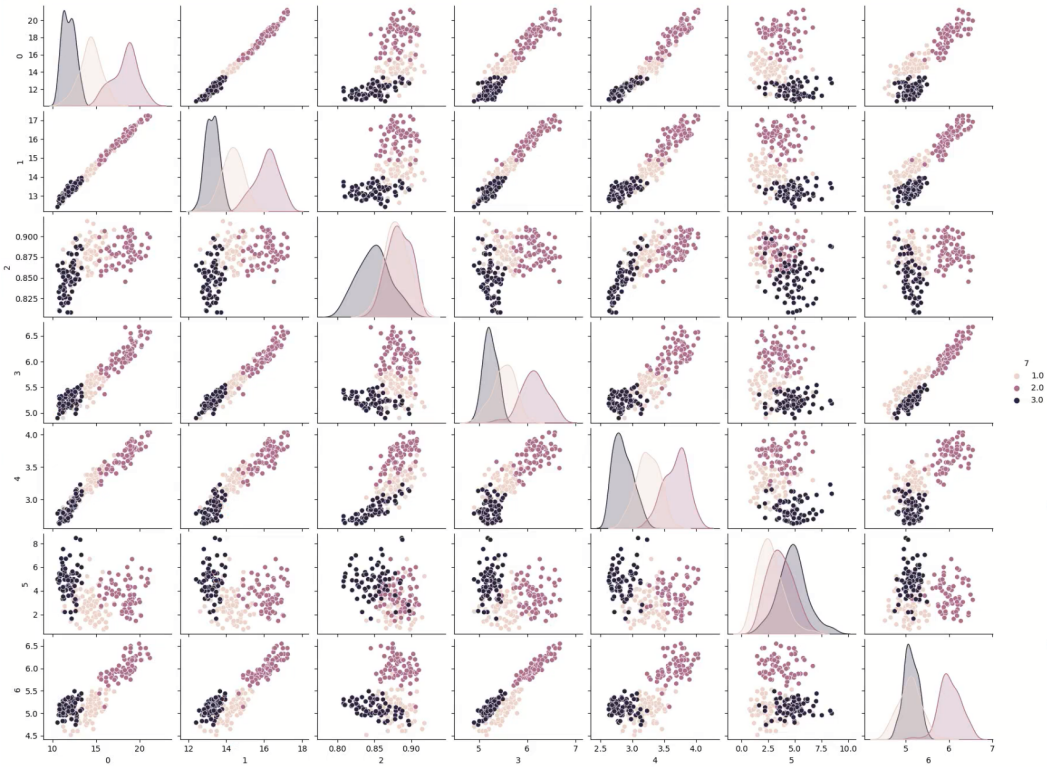
Q1

```
Neighbouring words at most 2 edits apart using Levenshtein distance.

Target: Kirkland
['Kirikland']


Target: Valpariso
['Valapariso', 'Valparaiso']


Target: Lody
['Cody', 'Lodi', 'Lordy', 'Lowdy', 'Codi', 'Laudy']


Target: Bakersfeild
['Bakersfield']


Target: Fort Le
['Fort Lee']
```

Discussion:

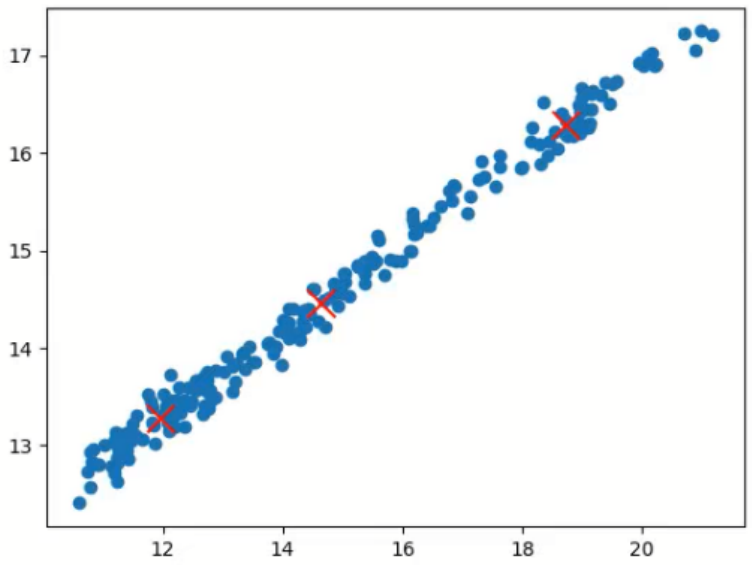
I used the Levenshtein distance to implement this knn style function to determine which cities have spelling similar to the five randomly selected target words. I decided to use this distance metric over cosine similarity, which is another distance metric for string comparison, because cosine compares vectors of data and would have been more appropriate when we did NLP on word vectors in the previous module. However, since the goal was to compare one word at a time to see how far apart it was for each of the words in the Misspelled City Names dataset, I decided to use Levenshtein distance which supplies a value of how many edits need to be done to change the compared word to the target. I implemented it in the function knn_lev_dist, which simply takes the target word and finds the distance from that word to all words in the input list. The function then returns the similar words (minus target word) that are at most k edits away from the target word. I therefore refer to this as a knn style function because it does not do any further discrimination, for e.g. with the target word “Lody” similar words provided are “Cody” which could truly be a misspelled word, or a validly spelled city. The other reason it is knn style is because it returns based on k edits away instead of the try k most similar words, e.g. if k=2 then 2 similar words would be returned in knn.

Q2



Discussion:

For section B, I decided the data does not need feature engineering or transformation because the kde on the pairwise plot (above) shows the data is normal, and there are no serious outliers on the data. So I proceeded to run kmeans clustering, with k=3 clusters on the data and got the below result showing the cluster centroids. In a true unsupervised example I would have calculated the entropy for k=1 to 10 and plotted that and then used the elbow method to determine the best k.



Q3

Manhattan distance matrix on iris data:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	1.000000	0.183616	0.175847	0.283898	0.086394	1.712100	1.342043	1.590160	1.351224	1.411252	2.154661	1.891008	2.228814	2.435499	2.239642
1	0.183616	1.000000	0.158898	0.433616	0.270009	1.695151	1.491761	1.573211	1.334275	1.394303	2.137712	1.874058	2.211864	2.418550	2.222693
2	0.175847	0.283898	1.000000	0.308616	0.262241	1.536252	1.332863	1.414313	1.175377	1.235405	1.978814	1.715160	2.052966	2.259652	2.063795
3	0.086394	1.712100	1.342043	1.000000	0.314736	1.678202	1.169256	1.556262	1.233992	1.294021	2.204096	1.857109	2.194915	2.318267	2.122411
4	1.590160	1.351224	1.411252	2.154661	1.000000	1.715160	1.289548	1.593220	1.354284	1.414313	2.157721	1.894068	2.231874	2.438559	2.242702
5	1.891008	2.228814	2.435499	2.239642	0.158898	1.000000	1.425612	0.489171	1.027542	0.800847	0.914783	0.456685	0.572269	0.723399	0.694209
6	0.433616	0.270009	1.695151	1.491761	1.573211	1.334275	1.000000	1.303672	0.398070	0.624765	1.951507	1.604520	1.942326	1.815678	1.869821
7	1.394303	2.137712	1.874058	2.211864	2.418550	2.222693	0.308616	1.000000	0.905603	0.678908	0.647834	0.300847	0.638653	0.845339	0.649482
8	0.262241	1.536252	1.332863	1.414313	1.175377	1.235405	1.978814	1.715160	1.000000	0.226695	1.553437	1.206450	1.544256	1.417608	1.471751
9	2.052966	2.259652	2.063795	0.314736	1.678202	1.169256	1.556262	1.233992	1.294021	1.000000	1.326742	0.979755	1.317561	1.190913	1.245056
10	2.204096	1.857109	2.194915	2.318267	2.122411	1.715160	1.289548	1.593220	1.354284	1.414313	1.000000	0.458098	0.342514	0.780838	0.303908
11	2.157721	1.894068	2.231874	2.438559	2.242702	1.425612	0.489171	1.027542	0.800847	0.914783	0.456685	1.000000	0.337806	0.627825	0.348635
12	0.572269	0.723399	0.694209	1.303672	0.398070	0.624765	1.951507	1.604520	1.942326	1.815678	1.869821	0.905603	1.000000	0.540019	0.155838
13	0.678908	0.647834	0.300847	0.638653	0.845339	0.649482	0.226695	1.553437	1.206450	1.544256	1.417608	1.471751	1.326742	1.000000	0.612524
14	0.979755	1.317561	1.190913	1.245056	0.458098	0.342514	0.780838	0.303908	0.337806	0.627825	0.348635	0.540019	0.155838	0.612524	1.000000

Euclidean distance matrix on iris data:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	1.000000	0.119064	0.106142	0.189397	0.052868	0.925751	0.793218	0.917355	0.726987	0.764937	1.282314	1.059546	1.249968	1.276547	1.253929
1	0.119064	1.000000	0.129515	0.300940	0.148914	0.886060	0.860292	0.905527	0.766007	0.791233	1.261115	1.035168	1.220406	1.248549	1.231295
2	0.106142	0.189397	1.000000	0.209022	0.142879	0.844406	0.746198	0.864146	0.664602	0.703503	1.230369	0.997445	1.185672	1.195425	1.192999
3	0.052868	0.925751	0.793218	1.000000	0.187579	0.949349	0.654196	0.913079	0.634463	0.692853	1.286515	1.067305	1.261169	1.273065	1.256370
4	0.917355	0.726987	0.764937	1.282314	1.000000	0.930819	0.784050	0.905903	0.721122	0.759349	1.269729	1.051568	1.242892	1.278998	1.244012
5	1.059546	1.249968	1.276547	1.253929	0.129515	1.000000	0.792330	0.348467	0.568873	0.495622	0.546043	0.293916	0.412461	0.373941	0.453388
6	0.300940	0.148914	0.886060	0.860292	0.905527	0.766007	1.000000	0.687139	0.226692	0.334670	0.996983	0.819002	0.980302	0.943598	0.948698
7	0.791233	1.261115	1.035168	1.220406	1.248549	1.231295	0.209022	1.000000	0.487087	0.376752	0.374858	0.193152	0.379252	0.501671	0.363324
8	0.142879	0.844406	0.746198	0.864146	0.664602	0.703503	1.230369	0.997445	1.000000	0.137922	0.818123	0.614366	0.784889	0.754590	0.763322
9	1.185672	1.195425	1.192999	0.187579	0.949349	0.654196	0.913079	0.634463	0.692853	1.000000	0.708094	0.514634	0.687280	0.673487	0.667743
10	1.286515	1.067305	1.261169	1.273065	1.256370	0.930819	0.784050	0.905903	0.721122	0.759349	1.000000	0.267385	0.197595	0.446715	0.160071
11	1.269729	1.051568	1.242892	1.278998	1.244012	0.792330	0.348467	0.568873	0.495622	0.546043	0.293916	1.000000	0.195736	0.347478	0.199455
12	0.412461	0.373941	0.453388	0.687139	0.226692	0.334670	0.996983	0.819002	0.980302	0.943598	0.948698	0.487087	1.000000	0.281855	0.082740
13	0.376752	0.374858	0.193152	0.379252	0.501671	0.363324	0.137922	0.818123	0.614366	0.784889	0.754590	0.763322	0.708094	1.000000	0.322519
14	0.514634	0.687280	0.673487	0.667743	0.267385	0.197595	0.446715	0.160071	0.195736	0.347478	0.199455	0.281855	0.082740	0.322519	1.000000