Masters in collective intelligence

Module: Programming, Data science and Statistics. Lab 1

School of Collective Intelligence

Pr. Ikram Chairi
Dr. Moad Hicham Safhi

INTRODUCTION

This lab covers:

- Setting up the python dev environment.
- Tutorial: Hands on python syntax.
- Features selection Lab:
    - Removing features with low variance.
    - Interaction effect: Covariance Matrix & Correlation.

Setup environment

Things to install:

- Visual Studio Code (i assume you all have it)
- Python >= 3.6
- Jupyter notebook (you can install it during the activity session)

Setup environment

Things to install:

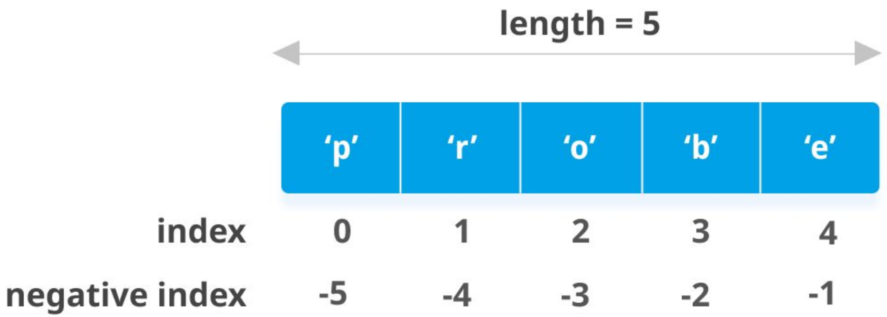
- Python interpreter: https://www.python.org/downloads/
- VSCode Python extension: From VSCode extensions.
- Jupyter notebook: https://jupyter.org/install
- Pandas and matplotlib packages:

  pip install pandas

Introduction to Python syntax

Python basic elements:

- Base types: integer, float, boolean, string, bytes.
- Lists: l1 = [“a”, 1, 10, [“um”, “6p”]]



Introduction to Python syntax

Python basic elements:

- Data structures:
    - List [2,7,9]
    - Tuple (2,7,9)
    - Dict {“key”: “value”}
    - Set {“key1”, “key2”}

Introduction to Python syntax

Python doesn’t use brackets {} !

```r
## R:
x <- 10
if(x > 5){
  print("x is greather than x")
}else{
  print("x is 5 or less")
}
```

```python
## Python (>=3.0):
x = 10
if x > 5:
    print("x is greather than x")
else:
    print("x is 5 or less")
```

Introduction to Python syntax

Task	Python	R
Variable assignment	x = 5	x <- 5 or x = 5 or 5 -> x
Vector creation	x = [1, 2, 3]	x <- c(1, 2, 3)
Array/Matrix Creation	import numpy as np X = np.array([[1, 2], [3, 4]])	X <- matrix(c(1,2,3,4), nrow=2, ncol=2)

Function Definition	def my_func(x):
	    return x*2
		My_func = function <- function(x){
		 return(x*2)
		}

Conditional statements	If x > 10: print(“x is greater than 10”)	if(x>10){ print(“x is greater than 10”) }

Introduction to Python syntax

Task	Python	R
For loop	For i in range(5): print(i)	for(i in 1:5){ print(i) }
While loop	While x<5: x+=1	while(x<5) { x <- x+i }
Function application to Vector/Array	x = [1, 2, 3] Y = list(map(lambda a: a*2, x))	X <- c(1, 2, 3) Y <- sapply(x, function(a){a*2})
Indexing (0-based vs 1-based)	x=[‘a’, ‘b’, ‘c’] first_elem = x[0]	x=c(‘a’, ‘b’, ‘c’) first_elem = x[1]

Introduction to Python syntax

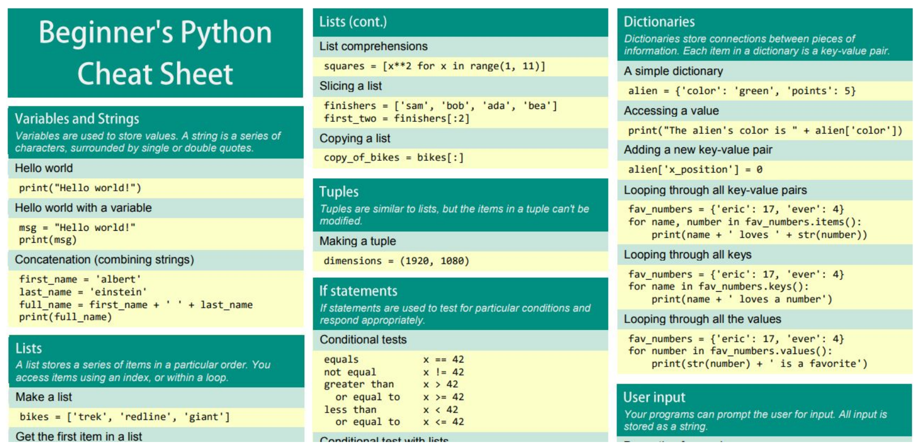
Python 3 Cheat Sheet

- https://canvas.harvard.edu/files/3517549/download?download_frd=1

- https://perso.limsi.fr/pointal/_media/python:cours:mementopython3-english.pdf

Or:

- Just type in Google: Python 3 Cheat Sheet



Beginner's Python Cheat Sheet

Variables and Strings

Variables are used to store values. A string is a series of characters, surrounded by single or double quotes.

Hello world

```
print("Hello world!")
```

Hello world with a variable

```
msg = "Hello world!"
print(msg)
```

Concatenation (combining strings)

```
first_name = 'albert'
last_name = 'einstein'
full_name = first_name + ' ' + last_name
print(full_name)
```

Lists

A list stores a series of items in a particular order. You access items using an index, or within a loop.

Make a list

```
bikes = ['trek', 'redline', 'giant']
```

Get the first item in a list

Lists (cont.)

List comprehensions

```
squares = [x**2 for x in range(1, 11)]
```

Slicing a list

```
finishers = ['sam', 'bob', 'ada', 'bea']
first_two = finishers[:2]
```

Copying a list

```
copy_of_bikes = bikes[:]
```

Tuples

Tuples are similar to lists, but the items in a tuple can't be modified.

Making a tuple

```
dimensions = (1920, 1080)
```

If statements

If statements are used to test for particular conditions and respond appropriately.

Conditional tests

```
equals              x == 42
not equal           x != 42
greater than        x > 42
  or equal to       x >= 42
less than           x < 42
  or equal to       x <= 42
```

Dictionaries

Dictionaries store connections between pieces of information. Each item in a dictionary is a key-value pair.

A simple dictionary

```
alien = {'color': 'green', 'points': 5}
```

Accessing a value

```
print("The alien's color is " + alien['color'])
```

Adding a new key-value pair

```
alien['x_position'] = 0
```

Looping through all key-value pairs

```
fav_numbers = {'eric': 17, 'ever': 4}
for name, number in fav_numbers.items():
    print(name + ' loves ' + str(number))
```

Looping through all keys

```
fav_numbers = {'eric': 17, 'ever': 4}
for name in fav_numbers.keys():
    print(name + ' loves a number')
```

Looping through all the values

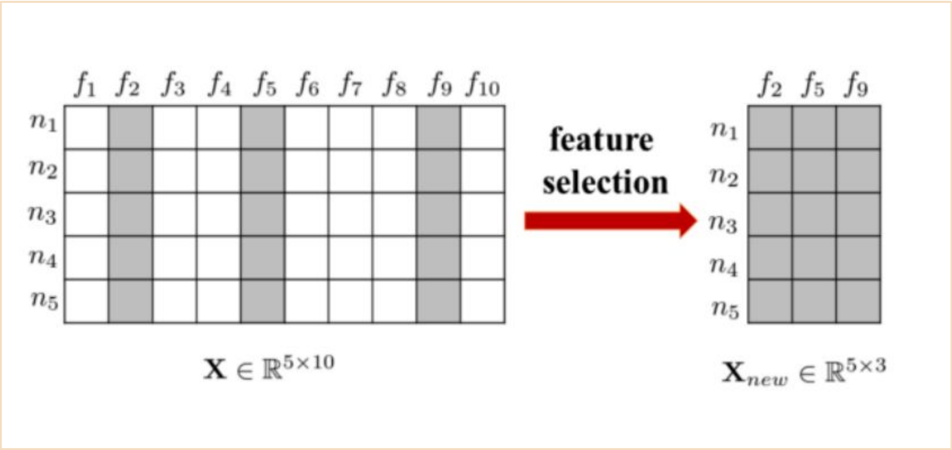
```
fav_numbers = {'eric': 17, 'ever': 4}
for number in fav_numbers.values():
    print(str(number) + ' is a favorite')
```

User input

Your programs can prompt the user for input. All input is stored as a string.

Feature Selection

Is the process of reducing the number of input variables when developing a predictive model



Feature Selection

Removing features with low variance:

- Variance tells us about the spread of the data.
- It tells us how far the points are from the mean.
- Features with low variance often contain mostly constant values or very similar values across all samples.

Feature Selection

Removing features with low variance:

- Python example:

```
from sklearn import datasets
from sklearn.feature_selection import VarianceThreshold
import numpy as np

iris = datasets.load_iris()
X = iris.data
y = iris.target

feature_names = iris.feature_names

thresholder = VarianceThreshold(threshold=.5)
X_high_variance = thresholder.fit_transform(X)

selected_indices = thresholder.get_support(indices=True)
selected_feature_names = np.array(feature_names)[selected_indices]

print("Selected features after applying VarianceThreshold:")
print(selected_feature_names)
```

```
Selected features after applying VarianceThreshold:
['sepal length (cm)' 'petal length (cm)' 'petal width (cm)']
```

Feature Selection

Covariance Matrix:

- Covariance is a measure of how two variables change together.
- In the context of feature selection and data analysis, the covariance matrix provides valuable insights into the relationships between variables within a dataset.

$$Var(X) = \frac{\sum (X_i - X)^2}{N} = \frac{\sum x_i}{N}$$

$$Cov(X, Y) = \frac{\sum (X_i - X)(Y_i - Y)}{N} = \frac{\sum x_i y_i}{N}$$

Feature Selection

Covariance Matrix:

- One straightforward approach to feature selection using covariance matrices is to calculate the covariance between each feature and the target variable.
- Features with high covariance with the target variable may be considered important for predicting the target and thus selected for inclusion in the model.
- Features with low covariance with the target variable may be considered less informative and thus excluded from the model.

Feature Selection

Covariance Matrix:

The covariance matrix, is a square matrix where each element represents the covariance between two variables.

Specifically, the (i, j)th element of the covariance matrix represents the covariance between the ith and jth variables in the dataset.

Covariance Matrix

$$\begin{bmatrix} Var(x_1) & \cdots & Cov(x_n, x_1) \\ \vdots & \ddots & \vdots \\ Cov(x_n, x_1) & \cdots & Var(x_n) \end{bmatrix}$$

Feature Selection

Covariance Matrix:

Suppose there are 3 dimensions, denoted as X, Y, Z.

The covariance:

$$COV = \begin{bmatrix} COV(X,X) & COV(X,Y) & COV(X,Z) \\ COV(Y,X) & COV(Y,Y) & COV(Y,Z) \\ COV(Z,X) & COV(Z,Y) & COV(Z,Z) \end{bmatrix}$$

- Note the diagonal is the covariance of each dimension with respect to itself, which is just the variance of each random variable.
- Also COV(X,Y) = COV(Y,X)

Feature Selection

Covariance Matrix:

using python:

```
Import numpy as np

np.cov(...)
```

```
help(np.cov)
```

Feature Selection

Covariance Matrix:

using python:

```
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris

# Load the Iris dataset
iris = load_iris()
X = iris.data
y = iris.target
feature_names = iris.feature_names

# Convert to DataFrame for easier manipulation
df = pd.DataFrame(X, columns=feature_names)
df['target'] = y

# Calculate the covariance matrix
cov_matrix = df.cov()

# Print the covariance matrix
print("Covariance Matrix of the iris dataset:")
cov_matrix
```

Feature Selection

Covariance Matrix:

```
Covariance Matrix of the iris dataset:
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
sepal length (cm)	0.685694	-0.042434	1.274315	0.516271	0.530872
sepal width (cm)	-0.042434	0.189979	-0.329656	-0.121639	-0.152349
petal length (cm)	1.274315	-0.329656	3.116278	1.295609	1.372483
petal width (cm)	0.516271	-0.121639	1.295609	0.581006	0.597315
target	0.530872	-0.152349	1.372483	0.597315	0.671141

Feature Selection

- **Correlation** is a statistic that measures the linear relationship between two continuous variables.
- Standardized measure bounded between -1 and 1.
- Easier to interpret as it is dimensionless and scale-invariant.
- Provides insights into the direction and strength of the relationship.

Example:

Pearson correlation:

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

Feature Selection

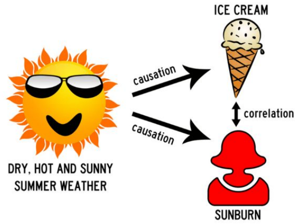
Correlation:

- Correlation measures can be used for feature selection:
  - Good variables correlate highly with the target.
  - Highly correlated variables might contain redundant information. No need to keep them all.

Feature Selection

Correlation:

- It's important to note that correlation does not imply causation.
- Just because two variables are correlated does not mean that changes in one variable cause changes in the other.
- Correlation simply measures the degree of association between variables.
- A correlation coefficient of 0 indicates no linear relationship between two variables, though they may still have non-linear associations.



Feature Selection

Correlation:

Pearson correlation is widely used correlation.

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

Feature Selection

Correlation:

It’s easy to implement Pearson correlation.

For this lab we can use the built-in function of pandas package.

df.corr()

By default it uses Pearson correlation. We can pass other correlation indices:

df.corr(method= “spearman”)

Feature Selection

In [23]:

```python
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris


iris = load_iris()
# then convert it to pandas dataframe:
iris_pandas = pd.DataFrame(data=iris.data, columns=iris.feature_names)

# or:
# iris_pandas = pd.read_csv('path_to_iris.csv')

# data exploration:
iris_pandas.head()
iris_pandas.tail()
iris_pandas.info()
iris_pandas.describe()



#
pearson = iris_pandas.corr()

pearson
```

Feature Selection

Pearson correlation

Out[23]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
sepal length (cm)	1.000000	-0.117570	0.871754	0.817941
sepal width (cm)	-0.117570	1.000000	-0.428440	-0.366126
petal length (cm)	0.871754	-0.428440	1.000000	0.962865
petal width (cm)	0.817941	-0.366126	0.962865	1.000000

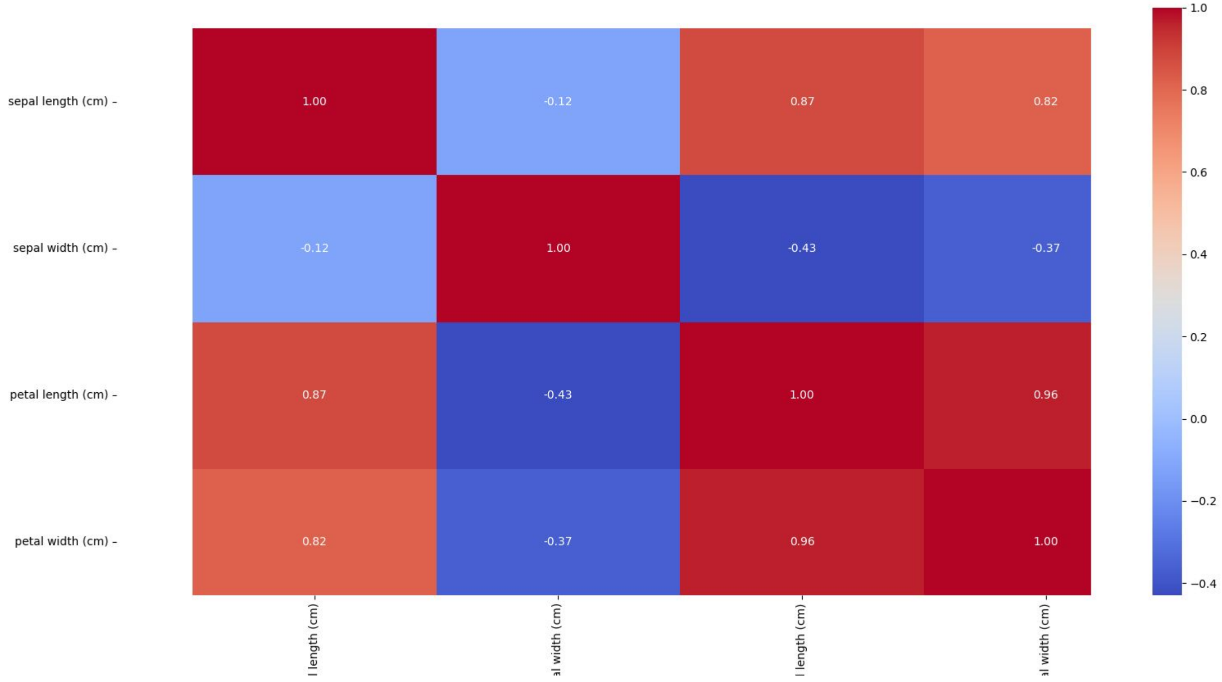
Feature Selection

Visualize it as heatmap:

```
#
pearson = iris_pandas.corr()

import seaborn as sns
sns.heatmap(pearson, annot=True, cmap='coolwarm', fmt=".2f")
plt.show()
```

Feature Selection

Visualize it as heatmap:



Exercice

Dataset example: Housing prices dataset.

- Import the dataset
- Do data exploration
- Calculate correlation between features
- Detect highly correlated variables. For example consider 0.75 threshold.
- Detect highly correlated variables with the target.

Masters in collective intelligence

Module: Programming, Data science and Statistics. Lab 2

School of Collective Intelligence

Pr. Ikram Chairi
Dr. Moad Hicham Safhi

Feature Selection

Quick Recap:

Previous Session Overview

INTRODUCTION

This lab covers:

- Statistical Tests for feature selection

Statistical Tests for Feature Selection

Statistical tests

- ***<u>Prerequisite concepts:</u>***
    - Hypothesis Testing
    - Null Hypothesis H0
    - Alternate Hypothesis H1
    - P-value
    - Degree of freedom

Statistical Tests for Feature Selection

Statistical tests

- Statistical tests are tools used to assess whether observed data support or refute hypotheses about population parameters based on sample data.
- They help determine if there are significant differences between samples or between a sample and a population.

Statistical Tests for Feature Selection

Statistical tests

- Descriptive statistics, such as mean, median, mode, range, or standard deviation, can be used to summarize data and provide insights into its characteristics.
- Mean is often preferred for statistical testing due to its sensitivity to changes in data values.

Statistical Tests for Feature Selection

Statistical tests

- Statistical methods yield numerical outputs that are compared with a significance level, typically represented by the p-value.
- If the calculated test statistic is greater than the p-value, it suggests acceptance of the null hypothesis, indicating no significant difference.
- Conversely, if the test statistic is less than the p-value, the null hypothesis is rejected, suggesting a significant difference exists.

Statistical Tests for Feature Selection

Statistical tests

- Understanding how to interpret statistical results is essential for making informed decisions in research and data analysis.

Statistical Tests for Feature Selection

Statistical tests

- The steps for conducting each statistical test are outlined below:
    - Calculate the test statistic using the appropriate mathematical formula.
    - Determine the critical value using statistical tables or software.
    - Utilize the critical value to calculate the p-value.
    - If the p-value is greater than 0.05, we accept the null hypothesis; otherwise, we reject it.

Statistical Tests for Feature Selection

Statistical tests

- Statistical tests for feature selection are techniques used to identify the most relevant features in a dataset for predictive modeling or analysis.
- Some common statistical tests for feature selection include:
  - Correlation Analysis
  - **Chi-Squared test / chi-2**
  - ANOVA (Analysis of variance)
  - T-Test
  - RFE

Statistical Tests for Feature Selection

Statistical tests

- The chi-squared (χ^2) test is a statistical method used for analyzing categorical data to determine if there is a significant association between **two categorical variables**.
- It assesses whether the **observed** frequency distribution of a categorical variable differs from the **expected** frequency distribution under the assumption of independence between the variables.

Statistical Tests for Feature Selection

Chi-squared test

- Steps to do chi-2 test:
  - Formulate hypothesis.
  - Calculate Expected Frequencies.
  - Compute the Test Statistic.
  - Determine Degrees of Freedom.
  - Find Critical value.
  - Make decision.

Statistical Tests for Feature Selection

Chi-squared test

- The null hypothesis (H0) states that there is no association between the two categorical variables.
- The alternative hypothesis (Ha) suggests that there is an association.

Statistical Tests for Feature Selection

Chi-squared test

- **Calculate Expected Frequencies:**
- Compute the expected frequency for each cell in the contingency table if the null hypothesis were true.
- This is done based on the marginal totals and assuming independence between the variables.

Statistical Tests for Feature Selection

Chi-squared test

- **Calculate Expected Frequencies:**
- Suppose we have a dataset that records the outcomes of a survey where individuals were asked about their favorite type of pet (cats, dogs, or birds) and their gender (male or female).
- We want to determine if there is an association between the preferred pet type and gender.

	Cats	Dogs	Birds
Male	20	30	10
Female	15	25	30

Statistical Tests for Feature Selection

Chi-squared test

- **NB:**

  The previous pairwise contingency table can also be derived from a dataset containing features. Example:

user_id	fav_pet	gender
1	Cat	Male
2	Dog	Male
3	Bird	Male
4	Cat	Female
5	Dog	Female
6	Bird	Female

pivot →

	Cats	Dogs	Birds
Male	1	1	1
Female	1	1	1

Statistical Tests for Feature Selection

Chi-squared test

- **Calculate Expected Frequencies:**
- Compute the row and column totals for the observed frequencies.

	Cats	Dogs	Birds	Raw total
Male	20	30	10	60
Female	15	25	30	60
Column total	35	55	30	120

Statistical Tests for Feature Selection

Chi-squared test

- **Calculate Expected Frequencies:**
- Use the formula for expected frequencies:
    - If two variables x1 and x2 are independent, then:

      $$P(x1\ U\ x2) = p(x1) * p(x2)$$

    - Therefore the expected frequency table for our variables considering they are independent is expressed as following:

	Cats	Dogs	Birds
Male	p(Male) * p(Cats)	p(Male) * p(Dogs)	p(Male) * p(Birds)
Female	p(Female) * p(Cats)	p(Female) * p(Dogs)	p(Female) * p(Birds)

Statistical Tests for Feature Selection

Chi-squared test

- **Calculate Expected Frequencies:**
- Use the formula for expected frequencies:
    - E1(Male, Cats) = p(Cats) * p(Male) = 35/120 * 60/120 = 2100/120 = 17,5...

	Cats	Dogs	Birds	Raw total
Male	20	30	10	60
Female	15	25	30	60
Column total	35	55	30	120

Statistical Tests for Feature Selection

Chi-squared test

- **Calculate Expected Frequencies:**
- Similarly, we calculate expected frequencies for all other cells in the contingency table.
- Replace the observed frequencies with the calculated expected frequencies in the contingency table.

	Cats	Dogs	Birds
Male	17.5		
Female			

Statistical Tests for Feature Selection

Chi-squared test

- Calculate the chi-2 score:

$$\chi^2 = \sum \frac{\left(O_{ij} - E_{ij}\right)^2}{E_{ij}}$$

In this example it’s: 4.16

Then check the chi-square table to decide either to accept or reject the null hypothesis.

$$4.16 < 5.99$$

Statistical Tests for Feature Selection

Chi-squared test

- chi-square table alternative:

```python
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats

# Degrees of freedom and significance level
df = 2
alpha = 0.05

# Critical value from the Chi-Square distribution table
critical_value = stats.chi2.ppf(1 - alpha, df)
critical_value
```

```
5.991464547107979
```

Statistical Tests for Feature Selection

Chi-squared test

- Since our calculated chi-squared statistic is less than the critical value, we fail to reject the null hypothesis.
- Therefore, based on the chi-squared test, we do not have enough evidence to conclude that there is a significant association between preferred pet type and gender at the 0.05 significance level.

Statistical Tests for Feature Selection

Chi-squared test

- Using python:

```python
import numpy as np
from scipy.stats import chi2_contingency

observed = np.array([[20, 30, 10],
                     [15, 25, 20]])


chi2, p_value, dof, expected = chi2_contingency(observed)

print("Expected frequencies:")
print(expected)

print("Chi-squared statistic:", chi2)
print("P-value:", p_value)
print("Degrees of freedom:", dof)

alpha = 0.05

if p_value <= alpha:
    print('Dependent (reject H0)')
else:
    print('Independent (H0 holds true)')
```

```
Expected frequencies:
[[17.5 27.5 15. ]
 [17.5 27.5 15. ]]
Chi-squared statistic: 4.5021645021645025
P-value: 0.10528521784009062
Degrees of freedom: 2
Independent (H0 holds true)
```

Statistical Tests for Feature Selection

Chi-squared test

- Example:

```python
import pandas as pd
from sklearn.feature_selection import SelectKBest, chi2

# https://www.kaggle.com/datasets/m1thr4nd1r/tennis?resource=download
df = pd.read_csv('~/Downloads/tennis.csv')
df.head()
```

	day	outlook	temp	humidity	wind	play
0	D1	Sunny	Hot	High	Weak	No
1	D2	Sunny	Hot	High	Strong	No
2	D3	Overcast	Hot	High	Weak	Yes
3	D4	Rain	Mild	High	Weak	Yes
4	D5	Rain	Cool	Normal	Weak	Yes

Statistical Tests for Feature Selection

- Example 1:

```python
import pandas as pd
from sklearn.feature_selection import SelectKBest, chi2

# https://www.kaggle.com/datasets/m1thr4nd1r/tennis?resource=download
df = pd.read_csv('~/Downloads/tennis.csv')
df.head()

# remove column names & day id:
df = df = df.drop('day', axis=1)

# Convert categorical variable into dummy/indicator variables:
df = pd.get_dummies(df, columns=df.columns, drop_first=True)


x = df.drop('play_Yes', axis=1)
y = df['play_Yes']


selector = SelectKBest(k=3, score_func=chi2)
selector.fit(x, y)

print(selector.scores_)

selected_features_idx = selector.get_support(indices=True)
selected_features_idx

selected_features = x.columns[selected_features_idx]
selected_features

x[selected_features].head()
```

```
[0.04       1.28444444 0.35555556 0.01481481 1.4        0.4       ]
```

	outlook_Sunny	humidity_Normal	wind_Weak
0	1	0	1
1	1	0	0
2	0	0	1
3	0	0	1
4	0	1	1

Statistical Tests for Feature Selection

```
from sklearn.feature_selection import SelectKBest, chi2
from sklearn.preprocessing import LabelEncoder
import pandas as pd

# https://www.kaggle.com/datasets/m1thr4nd1r/tennis?resource=download
df = pd.read_csv('~/Downloads/tennis.csv')
df.head()

# remove column names & day id:
df = df = df.drop('day', axis=1)

x = df.drop('play', axis=1)
y = df['play']


# Encode categorical variables
label_encoders = {}
for column in x.columns:
    if x[column].dtype == 'object':
        label_encoders[column] = LabelEncoder()
        x[column] = label_encoders[column].fit_transform(x[column])

selector = SelectKBest(k=3, score_func=chi2)
selector.fit(x, y)

print(selector.scores_)

selected_features_idx = selector.get_support(indices=True)
selected_features_idx

selected_features = x.columns[selected_features_idx]
selected_features

x[selected_features].head()
```

```
[2.02814815 0.02222222 1.4        0.4        ]
```

- Example 2:


```
[2.02814815 0.02222222 1.4        0.4        ]
```


	outlook	humidity	wind
0	2	0	1
1	2	0	0
2	0	0	1
3	1	0	1
4	1	1	1

Statistical Tests for Feature Selection

ANOVA

- Analysis of Variance (ANOVA) is a statistical technique used to analyze the variation between groups and within groups.
- In the context of feature selection, ANOVA assesses whether there are statistically significant differences in the means of numerical features across different categories of a categorical target variable.

Statistical Tests for Feature Selection

ANOVA vs Chi-2

- ANOVA is typically used when the data consists of continuous numerical variables (e.g., measurements, scores) and involves comparing the means of these variables across multiple groups or categories.
- The chi-squared test is used when the data consists of categorical variables and involves analyzing the association or independence between these variables.

Statistical Tests for Feature Selection

ANOVA vs Chi-2

- ANOVA is used to test for differences in means across multiple groups or conditions. It assesses whether there are statistically significant differences in the means of the numerical variable(s) among the groups.
- The chi-squared test is used to test for association or independence between two categorical variables. It determines whether there is a significant relationship between the categories of the variables.

Statistical Tests for Feature Selection

ANOVA

- F Statistic used to compare two variances, s1 and s2, by dividing them.
- The result is always a positive number (because variances are always positive).

Statistical Tests for Feature Selection

ANOVA

- The equation for comparing two variances with the F test is:

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Squares (MS)	F
Within	$SSW = \sum_{j=1}^{k}\sum_{j=1}^{l}(X-\bar{X}_j)^2$	$df_w = k-1$	$MSW = \frac{SSW}{df_w}$	$F = \frac{MSB}{MSW}$
Between	$SSB = \sum_{j=1}^{k}(\bar{X}_j-\bar{X})^2$	$df_b = n-k$	$MSB = \frac{SSB}{df_b}$	
Total	$SST = \sum_{j=1}^{n}(\bar{X}_j-\bar{X})^2$	$df_t = n-1$		

https://byjus.com/anova-formula/

Statistical Tests for Feature Selection

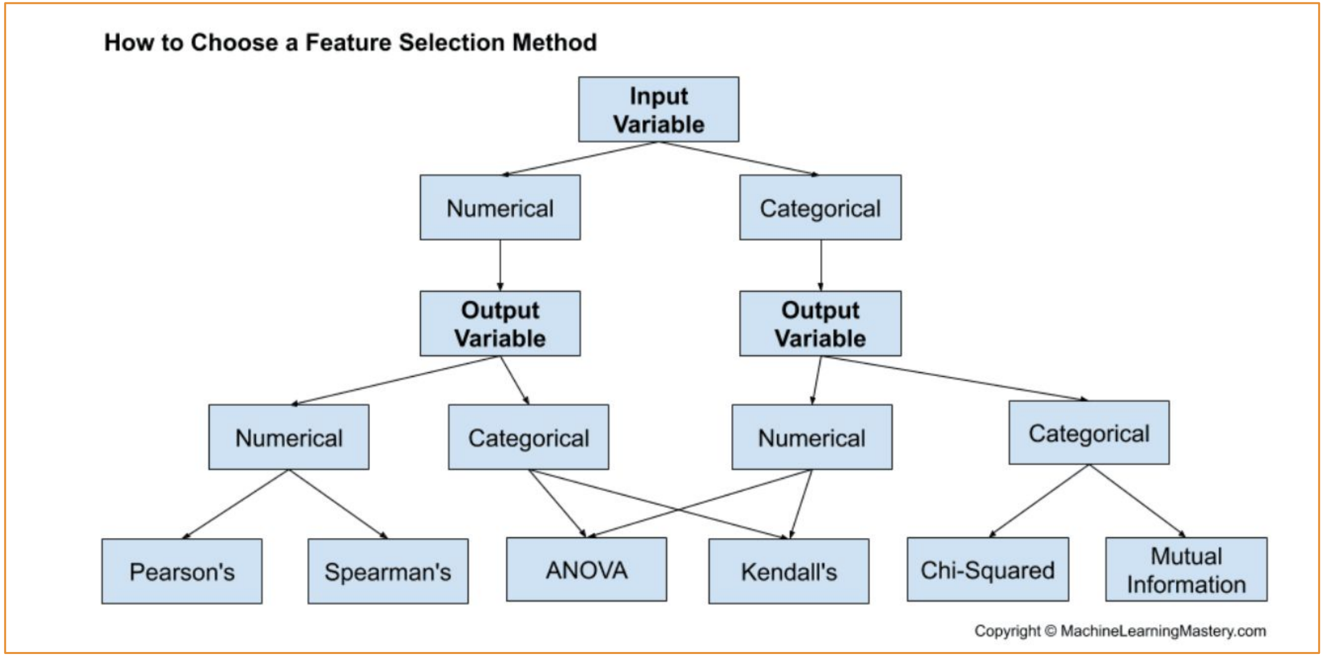
ANOVA

```python
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_classif

# Load the Iris dataset
iris = load_iris()
X = iris.data  # Features
y = iris.target  # Target variable

# Convert data into a DataFrame (optional but helpful for visualization)
df = pd.DataFrame(X, columns=iris.feature_names)
df['target'] = y

# Perform ANOVA for feature selection
selector = SelectKBest(score_func=f_classif, k=2)  # Select top 2 features
X_selected = selector.fit_transform(X, y)

# Get the selected feature indices
selected_feature_indices = selector.get_support(indices=True)
selected_features = df.columns[selected_feature_indices[:-1]]  # Exclude the target variable

# Print selected features
print("Selected features:", selected_features)
```

```
Selected features: Index(['petal length (cm)'], dtype='object')
```

Statistical Tests for Feature Selection

How to choose a Feature Selection Method:



Masters in collective intelligence

Module: Programming, Data science and Statistics. Lab 3

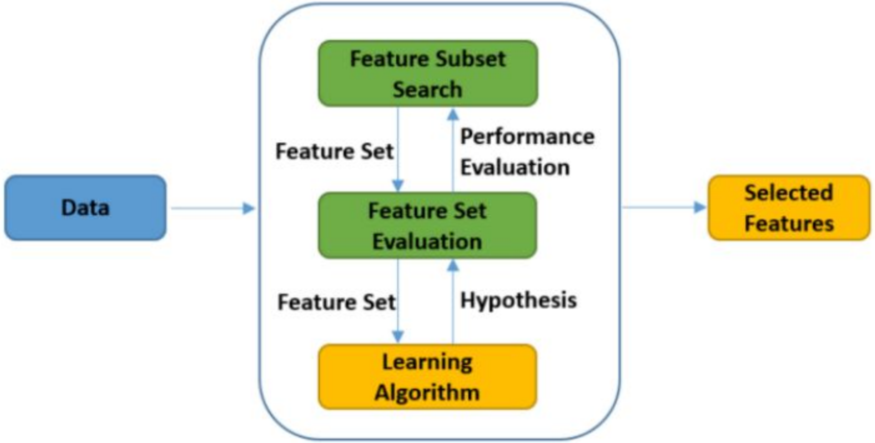
School of Collective Intelligence

Pr. Ikram Chairi
Dr. Moad Hicham Safhi

Recursive Feature Elimination

Recursive Feature Elimination

3- Wrapper Methods



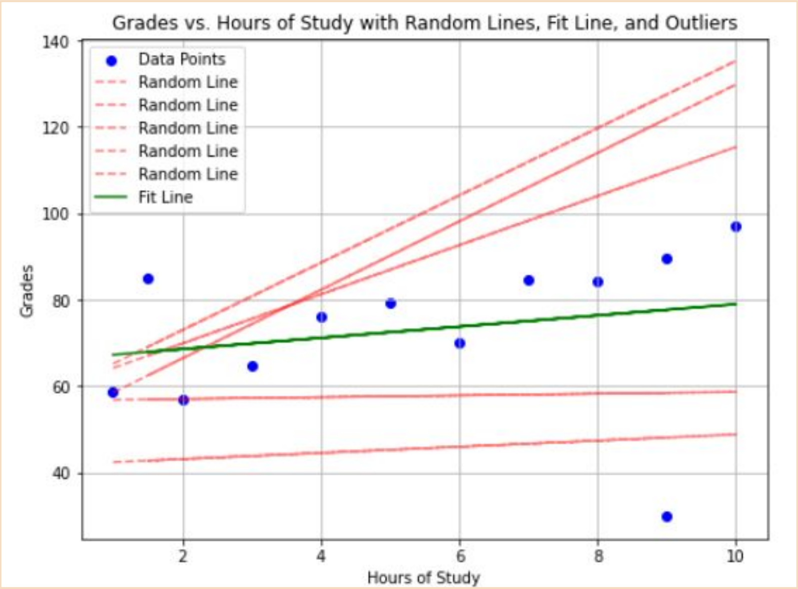
- Step 1: search for a subset of features
- Step 2: evaluate the selected features
- Repeat Step 1 and Step 2 until stopped

Slide credit :
Montreal university & MILA course - IFT6758 - Data Science
Slides here

Recursive Feature Elimination

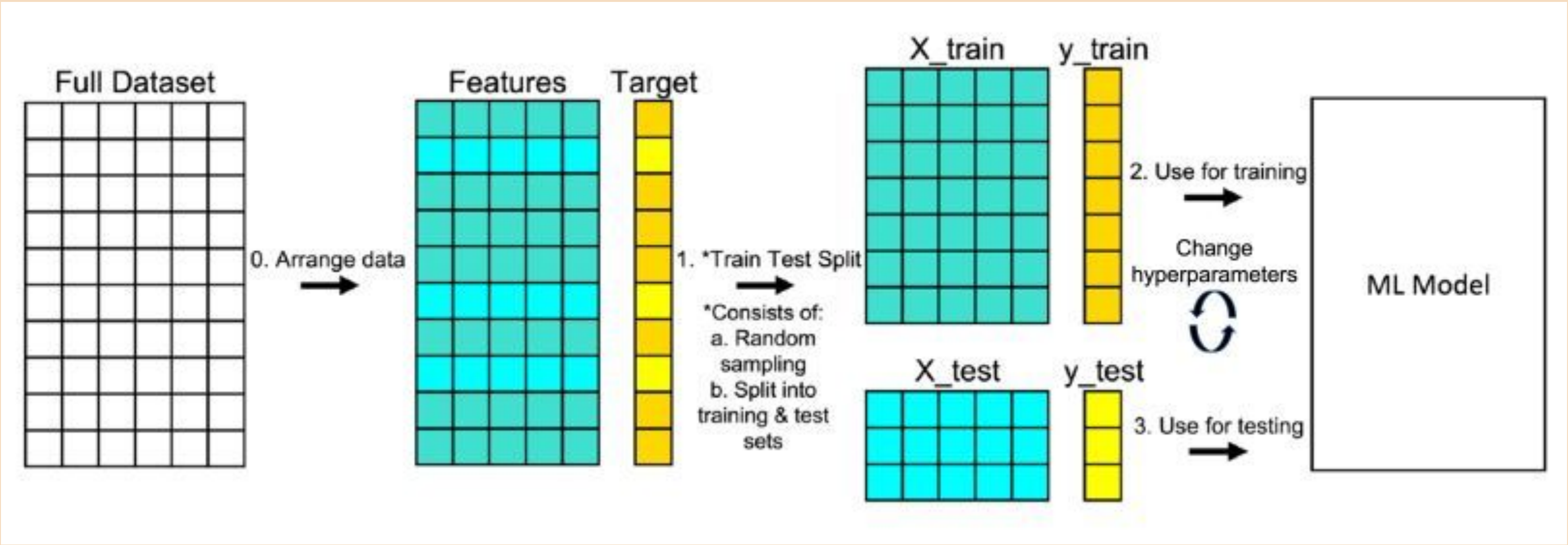
Machine learning

- **Data Modeling: Linear regression:**
- Regression is an algorithmic process that begins with a random line (or curve) and iteratively improves it by minimizing the error between predicted and actual outcomes.
- This improvement is achieved through calculating and optimizing the error, aiming to make the model's predictions progressively more accurate over time.



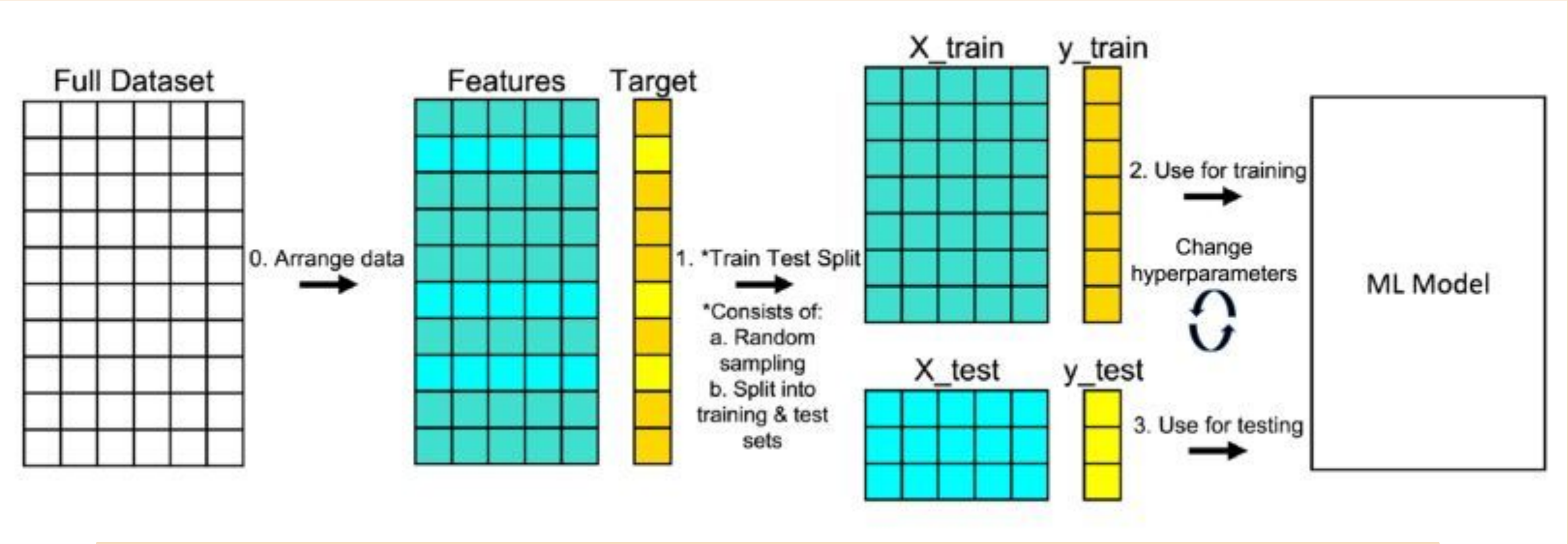
Recursive Feature Elimination

Machine learning



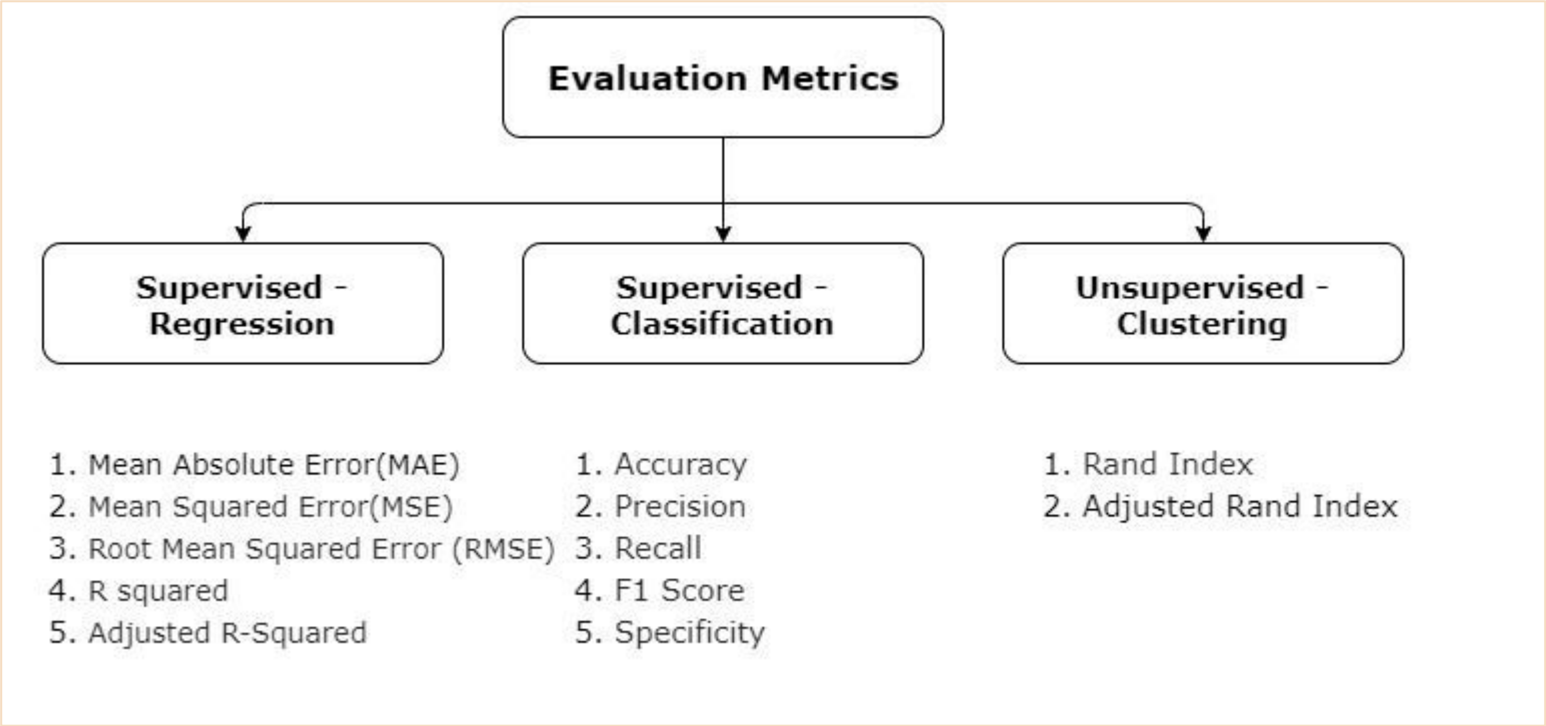
Recursive Feature Elimination

Machine learning




```
# Sperate train and test data
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Recursive Feature Elimination



Recursive Feature Elimination

Task 1:

To Do:

- Task 0: Execute each cell sequentially and ensure comprehension of the code in each.
- Task 1: Apply the same code to the diabetes dataset. Remember to adjust the evaluation method since the target variable is continuous.
- Task 2: Using the custom_rfe function, implement forward feature selection (custom_fss).

Masters in collective intelligence

Module: Programming, Data science and Statistics. Lab 4

School of Collective Intelligence

Pr. Ikram Chairi
Dr. Moad Hicham Safhi

Quizz

Questions

- Can you explain the difference between filter, wrapper, and embedded methods in feature selection?

- What are the advantages and disadvantages of using feature selection techniques?

Dimensionality Reduction

Dimensionality Reduction

Introduction

- **Feature selection:**
  - Select a subset of features
  - The measurement units (length, weight, etc.) of the features are preserved.
- **Dimensionality reduction:**
  - **Transform features into a smaller set.**
  - **The measurement units (length, weight, etc.) of the features are lost.**

Dimensionality Reduction

Dimensionality Reduction

- PCA is a powerful technique used to reduce the dimensionality of datasets while preserving most of the essential information.
- PCA seeks to find a **new set of variables**, called principal components, that capture **the maximum variance in the data**.

Dimensionality Reduction

Dimensionality Reduction

- Consider the following 3D points

$$\begin{bmatrix}1\\2\\3\end{bmatrix} \quad \begin{bmatrix}2\\4\\6\end{bmatrix} \quad \begin{bmatrix}4\\8\\12\end{bmatrix} \quad \begin{bmatrix}3\\6\\9\end{bmatrix} \quad \begin{bmatrix}5\\10\\15\end{bmatrix} \quad \begin{bmatrix}6\\12\\18\end{bmatrix}$$

- If each component is stored in a byte, we need $18 = 3 \times 6$ bytes

Dimensionality Reduction

Dimensionality Reduction

- Looking closer, we can see that all the points are related geometrically
  - they are all in the same direction, scaled by a factor:

$$\begin{bmatrix}1\\2\\3\end{bmatrix} = 1 \times \begin{bmatrix}1\\2\\3\end{bmatrix} \qquad \begin{bmatrix}4\\8\\12\end{bmatrix} = 4 \times \begin{bmatrix}1\\2\\3\end{bmatrix} \qquad \begin{bmatrix}5\\10\\15\end{bmatrix} = 5 \times \begin{bmatrix}1\\2\\3\end{bmatrix}$$

$$\begin{bmatrix}2\\4\\6\end{bmatrix} = 2 \times \begin{bmatrix}1\\2\\3\end{bmatrix} \qquad \begin{bmatrix}3\\6\\9\end{bmatrix} = 3 \times \begin{bmatrix}1\\2\\3\end{bmatrix} \qquad \begin{bmatrix}6\\12\\18\end{bmatrix} = 6 \times \begin{bmatrix}1\\2\\3\end{bmatrix}$$

Dimensionality Reduction

Dimensionality Reduction

$$\begin{bmatrix}1\\2\\3\end{bmatrix} = 1 \times \begin{bmatrix}1\\2\\3\end{bmatrix} \qquad \begin{bmatrix}4\\8\\12\end{bmatrix} = 4 \times \begin{bmatrix}1\\2\\3\end{bmatrix} \qquad \begin{bmatrix}5\\10\\15\end{bmatrix} = 5 \times \begin{bmatrix}1\\2\\3\end{bmatrix}$$

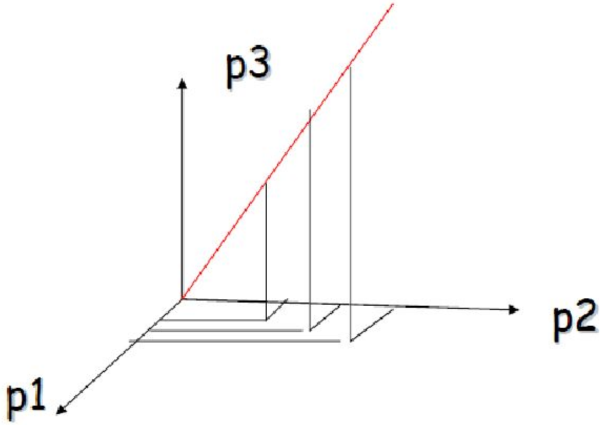
$$\begin{bmatrix}2\\4\\6\end{bmatrix} = 2 \times \begin{bmatrix}1\\2\\3\end{bmatrix} \qquad \begin{bmatrix}3\\6\\9\end{bmatrix} = 3 \times \begin{bmatrix}1\\2\\3\end{bmatrix} \qquad \begin{bmatrix}6\\12\\18\end{bmatrix} = 6 \times \begin{bmatrix}1\\2\\3\end{bmatrix}$$

- They can be stored using only 9 bytes (50% savings!):
  - Store one direction (3 bytes) + the multiplying constants (6 bytes)

Dimensionality Reduction

Dimensionality Reduction

- View points in 3D space

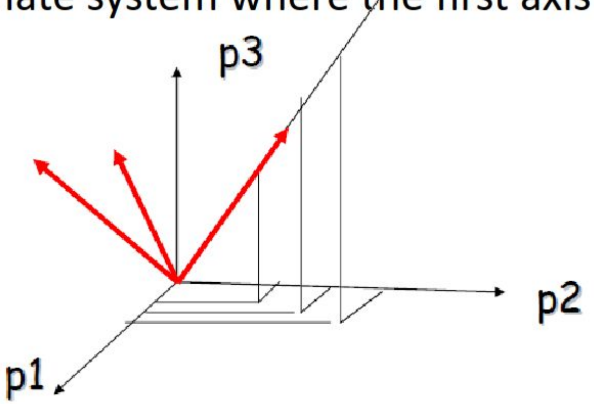


- In this example, all the points happen to lie on one line
  - a 1D subspace of the original 3D space

Dimensionality Reduction

Dimensionality Reduction

- Consider a new coordinate system where the first axis is along the direction of the line

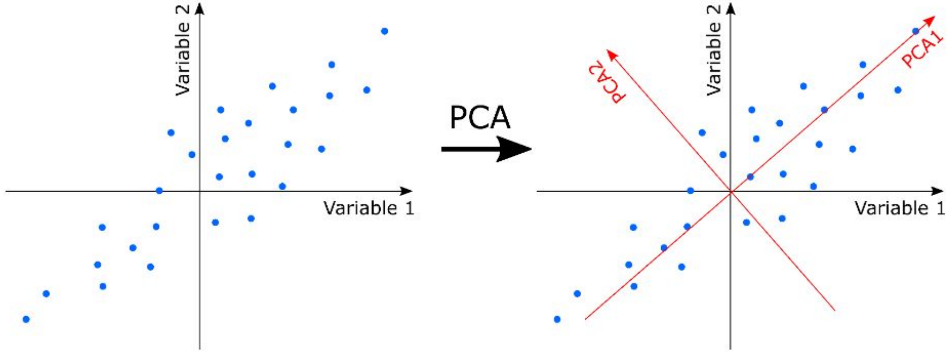


- In the new coordinate system, every point has only one non-zero coordinate
  - we only need to store the direction of the line (a 3 bytes point) and the nonzero coordinates for each point (6 bytes)

Dimensionality Reduction

Dimensionality Reduction

- Given a set of points, how can we know if they can be compressed similarly to the previous example?
  - We can look into the correlation between the points by the tool of PCA



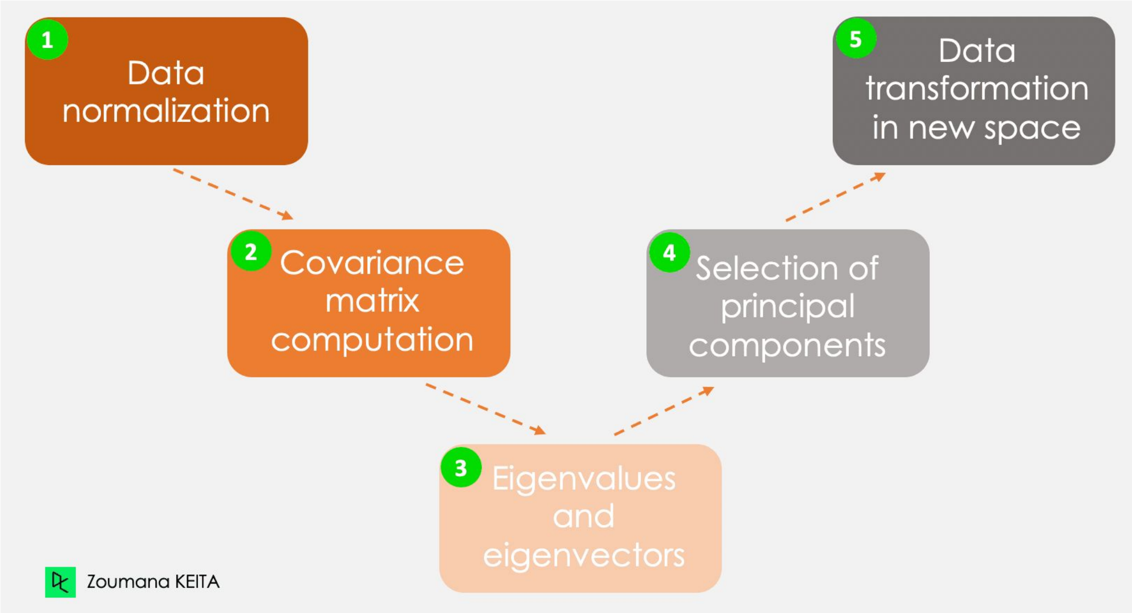
Dimensionality Reduction

Dimensionality Reduction

- In the previous example, PCA rebuilds the coordination system for the data by selecting:
  - The direction with the largest variance as the first new base direction;
  - The direction with the second largest variance as the second new direction ;
  - And so on;

Dimensionality Reduction

Dimensionality Reduction



```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

# Load the Iris dataset
iris = load_iris()
X = iris.data  # Features
y = iris.target  # Target variable

# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Perform PCA
pca = PCA(n_components=2)  # Reduce to 2 principal components for visualization
X_pca = pca.fit_transform(X_scaled)

# Plot PCA results
plt.figure(figsize=(8, 6))
for target in np.unique(y):
    plt.scatter(X_pca[y == target, 0], X_pca[y == target, 1], label=iris.target_names[target])
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.title('PCA of Iris Dataset')
plt.legend()
plt.grid(True)
plt.show()
```