ibimbing

BREAST CANCER CLASSIFICATION

RANDOM FOREST CLASSIFIER & TREE DECISION

by Rofiqo Azzahra

https://www.linkedin.com/in/rofiqo-azzahra

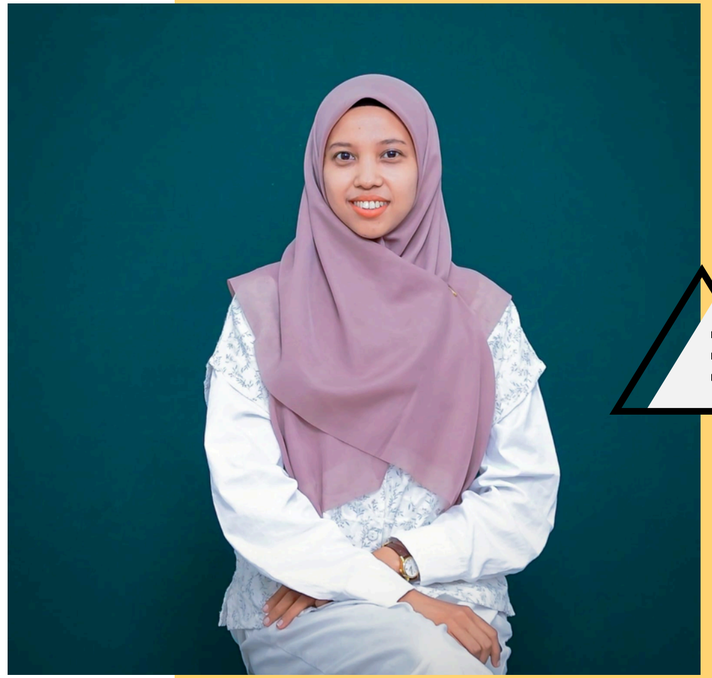
LIST OF CONTENTS

Introducing

ABOUT ME

Hello, everyone! I am Rofiqo Azzahra, a Bachelor of Information Systems with a 3.66 GPA. I'm a Project/Document Administrator with over three years of experience in the telecommunications industry.

I am highly interested in Data Science because of its ability to analyze complex data and drive impactful decisions. I'm excited to learn more and apply my skills in this field.



Rofiqo Azzahra

https://www.linkedin.com/in/rofiqo-azzahra

TOOLS USED

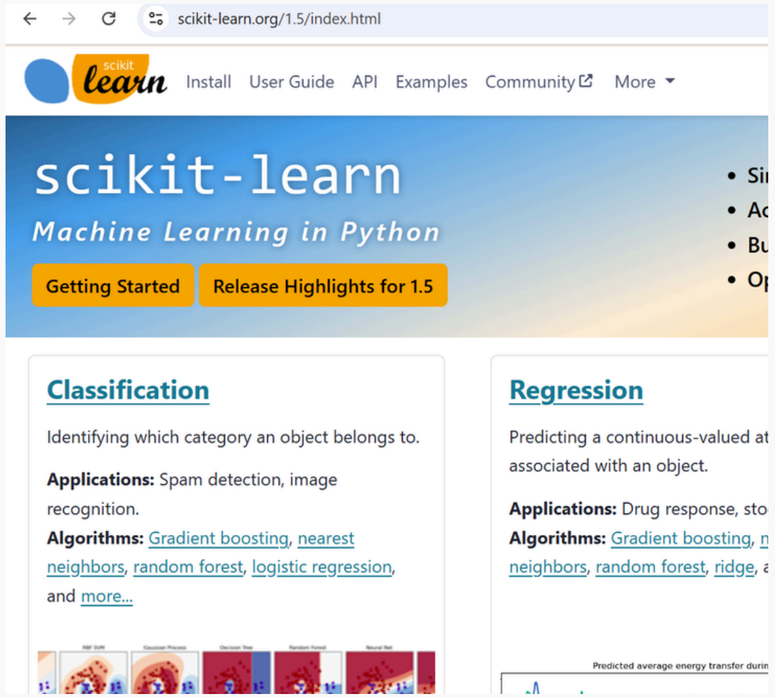
DESCRIPTION

- **Breast cancer** is a disease that occurs when cells in the breast grow out of control and form tumors. The tumors can spread to other parts of the body, which can be fatal.

- **Scikit-learn** is a free, open-source Python library that helps implement machine learning models. It's a popular choice for beginners because of its extensive documentation and ability to work well in production.

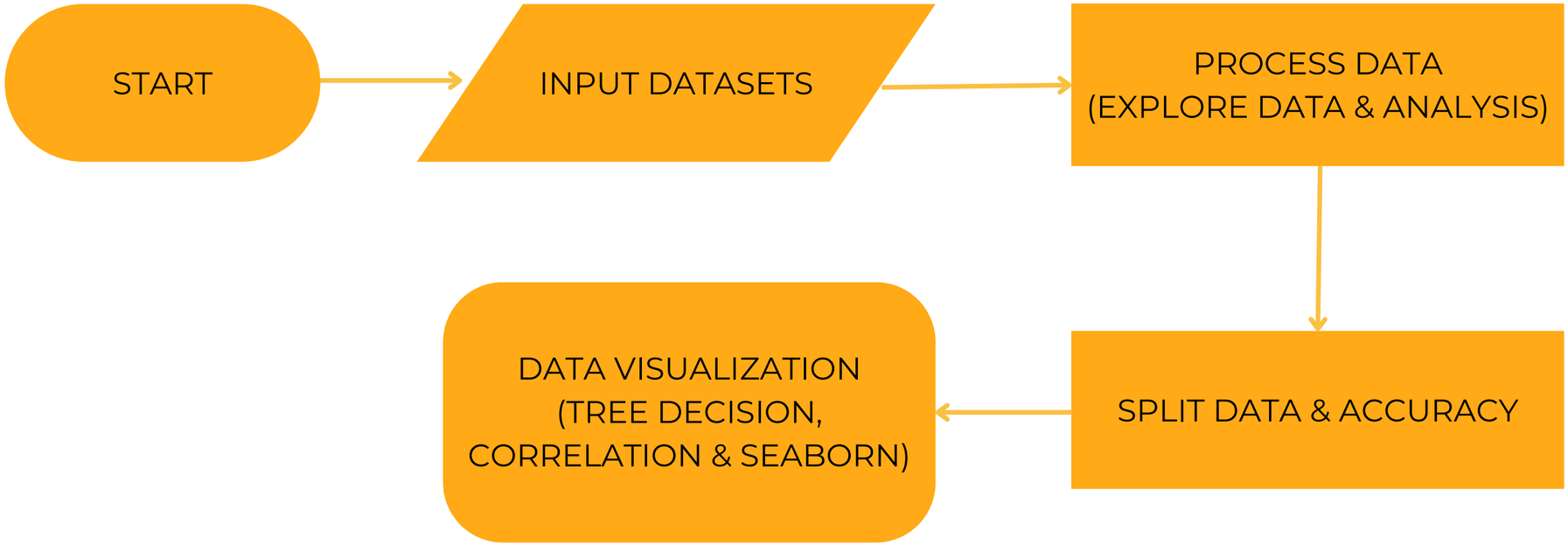
- **Pandas (styled as pandas)** is a software library written for the Python programming language for data manipulation and analysis.

- **Google Colab** is a hosted Jupyter Notebook service that requires no setup to use and provides free of charge access to computing resources, including GPUs and TPUs. Colab is especially well suited to machine learning, data science, and education.



https://scikit-learn.org/1.5/index.html

FLOWCHART



https://www.linkedin.com/in/rofiqo-azzahra

DATA PROCESS

import libraries and load datasets

```
#import data from sklearn
from sklearn import datasets
import pandas as pd

#load datasets and convert into a Dataframe
cancer = datasets.load_breast_cancer()

x = cancer.data      #input for machine learning
y = cancer.target    #ouput of machine learning

#convert feature and target data into Dataframe
df_x = pd.DataFrame(x, columns=cancer.feature_names)
df_y = pd.Series(y, name='target')

#combine features and targets in one Dataframes
df = pd.concat([df_x, df_y], axis = 1)

df.head(10) #show 10 line of Dataframe
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419	0.07871	...
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.1812	0.05667	...
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.2069	0.05999	...
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.2597	0.09744	...
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.1809	0.05883	...
5	12.45	15.70	82.57	477.1	0.12780	0.17000	0.15780	0.08089	0.2087	0.07613	...
6	18.25	19.98	119.60	1040.0	0.09463	0.10900	0.11270	0.07400	0.1794	0.05742	...
7	13.71	20.83	90.20	577.9	0.11890	0.16450	0.09366	0.05985	0.2196	0.07451	...
8	13.00	21.82	87.50	519.8	0.12730	0.19320	0.18590	0.09353	0.2350	0.07389	...
9	12.46	24.04	83.97	475.9	0.11860	0.23960	0.22730	0.08543	0.2030	0.08243	...

overview

DATA PROCESS

```python
#Show basic information of data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   mean radius             569 non-null    float64
 1   mean texture            569 non-null    float64
 2   mean perimeter          569 non-null    float64
 3   mean area               569 non-null    float64
 4   mean smoothness         569 non-null    float64
 5   mean compactness        569 non-null    float64
 6   mean concavity          569 non-null    float64
 7   mean concave points     569 non-null    float64
 8   mean symmetry           569 non-null    float64
 9   mean fractal dimension  569 non-null    float64
 10  radius error            569 non-null    float64
 11  texture error           569 non-null    float64
 12  perimeter error         569 non-null    float64
 13  area error              569 non-null    float64
 14  smoothness error        569 non-null    float64
 15  compactness error       569 non-null    float64
 16  concavity error         569 non-null    float64
 17  concave points error    569 non-null    float64
 18  symmetry error          569 non-null    float64
 19  fractal dimension error 569 non-null    float64
 20  worst radius            569 non-null    float64
 21  worst texture           569 non-null    float64
 22  worst perimeter         569 non-null    float64
 23  worst area              569 non-null    float64
 24  worst smoothness        569 non-null    float64
 25  worst compactness       569 non-null    float64
 26  worst concavity         569 non-null    float64
 27  worst concave points    569 non-null    float64
 28  worst symmetry          569 non-null    float64
 29  worst fractal dimension 569 non-null    float64
 30  target                  569 non-null    int64
dtypes: float64(30), int64(1)
memory usage: 137.9 KB
```

Explore Data and Analysis (EDA)

EXPLORE DATA AND ANALYSIS (EDA)

```
#View a stastistical description of the data
df.describe()
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800

Data MODELLING

Split Data

```
from sklearn.model_selection import train_test_split

#Split data
x_train, x_test, y_train, y_test = train_test_split(df_x, df_y, test_size = 0.2, random_state = 42)
```

Create & Train

```
from sklearn.ensemble import RandomForestClassifier

#Create and train randomly
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(x_train, y_train)
```

RandomForestClassifier

```
RandomForestClassifier(random_state=42)
```

DATA MODELLING

```python
from sklearn.metrics import accuracy_score, classification_report

#Make predictions
y_pred = model.predict(x_test)
accuracy = accuracy_score(y_test, y_pred)

#Evaluate the model
print("Classification Report: ")
print(classification_report(y_test, y_pred))
print(f"Accuracy: {accuracy * 100:.2f}%")
```

Accuracy

```
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.93      0.95        43
           1       0.96      0.99      0.97        71

    accuracy                           0.96       114
   macro avg       0.97      0.96      0.96       114
weighted avg       0.97      0.96      0.96       114

Accuracy: 96.49%
```

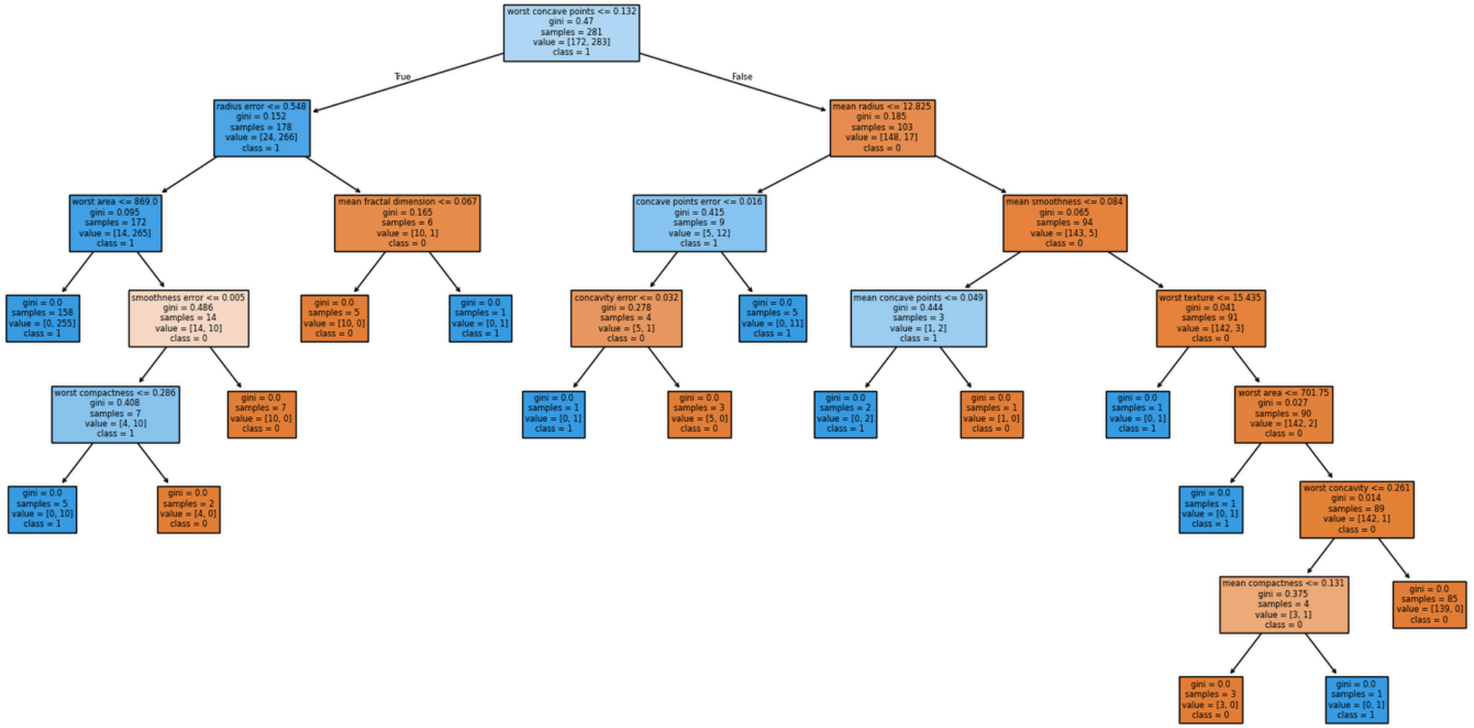
DATA VISUALIZATION

```
import matplotlib.pyplot as plt
import seaborn as sns

#
sns.countplot(x='target', data=df)
plt.title('Count of Target Classes')
plt.xlabel('Target Class')
plt.ylabel('Count')
plt.show()
```

DATA VISUALIZATION

```
import matplotlib.pyplot as plt
from sklearn import tree

plt.figure(figsize=(20, 10))
tree.plot_tree(model.estimators_[0], feature_names=df_x.columns, class_names=['0', '1'], filled=True)
plt.show()
```

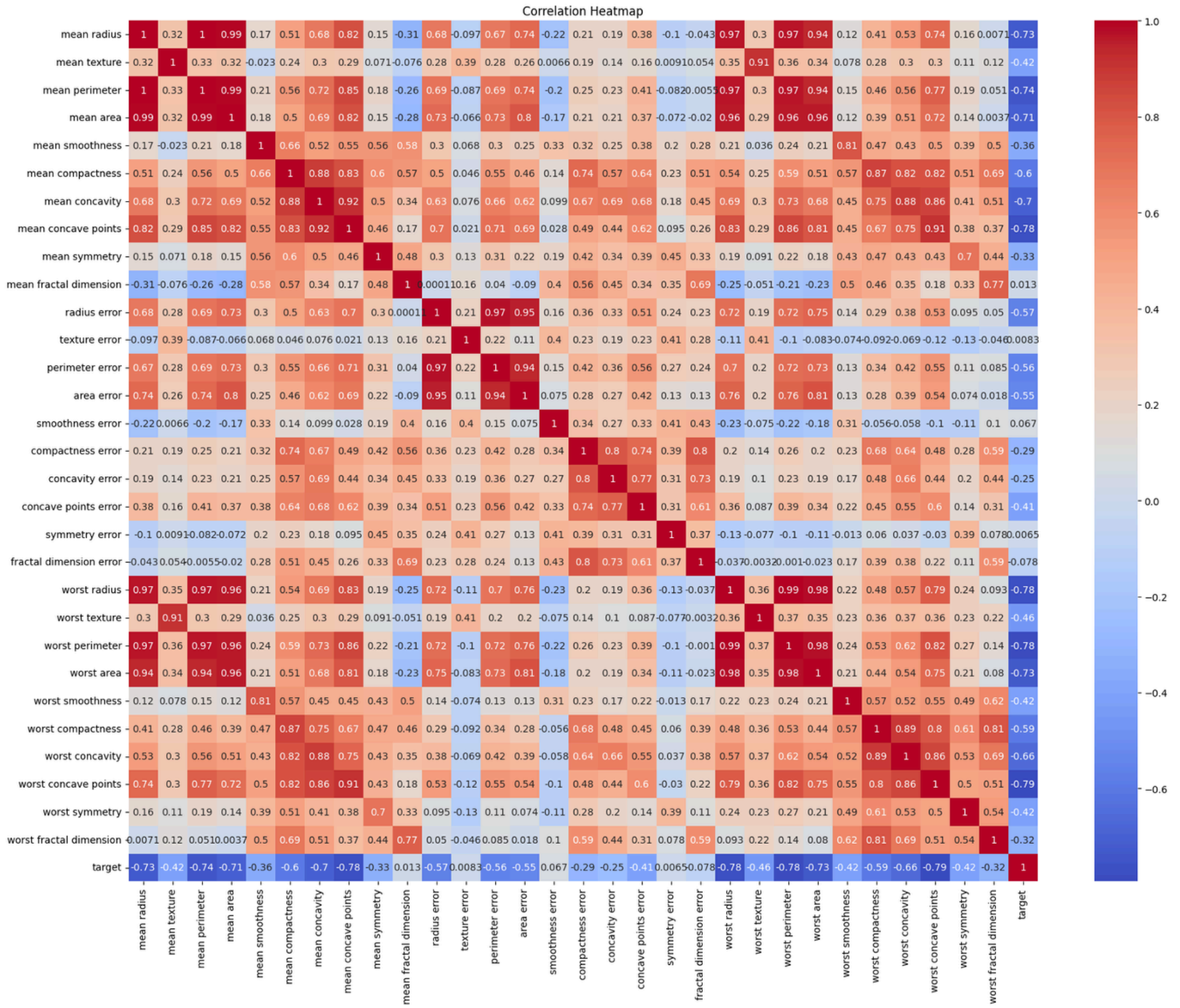
Decision Tree



DATA VISUALIZATION

```
corr = df.corr()
plt.figure(figsize=(21, 16))
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

Correlation



THANK YOU

https://github.com/raq00/python-of-ds

https://www.linkedin.com/in/rofiqo-azzahra