

Loan Prediction

**Artificial Intelligence project**



May 31, 2022

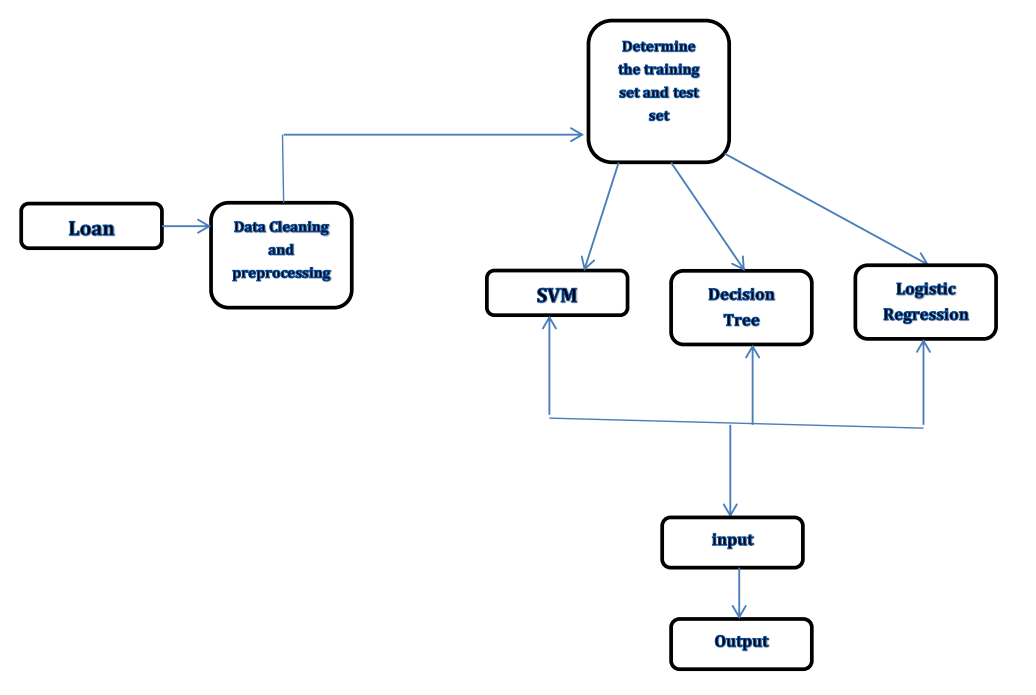
faculty of computer science

Ain Shams University

**TA: Verena Nashaat**

**Department: Main Stream**

* **Description:**
* We are designing a Loan Prediction system that predict if the user can take loan or not from the given features: Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History and others.
* So using the training dataset we will train our model and try to predict our target column that is “**Loan Status**” on the test dataset.
* **System Architecture:**

****

* **Algorithms:**
* Logistic Regression
* SVM (Support Vector Machine)
* Decision Tree
* Data Scaling
* KNN
* Random Forest
* **Dataset:**
* Train and test dataset would have the same columns except for the target column that is “**Loan Status**”.
* We have **614 rows** and **13 columns** in our dataset.
* **Python Libraries:**
* **Numpy** 🡪 useful for making arrays and processing
* **Pandas** 🡪 useful for preprocessing
* **Seaborn** 🡪 useful for making graphs and plots
* **Sklearn** 🡪 is a free machine learning library for Python. It features various algorithms like support vector machine, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy.
* **train\_test\_split 🡪** to split the data set into test and train samples, we also use different algorthims(SVM,Logisitic regression ,KNN, Decision Tree )
* **accuracy\_score** 🡪 to make predictions and compare their performance.
* **StandardScaler** 🡪 needed in data preprocessing specifically in data scaling.
* **confusion\_matrix** 🡪 to evaluate the accuracy of a classification.
* **sklearn.metrics** 🡪 implements functions assessing prediction error for specific purposes. These metrics are utility functions to measure classification performance
* pyplot 🡪 makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.
* **Preprocessing:**
* **Data Cleaning:** Before building our models, we need to make sure that the dataset is **clean** and ready-to-use.
* We Identified **missing values** in each column
* Nearly **21%** of data is missing, can't risk losing all of the data, so replacing the null values with average values will be a suitable solution.
* for numerical terms 🡪 **Mean,** for categorical terms 🡪 **Mode**
* **Label Encoding**: We used pandas function called **Replace** to replace values of “**Loan Status**” column to numerical values (0, 1), it will be helpful in our processing.
* We want to see what the different values in “**Dependents**” column are.
* We found four different values (**0, 1, 2, 3+**), then we will replace **3+** with **4** to be more accurate.
* **Data Visualization:** We need to find **relationship** between various columns in the dataset.
* We will replace **text format values** to **numerical values** in other columns such that our model can understand it.
* **separating the data and label:** We will drop “**Loan\_ID**” and “**Loan Status**”
* Then, we will store “**Loan Status**” label column in a separate variable.
* **Train Test Splitting:** We splitted the data into training data and test data, train data 🡪 **x\_train** variable, labels of train data 🡪 **y\_train** variable, all data except label variable 🡪 **x\_test** variable, corresponding label 🡪 **y\_test**
* Test data will be **10%** of the original data
* We will stratify our data based on Y (0’s and 1’s of Loan\_Status’s are splitted equally).
* Random\_state: Controls the shuffling applied to the data before applying the split.
* **Data Scaling:**
* Scaling is a technique of bringing down the values of all the independent features of our dataset on the same scale.
* It also makes training faster and improves performance.
* We used scaling not normalization because it fits all data while normalization fits only normally distributed data.

* **Data Selection:**
* Feature selection is the process of isolating the most consistent, non- redundant, and relevant features to use in model construction.
* Feature Methodically reducing the size of dataset is important as the size and variety of datasets continue to grow.
* The main goal of feature selection is to improve the performance of a predictive model and reduce the computational cost of modeling.
* **Logistic Regression:**
* This type of statistical model (also known as logit model) is often used for **classification** and **predictive** analytics. Logistic regression estimates the probability of an event occurring, such as **voted or didn’t vote**, based on a given dataset of **independent variables**. Since the outcome is a probability, the dependent variable is bounded between **0 and 1**.In logistic regression, a logit transformation is applied on the odds—that is, the probability of **success** divided by the probability of **failure**.
* The classification problem in logistic regression is a **discrete problem**.
* **SVM:**
* A support vector machine (SVM) is a **supervised** machine learning model that uses classification algorithms for **two-group** **classification problems**.
* We used **linear** SVM to split it into two categories.
* After giving an SVM model sets of **labeled training data** for each category, they’re able to categorize new text.
* After splitting the data to training data and test data we **check the accuracy** of SVM model on both so the new input will easily predicted using SVM classifier.
* We found that the accuracy of training data **almost equal** to accuracy of test data which means that our model **is not over fitted**.
* **Decision Tree:**
* Decision Trees are a type of **Supervised** Machine Learning where the data is continuously split according to a certain parameter.
* The tree can be explained by two entities, namely **decision nodes** and **leaves**.
* The leaves are the decisions or the final outcomes.
* The decision nodes are where the data is split.
* **KNN:**
* **K-Nearest Neighbors** (KNN) is a **standard** machine-learning method that has been extended to large-scale data mining efforts.
* The idea is that one uses a **large amount of training data**, where each data point is characterized by a set of variables.
* Conceptually, each point is plotted in a high-dimensional space, where each axis in the space corresponds to an individual variable.
* When we have a new (test) data point, we want to find out the K nearest neighbors that are closest (ie, most “**similar**” to it). The number K is typically chosen as the square root of N, the total number of points in the training data set.
* n\_neighbors : We pass the K value for this parameter to find the K nearest neighbors that are closest and predict the value depending on them.
* **Random Forest:**
* Random Forest is a popular machine learning algorithm that belongs to the **supervised** learning technique.
* It can be used for both **Classification** and **Regression** problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to **solve a complex problem** and to improve the **performance** of the model.
* **Classification report:**
* **Recall:** shows what the percentage of actual people take loan was correctly identified.
* **Precision:** shows what percentage of people take loan identifications was actually correct.
* **F1 Score:** A combination of precision and recall.
* **Support: The** number of samples each metric was calculated on.
* **Mean square error:**
* The mean squared error (MSE) tells you how close a regression line is to a set of points.
* It does this by taking the distances from the points to the regression line (these distances are the “errors”) and squaring them.
* The squaring is necessary to remove any negative signs.
* It also gives more weight to larger differences.
* It’s called the mean squared error as you’re finding the average of a set of errors.
* The lower the MSE, the better the forecast.
* **Confusion matrix:**
* A confusion matrix is a technique for summarizing the performance of a classification algorithm.
* **Model Accuracy:**

|  |  |  |
| --- | --- | --- |
| Models | Accuracy on training data | Accuracy on test data |
| Logistic Regression | **0.81** | **0.79** |
| SVM | **0.8** | **0.8** |
| Decision Tree | **0.82** | **0.79** |
| KNN | **0.8** | **0.82** |
| Random Forest | **0.88** | **0.66** |