CHEMENG 4H03

Assignment 2 – Principal Component Analysis

February 13th, 2023

Michael Djurdjevic, djurdjm, 400132129

Aron Markandaier, markanda, 400121110

Harsahib Matharoo, matharoh, 400185871

Problem 1:

1.

Feature scaling is an additional way of data preprocessing that is frequently employed. A technique called "feature scaling" is used to scale or "normalize" the values of the various features in a dataset such that they are all on the same scale or range. There are various techniques for feature scaling, including min-max scaling, quantile transformer scaling, power transformer scaling, unit vector scaling, and robust scaling (Roy, 2023). This estimator scales and translates each feature individually such that it is in the given range on the training set, e.g., between zero and one. This Scaler shrinks the data within the range of -1 to 1 if there are negative values. We can set the range like [0,1] or [0,5] or [-1,1].

Min-max scaling involves scaling the values of a feature to a range between 0 and 1, by subtracting the minimum value of the feature and dividing it by the range (i.e., the difference between the maximum and minimum values). Robust scaling options involve scaling the values of a feature to be between the 25th and 75th percentiles, to reduce the influence of outliers. The RobustScaler() technique uses the same principle of scaling as MinMaxScaler(). However, it uses the interquartile range instead of the min-max, which makes it more reliable regarding outliers (Roy, 2023).

Scikit-learn provides several modules for data preprocessing, including feature extraction, feature selection, and feature scaling. These modules are designed to work seamlessly with scikit-learn's machine learning algorithms, making it easy to build end-to-end machine learning pipelines (Chouhbi, 2020). Scikit-learn provides a range of functions for feature scaling, including StandardScaler, MinMaxScaler, MaxAbsScaler, RobustScaler, and QuantileTransformer. These functions can be used to standardize the scale and distribution of the input features in a dataset, which is an important preprocessing step for many machine learning algorithms (Chouhbi, 2020).

2.

Feature scaling can be applied to a wide range of data types, including numerical data (both continuous and discrete), categorical data that has been numerically encoded, and ordinal data. One of the main advantages of feature scaling is that it can help to improve the performance and convergence of machine learning models, by reducing the impact of features with larger magnitudes. It also gives better error surface shape, and another fact is that weight decay and optimization can be achieved efficiently. However, one potential disadvantage of feature scaling is that it can distort the shape of the data distribution, particularly if there are extreme values or outliers in the dataset (G, 2018).

Personal Example -

Assume we have a dataset of student exam scores with two features: science score and reading score. The science score ranges from 50 to 100 and the reading score ranges from 70 to 90. We would like to use a machine learning to predict the students' overall exam performance based on these features, but however, before proceeding to this step, we need to preprocess the data first.

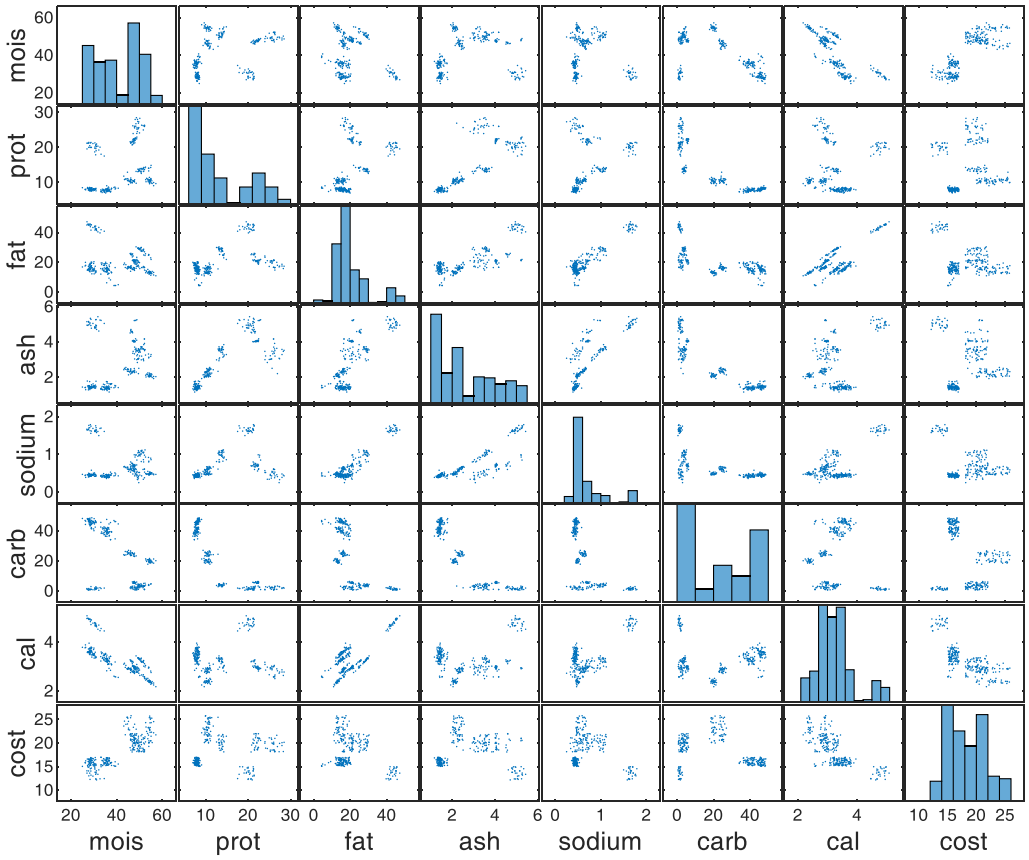
We can use standardization to scale both features to have a mean of 0 and a standard deviation of 1. To do this, we subtract the mean of each feature from every value of that feature, then divide by the standard deviation of the feature:

Scaled science score = (science score - mean(science score)) / standard deviation(science score) Scaled reading score = (reading score - mean(reading score)) / standard deviation(reading score)

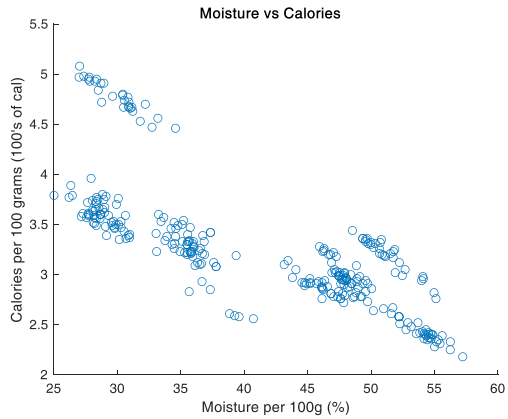
After scaling, both features will have a mean of 0 and a standard deviation of 1. This will ensure that the algorithm is not biased towards features with larger scales and will help it converge faster during the training part. Ensures that no one feature dominates the others.

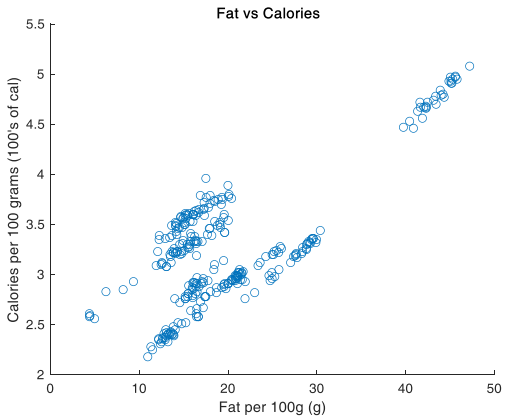
Problem 2:

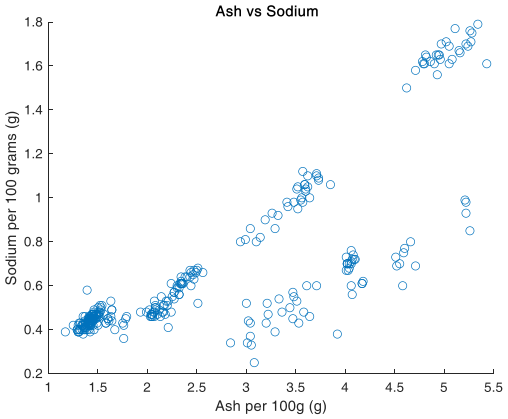
1.



Looking at the above we can observe many correlated variables, three examples are moisture and calories, fat and calories, and ash and sodium.







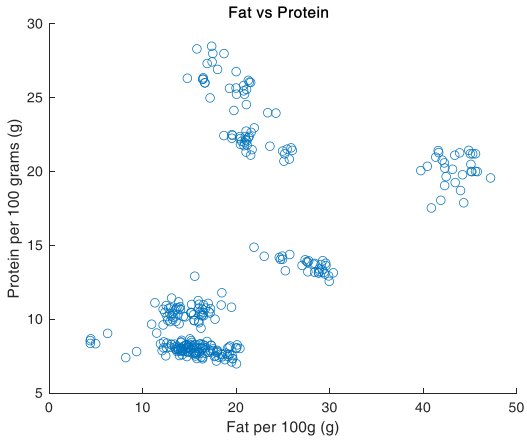
2.

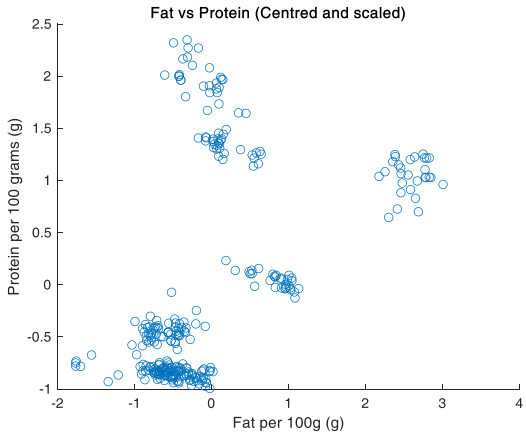
Means

mois = 40.9030666666667	prot = 13.3735666666667	fat = 20.2295333333333
ash = 2.63323333333333	sod = 0.669400000000000	carb = 22.8647666666667
cal = 3.27100000000000	cost = 18.3364666666667	

Standard Deviations

mois = 9.55298664161135	prot = 6.43439235375773	fat = 8.97565830020072
ash = 1.26972354148250	sod = 0.370357740822696	carb = 18.0297224581694
cal = 0.620034253018644	cost = 3.21162502479485	





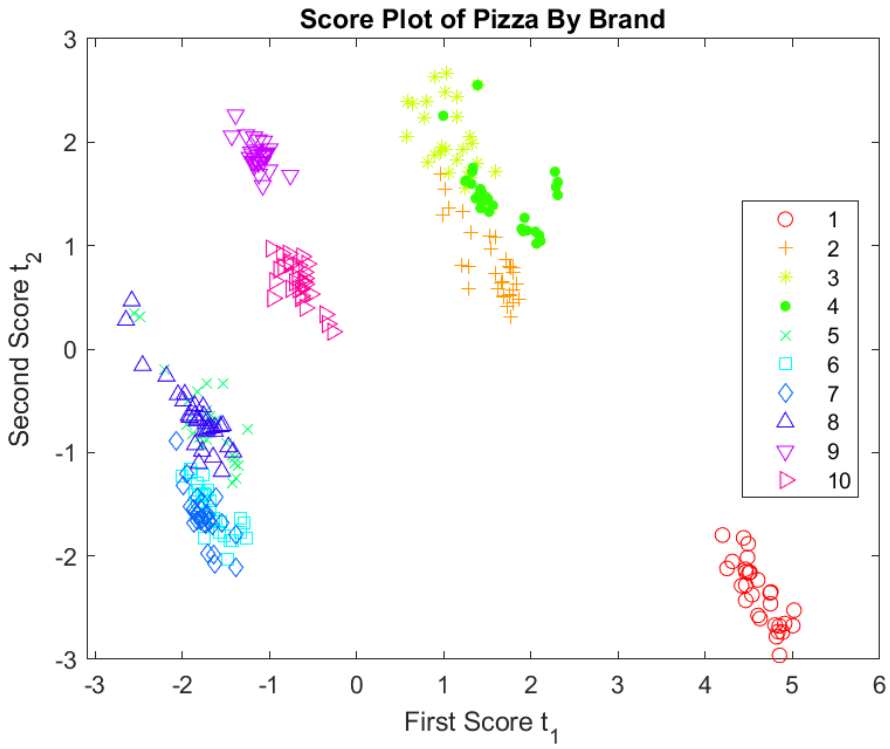
There seems to be different distinct groups of points which all have slightly different trends. Some of these trends include the group on the top there is a negative correlation between fat and protein while the group on the bottom left has a more positive correlation. This is also compared with the group on the far right which exhibits no visible correlation. Both plots look relatively similar after centering and scaling as there wasn't much affect from any outliers.

Problem 3:

1.

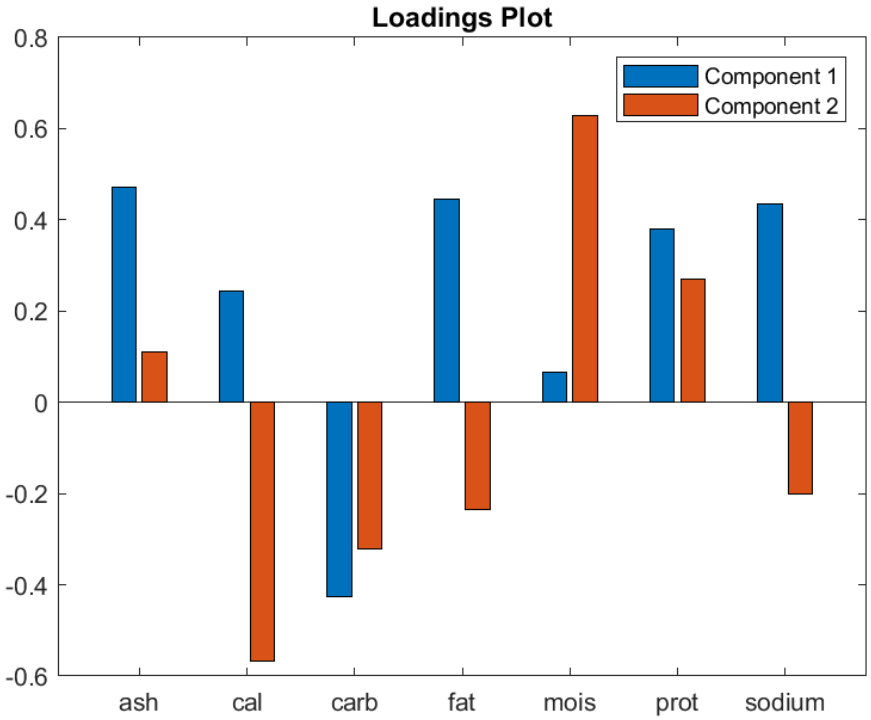
The R^2 for the 3 principal components are 0.5960, 0.9232, and 0.9824. Which means by the third component most of the variance has been realized by the three components. We can see from the first component a lot of the variance has been realized, with a major jump by adding the second component, and finally a decent increase with the third. However, with the third components resulting in an R^2 of 0.98 this means a fourth component will still increase the R^2, but it will be extremely diminishing in return and not substantially explain additional variance in the data.

2.



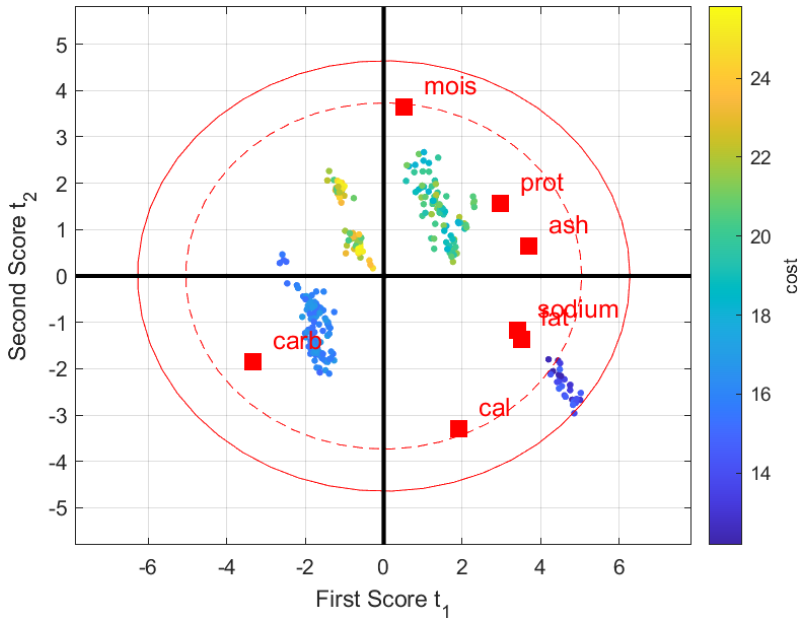
Brands tend to be generally consistent with their first and second component score resulting in groups of points with the same brand. Some brands also overlap in their scores, such as brands 5, 6,7, and 8. Brands to tend to vary more on the second components compared to the first.

3.



From the loading plot it's observed that ash, calories, fat, protein, and sodium have some positive correlation which is represented by component 1 while showing negative correlation with carbohydrates. We can also observe moisture and protein have positive correlation with high loading contributions to component 2, and negative correlation with calories, carbohydrates, fat, and sodium.

4.



A lower score for the first component and a higher score for the second component will result in a higher pizza price.

5.

To have a higher price we can observe that a lower sodium, fat, and calories, along with a higher moisture content will yield in the pricier pizza. This is because the loading vectors of sodium, fat, and calories point in the opposite direction of the priciest pizzas, and the loadings vector for moisture points very closely to the direction of the priciest pizzas.

Problem 4:

	NIPALS algorithm						Eigenvalue decomposition algorithm					
t =		p =		R2 =			t =		p =		R2 =	
5.002	-2.6747	0.0647	0.6283	0.596	0.9232		5.002	2.6747	0.0647	-0.6283	0.5960	0.9232
5.0154	-2.5251	0.3788	0.2697				5.0154	2.5251	0.3788	-0.2697		
4.7974	-2.6692	0.4467	-0.2344				4.7974	2.6692	0.4467	0.2344		
4.4621	-2.2812	0.4719	0.111				4.4621	2.2812	0.4719	-0.111		
4.4644	-2.1556	0.4357	-0.2017				4.4644	2.1556	0.4357	0.2017		
4.4973	-2.1644	-0.4249	-0.3203				4.4973	2.1644	-0.4249	0.3203		
4.3082	-2.0536	0.2445	-0.5675				4.3082	2.0536	0.2445	0.5675		
4.7499	-2.3492						4.7499	2.3492				
4.8465	-2.6767						4.8465	2.6767				
4.9082	-2.6541						4.9082	2.6541				
4.8329	-2.732						4.8329	2.732				
4.8832	-2.7379						4.8832	2.7379				
4.6031	-2.575						4.6031	2.575				
4.4142	-2.2879						4.4142	2.2879				
4.4593	-2.1293						4.4593	2.1293				
4.7455	-2.463						4.7455	2.463				
4.4377	-1.8253						4.4377	1.8253				
4.195	-1.7985						4.195	1.7985				
4.4894	-1.8829						4.4894	1.8829				
4.5999	-2.2313						4.5999	2.2313				
4.8492	-2.9614						4.8492	2.9614				

Sample output for both the NIPALS and eigenvalue decomposition algorithms are shown above. The only notable difference is that the second component of the score vector t, and the loading vector p, was inverted. This does not affect our PCA model however as:

$$\hat{X} = \boldsymbol{t}\,\boldsymbol{p}^T \equiv (-\boldsymbol{t})(-\boldsymbol{p}^T)$$

If there were some missing data in the dataset however only the NIPALS algorithm would be able to work correctly. The difference in sign is due to NIPALS being an iterative method that is performing a minimization optimization on the residuals. This could cause the algorithm to converge to the inverted eigenvector case.

References

Chouhbi, K. (2020, July 8). Preprocessing data: Feature scaling. Medium. Retrieved February 11, 2023, from https://towardsdatascience.com/preprocessing-data-feature-scaling-cc28c508e8af

Roy, B. (2023, February 4). All about feature scaling. Medium. Retrieved February 11, 2023, from https://towardsdatascience.com/all-about-feature-scaling-bcc0ad75cb35

-, G. (2018, June 26). Feature scaling: From The GENESIS. Retrieved February 11, 2023, from https://www.fromthegenesis.com/feature-scaling/