****

**MAWLANA BHASHANI SCIENCE AND TECHNOLOGY UNIVERSITY**

**Lab Report**

**Department:** Computer Science and Engineering

**Course Title:**  Artificial Intelligence and Machine Learning Lab

**Course Code:** CSE 4102

**Report Tittle:** Applying various Machine learning algorithm in the dataset.

**Submitted By:**

**Shamim Sorkar (CE19044)**

**Submitted To:**

**Lubna Yasmin Pinky**

Assistant Professor,

Dept. of CSE, MBSTU.

**Name of Experiment:** There is a dataset given which contains the information of various users obtained from the social networking sites. There is a car making company that has recently launched a new SUV car. So, the company wanted to check how many users from the dataset, wants to purchase the car.

**Introduction:** In Machine Learning algorithm, we firstly train the system with train data then predict the output. Supervised learning algorithm mainly used for classification problems. We start by preprocessing the data, which involves reading the dataset, handling missing values, and scaling the features using StandardScaler. Then, we split the data into training and test sets using train\_test\_split. Next, we define a function to evaluate the performance of classification models based on accuracy, precision, recall, F1-score, and confusion matrix. We implement several classification algorithms such as Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), and k-Nearest Neighbors (kNN).

**Data Description:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| User\_ID | Gender | Age | EstimatedSalary | Purchased |
| 15624510 | Male | 19 | 15000 | 0 |
| 15810944 | Male | 35 | 20000 | 0 |
| 15668575 | Female | 26 | 43000 | 0 |
| 15603246 | Female | 27 | 57000 | 0 |
| 15804002 | Male | 19 | 76000 | 0 |
| 15728773 | Male | 27 | 58000 | 0 |
| 15598044 | Female | 27 | 84000 | 0 |
| 15694829 | Female | 32 | 150000 | 1 |
| 15600575 | Male | 25 | 33000 | 0 |
| 15727311 | Female | 35 | 65000 | 0 |
| 15570769 | Female | 26 | 80000 | 0 |
| 15606274 | Female | 26 | 52000 | 0 |
| 15746139 | Male | 20 | 86000 | 0 |
| 15704987 | Male | 32 | 18000 | 0 |
| 15628972 | Male | 18 | 82000 | 0 |
| 15697686 | Male | 29 | 80000 | 0 |
| 15733883 | Male | 47 | 25000 | 1 |
| 15617482 | Male | 45 | 26000 | 1 |
| 15704583 | Male | 46 | 28000 | 1 |
| 15621083 | Female | 48 | 29000 | 1 |
| 15649487 | Male | 45 | 22000 | 1 |
| 15736760 | Female | 47 | 49000 | 1 |

**Source Code:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler

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

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score, confusion\_matrix

from sklearn.linear\_model import LogisticRegression

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.svm import SVC

from sklearn.neighbors import KNeighborsClassifier

from matplotlib.colors import ListedColormap

data = pd.read\_csv("dataset.csv")

data.dropna(inplace=True)

X = data[['Age', 'EstimatedSalary']]

y = data['Purchased']

scaler = StandardScaler()

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

def evaluate\_model(model, X\_test, y\_test):

y\_pred = model.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

f1 = f1\_score(y\_test, y\_pred)

conf\_matrix = confusion\_matrix(y\_test, y\_pred)

f1 = precision = recall = 1

if conf\_matrix.shape[0] < 2 or conf\_matrix.shape[1] < 2:

tp = fp = fn = tn = 0

else:

tp = conf\_matrix[0, 0]

fp = conf\_matrix[0, 1]

fn = conf\_matrix[1, 0]

tn = conf\_matrix[1, 1]

precision = tp / (tp + fp) if tp + fp > 0 else 0

recall = tp / (tp + fn) if tp + fn > 0 else 0

f1 = (2 \* precision \* recall) / (precision + recall)

return accuracy, precision, recall, f1, conf\_matrix

results = {}

confusion\_matrices = {}

log\_reg\_model = LogisticRegression()

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

log\_reg\_results\_train = evaluate\_model(log\_reg\_model, X\_train, y\_train)

log\_reg\_results\_test = evaluate\_model(log\_reg\_model, X\_test, y\_test)

results['Logistic Regression'] = log\_reg\_results\_test[:-1]

confusion\_matrices['Logistic Regression'] = log\_reg\_results\_test[-1]

decision\_tree\_model = DecisionTreeClassifier()

decision\_tree\_model.fit(X\_train, y\_train)

decision\_tree\_results\_train = evaluate\_model(decision\_tree\_model, X\_train, y\_train)

decision\_tree\_results\_test = evaluate\_model(decision\_tree\_model, X\_test, y\_test)

results['Decision Tree'] = decision\_tree\_results\_test[:-1]

confusion\_matrices['Decision Tree'] = decision\_tree\_results\_test[-1]

random\_forest\_model = RandomForestClassifier(n\_estimators=100, random\_state=42)

random\_forest\_model.fit(X\_train, y\_train)

random\_forest\_results\_train = evaluate\_model(random\_forest\_model, X\_train, y\_train)

random\_forest\_results\_test = evaluate\_model(random\_forest\_model, X\_test, y\_test)

results['Random Forest'] = random\_forest\_results\_test[:-1]

confusion\_matrices['Random Forest'] = random\_forest\_results\_test[-1]

svm\_model = SVC(kernel='linear', random\_state=42)

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

svm\_results\_train = evaluate\_model(svm\_model, X\_train, y\_train)

svm\_results\_test = evaluate\_model(svm\_model, X\_test, y\_test)

results['SVM'] = svm\_results\_test[:-1]

confusion\_matrices['SVM'] = svm\_results\_test[-1]

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

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

knn\_results\_train = evaluate\_model(knn\_model, X\_train, y\_train)

knn\_results\_test = evaluate\_model(knn\_model, X\_test, y\_test)

results['kNN'] = knn\_results\_test[:-1]

confusion\_matrices['kNN'] = knn\_results\_test[-1]

for model\_name, conf\_matrix in confusion\_matrices.items():

print(f"\nConfusion Matrix for {model\_name} (Test set):\n{conf\_matrix}")

metrics\_df = pd.DataFrame(results, index=['Accuracy', 'Precision', 'Recall', 'F1-Score']).T

print("Test Set Results:\n", metrics\_df)

for model\_name, model in {

'Logistic Regression': log\_reg\_model,

'Decision Tree': decision\_tree\_model,

'Random Forest': random\_forest\_model,

'SVM': svm\_model,

'kNN': knn\_model

}.items():

plt.figure(figsize=(12, 5))

# Plot training set

plt.subplot(1, 2, 1)

x\_set\_train, y\_set\_train = X\_train, y\_train

x1\_train, x2\_train = np.meshgrid(np.arange(start=x\_set\_train[:, 0].min() - 1, stop=x\_set\_train[:, 0].max() + 1, step=0.01),

np.arange(start=x\_set\_train[:, 1].min() - 1, stop=x\_set\_train[:, 1].max() + 1, step=0.01))

plt.contourf(x1\_train, x2\_train, model.predict(np.array([x1\_train.ravel(), x2\_train.ravel()]).T).reshape(x1\_train.shape),

alpha=0.75, cmap=ListedColormap(('purple', 'green')))

plt.xlim(x1\_train.min(), x1\_train.max())

plt.ylim(x2\_train.min(), x2\_train.max())

for i, j in enumerate(np.unique(y\_set\_train)):

plt.scatter(x\_set\_train[y\_set\_train == j, 0], x\_set\_train[y\_set\_train == j, 1],

c=ListedColormap(('purple', 'green'))(i), label=j)

plt.title(f'{model\_name} (Training set)')

plt.xlabel('Age')

plt.ylabel('Estimated Salary')

plt.legend()

# Plot test set

plt.subplot(1, 2, 2)

x\_set\_test, y\_set\_test = X\_test, y\_test

x1\_test, x2\_test = np.meshgrid(np.arange(start=x\_set\_test[:, 0].min() - 1, stop=x\_set\_test[:, 0].max() + 1, step=0.01),

np.arange(start=x\_set\_test[:, 1].min() - 1, stop=x\_set\_test[:, 1].max() + 1, step=0.01))

plt.contourf(x1\_test, x2\_test, model.predict(np.array([x1\_test.ravel(), x2\_test.ravel()]).T).reshape(x1\_test.shape),

alpha=0.75, cmap=ListedColormap(('purple', 'green')))

plt.xlim(x1\_test.min(), x1\_test.max())

plt.ylim(x2\_test.min(), x2\_test.max())

for i, j in enumerate(np.unique(y\_set\_test)):

plt.scatter(x\_set\_test[y\_set\_test == j, 0], x\_set\_test[y\_set\_test == j, 1],

c=ListedColormap(('purple', 'green'))(i), label=j)

plt.title(f'{model\_name} (Test set)')

plt.xlabel('Age')

plt.ylabel('Estimated Salary')

plt.legend()

plt.show()

**Output:**

Confusion Matrix for Logistic Regression (Test set):

[[5]]

Confusion Matrix for Decision Tree (Test set):

[[3 2]

[0 0]]

Confusion Matrix for Random Forest (Test set):

[[3 2]

[0 0]]

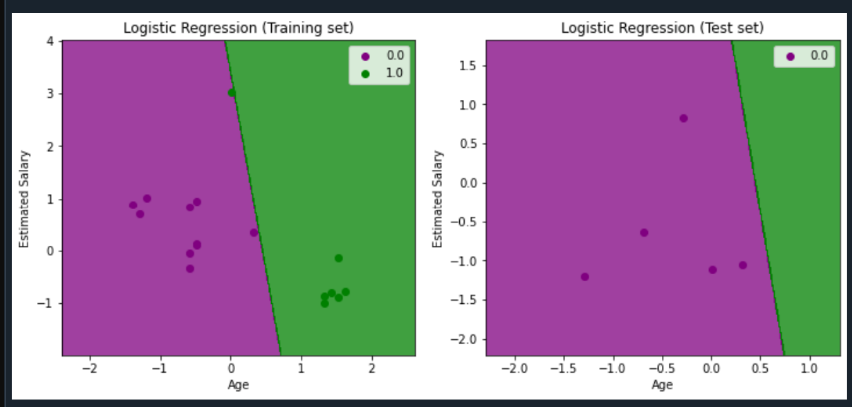
Confusion Matrix for SVM (Test set):

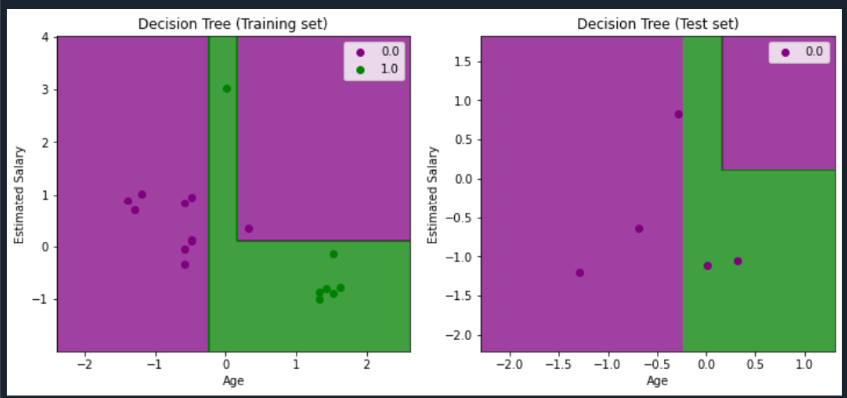
[[5]]

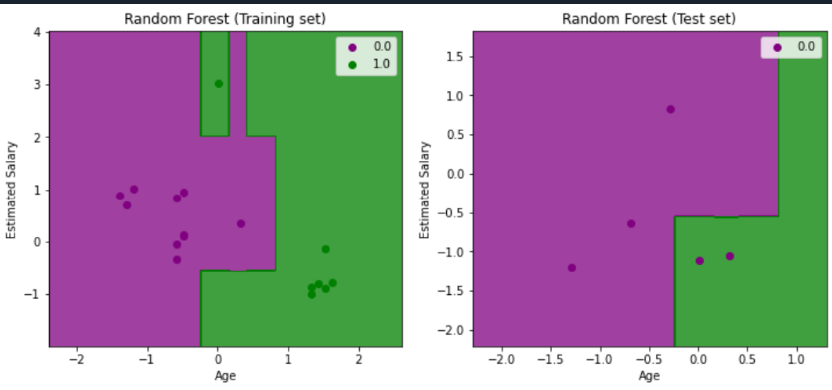
Confusion Matrix for kNN (Test set):

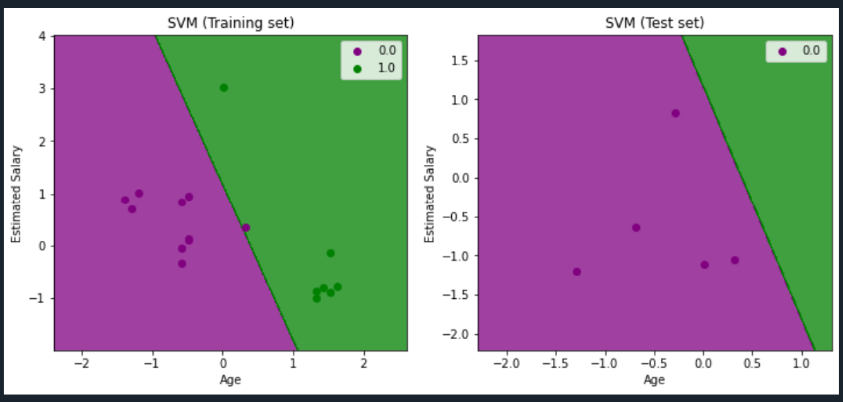
[[4 1]

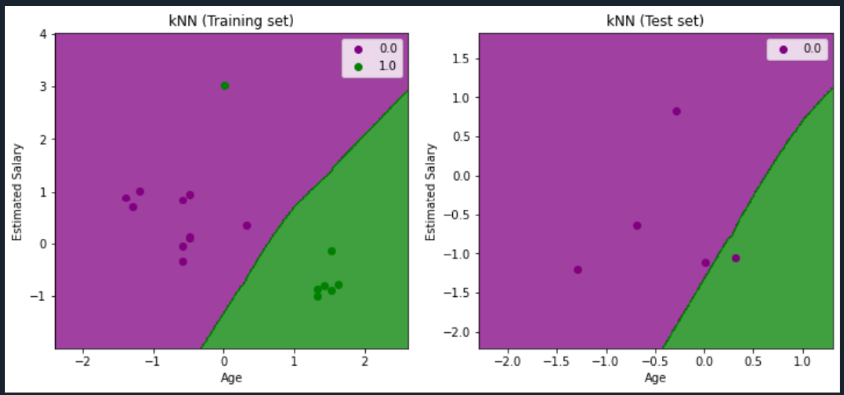
[0 0]]











**Test Set Results:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| **Logistic Regression** | 1 | 1 | 1.0 | 1.00 |
| **Decision Tree** | 0.6 | 0.6 | 1.0 | 0.75 |
| **Random Forest** | 0.6 | 0.6 | 1.0 | 0.75 |
| **SVM** | 1 | 1 | 1.0 | 1.00 |
| **kNN** | 0.8 | 0.8 | 1.0 | 0.89 |

**Conclusion:** In conclusion, we have successfully trained and evaluated multiple classification models on the given dataset. Each model's performance metrics have been calculated and compared, allowing us to assess their effectiveness in predicting customer purchases based on age and estimated salary.