COURSEWORK REPORT - 1

DATA70132 : STATISTICS AND MACHINE LEARNING 2

2021-2022

Submitted By : 10934372

CONTENT

DIMENSIONALITY REDUCTION

1. Dimensionality Reduction

Dimensionality reduction is the process of simplifying a high-dimensional dataset by reducing the features under consideration, to create a low-dimensional dataset. While reducing the dimensionality of the data, we need to make sure that it maintains its informational integrity for further modelling. Dimensionality reduction is only useful as long as the predictive accuracy of the model is maintained or improved. The ultimate goal of dimensionality reduction is to make the data more comprehensible.

This can be seen in the case of the *Ansur Male* dataset being used; it consists of 109 features, each explaining an attribute of an individual in the army. However, some of the features may be highly correlated with other features thus providing no additional information. Using dimensionality reduction in this case, we can ignore these redundant attributes and include only the important attributes while modelling. This prevents our modelling from being plagued by the "Curse of Dimensionality" [3].

2. Principal Component Analysis

Principal Component Analysis is an unsupervised linear dimensionality reduction technique. It is a statistical procedure to transform a set of values of possibly correlated features into a set of values of linearly uncorrelated features. It does so by projecting the data in a high-dimensional space onto a low-dimensional subspace [4].

PCA reduces the dimensionality of the dataset by selecting the most important features that capture maximum information about the dataset, and eliminates the ones that show fewer variations. The features are selected on the basis of variance they cause in the output such that the first principal component selected has the maximum variance, then the second component and so on.

3. Data pre-processing and variable selection

The first step when dealing with a data problem is to perform data pre-processing to ensure that the quality of the data is trustworthy. For this dataset, on checking the missing value percentages, it was found that the *Ethnicity* column had a high percentage, therefore it was removed. No other columns had missing values. Also no duplicate records were found. For feature selection, the non-numerical columns and the *subjectid* column were removed, since PCA should preferably be used for numerical values that belong on a coordinate plane [1]. The target variable *Component* was also separated from the dataset.

Following the pre-processing, 10 features were randomly selected to be used for PCA, ensuring that they covered the most commonly used anthropometric measurements [2].

Once we had the cleaned dataset with the selected features, an exploratory data analysis was performed to understand the data being used. From Figure 3.1 we can see that the features are correlated to each other and Figure 3.2 shows the correlation matrix for all the features.

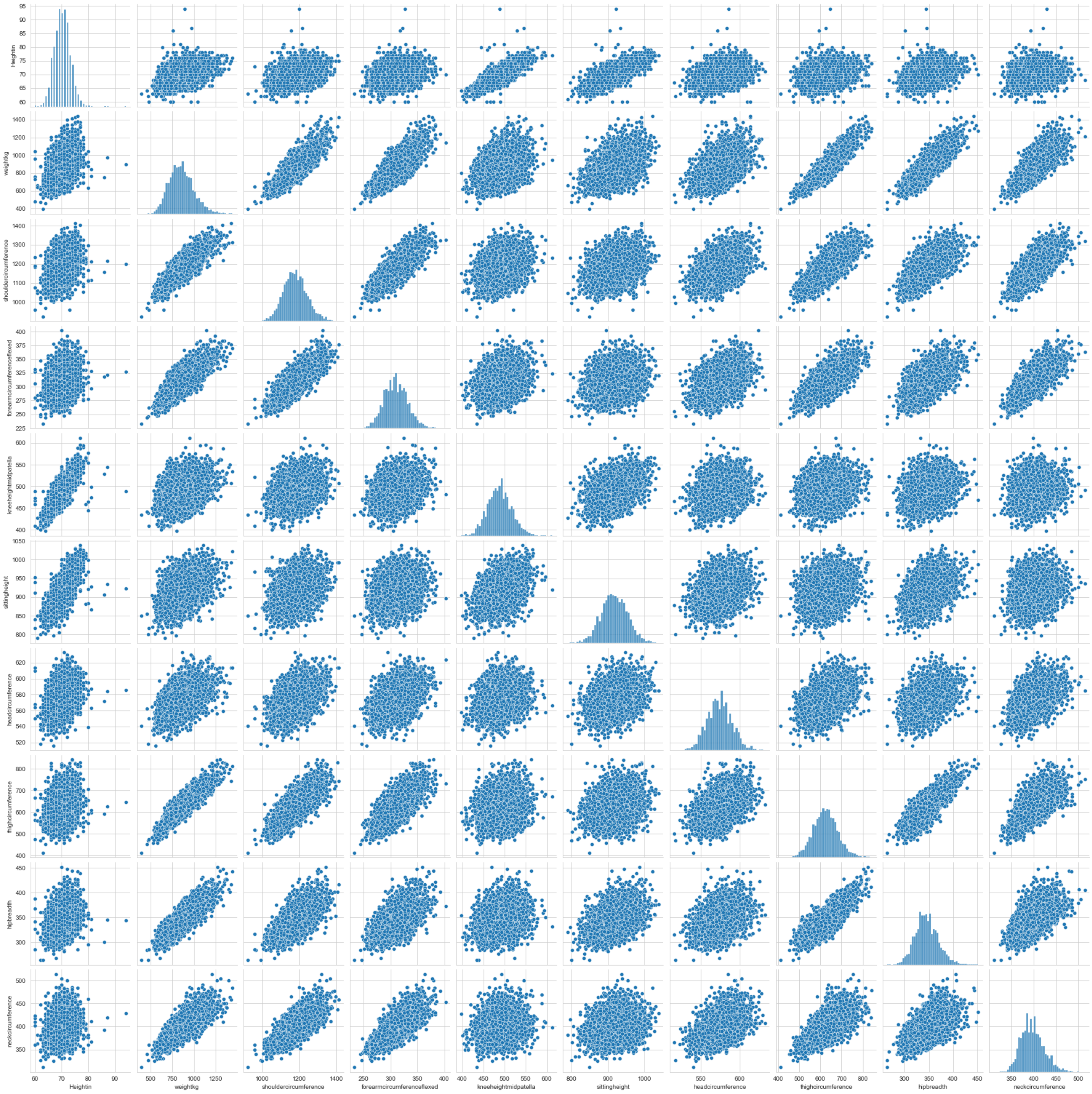


Figure 3.1 : Pair plot for the dataset

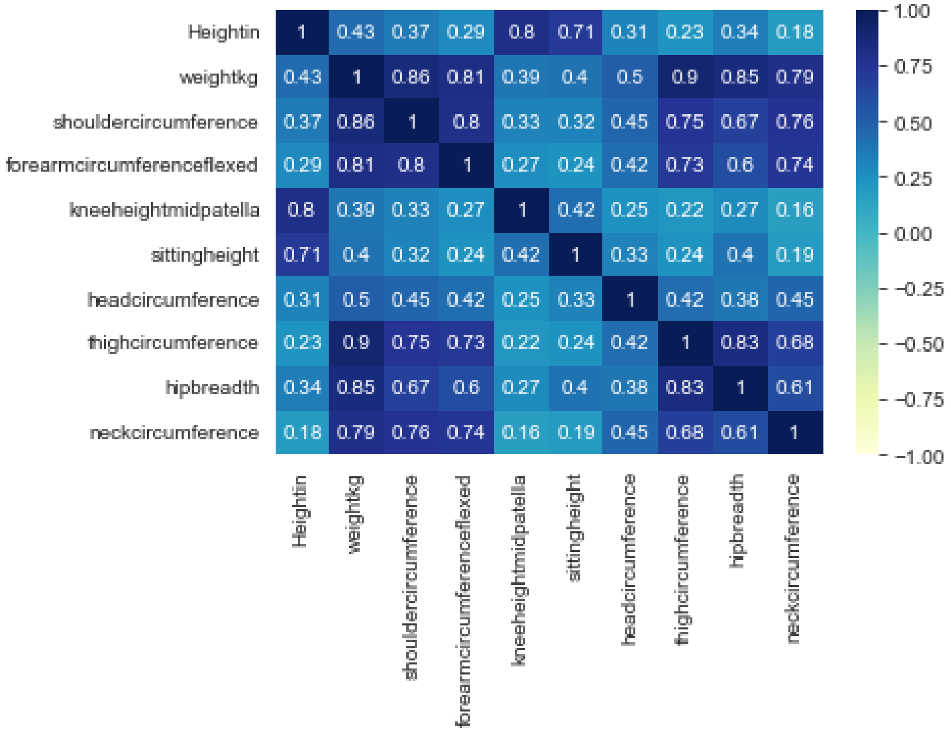


Figure 3.2 : Heatmap for the selected features

4. PCA results

After performing PCA on the dataset we see that:

1. The principal components obtained are uncorrelated unlike the original features as shown in the pairplot in Figure 4.1
2. The first 5 principal components are enough to give 92% of the information representation, therefore we can keep these for further analysis instead of the original 10 features if we want to retain maximum information for predictive modelling (Figure 4.2)
3. If we only want to analyse the data for statistical use, using the first 3 PC would suffice since they capture about 80% of the information (Figure 4.2)
4. From the scree plot (Figure 4.3) we find the elbow at PC 3, meaning using minimum 3 principal components we can sufficiently describe the data, thus effectively reducing the dimensionality from 10 to 3
5. The loadings for each principal component for each feature can be seen in the Figure 4.4
6. From Figure 4.5 we see that when we use the first two PC to understand the data based on the target variable class, a clear cluster separation is not visible
7. From Figure 4.5 we see that the target class “Regular Army” and “Army National Guard” show greater variability than “Army Reserve”

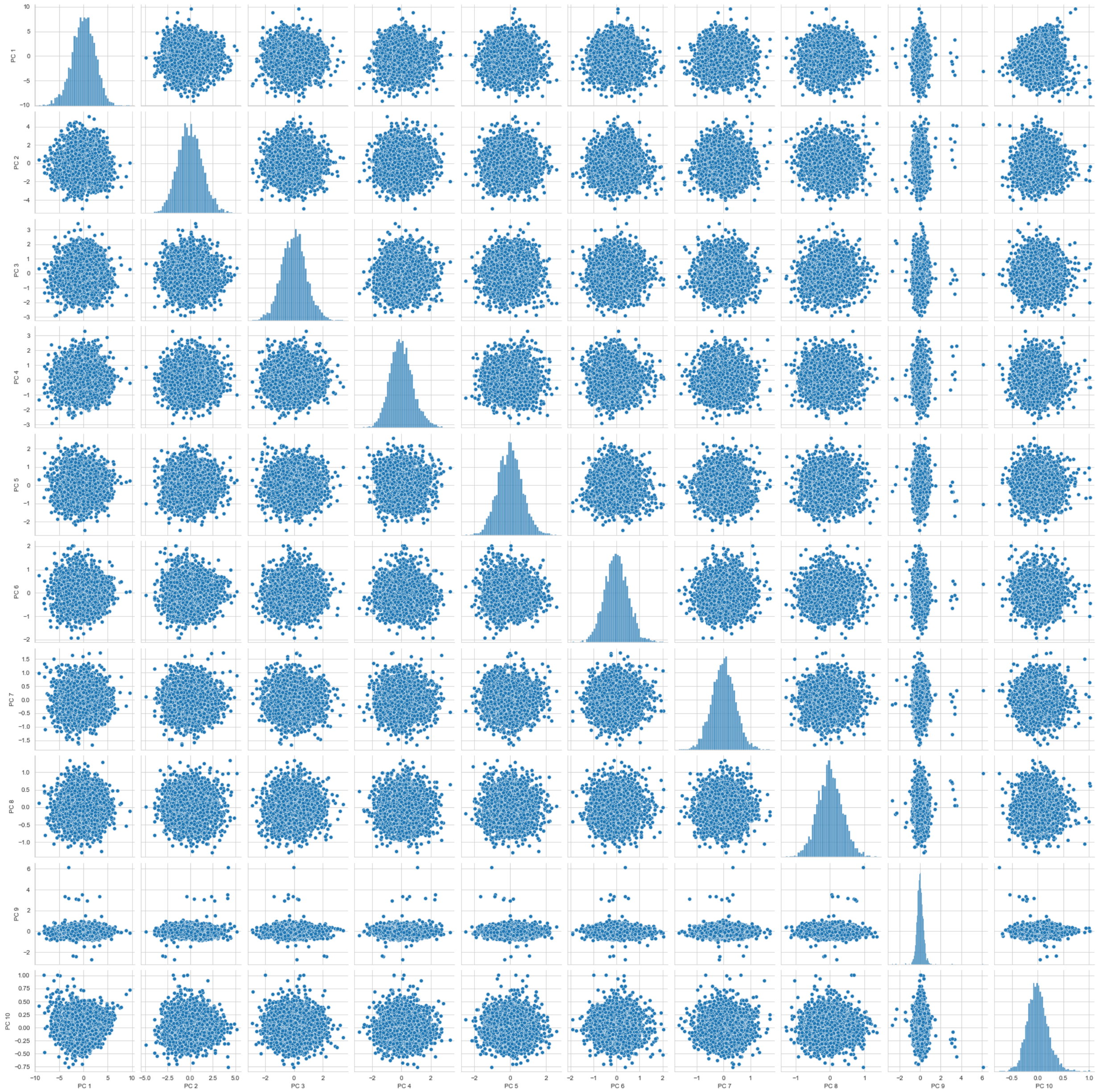


Figure 4.1 : Pair plot for Principal components

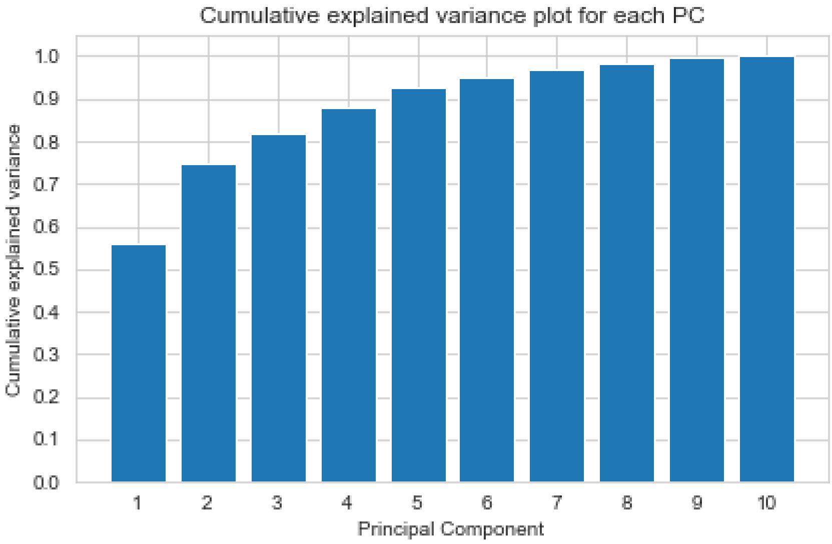


Figure 4.2: Cumulative Explained variance plot for each principal Component

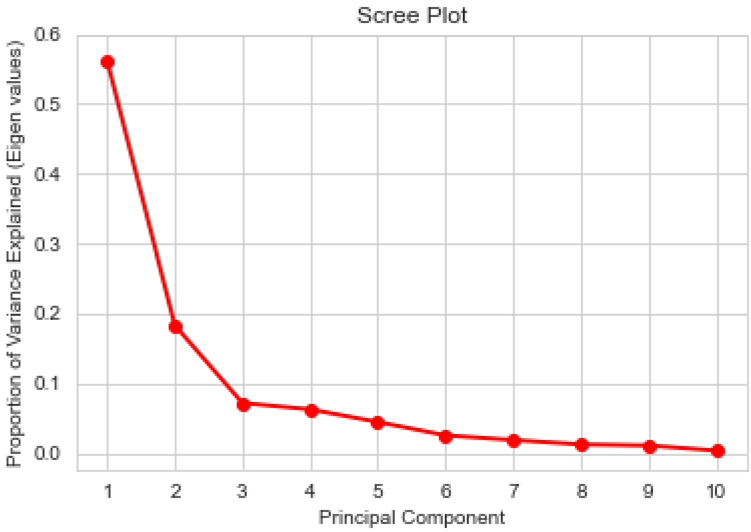
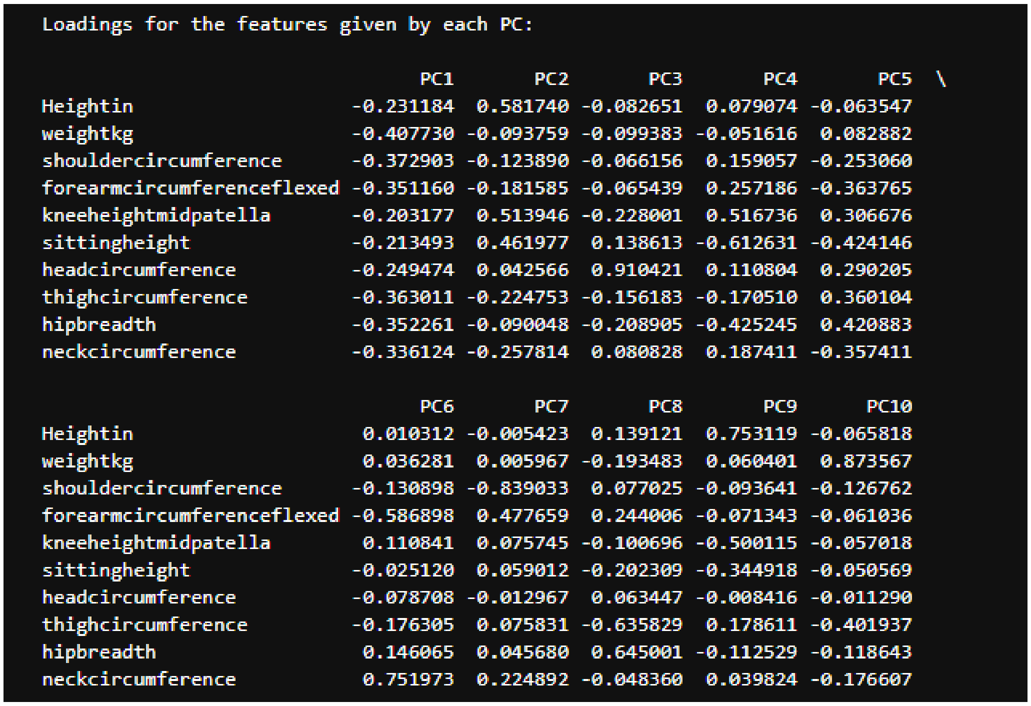


Figure 4.3: Scree Plot for PCA



```
Loadings for the features given by each PC:

                                 PC1       PC2       PC3       PC4       PC5  \
Heightin                   -0.231184  0.581740 -0.082651  0.079074 -0.063547
weightkg                   -0.407730 -0.093759 -0.099383 -0.051616  0.082882
shouldercircumference      -0.372903 -0.123890 -0.066156  0.159057 -0.253060
forearmcircumferenceflexed -0.351160 -0.181585 -0.065439  0.257186 -0.363765
kneeheightmidpatella       -0.203177  0.513946 -0.228001  0.516736  0.306676
sittingheight              -0.213493  0.461977  0.138613 -0.612631 -0.424146
headcircumference          -0.249474  0.042566  0.910421  0.110804  0.290205
thighcircumference         -0.363011 -0.224753 -0.156183 -0.170510  0.360104
hipbreadth                 -0.352261 -0.090048 -0.208905 -0.425245  0.420883
neckcircumference          -0.336124 -0.257814  0.080828  0.187411 -0.357411

                                 PC6       PC7       PC8       PC9      PC10
Heightin                    0.010312 -0.005423  0.139121  0.753119 -0.065818
weightkg                    0.036281  0.005967 -0.193483  0.060401  0.873567
shouldercircumference      -0.130898 -0.839033  0.077025 -0.093641 -0.126762
forearmcircumferenceflexed -0.586898  0.477659  0.244006 -0.071343 -0.061036
kneeheightmidpatella        0.110841  0.075745 -0.100696 -0.500115 -0.057018
sittingheight              -0.025120  0.059012 -0.202309 -0.344918 -0.050569
headcircumference          -0.078708 -0.012967  0.063447 -0.008416 -0.011290
thighcircumference         -0.176305  0.075831 -0.635829  0.178611 -0.401937
hipbreadth                  0.146065  0.045680  0.645001 -0.112529 -0.118643
neckcircumference           0.751973  0.224892 -0.048360  0.039824 -0.176607
```

Figure 4.4: Loadings for the features for each Principal Component

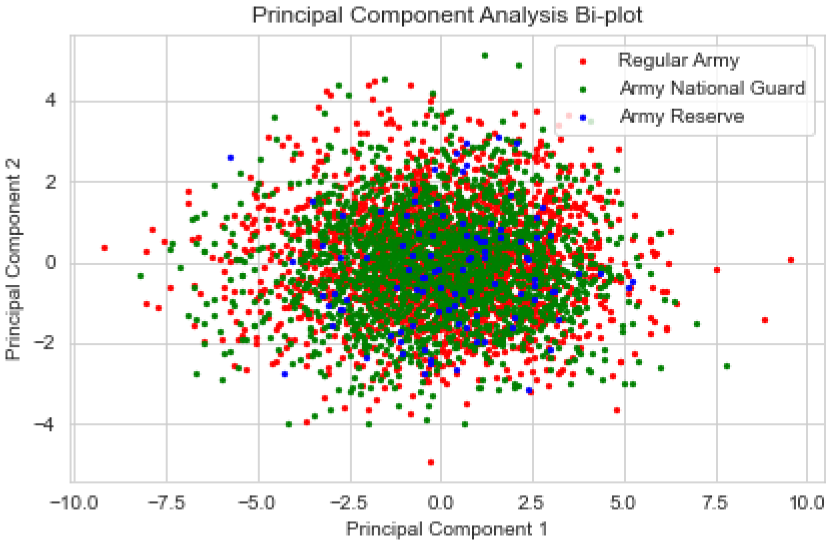


Figure 4.5: Bi-plot for dataset using the first two principal components and target variable

5. Factor Analysis

Factor Analysis is an exploratory data analysis method used to search important underlying factors or latent features from a set of features. It is an unsupervised algorithm used for dimensionality reduction. It extracts the maximum common variance from all features being evaluated and puts them into a common score.

In factor analysis the observed variables are represented as a linear combination of factors and error terms. Each factor explains a particular amount of variance in the observed features. The factors are generated from the observed variables and represent the common variance (variance due to correlation among the observed variables).

It comprises three steps:

1. The Bartlett's test : It is used to evaluate the factorability of the dataset, if the test is found statistically insignificant, factor analysis cannot be performed.
2. Choosing the number of factors: This decision is based on eigenvalues and is done using the scree plot. Since the data is standardised, the variance of a feature is 1. Therefore we select factors whose eigenvalues (variance) are greater than 1.
3. Performing factor analysis

6. Factor analysis results

On performing the factor analysis on the data we found that:

1. The Bartlett's test shows that the p-value is 0, making the result statistically significant therefore factors can be found
2. From Figure 6.1 we see that the eigenvalues drop below 1 from the 3rd factor. So, the optimal number of factors is 2.
3. The first two factors can represent 69% of the data as per the cumulative variance (Figure 6.3)
4. From Figure 6.2 we can see the loadings calculated by the two factors. Loadings indicate how much a factor explains a variable, it will range from -1 to 1.Values close to -1 or 1 indicate that the factor has an influence on these variables. Values close to 0 indicate that the factor has a lower influence on the variable.
5. In Factor 0, we can see that the features 'thighcircumference', 'shouldercircumference', 'neckcircumference', 'forearmcircumferenceflexed' and 'weightkg' have higher loadings than other variables. This means Factor 0 describes a person well based on their body circumferences and weight. (Figure 6.2)
6. Factor 1 on the other hand describes a person better on the basis of height, since 'kneeheightmidpatella' has a higher loading than others. (Figure 6.2)

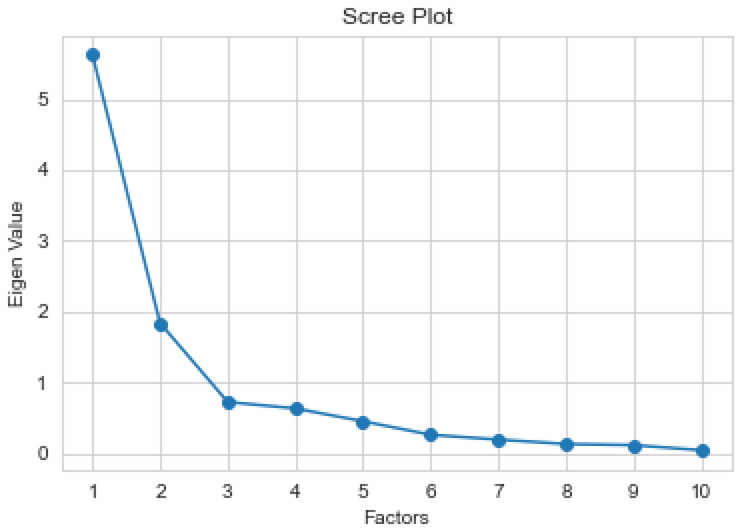
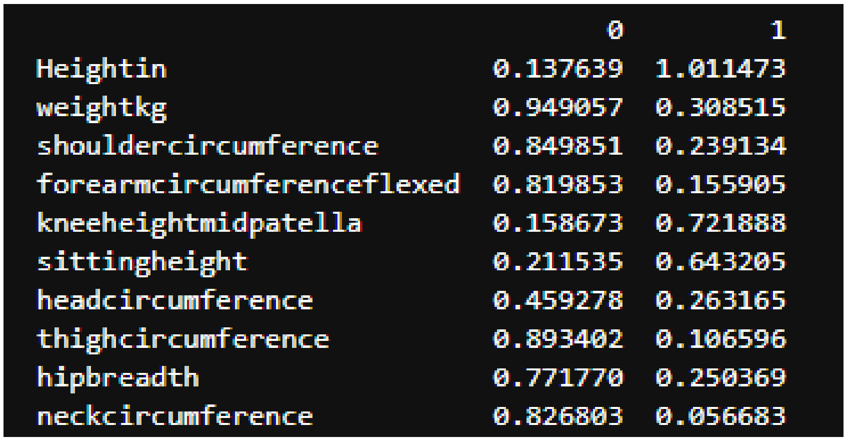
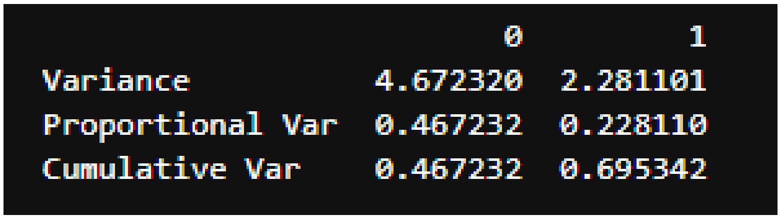


Figure 6.1 : Scree plot for factor analysis



```
                                  0         1
Heightin                   0.137639  1.011473
weightkg                   0.949057  0.308515
shouldercircumference      0.849851  0.239134
forearmcircumferenceflexed 0.819853  0.155905
kneeheightmidpatella       0.158673  0.721888
sittingheight              0.211535  0.643205
headcircumference          0.459278  0.263165
thighcircumference         0.893402  0.106596
hipbreadth                 0.771770  0.250369
neckcircumference          0.826803  0.056683
```

Figure 6.2 : Loadings for each factor for the selected features



```
                         0         1
Variance          4.672320  2.281101
Proportional Var  0.467232  0.228110
Cumulative Var    0.467232  0.695342
```

Figure 6.3: Variances for the factors generated in Factor analysis

7. Factor analysis vs PCA results

On comparing the two procedures and the results we get from them we can see that PCA does a better job at preserving the informational integrity of the data. The cumulative variance is greater in case of PCA even if we use only the first 2 principal components.

APPENDIX

1. Code for Data Pre-processing and EDA

```
# import the necessary libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
sns.set_style("whitegrid", {'axes.grid' : True})

# Read the data
data = pd.read_csv('./ANSUR_II_MALE_Public.csv', encoding='latin-1')
data.head()


# Data Pre-processing

# 1) Check for missing values
# the count of missing values
missing_entry_count = data.isnull().sum()
# total values
total_entry_count = data.isnull().count()
# the percentage of missing values
missing_entry_percentage = round(missing_entry_count/total_entry_count *100, 2)
# dataframe of columns with the number of missing values and its percentage
missing_data_df = pd.DataFrame({'missing entries count': missing_entry_count,
'percentage of entries missing': missing_entry_percentage})
# print any features which have missing values, with number of missing values and
percentage of missing values
print("Columns with missing entries are :\n")
missing_data_df[missing_data_df['missing entries count']!=0]

# 2) check if there are any duplicate records
data[data.duplicated() == True].shape[0]

# 3) feature selection
# drop column Ethnicity, since missing values percentage is extremely high, no use
of imputation
data1 = data.drop("Ethnicity", axis=1)

# dropping non-numerical columns
data2 = data1.select_dtypes('number')
```

```
# remove ID column
data3 = data2.drop("subjectid", axis=1)

# select common 10 features used for anthropometric data
selected_features = ["Heightin", "weightkg", "shouldercircumference",
"forearmcircumferenceflexed", "kneeheightmidpatella",
                     "sittingheight", "headcircumference", "thighcircumference",
"hipbreadth", "neckcircumference"]
ansur_male = data3[selected_features]

# separate out target variable values
target_var = data['Component']


# Exploratory Data Analysis (EDA)

# 1) Univariate Analysis
# histogram plot for selected feature data
print("\nHistogram plots for the selected features:\n")
ax = ansur_male.hist(bins = 25, figsize = (20,20))
plt.tight_layout()
plt.show()
plt.savefig('./FiguresCW1/Histogram plot.pdf',format='pdf')

# 2) Bivariate Analysis
# pair plot for the selected features, to see if they have correlations and how
they affect each other
print("Pair plot for the selected features:\n")
sns.pairplot(ansur_male)
plt.show()
plt.savefig('./FiguresCW1/Pair plot 1.pdf',format='pdf')

# 3) Multivariate Analysis
# heat map to visualise correlation matrix for the selected features
print("\Heatmap for correlation between selected features:\n")
sns.heatmap(ansur_male.corr(), vmax=1, vmin=-1, cmap='YlGnBu', annot=True)
plt.show()
plt.savefig('./FiguresCW1/Heat map.pdf',format='pdf')
```

2. Code for Principal Component Analysis

```
# PRINCIPAL COMPONENT ANALYSIS (PCA)
# import PCA and standard scaler methods from sklearn
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

# using standard scaler to standardise the data
scaler = StandardScaler()
ansur_std = scaler.fit_transform(ansur_male)

# create PCA instance, fit and transform the standardised data with pca
pca = PCA()
pc = pca.fit_transform(ansur_std)
# store PCA result in a dataframe
pc_df = pd.DataFrame(pc, columns=['PC 1', 'PC 2', 'PC 3', 'PC 4', 'PC 5', 'PC 6',
'PC 7', 'PC 8', 'PC 9', 'PC 10'])

# pairplot of the principal component dataframe
sns.pairplot(pc_df)
plt.show()
plt.savefig('./FiguresCW1/Pair plot 2.pdf',format='pdf')

# calculate the explained variance ratio per component (the eigenvalues)
var_explained = pca.explained_variance_ratio_
print(var_explained)

# calculate the cumulative sum of the explained variance ratio for each principal
component
cum_var_exp = pca.explained_variance_ratio_.cumsum()
print(cum_var_exp)

# plot the bar graph for cumulative explained variance for each principal component
PC_values = np.arange(pca.n_components_) + 1
plt.bar(PC_values, cum_var_exp, align='center')
plt.title('Cumulative explained variance plot for each PC')
plt.ylabel('Cumulative explained variance')
plt.xlabel('Principal Component')
plt.xticks(np.arange(1, 11))
plt.yticks(np.arange(0, 1.1, step=0.1))
plt.tight_layout()
plt.show()
plt.savefig('./FiguresCW1/PCA cumulative plot.pdf',format='pdf')

# create the scree plot for the explained variance ratio for each principal
component
plt.plot(PC_values, pca.explained_variance_ratio_, 'ro-', linewidth=2)
plt.title('Scree Plot')
plt.xlabel('Principal Component')
plt.ylabel('Proportion of Variance Explained (Eigen values)')
plt.xticks(np.arange(1, 11))
plt.yticks(np.arange(0, 0.7, step=0.1))
plt.show()
plt.savefig('./FiguresCW1/PCA Scree plot.pdf',format='pdf')

# calculate the loadings for each feature given by each principal component (PC)
loadings = pd.DataFrame(pca.components_.T, columns=['PC1', 'PC2', 'PC3', 'PC4',
'PC5', 'PC6', 'PC7', 'PC8', 'PC9', 'PC10'], index=ansur_male.columns)
print("Loadings for the features given by each PC:\n")
print(loadings)

# create the bi-plot for the 2 principal components for feature : Component
plt.figure()
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.title("Principal Component Analysis Bi-plot")
targets = target_var.unique().tolist()
colours = ['r', 'g', 'b']
for target, colour in zip(targets, colours):
    indicesToKeep = data['Component'] == target
    plt.scatter(pc_df.loc[indicesToKeep, 'PC 1']
               , pc_df.loc[indicesToKeep, 'PC 2'], c = colour, s = 5)

plt.legend(targets)
plt.tight_layout()
plt.show()
plt.savefig('./FiguresCW1/PCA Bi-plot.pdf',format='pdf')
```

3. Code for Factor Analysis

```
# FACTOR ANALYSIS (FA)

# import the factor analyser libraries
from factor_analyzer import FactorAnalyzer
from factor_analyzer.factor_analyzer import calculate_bartlett_sphericity

# Step 1 : perform the Bartlett's test to check if factor analysis can be performed
for the data
chi_square_value, p_value = calculate_bartlett_sphericity(ansur_std)
chi_square_value, p_value
print("Bartlett's Test result? [True/False] : ", (p_value==0))
```

```
# Step 2 : select number of features using scree plot

# initialise the factor analyzer and fit the standardised data to it
fa = FactorAnalyzer(rotation = None, impute = "drop", n_factors =
ansur_std.shape[1])
fa.fit(ansur_std)

# get the eigenvalues from the factor analysis
ev,_ = fa.get_eigenvalues()

# create scree plot for eigenvalues of each factor
plt.scatter(range(1, ansur_std.shape[1]+1), ev)
plt.plot(range(1, ansur_std.shape[1]+1), ev)
plt.title('Scree Plot')
plt.xlabel('Factors')
plt.ylabel('Eigen Value')
plt.xticks(np.arange(1, 11))
plt.savefig('./FiguresCW1/FA Scree plot.pdf',format='pdf')
plt.show()

# Step 3: perform factor analysis on the data using the number of features
calculated in previous step

# initialise the factor analyzer and fit the standardised data to it
fa = FactorAnalyzer(n_factors=2, rotation='varimax')
fa.fit(ansur_std)

# print the loadings for each feature for each factor
print(pd.DataFrame(fa.loadings_, index=ansur_male.columns))

# print the variances for each factor
print(pd.DataFrame(fa.get_factor_variance(),index=['Variance','Proportional
Var','Cumulative Var']))
```

REFERENCES

1. Walker, B. (2019). PCA Is Not Feature Selection. *Medium*. [online]. Available from: https://towardsdatascience.com/pca-is-not-feature-selection-3344fb764ae6# [Accessed February 20, 2022].
2. Anon. Introduction to anthropometry. *Conflict.lshtm.ac.uk*. [online]. Available from: http://conflict.lshtm.ac.uk/page_120.htm [Accessed February 20, 2022].
3. Anon. (2020). Understanding the Curse of Dimensionality. *Great Learning*. [online]. Available from: https://www.mygreatlearning.com/blog/understanding-curse-of-dimensionality/ [Accessed February 20, 2022].
4. Sharma, A. (2020). Principal Component Analysis in Python. *Datacamp*. [online]. Available from: https://www.datacamp.com/community/tutorials/principal-component-analysis-in-python [Accessed February 21, 2022].