# How to build KNN Model

The part is to analyze the effectiveness of the k-nearest neighbors (KNN) algorithm in classifying loan defaulters based on their income and loan amount. The dataset used for this analysis consists of records containing information on individuals' income, loan amount, and whether they defaulted on their loan or not. The KNN algorithm is a popular choice for classification tasks due to its simplicity and effectiveness.

## Data Preprocessing:

Data preparation involves a series of essential tasks aimed at ensuring the accuracy, completeness, and relevance of data for analysis or machine learning models. These tasks encompass several key processes, including:

1)Data cleaning: This process addresses issues such as missing data, duplicated data, and outliers, which can adversely affect the quality of analysis or model training.

2)Data transformation: This step involves normalization, which scales the data to a standard range, ensuring that each feature contributes equally to the analysis or model.

3)Handling class imbalances: In situations where one class significantly outweighs the others, techniques such as resampling or adjusting class weights may be employed to address this imbalance.

4) Feature selection: Feature selection is a crucial step aimed at selecting a subset of relevant features from the original set of features in the dataset.

### Data Cleaning:

No data cleaning is required as there are no missing data, duplicated data, or outliers present in the dataset.

### Data Transformation:

Before building the model, it's essential to normalize the data since the unit of measurement might differ across features. This process ensures that all features contribute equally to the model. Here are the two steps involved:

Step 1: Visualize the distribution of each numerical column using a histogram to assess its normality. Look for a bell curve shape, which indicates a Gaussian distribution.

Step 2: Normalize the features based on the observed distribution. Since columns "Income" and "Loan Amount" exhibit non-Gaussian distributions, Min-Max Normalization will be applied to these columns. This technique scales the values between a specified range, typically 0 and 1, ensuring uniformity across features.

#Visualize the distribution of each numerical column in X

numeric\_columns = X.select\_dtypes(include=['float64', 'int64'])

numeric\_columns.hist(bins=20, figsize=(9, 6))

plt.suptitle('Histograms for Numerical Columns in X', y=0.95)

plt.show()

**A graph of a column with blue lines

Description automatically generated with medium confidence**

#Normalize the dataset

from sklearn.preprocessing import MinMaxScaler

scaler= MinMaxScaler().fit(X\_train) # the scaler is fitted to the training set

normalized\_X\_train= scaler.transform(X\_train) # the scaler is applied to the training set

normalized\_X\_test= scaler.transform(X\_test) # the scaler is applied to the test set

print('X train before Normalization')

print(X\_train[0:5])

print('\nX train after Normalization')

print(normalized\_X\_train[0:5])

**A screenshot of a computer

Description automatically generated**

### Handling class imbalances

There is no need to address class imbalances as the dataset does not exhibit any imbalance in class distribution.

### Feature Selection:

Feature selection is not required for this assignment as features and asked you to tag each row with yes for the default example and no for the not default examples.

## Data Tagging:

Tag (label) your data appropriately by typing yes or no under the column titled Default in the file loan4a2.csv based on the following rules.

Data tagging, also known as annotation or labeling, involves the process of appending tags to raw data. These tags serve as cues for machine learning models, guiding them on the desired target responses they are expected to predict.

A diagram of a financial flowchart

Description automatically generated A white paper with black text

Description automatically generated

## Methodology - K-Nearest Neighbors:

In this study, the KNN algorithm is applied to classify loan defaulters using the provided dataset. The dataset is first loaded into a panda DataFrame, and then the features (income and loan amount) and the target variable (default) are extracted. To ensure that the data is on a similar scale, it is normalized using Min-Max Normalization after dataset splitting.

The dataset is split into training and testing sets using a 70-30 split, with 70% of the data used for training and 30% for testing. The KNN classifier is then built using the training data, with k = 5 neighbors chosen based on the rule of thumb.

**Load Data**

# Ignore warnings to keep the output clean

import warnings

warnings.filterwarnings('ignore')

# Import libraries

from matplotlib.colors import ListedColormap

import matplotlib.pyplot as plt

from sklearn import datasets

import numpy as np

import pandas as pd

from sklearn import tree

from sklearn import metrics

# Command to display plots inline in Jupyter notebooks

%matplotlib inline

#loading a csv file

import pandas as pd

loan = pd.read\_csv("loan4a2.csv")

# Extract features (X) and target variable (y) from the dataset

X = loan.iloc[:, :-1]  # Extracts all rows and all columns except the last one

y = loan.iloc[:, -1]  # Extracts all rows and the last column

# Print the unique class labels in the target variable y

print('Class labels:', np.unique(y))

**Split data into train and test**

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=0)

X\_train= np.asarray(X\_train)

y\_train= np.asarray(y\_train)

X\_test= np.asarray(X\_test)

y\_test= np.asarray(y\_test)

#Print the size of training and test datasets

print(f'training set size: {X\_train.shape[0]} samples \ntest set size: {X\_test.shape[0]} samples')

#Normalize the dataset

from sklearn.preprocessing import MinMaxScaler

scaler= MinMaxScaler().fit(X\_train) # the scaler is fitted to the training set

normalized\_X\_train= scaler.transform(X\_train) # the scaler is applied to the training set

normalized\_X\_test= scaler.transform(X\_test) # the scaler is applied to the test set

print('X train before Normalization')

print(X\_train[0:5])

print('\nX train after Normalization')

print(normalized\_X\_train[0:5])

**Build the KNN classifier and Generate the evaluation metrics**

from sklearn.neighbors import KNeighborsClassifier

clf = KNeighborsClassifier(n\_neighbors=5, p=2, metric='minkowski')  #using Rule of thumb to choose k = 5

clf.fit(normalized\_X\_train, y\_train)

print("Train - Accuracy :", metrics.accuracy\_score(y\_train, clf.predict(normalized\_X\_train)))

print("Train - Confusion matrix :",metrics.confusion\_matrix(y\_train, clf.predict(normalized\_X\_train)))

print("Train - classification report :", metrics.classification\_report(y\_train, clf.predict(normalized\_X\_train)))

print("Test - Accuracy :", metrics.accuracy\_score(y\_test, clf.predict(normalized\_X\_test)))

print("Test - Confusion matrix :",metrics.confusion\_matrix(y\_test, clf.predict(normalized\_X\_test)))

print("Test - classification report :", metrics.classification\_report(y\_test, clf.predict(normalized\_X\_test)))

## Analysis and findings

A screenshot of a computer screen

Description automatically generated

The KNN classifier achieved perfect accuracy of 100% on both the training and testing sets, indicating that it correctly classified all instances in the dataset. The confusion matrices and classification reports for both the training and testing sets further confirm the high accuracy of the model, with precision, recall, and F1-score values of 1.00 for both classes (default and non-default).

These findings suggest that the KNN algorithm is highly effective in classifying loan defaulters based on their income and loan amount, achieving perfect accuracy on the provided dataset. However, further evaluation on larger and more diverse datasets is recommended to validate the robustness of the model. Additionally, exploring other classification algorithms and comparing their performance could provide further insights into the best approach for this task.

## Reference

Kudan, N. (2023, April 27). What is the purpose of tagging data? Toloka. Retrieved from https://toloka.ai/blog/machine-learning-tagging-text/#:~:text=Data%20tagging%20(sometimes%20called%20annotation,tagged%20info%20constitutes%20a%20dataset.

Kumar, V. G. R. (2024, January 6). Standardization and Normalization Techniques in Machine Learning: StandardScaler(), MinMaxScaler(), Normalizer()&RobustScaler(). Medium. https://medium.com/@vinodkumargr/07-standardization-and-normalization-techniques-in-machine-learning-standardscaler-3890a89bddbf