NTNU
Kunnskap for en bedre verden

Department of Computer Science

IT3212 - Data-Driven Software

Assignment 1

Authors:
Birk Strand Bjørnaa
Carl Edward Storlien
Christian Stensøe
Åsmund Løvoll

25.11.2024

Table of Contents

List of Figures

List of Tables

1 Changelog

For the revised version of Assignment 1, we have addressed the feedback given. Below is a list of improvements.

- Explained why feature scaling is necessary and how it impacts the model. - Justified the data splitting. - Explained the importance of splitting the data and how it prevents overfitting. - Properly explained the usage of PCA. - Discussed the PCA results and added visuals. - Improved explanation of handling missing values. - Improved the section on encoding and feature scaling. Specified the categorical columns. - Fixed citing issues. - Improved formatting of sections.

2 Data Exploration

For the data exploration tasks, we chose to use the dataset `smoking_drinking_dataset.csv`.

2.1 a)

2.1.1 First Few Rows

The first five rows are of the dataset[1] shown in table 1.

Table 1: First five rows

Feature	**0**	**1**	**2**	**3**	**4**
sex	Male	Male	Male	Male	Male
age	35	30	40	50	50
height	170	180	165	175	165
weight	75	80	75	80	60
waistline	90.0	89.0	91.0	91.0	80.0
sight_left	1.0	0.9	1.2	1.5	1.0
sight_right	1.0	1.2	1.5	1.2	1.2
hear_left	1.0	1.0	1.0	1.0	1.0
hear_right	1.0	1.0	1.0	1.0	1.0
SBP	120.0	130.0	120.0	145.0	138.0
DBP	80.0	82.0	70.0	87.0	82.0
BLDS	99.0	106.0	98.0	95.0	101.0
tot_chole	193.0	228.0	136.0	201.0	199.0
HDL_chole	48.0	55.0	41.0	76.0	61.0
LDL_chole	126.0	148.0	74.0	104.0	117.0
triglyceride	92.0	121.0	104.0	106.0	104.0
hemoglobin	17.1	15.8	15.8	17.6	13.8
urine_protein	1.0	1.0	1.0	1.0	1.0
serum_creatinine	1.0	0.9	0.9	1.1	0.8
SGOT_AST	21.0	20.0	47.0	29.0	19.0
SGOT_ALT	35.0	36.0	32.0	34.0	12.0
gamma_GTP	40.0	27.0	68.0	18.0	25.0
SMK_stat_type_cd	1.0	3.0	1.0	1.0	1.0
DRK_YN	Y	N	N	N	N

[1] Soo.Y 2024.

2.1.2 Summary Statistics

There are `991346` rows in the dataset. Summary statistics of the features are shown in table 2. The categorical features, which are stated in subsection 2.2.3, are removed from the summary statistics.

Table 2: Summary statistics

Variable	Mean	Std	Min	25%	75%	Max
age	47.61	14.18	20.00	35.00	60.00	85.00
height	162.24	9.28	130.00	155.00	170.00	190.00
weight	63.28	12.51	25.00	55.00	70.00	140.00
waistline	81.23	11.85	8.00	74.10	87.80	999.00
sight_left	0.98	0.61	0.10	0.70	1.20	9.90
sight_right	0.98	0.60	0.10	0.70	1.20	9.90
SBP	122.43	14.54	67.00	112.00	131.00	273.00
DBP	76.05	9.89	32.00	70.00	82.00	185.00
BLDS	100.42	24.18	25.00	88.00	105.00	852.00
tot_chole	195.56	38.66	30.00	169.00	219.00	2344.00
HDL_chole	56.94	17.24	1.00	46.00	66.00	8110.00
LDL_chole	113.04	35.84	1.00	89.00	135.00	5119.00
triglyceride	132.14	102.20	1.00	73.00	159.00	9490.00
hemoglobin	14.23	1.58	1.00	13.20	15.40	25.00
urine_protein	1.09	0.44	1.00	1.00	1.00	6.00
serum_creatinine	0.86	0.48	0.10	0.70	1.00	98.00
SGOT_AST	25.99	23.49	1.00	19.00	28.00	9999.00
SGOT_ALT	25.76	26.31	1.00	15.00	29.00	7210.00
gamma_GTP	37.14	50.42	1.00	16.00	39.00	999.00

2.1.3 Data Types

Table 3 shows the data types of the features as recognized by the `Pandas` package.

Feature	**Data Type**
sex	object
age	int64
height	int64
weight	int64
waistline	float64
sight_left	float64
sight_right	float64
hear_left	float64
hear_right	float64
SBP	float64
DBP	float64
BLDS	float64
tot_chole	float64
HDL_chole	float64
LDL_chole	float64
triglyceride	float64
hemoglobin	float64
urine_protein	float64
serum_creatinine	float64
SGOT_AST	float64
SGOT_ALT	float64
gamma_GTP	float64
SMK_stat_type_cd	float64
DRK_YN	object

Table 3: Data types of features

2.2 b)

2.2.1 Missing Values

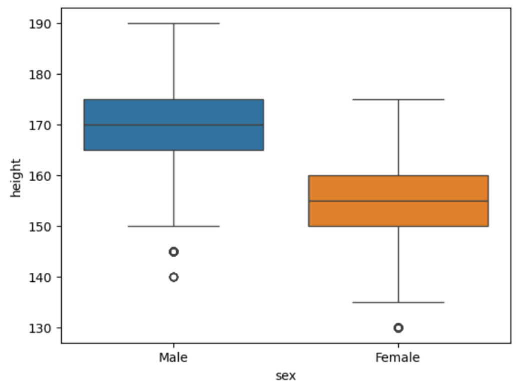
The `smoking_drinking_dataset.csv` dataset has <u>no</u> missing values. The Pandas method `isna()` was used to inspect this.

2.2.2 Outliers

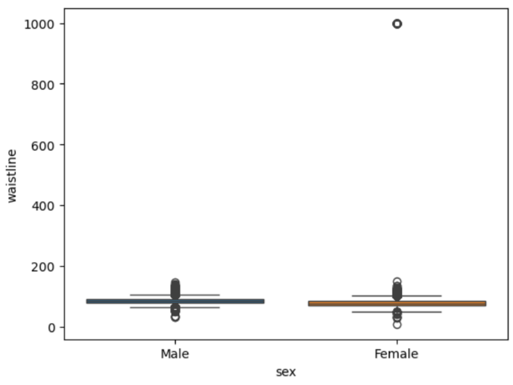
The univariate outliers in the dataset `smoking_drinking_dataset.csv` can be identified by using a box plot for the features. For this task, the `seaborn` library in Python was used to generate the box plots.

Univariate outliers have been identified in almost all of the features in this dataset:
`height`, `weight`, `waistline`, `sight_left`, `sight_right`, `SBP`, `DBP`, `BLDS`, `tot_chole`, `HDL_chole`, `LDL_chole`, `triglyceride`, `hemoglobin`, `serum_creatinine`, `SGOT_AST`, `SGOT_ALT` and `gamma_GTP`.

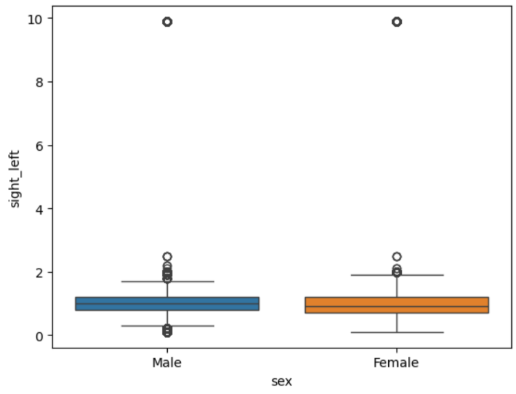
Figure 1 shows some of the box plots indicating univariate outliers.



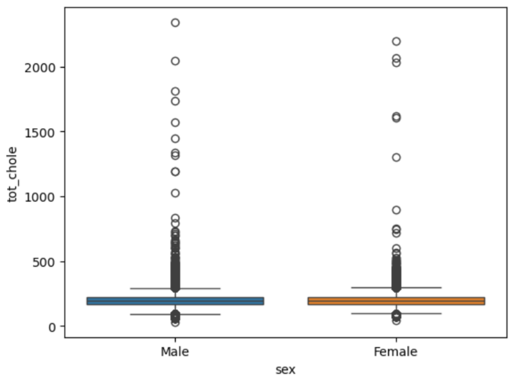
(a) Box plot of `height`



(b) Box plot of `waistline`



(c) Box plot of `sight_left`



(d) Box plot of `tot_chole`

Figure 1: Examples of univariate outliers identified with box plots.

2.2.3 Unique Values

The unique values of the categorical features are shown in table 4. This was confirmed using `Pandas'` built-in function `unique()`, and information on what the categorical values represented was found on Soo.Y 2024.

Categorical Feature	**Original Values**
Sex	Female, Male
hear_left	1.0 (normal), 2.0 (abnormal)
hear_right	1.0 (normal), 2.0 (abnormal)
urine_protein	1 (-), 2 (+/-), 3 (+1), 4 (+2), 5 (+3), 6 (+4)
SMK_stat_type_cd	1.0 (never), 2.0 (used to smoke but quit), 3.0 (still smoke)
DRK_YN	N, Y

Table 4: Unique values of categorical features

3 Data Cleaning

3.1 a), b) and c)

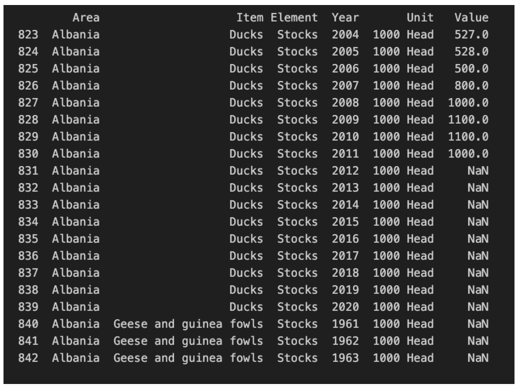
3.1.1 Handling Missing Values

There are generally two ways of dealing with missing data in a dataset: If the dataset is large, the missing values can be ignored. Alternatively, one can fill in the missing values.

In the dataset `food-bank`, and the csv-file `live1.csv`, we chose to remove all the null values. This is because we found that only 1.88% of the elements contained null values, and we we considered this percentage small enough that the most logical solution was deletion. The deletion was performed by creating a dataframe with `pandas`-function `read`, and then using the `numpy` library's `dropna`-function.

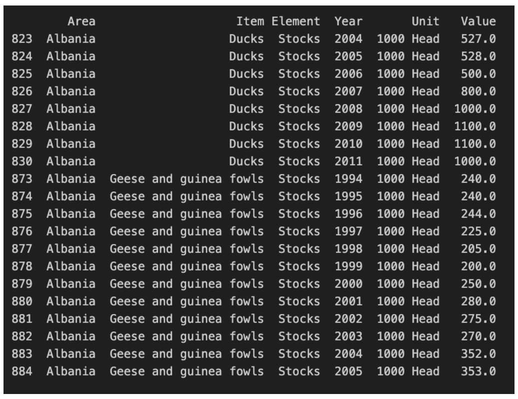
Upon inspection of the dataset, we learned that the missing values were contained to certain areas of the dataset. An example is Ducks in Albania. As the only data available was from 1994 to 2011, we decided that this data range was insufficient to assess trends in duck stock for the missing years (1961-1993 and 2011-2020). Furthermore, the data were very similar when looking past the stock and year, which means that the consequence of removing the rows with missing data did not lead to losing other important data. If the dataset had been more complex, such that deletion of rows had lead to loss of other relevant data, we would be more inclined to use another method for handling missing values.

We initially considered handling the missing values with *interpolation*, which is used to estimate the values of unknown data points between existing ones. As we learned upon inspection that the missing values were contained to certain areas, and were several in a row, it was deemed unfit for this dataset.
Numerical imputation was also considered as an approach, but as we wouldn't be able to take changing trends over time into account, we deemed this approach unfit as well. There was no clear relationship between the datapoints, and thus we were not confident that either mean or median interpolation, or other regression models would yield correct results.



```
        Area                  Item Element  Year       Unit   Value
823  Albania                 Ducks  Stocks  2004  1000 Head   527.0
824  Albania                 Ducks  Stocks  2005  1000 Head   528.0
825  Albania                 Ducks  Stocks  2006  1000 Head   500.0
826  Albania                 Ducks  Stocks  2007  1000 Head   800.0
827  Albania                 Ducks  Stocks  2008  1000 Head  1000.0
828  Albania                 Ducks  Stocks  2009  1000 Head  1100.0
829  Albania                 Ducks  Stocks  2010  1000 Head  1100.0
830  Albania                 Ducks  Stocks  2011  1000 Head  1000.0
831  Albania                 Ducks  Stocks  2012  1000 Head     NaN
832  Albania                 Ducks  Stocks  2013  1000 Head     NaN
833  Albania                 Ducks  Stocks  2014  1000 Head     NaN
834  Albania                 Ducks  Stocks  2015  1000 Head     NaN
835  Albania                 Ducks  Stocks  2016  1000 Head     NaN
836  Albania                 Ducks  Stocks  2017  1000 Head     NaN
837  Albania                 Ducks  Stocks  2018  1000 Head     NaN
838  Albania                 Ducks  Stocks  2019  1000 Head     NaN
839  Albania                 Ducks  Stocks  2020  1000 Head     NaN
840  Albania  Geese and guinea fowls  Stocks  1961  1000 Head     NaN
841  Albania  Geese and guinea fowls  Stocks  1962  1000 Head     NaN
842  Albania  Geese and guinea fowls  Stocks  1963  1000 Head     NaN
```

(a) Part of dataset before deletion.



```
        Area                  Item Element  Year       Unit   Value
823  Albania                 Ducks  Stocks  2004  1000 Head   527.0
824  Albania                 Ducks  Stocks  2005  1000 Head   528.0
825  Albania                 Ducks  Stocks  2006  1000 Head   500.0
826  Albania                 Ducks  Stocks  2007  1000 Head   800.0
827  Albania                 Ducks  Stocks  2008  1000 Head  1000.0
828  Albania                 Ducks  Stocks  2009  1000 Head  1100.0
829  Albania                 Ducks  Stocks  2010  1000 Head  1100.0
830  Albania                 Ducks  Stocks  2011  1000 Head  1000.0
873  Albania  Geese and guinea fowls  Stocks  1994  1000 Head   240.0
874  Albania  Geese and guinea fowls  Stocks  1995  1000 Head   240.0
875  Albania  Geese and guinea fowls  Stocks  1996  1000 Head   244.0
876  Albania  Geese and guinea fowls  Stocks  1997  1000 Head   225.0
877  Albania  Geese and guinea fowls  Stocks  1998  1000 Head   205.0
878  Albania  Geese and guinea fowls  Stocks  1999  1000 Head   200.0
879  Albania  Geese and guinea fowls  Stocks  2000  1000 Head   250.0
880  Albania  Geese and guinea fowls  Stocks  2001  1000 Head   280.0
881  Albania  Geese and guinea fowls  Stocks  2002  1000 Head   275.0
882  Albania  Geese and guinea fowls  Stocks  2003  1000 Head   270.0
883  Albania  Geese and guinea fowls  Stocks  2004  1000 Head   352.0
884  Albania  Geese and guinea fowls  Stocks  2005  1000 Head   353.0
```

(b) Part of dataset after deletion.

Figure 2: Comparison of dataset before and after deletion.

4 Handling Outliers

4.1 a)

4.1.1 Detecting Outliers

To detect and handle outliers in the dataset `smoking_drinking_dataset.csv`, the IQR (Interquartile Range) method is used in this task. The IQR method marks data points outside the desired range of inliers, or "normal" data points. Usually, a data point is marked as outlier if it lies more than 1.5 times the IQR below the first quartile, or more than 1.5 times the IQR above the third quartile. IQR is the range of values between the first and third quartile.

When first applying the IQR method, we noticed that males and females have different range of values in some of the features. Therefore, we modified the IQR method to calculate Q1 and Q3 for males and females separately.

We also tried the `IsolationForest` class from `sklearn.ensemble` on the dataset. We set the hyperparameter `contamination` to 0.05 to identify **5%** of the dataset as anomalies. The machine learning model identifies both univariate and multivariate outliers. After using the methods `decision_function()` and `predict()`, we confirmed the model had identified the correct proportion of the dataset. However, setting the ideal hyperparameters becomes the hardest part. We chose not to explore it any further.

4.2 b)

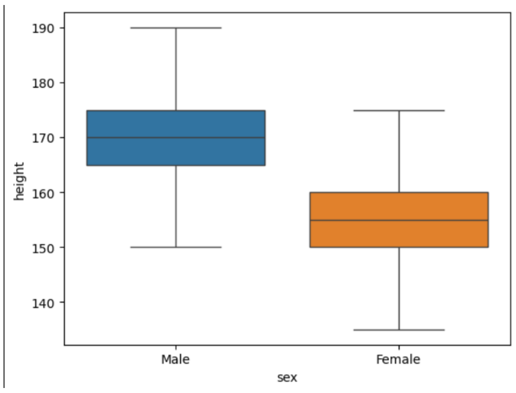
4.2.1 Handling the Outliers

The IQR method marked **36%** of the entire dataset as outliers. Because this portion constitutes too large a part of the dataset, the outliers should not be removed. It would reduce the size of the dataset too much which can impact how well our final model is fit.

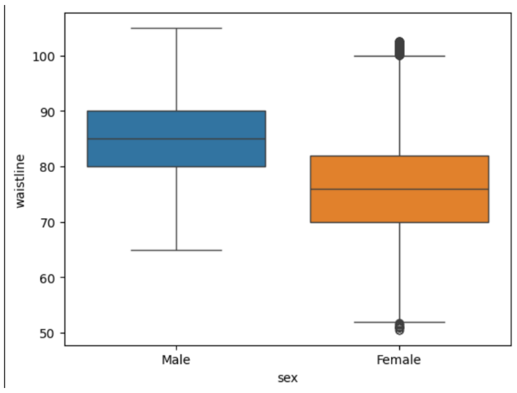
`36%` is a very high ratio of outliers, which might suggest that the dataset has very high variability and that we should try to increase the threshold to only capture the extreme outliers. We increased the threshold to `2 * IQR` and got a ratio of **21%**. We stuck with the standard `1.5 * IQR`.

We considered two methods of handling the outliers that didn't involve removal. One method is capping the outlier values to the lower or upper bound of the IQR method (setting their value as close to the inlier-outlier border as possible). Another method is setting the outlier values to the mean values of the feature.

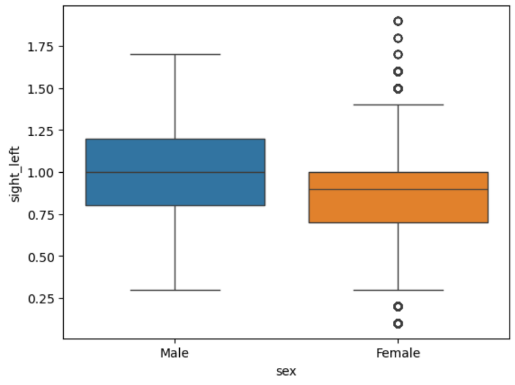
Since we do not know enough about this domain (body signals), we chose to set the outlier values to the mean of the feature, with respect to the sex. After exploring the dataset, it looked like a large portion of the outliers were measurement errors instead of naturally occurring extremes. It therefore makes sense to set the values to the mean instead of capping since it's not necessarily an actual extreme value.



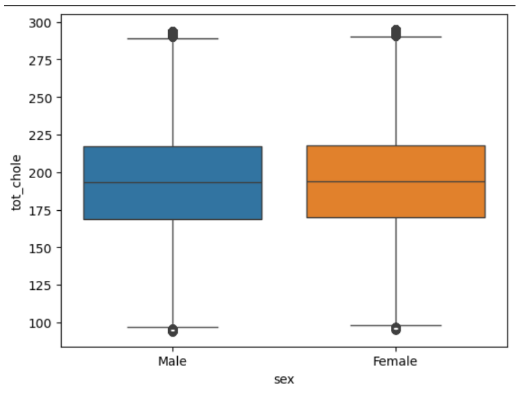
(a) Box plot of `height` after handling outliers



(b) Box plot of `waistline` after handling outliers



(c) Box plot of `sight_left` after handling outliers



(d) Box plot of `tot_chole` after handling outliers

Figure 3: Examples of features after handling outliers

Figure 3 shows the same box plots as in figure 1 after handling the outliers. Table 5 shows the summary statistics after handling the outliers.

From table 5, one can quickly observe that all the extreme max values are gone, compared to the original summary in table 2.

Table 5: Summary statistics after handling outliers

Variable	**Mean**	**Std**	**Min**	**25%**	**75%**	**Max**
age	47.61	14.18	20.00	35.00	60.00	85.00
height	162.25	9.28	135.00	155.00	170.00	190.00
weight	62.60	11.25	35.00	55.00	70.00	90.00
waistline	80.96	9.15	50.50	74.50	87.00	105.00
sight_left	0.95	0.32	0.10	0.70	1.20	1.90
sight_right	0.94	0.28	0.30	0.80	1.20	1.70
hear_left	1.03	0.17	1.00	1.00	1.00	2.00
hear_right	1.03	0.17	1.00	1.00	1.00	2.00
SBP	121.87	13.34	80.00	112.00	130.00	160.00
DBP	75.71	9.27	49.00	70.00	81.00	105.00
BLDS	96.23	11.46	63.00	89.00	103.11	135.00
tot_chole	194.33	35.73	94.00	169.00	218.00	295.00
HDL_chole	56.16	13.63	18.00	46.00	65.00	101.00
LDL_chole	112.05	32.59	20.00	89.00	134.00	204.00
triglyceride	117.15	57.98	1.00	73.00	152.00	332.00
hemoglobin	14.31	1.43	10.60	13.20	15.40	18.10
urine_protein	1.09	0.44	1.00	1.00	1.00	6.00
serum_creatinine	0.85	0.19	0.30	0.70	1.00	1.50
SGOT_AST	23.57	6.56	4.00	19.00	28.00	47.00
SGOT_ALT	22.11	10.22	1.00	15.00	28.00	60.00
gamma_GTP	28.30	18.09	1.00	16.00	35.00	103.00
SMK_stat_type_cd	1.61	0.82	1.00	1.00	2.00	3.00

5 Data Transformation

5.1 a)

5.1.1 Encoding Categorical Data

In task 4 we have used the `live1.csv` dataset. Label encoding is commonly used when dealing with categorical variables in datasets that need to be processed by machine learning algorithms. Most machine learning models work with numerical input data, so categorical text data must be converted into numbers. Label encoding transforms each unique category in a column into an integer value, enabling models to process the information.

For this dataset we did not consider one-hot encoding as it is less efficient for this particular dataset. One-hot encoding creates additional binary columns for each unique category, which can increase the dimensionality of the dataset, especially when the categorical columns like Area or Item contain many unique values. This leads to columns and rows where most values are zero, consuming more memory. We therefore concluded that label encoding was the correct way to transform the data.

For this particular dataset we removed the feature named **Element** which had the same value in the entire dataset, a so-called dimensionality reduction. **Unit** was also removed after we scaled the **Values** feature with a factor of 1000 due to the **Unit** feature implying what factor the **Value** was multiplied with. This made the **Values** feature more scalable. As shown in Tables 6 and 7, we used the LabelEncoder() method from `sklearn.preprocessing` library to label the features. The method labels alphabetically, as the tables show.

Encoded Value	Area
0	Afghanistan
1	Africa
2	Albania
3	Algeria
4	Americas
...	...
241	Zambia
242	Zimbabwe

(a) Mapping of Area encoded values

Encoded Value	Item
0	Asses
1	Beehives
2	Buffaloes
3	Camels
4	Cattle
...	...
13	Sheep

(b) Mapping of Item encoded values

Area	Item	Year	Value
0.0	0.0	0.000000	0.000039
0.0	0.0	0.016949	0.000026
0.0	0.0	0.033898	0.0000305
0.0	0.0	0.050847	0.000035
0.0	0.0	0.067797	0.000039

(c) First five rows of scaled data

Figure 4: Mapping of Area and Item encoded values and scaled data

5.2 b)

5.2.1 Feature Scaling

For our task, we used the normalizing technique which includes the use of MinMaxScaler(). We used the `.fittransform` method for scaling the features in the dataset.

Feature scaling is used in machine learning because many algorithms are sensitive to the range of features. Models that rely on distance metrics can be skewed by features with larger values, causing unequal contributions, for instance, with dimensionality reduction with PCA. Also, gradient-based methods converge faster and more reliably when features are on similar scales. Unscaled data can lead to inefficient learning and poor performance. Scaling also helps to fairly penalize features and improve model generalization. Summarized, feature scaling standardizes features, leading to faster, more accurate models.

6 Data Splitting

6.1 a)

6.1.1 Splitting the Data

The method we used to split the data was firstly to randomize the data such that the data split would be fair, and unbiased. Secondly, we used the length of the dataset and a factor of 0.8 so split the set in a 80-20 train-test split. This choice of ratio balances having sufficient data to train the model effectively while retaining a large enough test set to provide a reliable assessment of model performance. By modifying the splitting factor, other configurations, such as a 70-30 split, can easily be implemented if a different balance between training and evaluation data is desired.

6.2 b)

6.2.1 The Importance of Splitting

Splitting the data into training, validation, and test sets is important for building reliable machine-learning models. The model is trained on one subset of data and evaluated on another subset. This helps prevent overfitting where the model learns patterns specific to the training data but fails to generalize to new data. The test set is a final evaluation to check how well the model performs on completely unseen data, simulating real-world use. This separation ensures that performance metrics provide a more accurate reflection of how the model will behave in practice, preventing misleadingly high results due to memorization of the training data.

Randomizing the data ensures that each subset represents the distribution of the entire dataset, minimizing the risk of bias that could skew the results. Additionally, splitting the data helps avoid overfitting, where the model might memorize specific patterns in the training data, resulting in poor generalization to new examples. By ensuring the model is validated on unseen data, the performance metrics obtained more reflect its expected behavior in deployment.

7 Bonus

7.1 Applying PCA to the Dataset

Principal Component Analysis (PCA) is a dimensionality reduction technique that transforms a high-dimensional dataset into a lower-dimensional space by identifying the principal components that capture the most variance. [2].

We used `PCA()` from the `sklearn.decomposition` library to apply PCA to the `smoking_drinking_dataset.csv` dataset, ensuring the features were scaled using `StandardScaler` to normalize values to a mean of 0 and standard deviation of 1[3].

Figure 5 shows how much information is retained with n number of components.

To calculate these values, we set `n_components = 23`. We then used the `explained_variance_ratio_` attribute on the `pca` object to store an array of the variance ratios explained by each principal component. Next, we applied the `.cumsum()` method to store the cumulative variance explained up to each component, saving it to the variable `cumulative_explained_variance`.

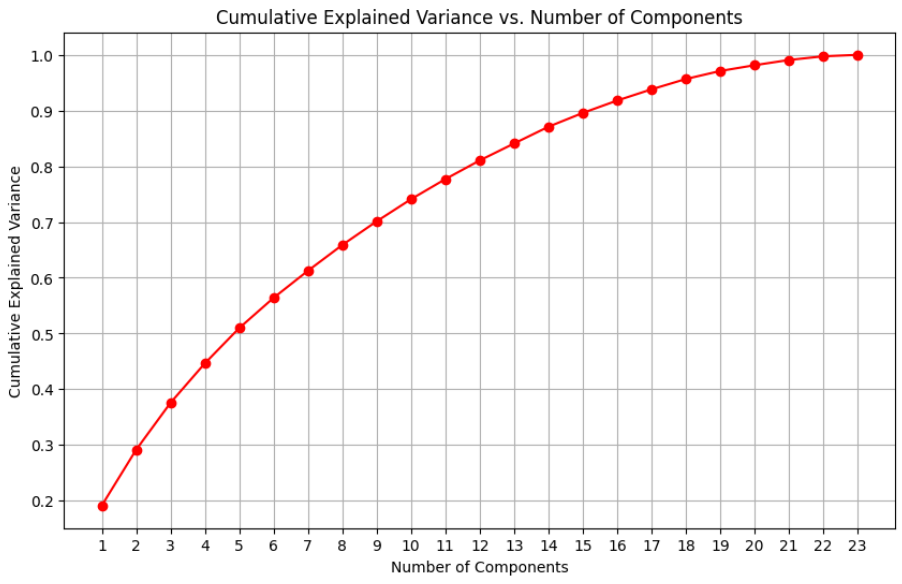
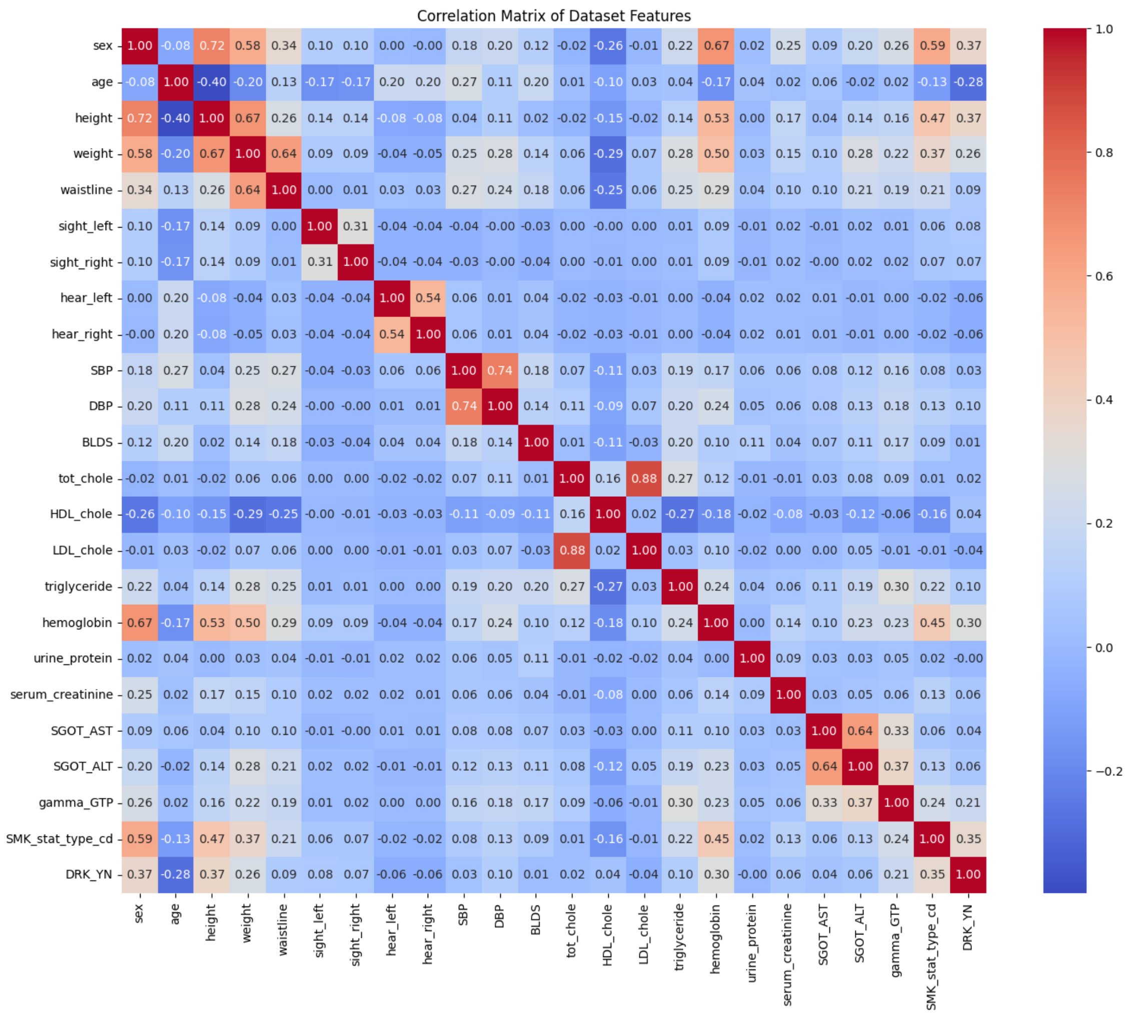


Figure 5: Variance explained by principal components

We observe that as the number of principal components increases, each successive component contributes less to the cumulative explained variance. For instance, the first component accounts for 19.14% of the variance, the tenth component adds 3.96%, and the final component contributes only 0.26%. We can also notice that we need more than 11 principal components for PCA to explain more than 80% of the variance.

This demonstrates how effectively PCA summarizes information within the first few components. Observing the features in our dataset, we notice that we have measurements for **waistline**, **weight**, and **height**. Before performing any calculations, we might guess that **waistline** and **weight**, as well as **height** and **weight**, could have a high correlation, meaning they tend to vary together (when one increases, so does the other). Looking at Figure 6, we can confirm this with correlation values of 0.64 and 0.67, respectively, indicating a relatively high correlation. Here we used the `.corr()` method on our dataframe to calculate the correlation matrix.

[2]Warya, Aish (n.d.). Principal Component Analysis(PCA)
[3]Warya, Aish (n.d.). Principal Component Analysis(PCA)



	sex	age	height	weight	waistline	sight_left	sight_right	hear_left	hear_right	SBP	DBP	BLDS	tot_chole	HDL_chole	LDL_chole	triglyceride	hemoglobin	urine_protein	serum_creatinine	SGOT_AST	SGOT_ALT	gamma_GTP	SMK_stat_type_cd	DRK_YN
sex	1.00	-0.08	0.72	0.58	0.34	0.10	0.10	0.00	-0.00	0.18	0.20	0.12	-0.02	-0.26	-0.01	0.22	0.67	0.02	0.25	0.09	0.20	0.26	0.59	0.37
age	-0.08	1.00	-0.40	-0.20	0.13	-0.17	-0.17	0.20	0.20	0.27	0.11	0.20	0.01	-0.10	0.03	0.04	-0.17	0.04	0.02	0.06	-0.02	0.02	-0.13	-0.28
height	0.72	-0.40	1.00	0.67	0.26	0.14	0.14	-0.08	-0.08	0.04	0.11	0.02	-0.02	-0.15	-0.02	0.14	0.53	0.00	0.17	0.04	0.14	0.16	0.47	0.37
weight	0.58	-0.20	0.67	1.00	0.64	0.09	0.09	-0.04	-0.05	0.25	0.28	0.14	0.06	-0.29	0.07	0.28	0.50	0.03	0.15	0.10	0.28	0.22	0.37	0.26
waistline	0.34	0.13	0.26	0.64	1.00	0.00	0.01	0.03	0.03	0.27	0.24	0.18	0.06	-0.25	0.06	0.25	0.29	0.04	0.10	0.10	0.21	0.19	0.21	0.09
sight_left	0.10	-0.17	0.14	0.09	0.00	1.00	0.31	-0.04	-0.04	-0.04	-0.00	-0.03	0.00	-0.00	0.00	0.01	0.09	-0.01	0.02	-0.01	0.02	0.01	0.06	0.08
sight_right	0.10	-0.17	0.14	0.09	0.01	0.31	1.00	-0.04	-0.04	-0.03	-0.00	-0.04	0.00	-0.01	0.00	0.01	0.09	-0.01	0.02	-0.00	0.02	0.02	0.07	0.07
hear_left	0.00	0.20	-0.08	-0.04	0.03	-0.04	-0.04	1.00	0.54	0.06	0.01	0.04	-0.02	-0.03	-0.01	0.00	-0.04	0.02	0.02	0.01	-0.01	0.00	-0.02	-0.06
hear_right	-0.00	0.20	-0.08	-0.05	0.03	-0.04	-0.04	0.54	1.00	0.06	0.01	0.04	-0.02	-0.03	-0.01	0.00	-0.04	0.02	0.01	0.01	-0.01	0.00	-0.02	-0.06
SBP	0.18	0.27	0.04	0.25	0.27	-0.04	-0.03	0.06	0.06	1.00	0.74	0.18	0.07	-0.11	0.03	0.19	0.17	0.06	0.06	0.08	0.12	0.16	0.08	0.03
DBP	0.20	0.11	0.11	0.28	0.24	-0.00	-0.00	0.01	0.01	0.74	1.00	0.14	0.11	-0.09	0.07	0.20	0.24	0.05	0.06	0.08	0.13	0.18	0.13	0.10
BLDS	0.12	0.20	0.02	0.14	0.18	-0.03	-0.04	0.04	0.04	0.18	0.14	1.00	0.01	-0.11	-0.03	0.20	0.10	0.11	0.04	0.07	0.11	0.17	0.09	0.01
tot_chole	-0.02	0.01	-0.02	0.06	0.06	0.00	0.00	-0.02	-0.02	0.07	0.11	0.01	1.00	0.16	0.88	0.27	0.12	-0.01	-0.01	0.03	0.08	0.09	0.01	0.02
HDL_chole	-0.26	-0.10	-0.15	-0.29	-0.25	-0.00	-0.01	-0.03	-0.03	-0.11	-0.09	-0.11	0.16	1.00	0.02	-0.27	-0.18	-0.02	-0.08	-0.03	-0.12	-0.06	-0.16	0.04
LDL_chole	-0.01	0.03	-0.02	0.07	0.06	0.00	0.00	-0.01	-0.01	0.03	0.07	-0.03	0.88	0.02	1.00	0.03	0.10	-0.02	0.00	0.00	0.05	-0.01	-0.01	-0.04
triglyceride	0.22	0.04	0.14	0.28	0.25	0.01	0.01	0.00	0.00	0.19	0.20	0.20	0.27	-0.27	0.03	1.00	0.24	0.04	0.06	0.11	0.19	0.30	0.22	0.10
hemoglobin	0.67	-0.17	0.53	0.50	0.29	0.09	0.09	-0.04	-0.04	0.17	0.24	0.10	0.12	-0.18	0.10	0.24	1.00	0.00	0.14	0.10	0.23	0.23	0.45	0.30
urine_protein	0.02	0.04	0.00	0.03	0.04	-0.01	-0.01	0.02	0.02	0.06	0.05	0.11	-0.01	-0.02	-0.02	0.04	0.00	1.00	0.09	0.03	0.03	0.05	0.02	-0.00
serum_creatinine	0.25	0.02	0.17	0.15	0.10	0.02	0.02	0.02	0.01	0.06	0.06	0.04	-0.01	-0.08	0.00	0.06	0.14	0.09	1.00	0.03	0.05	0.06	0.13	0.06
SGOT_AST	0.09	0.06	0.04	0.10	0.10	-0.01	-0.00	0.01	0.01	0.08	0.08	0.07	0.03	-0.03	0.00	0.11	0.10	0.03	0.03	1.00	0.64	0.33	0.06	0.04
SGOT_ALT	0.20	-0.02	0.14	0.28	0.21	0.02	0.02	-0.01	-0.01	0.12	0.13	0.11	0.08	-0.12	0.05	0.19	0.23	0.03	0.05	0.64	1.00	0.37	0.13	0.06
gamma_GTP	0.26	0.02	0.16	0.22	0.19	0.01	0.02	0.00	0.00	0.16	0.18	0.17	0.09	-0.06	-0.01	0.30	0.23	0.05	0.06	0.33	0.37	1.00	0.24	0.21
SMK_stat_type_cd	0.59	-0.13	0.47	0.37	0.21	0.06	0.07	-0.02	-0.02	0.08	0.13	0.09	0.01	-0.16	-0.01	0.22	0.45	0.02	0.13	0.06	0.13	0.24	1.00	0.35
DRK_YN	0.37	-0.28	0.37	0.26	0.09	0.08	0.07	-0.06	-0.06	0.03	0.10	0.01	0.02	0.04	-0.04	0.10	0.30	-0.00	0.06	0.04	0.06	0.21	0.35	1.00

Figure 6: Correlation matrix for the features in the dataset.

References

Codest, Session (2024). *Anomaly detection with isolation forest in scikit-learn.* URL: https://medium.com/mlthinkbox/anomaly-detection-with-isolation-forest-in-scikit-learn-99417dcc3971 (visited on 2nd Oct. 2024).

Dey, Akash (2024). *How to handle outliers.* URL: https://www.kaggle.com/code/aimack/how-to-handle-outliers (visited on 2nd Oct. 2024).

Soo.Y (2024). *Smoking and Drinking Dataset with body signal.* URL: https://www.kaggle.com/datasets/sooyoungher/smoking-drinking-dataset (visited on 2nd Oct. 2024).

Warya, Aish (2024). *Principal Component Analysis (PCA).* URL: https://www.geeksforgeeks.org/principal-component-analysis-pca/ (visited on 7th Oct. 2024).