Classification problem with the example:

Structured guide to complete the assignment: I will break it down, step by step, from choosing a dataset to developing the three models, namely Logistic Regression, k-Nearest Neighbors, and Decision Tree, followed by the evaluation of these models and then documenting the results.  
  
Step-by-Step Guide  
1. Selection of Dataset  
An appropriate dataset for classification should be selected. Following are a few examples that one can use:  
  
UCI Machine Learning Repository: This repository contains a list of datasets; among them, Iris, Wine, and Breast Cancer.  
Kaggle datasets: You might get examples on student performance, customer segmentation, etc.  
I am going to use, for this example, the Breast Cancer dataset from the UCI repository. This dataset fits well into a binary classification problem. The features describe characteristics of tumors, and the target variable describes whether the tumor is malignant or benign.

2. Model Development:

Code for the classification problem

# Import necessary libraries

import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LogisticRegression

from sklearn.neighbors import KNeighborsClassifier

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import classification\_report

# Load the dataset (Breast Cancer dataset from sklearn)

from sklearn.datasets import load\_breast\_cancer

data = load\_breast\_cancer()

df = pd.DataFrame(data.data, columns=data.feature\_names)

df['target'] = data.target

# Check the first few rows of the dataset

print(df.head())

# Split the dataset into training and testing sets

X = df.drop('target', axis=1)

y = df['target']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Standardize the features

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

# Initialize and train the Logistic Regression model

log\_model = LogisticRegression(max\_iter=1000)

log\_model.fit(X\_train, y\_train)

log\_pred = log\_model.predict(X\_test)

# Initialize and train the k-Nearest Neighbors model

knn\_model = KNeighborsClassifier(n\_neighbors=5)

knn\_model.fit(X\_train, y\_train)

knn\_pred = knn\_model.predict(X\_test)

# Initialize and train the Decision Tree model

dt\_model = DecisionTreeClassifier(random\_state=42)

dt\_model.fit(X\_train, y\_train)

dt\_pred = dt\_model.predict(X\_test)

# Evaluate the models

print("Logistic Regression Performance:\n", classification\_report(y\_test, log\_pred))

print("k-Nearest Neighbors Performance:\n", classification\_report(y\_test, knn\_pred))

print("Decision Tree Performance:\n", classification\_report(y\_test, dt\_pred))

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated