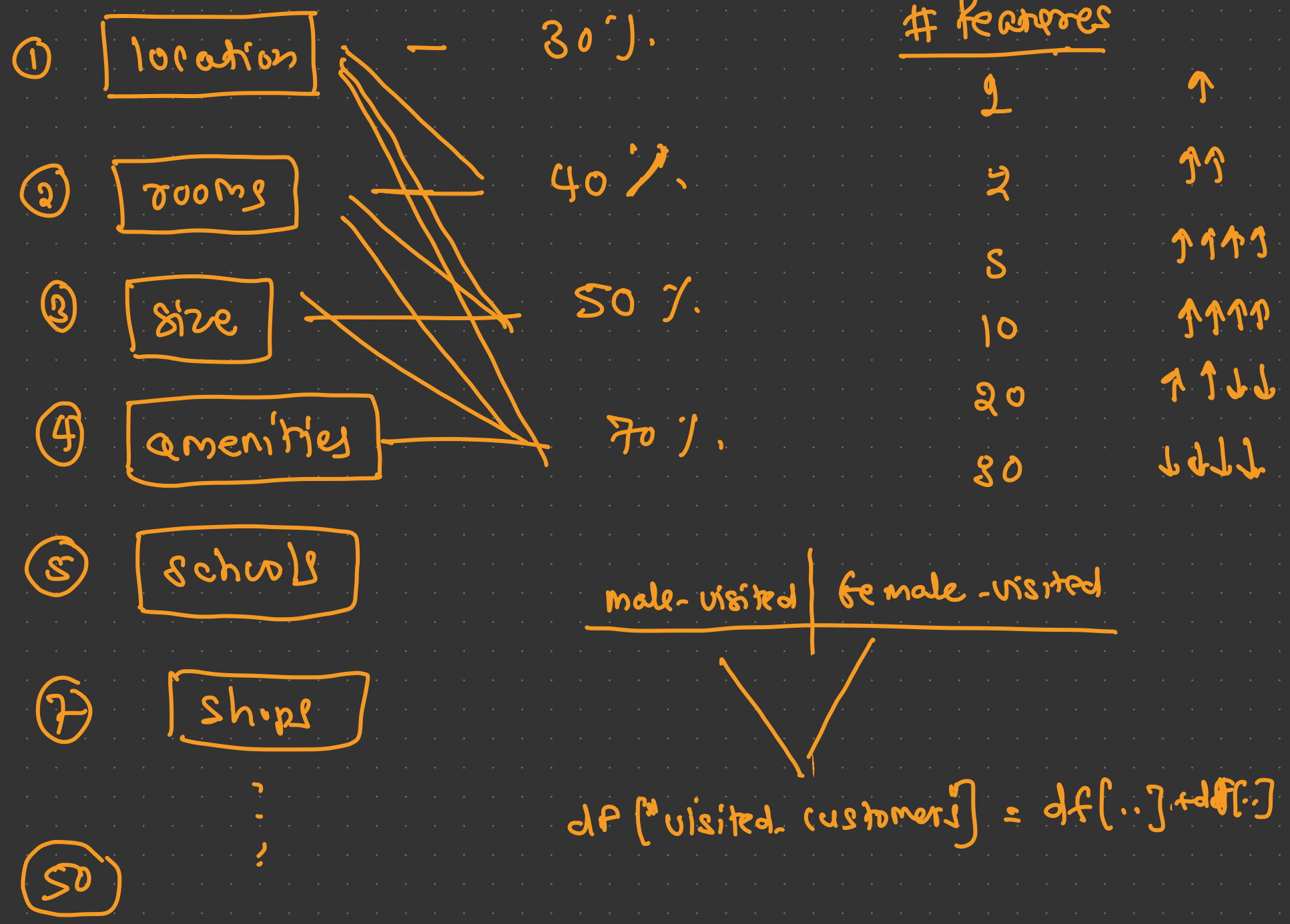
data → Feature Engineering → create model → evaluate model

Feature Engineering

way to find relavent features

important

significant


① location
② rooms
③ size
④ amenities
⑤ schools
⑦ shops
50
30 %.
40 %.
50 %.
70 %.
features
1
2
5
10
20
30
male-visited
female-visited
df['visited_customers'] = df[..] + df[..]

Introduction

Feature Extraction

Feature selection

- Feature Engineering is the process of creating new features or transforming existing features to improve the performance of a machine-learning model
- It involves selecting relevant information from raw data and transforming it into a format that can be easily understood by a model

significant - correlation analysis

- The goal is to improve model accuracy by providing more meaningful and relevant information
- The success of machine learning models heavily depends on the quality of the features used to train them
- Feature engineering involves a set of techniques that enable us to create new features by combining or transforming the existing ones

correlation analysis

- These techniques help to highlight the most important patterns and relationships in the data, which in turn helps the machine learning model to learn from the data more effectively

Need for Feature Engineering

- **Improve User Experience**
  - The primary reason we engineer features is to enhance the user experience of a product or service
  - By adding new features, we can make the product more intuitive, efficient, and user-friendly, which can increase user satisfaction and engagement → model accuracy increases
- **Competitive Advantage**
  - Another reason we engineer features is to gain a competitive advantage in the marketplace
  - By offering unique and innovative features, we can differentiate our product from competitors and attract more customers
- **Meet Customer Needs**
  - We engineer features to meet the evolving needs of customers
  - By analyzing user feedback, market trends, and customer behavior, we can identify areas where new features could enhance the product's value and meet customer needs
- **Future-Proofing**
  - Engineering features can also be done to future-proof a product or service
  - By anticipating future trends and potential customer needs, we can develop features that ensure the product remains relevant and useful in the long term

Feature Engineering Steps

- Data Cleansing → deal with NA, fix datatype, shuffle, balance, ...
  - Data cleansing (also known as data cleaning or data scrubbing) involves identifying and removing or correcting any errors or inconsistencies in the dataset. This step is important to ensure that the data is accurate and reliable.
- Data Transformation
- Feature Extraction → if required features are missing, create them
- Feature Selection → correlation analysis
  - Feature selection involves selecting the most relevant features from the dataset for use in machine learning
  - This can include techniques like correlation analysis, mutual information, and stepwise regression, domain knowledge, intuition
- Feature Iteration
  - Feature iteration involves refining and improving the features based on the performance of the machine learning model
  - This can include techniques like adding new features, removing redundant features and transforming features in different ways → model evaluation → optimization

Feature Engineering Process

- Feature Creation
- Feature Transformation
- Feature Extraction
- Feature Selection
- Feature Scaling

Feature Creation

- Feature Creation is the process of generating new features based on domain knowledge or by observing patterns in the data. It is a form of feature engineering that can significantly improve the performance of a machine-learning model
- Types of Feature Creation
  - Domain-Specific: Creating new features based on domain knowledge, such as creating features based on business rules or industry standards.
  - Data-Driven: Creating new features by observing patterns in the data, such as calculating aggregations or creating interaction features.
  - Synthetic: Generating new features by combining existing features or synthesizing new data points.
- Merits
  - Improves Model Performance: By providing additional and more relevant information to the model, feature creation can increase the accuracy and precision of the model.
  - Increases Model Robustness: By adding additional features, the model can become more robust to outliers and other anomalies.
  - Improves Model Interpretability: By creating new features, it can be easier to understand the model's predictions.
  - Increases Model Flexibility: By adding new features, the model can be made more flexible to handle different types of data.

Feature Transformation

- Feature Transformation is the process of transforming the features into a more suitable representation for the machine learning model. This is done to ensure that the model can effectively learn from the data
- Types of Feature Transformation:
  - Normalization: Rescaling the features to have a similar range, such as between 0 and 1, to prevent some features from dominating others.
  - Scaling: Scaling is a technique used to transform numerical variables to have a similar scale, so that they can be compared more easily. Rescaling the features to have a similar scale, such as having a standard deviation of 1, to make sure the model considers all features equally. → Standard Scalar
  - Encoding: Transforming categorical features into a numerical representation. Examples are one-hot encoding and label encoding.
  - Transformation: Transforming the features using mathematical operations to change the distribution or scale of the features. Examples are logarithmic, square root, and reciprocal transformations. → Polynomial Features
- Merits
  - Improves Model Performance: By transforming the features into a more suitable representation, the model can learn more meaningful patterns in the data.
  - Increases Model Robustness: Transforming the features can make the model more robust to outliers and other anomalies.
  - Improves Computational Efficiency: The transformed features often require fewer computational resources.
  - Improves Model Interpretability: By transforming the features, it can be easier to understand the model's predictions.

Feature Extraction

- It is the process of creating new features from existing ones to provide more relevant information to the machine learning model. This is done by transforming, combining, or aggregating existing features
- Types of Feature Extraction:
  - Dimensionality Reduction: Reducing the number of features by transforming the data into a lower-dimensional space while retaining important information. Examples are PCA and t-SNE.
  - Feature Combination: Combining two or more existing features to create a new one. For example, the interaction between two features.
  - Feature Aggregation: Aggregating features to create a new one. For example, calculating the mean, sum, or count of a set of features.
  - Feature Transformation: Transforming existing features into a new representation. For example, log transformation of a feature with a skewed distribution.
- Merits
  - Improves Model Performance: By creating new and more relevant features, the model can learn more meaningful patterns in the data.
  - Reduces Overfitting: By reducing the dimensionality of the data, the model is less likely to overfit the training data.
  - Improves Computational Efficiency: The transformed features often require fewer computational resources.
  - Improves Model Interpretability: By creating new features, it can be easier to understand the model's predictions.

Feature Selection → Correlation analysis

- It is the process of selecting a subset of relevant features from the dataset to be used in a machine-learning model. It is an important step in the feature engineering process as it can have a significant impact on the model's performance
- Types of Feature Selection: → correlation
  - Filter Method: Based on the statistical measure of the relationship between the feature and the target variable. Features with a high correlation are selected.
  - Wrapper Method: Based on the evaluation of the feature subset using a specific machine learning algorithm. The feature subset that results in the best performance is selected. → p-value
  - Embedded Method: Based on the feature selection as part of the training process of the machine learning algorithm.
- Merits
  - Reduces Overfitting: By using only the most relevant features, the model can generalize better to new data.
  - Improves Model Performance: Selecting the right features can improve the accuracy, precision, and recall of the model.
  - Decreases Computational Costs: A smaller number of features requires less computation and storage resources.
  - Improves Interpretability: By reducing the number of features, it is easier to understand and interpret the results of the model.

Feature Scaling

- It is the process of transforming the features so that they have a similar scale. This is important in machine learning because the scale of the features can affect the performance of the model
  - → speed to build model - time
  - accuracy
- Types of Feature Scaling:
  - Min-Max Scaling: Rescaling the features to a specific range, such as between 0 and 1, by subtracting the minimum value and dividing by the range. → MinMaxScaler
  - Standard Scaling: Rescaling the features to have a mean of 0 and a standard deviation of 1 by subtracting the mean and dividing by the standard deviation. → StandardScaler
  - Robust Scaling: Rescaling the features to be robust to outliers by dividing them by the interquartile range.
- Merits
  - Improves Model Performance: By transforming the features to have a similar scale, the model can learn from all features equally and avoid being dominated by a few large features.
  - Increases Model Robustness: By transforming the features to be robust to outliers, the model can become more robust to anomalies.
  - Improves Computational Efficiency: Many machine learning algorithms, such as k-nearest neighbors, are sensitive to the scale of the features and perform better with scaled features.
  - Improves Model Interpretability: By transforming the features to have a similar scale, it can be easier to understand the model's predictions.

PCA

Dimensionality Reduction

high dimension space → low dimension space

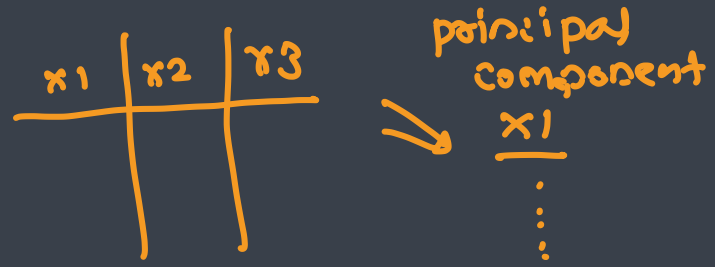
Curse of Dimensionality

- The Curse of Dimensionality refers to the phenomenon where the efficiency and effectiveness of algorithms deteriorate as the dimensionality of the data increases exponentially
- In high-dimensional spaces, data points become sparse, making it challenging to discern meaningful patterns or relationships due to the vast amount of data required to adequately sample the space
- The Curse of Dimensionality significantly impacts machine learning algorithms in various ways
- It leads to increased computational complexity, longer training times, and higher resource requirements
- Moreover, it escalates the risk of overfitting and spurious correlations, hindering the algorithms' ability to generalize well to unseen data → reduce testing accuracy

Dimensionality Issue

↳ features

- In machine learning classification problems, there are often too many factors on the basis of which the final classification is done
- These factors are basically variables called features
- The higher the number of features, the harder it gets to visualize the training set and then work on it
- Sometimes, most of these features are correlated, and hence redundant
- This is where dimensionality reduction algorithms come into play
- Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables
- It can be divided into feature selection and feature extraction



Advantages of Dimensionality Reduction

- It helps in data compression, and hence reduced storage space
- It reduces computation time
- It also helps remove redundant features, if any

↓

'interdependent

Disadvantages of Dimensionality Reduction

- It may lead to some amount of data loss
- PCA tends to find linear correlations between variables, which is sometimes undesirable
- PCA fails in cases where mean and covariance are not enough to define datasets
- We may not know how many principal components to keep- in practice, some thumb rules are applied

covariance

set of new features

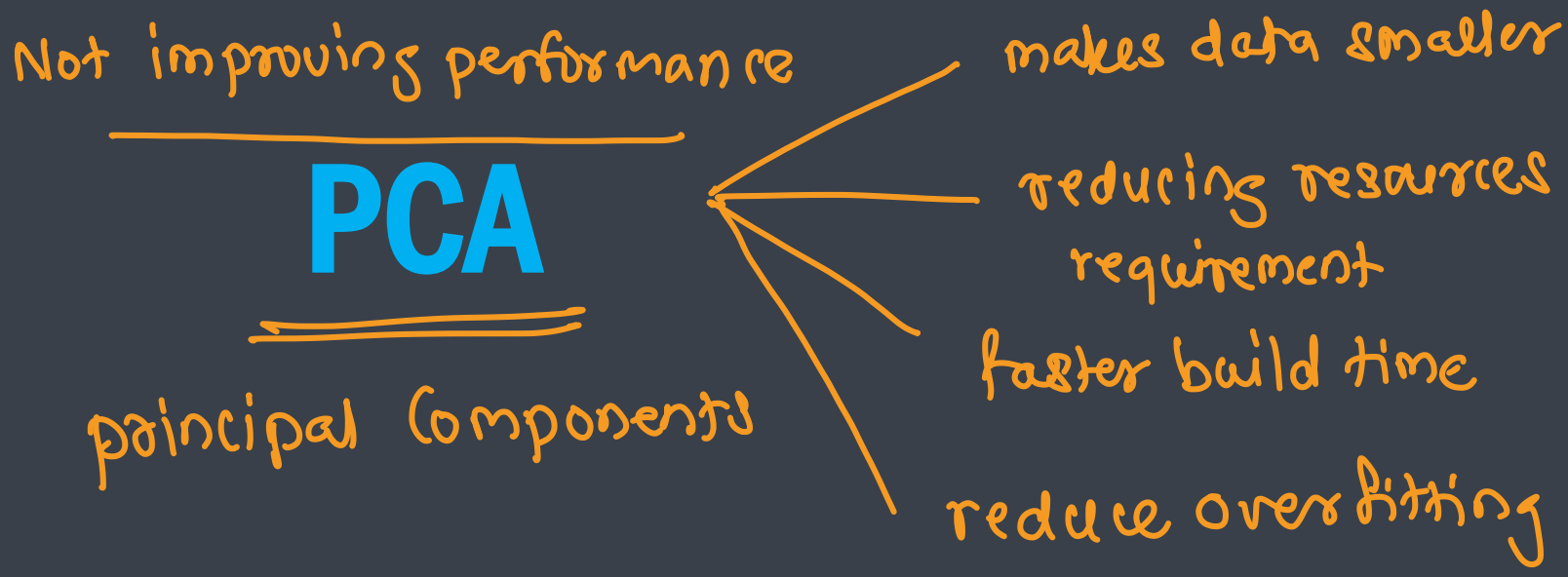
lower dimensions

Components of dimensionality reduction

- There are two components of dimensionality reduction:
- **Feature selection:** In this, we try to find a subset of the original set of variables, or features, to get a smaller subset which can be used to model the problem. It usually involves three ways:
  - Filter
  - Wrapper
  - Embedded
- **Feature extraction:** This reduces the data in a high dimensional space to a lower dimension space, i.e. a space with lesser no. of dimensions.

Methods of Dimensionality Reduction

- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)
- Generalized Discriminant Analysis (GDA)


Not improving performance
PCA
principal components
makes data smaller
reducing resources requirement
faster build time
reduce overfitting

Overview

- Is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets
- Reduces dimensions by transforming a large set of variables into a smaller one that still contains most of the information in the large set
- Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity
- Because smaller data sets are easier to explore and visualize and make analyzing data much easier and faster for machine learning algorithms without extraneous variables to process

What is PCA?

- This method was introduced by Karl Pearson
- It works on a condition that while the data in a higher dimensional space is mapped to data in a lower dimension space, the variance of the data in the lower dimensional space should be maximum

Step 1: Standardization → Standard Scaler

- The aim of this step is to standardize the range of the continuous initial variables so that each one of them contributes equally to the analysis

$$z = \frac{value - mean}{standard\ deviation}$$

- Once the standardization is done, all the variables will be transformed to the same scale.

Step 2: Covariance Matrix computation

- The aim of this step is to understand how the variables of the input data set are varying from the mean with respect to each other
- In other words, to see if there is any relationship between them
- Because sometimes, variables are highly correlated in such a way that they contain redundant information
- So, in order to identify these correlations, we compute the covariance matrix

$$\text{cov matrix}(x, y) = \begin{bmatrix} (0,0) & (0,1) \\ \text{cov}(x,x) & \text{cov}(x,y) \\ \text{cov}(y,x) & \text{cov}(y,y) \\ (1,0) & (1,1) \end{bmatrix}$$

Step 2: Covariance Matrix computation

- The covariance matrix is a $p \times p$ symmetric matrix (where p is the number of dimensions) that has as entries the covariances associated with all possible pairs of the initial variables
- For example, for a 3-dimensional data set with 3 variables x, y, and z, the covariance matrix is a 3×3 matrix of this from

$$\begin{bmatrix} Cov(x,x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Cov(y,y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{bmatrix}$$

Step 2: Covariance Matrix computation

- Since the covariance of a variable with itself is its variance (Cov(a,a)=Var(a)), in the main diagonal (Top left to bottom right) we actually have the variances of each initial variable
- Since the covariance is commutative (Cov(a,b)=Cov(b,a)), the entries of the covariance matrix are symmetric with respect to the main diagonal, which means that the upper and the lower triangular portions are equal.
- if positive then
  - the two variables increase or decrease together (correlated)
- if negative then
  - One increases when the other decreases (Inversely correlated)

Step 3: Compute eigenvectors eigenvalues

$|A - \lambda I| = 0$, $\begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix}$

- Eigenvectors and eigenvalues are the linear algebra concepts that we need to compute from the covariance matrix in order to determine the principal components of the data
- Principal components are new variables that are constructed as linear combinations or mixtures of the initial variables
- These combinations are done in such a way that the new variables (i.e., principal components) are uncorrelated and most of the information within the initial variables is squeezed or compressed into the first components
- Organizing information in principal components this way, will allow you to reduce dimensionality without losing much information, and this by discarding the components with low information and considering the remaining components as your new variables
- the principal components are less interpretable and don't have any real meaning since they are constructed as linear combinations of the initial variables

Step 4: Feature vector

- choose whether to keep all these components or discard those of lesser significance (of low eigenvalues), and form with the remaining ones a matrix of vectors that we call Feature vector
- feature vector is simply a matrix that has as columns the eigenvectors of the components that we decide to keep

Statistical Calculations

Example

first component

- Calculate PCA for the following dataset

X	Y
4	11
8	4
13	5
7	14

Step 1: Calculate Mean

$$\bar{x} = (4+8+13+7)/4 = 32/4 = 8$$

$$\bar{y} = (11+4+5+14)/4 = 34/4 = 8.5$$

Step 2: Calculate Covariance Matrix

$$\begin{bmatrix} cov(x,x) & cov(x,y) \\ cov(y,x) & cov(y,y) \end{bmatrix}$$

x	y	$x-\bar{x}$	$y-\bar{y}$	$(x-\bar{x})(y-\bar{y})$	$(x-\bar{x})^2$	$(y-\bar{y})^2$
4	11	-4	2.5	-10	16	6.25
8	4	0	-4.5	0	0	20.25
13	5	5	-3.5	-17.5	25	12.25
7	14	-1	5.5	-5.5	1	30.25
				-33	42	69

$$cov(x,y) = \frac{\sum (x-\bar{x})(y-\bar{y})}{N} = \frac{-33}{4} = -8.25$$

$$cov(x,x) = var(x) = \frac{\sum (x-\bar{x})^2}{N} = \frac{42}{4} = 10.5$$

$$cov(y,y) = var(y) = \frac{\sum (y-\bar{y})^2}{N} = \frac{69}{4} = 17.25$$

$$\text{cov mat} = \begin{bmatrix} 10.5 & -8.25 \\ -8.25 & 17.25 \end{bmatrix}$$

Step 3: Calculate eigenvalues of Covariance Matrix

$|A - \lambda I| = 0$

$$\left| \begin{bmatrix} 10.5 & -8.25 \\ -8.25 & 17.25 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \right| = 0$$

$$(10.5 - \lambda)(17.25 - \lambda) - (-8.25 \times -8.25) = 0$$

$$\lambda^2 - 27.75\lambda + 113.06 = 0$$

choose the max

$$\text{roots} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} = \begin{matrix} \lambda = \boxed{22.7} \checkmark \\ \lambda = 4.96 \end{matrix}$$

Step 4: Calculate eigenvector

$$(A-\lambda I)\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = 0$$

$$\left(\begin{bmatrix} 10.5 & -8.25 \\ -8.25 & 17.25 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}\right)\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = 0$$

for $\lambda = 22.7$ (max value is considered)

$$12.2u_1 + 8.2u_2 = 0$$
$$8.25u_1 + 5.4u_2 = 0$$

$$u_1 = -0.67$$
$$u_2 = 1$$

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} -0.67 \\ 1 \end{bmatrix}$$

Step 5: Normalize unit eigenvector

$$e = \begin{bmatrix} u_1 / \|u\| \\ u_2 / \|u\| \end{bmatrix}$$

$$e = \begin{bmatrix} -0.67 / 1.2 \\ 1 / 1.2 \end{bmatrix}$$

$$\boxed{e = \begin{bmatrix} -0.55 \\ 0.83 \end{bmatrix}}$$

$$\|u\| = \sqrt{(u_1)^2 + (u_2)^2}$$

$$= \sqrt{(-0.67)^2 + 1}$$

$$= \sqrt{0.44 + 1} = \sqrt{1.44}$$

$$\boxed{\|u\| = 1.2}$$

Step 6: Calculate first principal component

$$\text{first component } (x, y_1) = e^T \begin{bmatrix} x_1 - \bar{x} \\ y_1 - \bar{y} \end{bmatrix}$$

$$\text{first comp}(4, 11) = [-0.55 \quad 0.83] \begin{bmatrix} 4-8 \\ 11-8.5 \end{bmatrix}$$

$$\text{First comp}(8, 4) = [-0.55 \quad 0.83] \begin{bmatrix} 8-8 \\ 4-8.5 \end{bmatrix}$$

$$\text{first comp}(13, 5) = [-0.55 \quad 0.83] \begin{bmatrix} 13-8 \\ 5-8.5 \end{bmatrix}$$

$$\text{first comp}(7, 14) = [-0.55 \quad 0.83] \begin{bmatrix} 7-8 \\ 14-8.5 \end{bmatrix}$$

x	y	First comp
4	11	4.275
8	4	3.73
13	5	-5.665
7	14	5.115

original | extracted