Faculty of Media Engineering and Technology
German University in Cairo
Dr. Nourhan Ehab

CSEN 903: Advanced Computer Lab, Winter 2025
Lab 2 Manual: Machine Learning Foundations

Part 1: Read and Watch Before the Lab

In the previous lab, we focused on the **data cleaning and preprocessing** phase. This step was essential to prepare the Titanic dataset so that it could be effectively used in machine learning (ML) models, an important stage since well-prepared data often has a greater impact on model performance than the choice of algorithm itself.

Machine learning is at the heart of modern AI systems, and it comes in different flavors, including some major ones like:

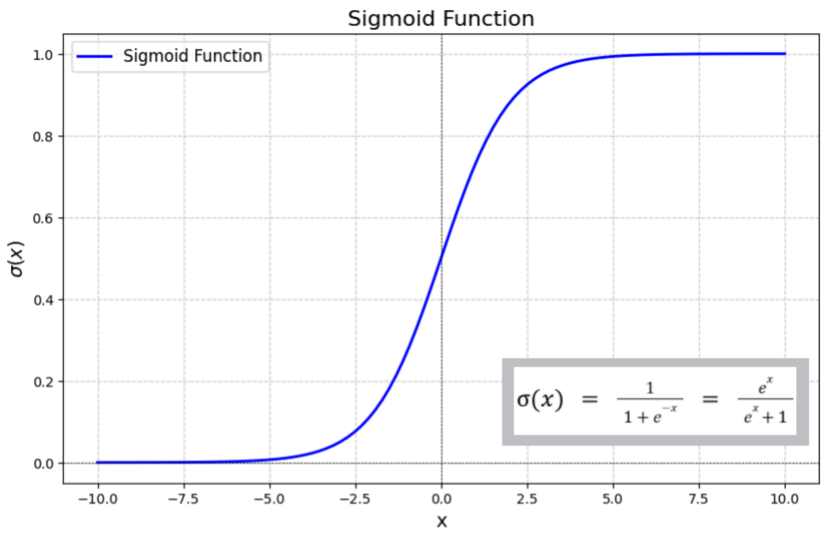
- **Supervised learning:** models learn from labeled data (input + expected output).
- **Unsupervised learning:** models find hidden patterns or groupings in unlabeled data.
- **Semi-supervised learning:** models learn from a small amount of labeled data combined with a large amount of unlabeled data, which is especially useful when labeling is expensive or time-consuming.

In this lab, we will focus on **supervised learning,** which can be used for 2 types of problems:

- **Classification:** where a category or discrete label is predicted (e.g., whether this email is spam or not? or which digit (0-9) is in an image).
- **Regression:** where a continuous value is predicted (e.g., predicting house prices, or a person's income).

We will be dealing with a **classification** problem, where the goal is to predict which category a new observation belongs to. Classification problems like in this lab can be approached using multiple ways as well, one of which is called **Logistic Regression.**

Although the term *regression* usually refers to modeling how a continuous dependent variable changes with respect to independent variables (as in linear regression), **logistic regression is actually a classification method**. The “regression” in its name comes from the fact that it models the probability of belonging to a class, not a continuous outcome itself. It can use functions like the **sigmoid function** to map predictions to values between 0 and 1, which can then be interpreted as probabilities. A threshold is applied to decide the final class label.



Sigmoid Function

Tools:

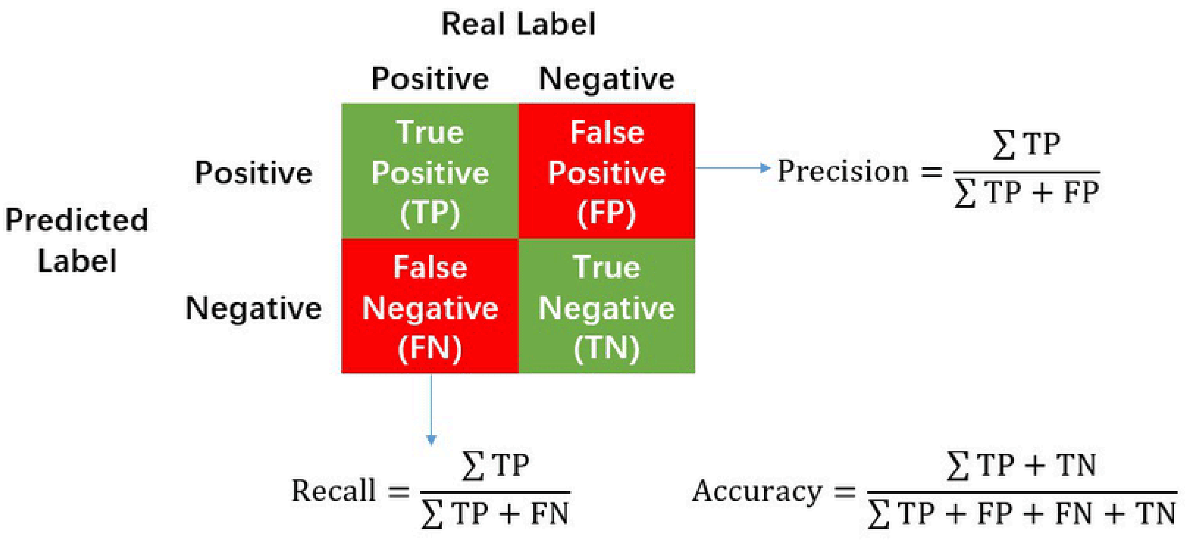
Scikit-learn is a popular Python library for machine learning. It provides easy-to-use tools for preprocessing data, training models, and evaluating their performance. In this lab, we'll use it to build and test a logistic regression model on the Titanic dataset.

Evaluation Metrics:

Evaluation Metrics are essential in machine learning because they tell us how well a model is performing beyond just looking at predictions. Different metrics highlight different strengths and weaknesses, helping us attain comprehensive insight about the model and avoid misleading conclusions. A few examples we will be inspecting are:

- **Accuracy** – The proportion of all predictions that the model classified correctly.
- **Precision** – The proportion of positive predictions that were actually correct, in simpler terms, of everything the model predicted as positive, how many were truly positive? Or "When the model says yes, how often is it right?"
- **Recall** – The proportion of actual positives that were correctly identified by the model, in simpler terms, of all the true positives in the data, how many did the model actually find? Or "When something is actually yes, how often does the model catch it?"

- **F1 Score** – The harmonic mean of precision and recall, balancing the two metrics.


Real Label
Positive
Negative
Predicted Label
Positive
Negative
True Positive (TP)
False Positive (FP)
False Negative (FN)
True Negative (TN)
$\text{Precision} = \frac{\sum TP}{\sum TP + FP}$
$\text{Recall} = \frac{\sum TP}{\sum TP + FN}$
$\text{Accuracy} = \frac{\sum TP + TN}{\sum TP + FP + FN + TN}$

Accuracy vs Recall

Note on metrics:

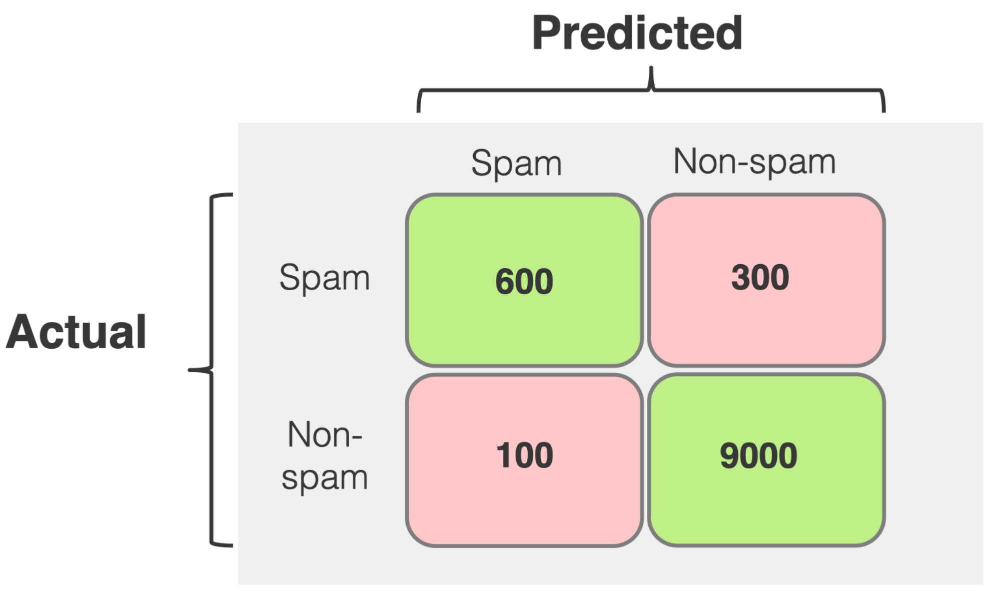
Consider Accuracy, a popular metric in ML and may seem like a great one to utilize; however, it may not be sufficient to draw conclusions. Take for example, a model that classifies cats and dogs; however, the model is so naive that it assumes "Dog" each time. Thus, when presented with a test data of 8 dogs and 2 cats, acc=80%

a test data of 3 dogs and 7 cats, acc=30% and so on, emphasizing that accuracy is not the best metric to be used on its own in some circumstances where the test dataset is not balanced, and that follows for the rest of the metrics, which have different trade-offs for their usage.

Confusion Matrix:

Precision and recall can be visualized using a **confusion matrix,** which is a table that summarizes how well a **classification model** performs by comparing its **predicted labels** to the **true labels**.

It shows where the model is correct and where it makes different kinds of mistakes.



Confusion matrix of email classification

By the end of this lab, you will learn to:

1. Load a dataset and split it into training and testing sets
2. Train classification models using Scikit-learn
3. Evaluate models using performance metrics like:
    - Accuracy (overall correctness)
    - Precision (how many predicted positives were correct)
    - Recall (how many actual positives were found)
    - F1 Score (balance between precision and recall)

We will use the **Titanic dataset** again, now focusing on predicting survival using selected features.

Required Readings:

1. Scikit-Learn User Guide – Supervised Learning
2. Train-Test Split Documentation
3. Metrics and Scoring

4. Logistic Regression
5. Overview of Accuray, Precision and Recall

Required Watch List:

1. Supervised and unsupervised Learning overview
2. Logistic Regression

Part 2: In-Lab Task: Notebook Link

1. Set Up Your Environment.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score, confusion_matrix, classification_report
```

2. Use the cleaned Titanic dataset from Lab 1 (or re-clean using the code below).

```
df = pd.read_csv('titanic.csv')

# Basic cleaning steps
df['Age'] = df['Age'].fillna(df['Age'].median())
df['Embarked'] = df['Embarked'].fillna(df['Embarked'].mode()[0])
df['Sex'] = df['Sex'].map({'male': 0, 'female': 1})
df = pd.get_dummies(df, columns=['Embarked'], drop_first=True)

df = df.drop(columns=['Cabin', 'Name', 'Ticket'])  # Remove irrelevant
columns
```

3. Define features and labels.

```
X = df[['Pclass', 'Sex', 'Age', 'Fare', 'SibSp', 'Parch']]  # You can modify
this
y = df['Survived']
```

4. Split the dataset into training and testing.

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

5. Train a classification model.

```
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
```

6. Evaluate the model.

```
y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("Precision:", precision_score(y_test, y_pred))
print("Recall:", recall_score(y_test, y_pred))
print("F1 Score:", f1_score(y_test, y_pred))

print("\nClassification Report:")
print(classification_report(y_test, y_pred))

print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))
```

7. Visualize the confusion matrix.

```
import seaborn as sns
import matplotlib.pyplot as plt
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt="d", cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```

Part 3: Graded Task

Task Description:

1. Use the **Titanic dataset** to train a classifier (e.g., Logistic Regression) that predicts survival.
2. Try using at least **two different feature sets** and compare performance.
3. Report the following metrics for both models:

    - Accuracy
    - Precision
    - Recall
    - F1 Score

4. Display a **confusion matrix** using `seaborn`.

Deliverables:

Submit a notebook with the completed above task that **must** include:

- The metrics of **each** model
    - Accuracy
    - Precision
    - Recall
    - F1 Score
- A visualization for the confusion matrix of **each** model

Submit an accessible notebook link containing all the completed tasks listed above via the form. The submission **deadline** is by the end of your lab.