

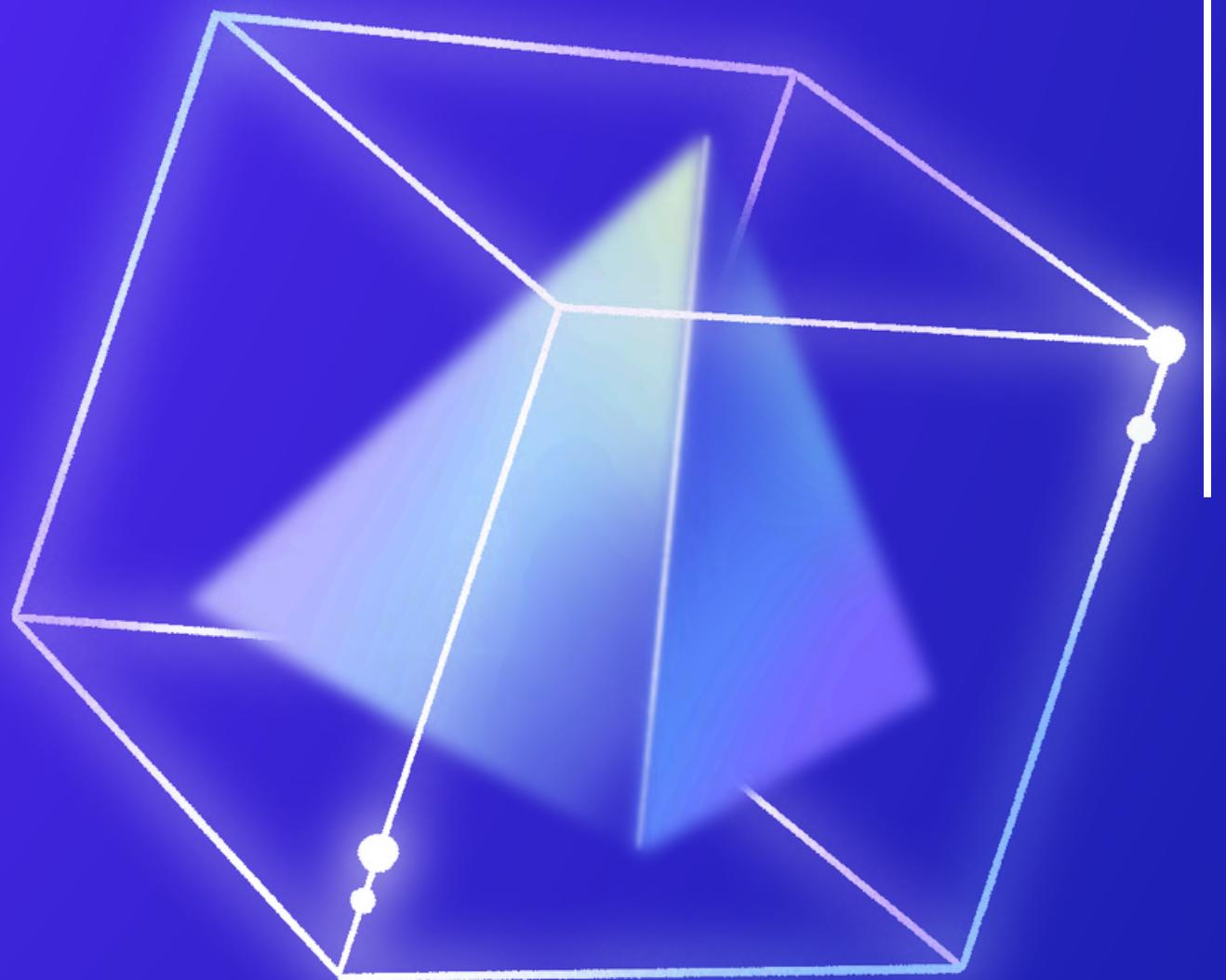
ADVANCED MACHINE LEARNING PROJECT



TABLE OF CONTENTS

- overview
- general information
- hyperparameters
- Cross-Validation
- results details
- GUI
- cover sheet

3
4
7
8
11
15
19



OVERVIEW

ANN & DT dataset link

[Bank Marketing Dataset](#) link

SVM dataset link

[Calories_Burnt_Prediction](#) link



GENERAL INFORMATION

Bank Marketing

Input variables:

bank client data:

1 - age (numeric)

2 - job : type of job (categorical)

3 - marital : marital status (categorical)

4 - education (categorical)

5 - default: has credit in default? (binary)

6 - balance: average yearly balance, in euros
numeric)

7 - housing: has housing loan? (binary)

8 - loan: has personal loan? (binary)

related with the last contact of the current campaign

9 - contact: contact communication type (categorical)

10 - day: last contact day of the month (numeric)

11 - month: last contact month of year (categorical)

12 - duration: last contact duration, in seconds (numeric)

other attributes:

13 - campaign: number of contacts performed during this campaign
and for this client (numeric)

14 - pdays: number of days that passed by after the client was last
contacted from a previous campaign (numeric)

15 - previous: number of contacts performed before this campaign and
for this client (numeric)

16 - poutcome: outcome of the previous marketing campaign
(categorical)

Output variable (desired target):

17 - y - has the client subscribed a term deposit? (binary)

GENERAL INFORMATION

Calories_Burnt

Input variables:

1. user_ID #not used
2. Gender (male or female)
3. Age
4. Height
5. Weight
6. Duration
7. Heart_Rate
8. Body_Temp

extract a new feature (bmr) used:

- Gender (male or female)
- Age
- Height
- Weight

Output variable (target):

1. Calories



PROJECT CYCLE



DATA
PRE_PROCESSING

DATA
VISUALIZATION

MODEL
TRAINING

GUI

HYPERPARAMETERS



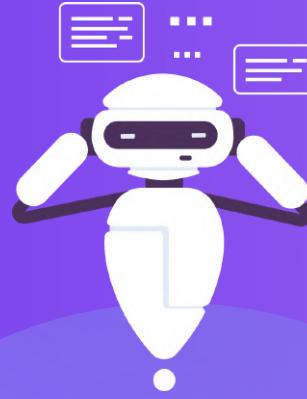
SVM

1. kernel=Radial Basis Function "rbf"
2. degree=2 , For polynomial kernels
3. epsilon=0.1 , It controls the tolerance for errors in the regression model.

ANN



1. Two hidden layers are used with 256 neurons in the first hidden layer and 32 neurons in the second hidden layer
2. ReLU activation function is used in the hidden layer , Sigmoid activation function is used in the output layer
3. Dropout layers with dropout rates of 0.5 and 0.3 are added after the first and second hidden layers, respectively.
4. Adam optimizer is used with a learning rate of 0.001
5. Binary cross-entropy loss function is used
6. EarlyStopping callback is applied with monitoring on validation loss, patience of 10 epochs
7. Training is performed for 100 epochs



DT

1. Decision Tree Hyperparameters:
 - min_samples_split: Minimum number of samples required to split a node (3).
 - max_depth: Maximum depth of the tree (3).
2. Grid Search Hyperparameters:
 - param_grid
 - min_samples_split: Range from 2 to 99.
 - max_depth: Values of 2, 3, and 4.
 - cv: Number of cross-validation folds (3).

CROSS-VALIDATION

1) SVM model

we used K fold cross validation

with 5 folds (num_folds = 5)

data is split into 5 equal parts, and the model is trained 5 times, each time using a different part as the validation set and the remaining parts as the training set

```
from sklearn.model_selection import cross_val_score, KFold
num_folds = 5
kf = KFold(n_splits=num_folds, shuffle=True, random_state=42)
cross_val_results = cross_val_score(regressor, X,Y, cv=kf)
print(f'Cross-Validation Results (Accuracy): {cross_val_results}')
print(f'Mean Accuracy: {cross_val_results.mean()}')
```

Cross-Validation Results (Accuracy): [0.98798532 0.98801809 0.98989697 0.9884515 0.98970963]
Mean Accuracy: 0.9888123018860295

GRIDSEARCHCV

1) DT model

cv=3 in GridSearchCV, 3-fold cross-validation during the hyperparameter search process. splitting the training data into 3 folds, training the model on 2 folds, and validating it on the remaining fold, repeating this process three times with different fold combinations.

```
] : from sklearn.model_selection import GridSearchCV

params = {'min_samples_split': list(range(2, 100)), 'max_depth': [2, 3, 4]}
grid_search_cv = GridSearchCV(DecisionTreeClassifier(random_state=42), params, verbose=1, cv=3)

grid_search_cv.fit(X_train, Y_train)

Fitting 3 folds for each of 294 candidates, totalling 882 fits
] : >      GridSearchCV          ⓘ ⓘ
] :   > estimator: DecisionTreeClassifier
] :     > DecisionTreeClassifier ⓘ
] : grid_search_cv.best_estimator_

] : >      DecisionTreeClassifier          ⓘ ⓘ
] :   DecisionTreeClassifier(max_depth=4, random_state=42)

] : from sklearn.metrics import accuracy_score

y_pred = grid_search_cv.predict(X_test)
accuracy_score(Y_test, y_pred)

] : 0.7917599641737573
```

PROJECT SCOPE



(Evaluate performance)

LOSS CURVE

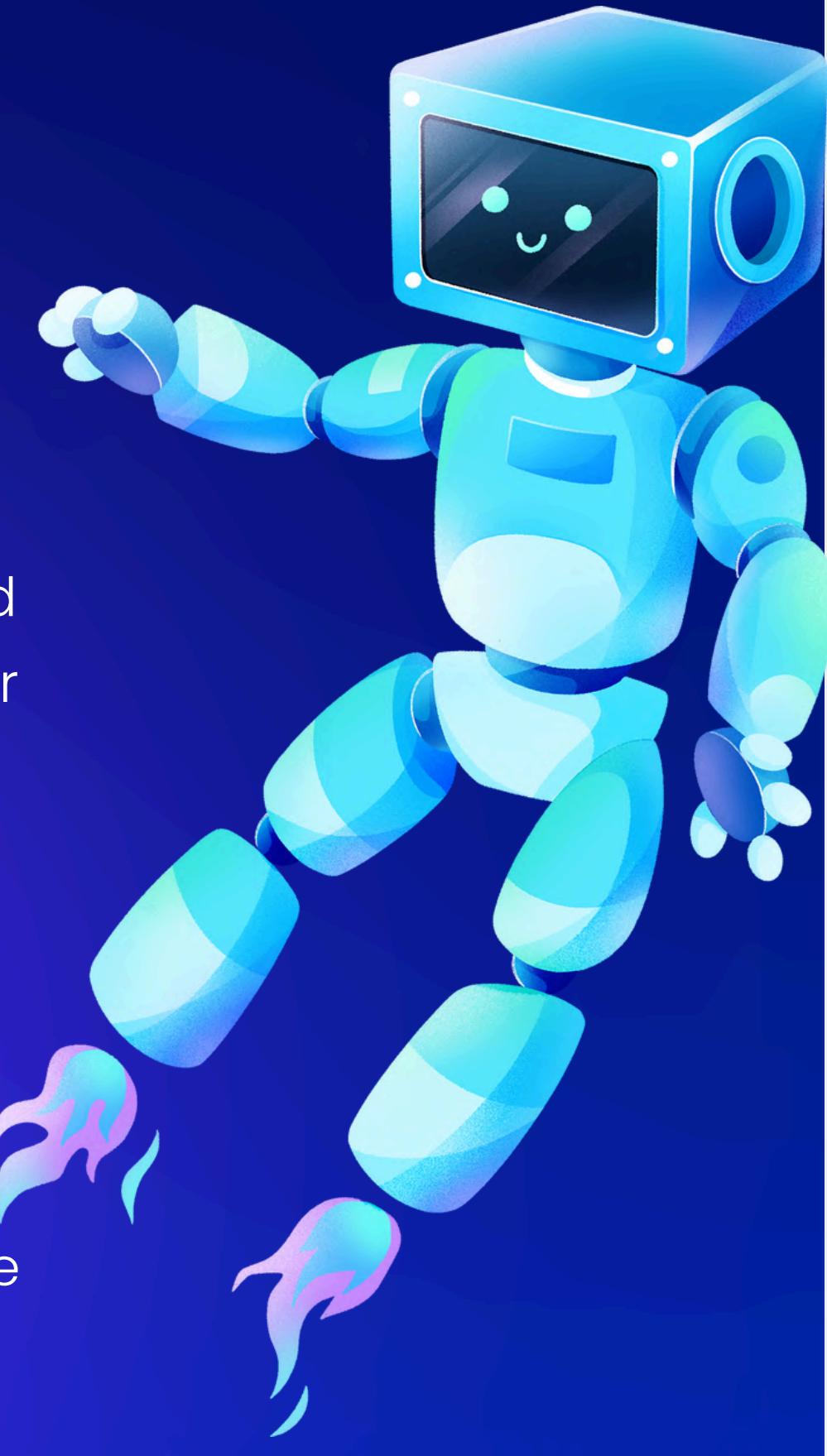
It helps visualize how the model's loss decreases during training and whether there are signs of overfitting (e.g., if the validation loss starts increasing while the training loss decreases).

ACCURACY

It helps visualize how the model's accuracy improves during training and whether there are signs of overfitting or underfitting based on the validation accuracy.

CONFUSION MATRIX

A confusion matrix is a useful tool for evaluating the performance of a classification model. It provides a summary of the model's predictions compared to the actual labels in the dataset.

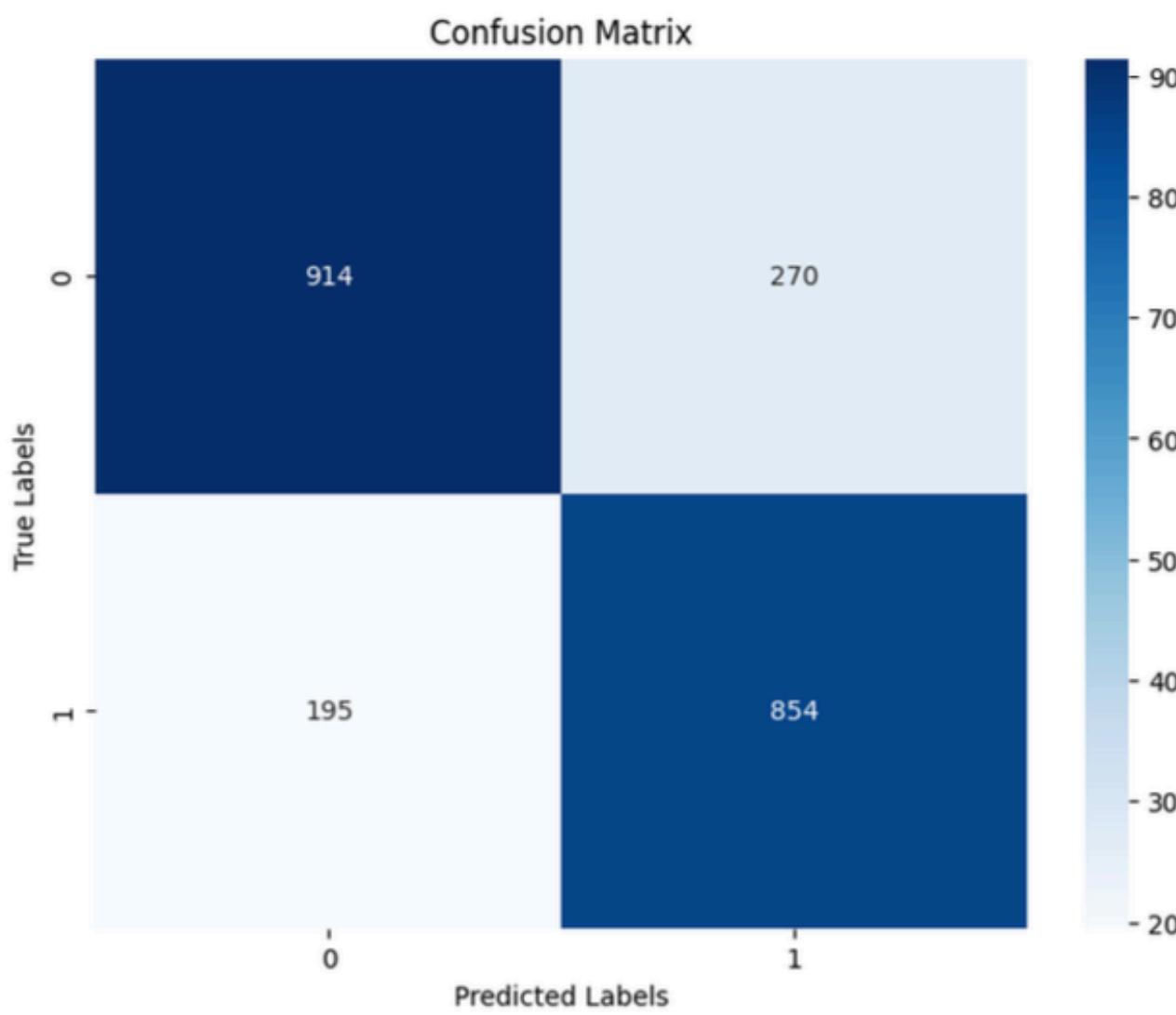


Model : DT

```
[58]: import seaborn as sns
from sklearn.metrics import confusion_matrix

conf_matrix = confusion_matrix(Y_test, Y_pred)

plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix')
plt.show()
```



```
[67]: from sklearn.metrics import accuracy_score

y_pred = grid_search_cv.predict(X_test)
accuracy_score(Y_test, y_pred)
```

```
[67]: 0.7917599641737573
```

```
[64]: Y_pred = classifier.predict(X_test)
from sklearn.metrics import accuracy_score
accuracy_score(Y_test, Y_pred)
```

```
[64]: 0.7917599641737573
```

Model : ANN

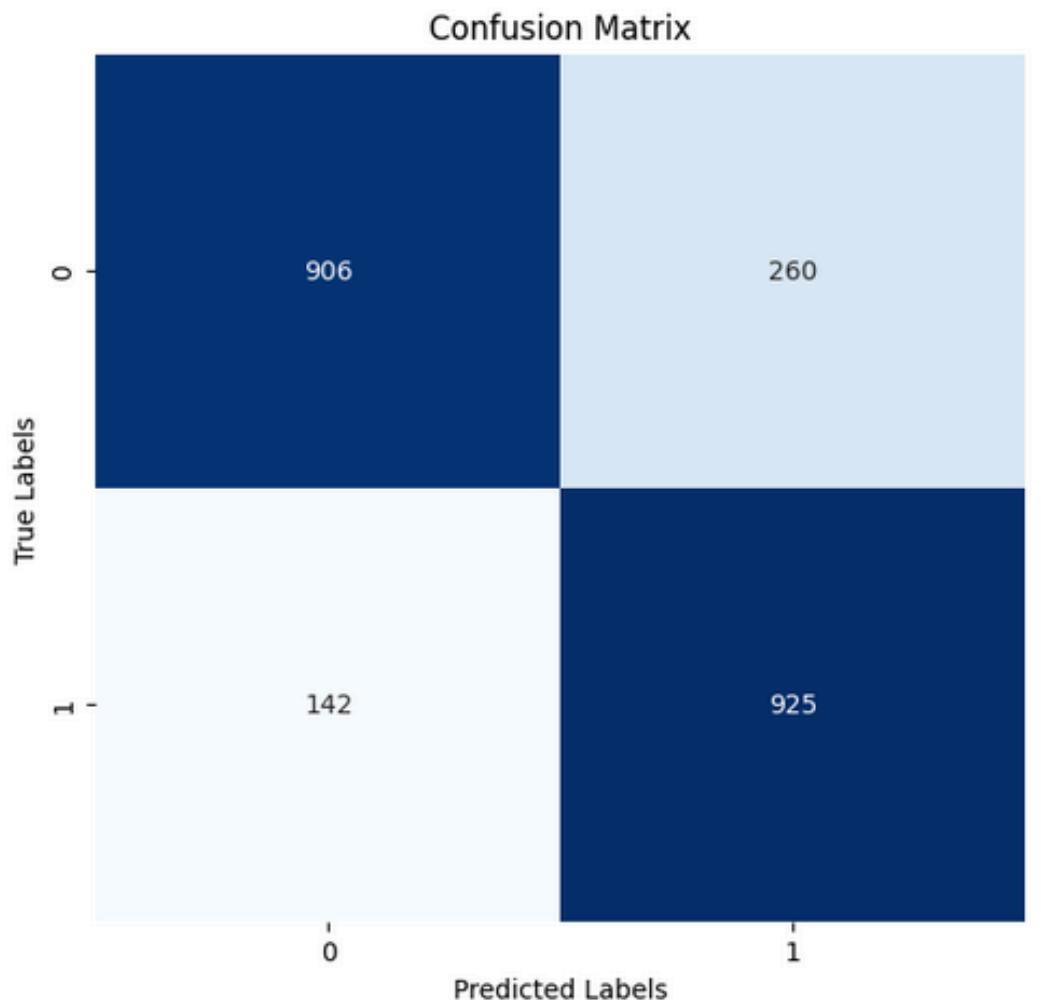
```
: val_loss, val_acc = model.evaluate(X_test, Y_test)
print("Validation Loss:", val_loss)
print("Validation Accuracy:", val_acc)
```

```
70/70 [=====] - 0s 4ms/step - loss: 0.4075 - accuracy: 0.8200
Validation Loss: 0.4074743092060089
Validation Accuracy: 0.8199731111526489
```

```
8]: import seaborn as sns
from sklearn.metrics import confusion_matrix

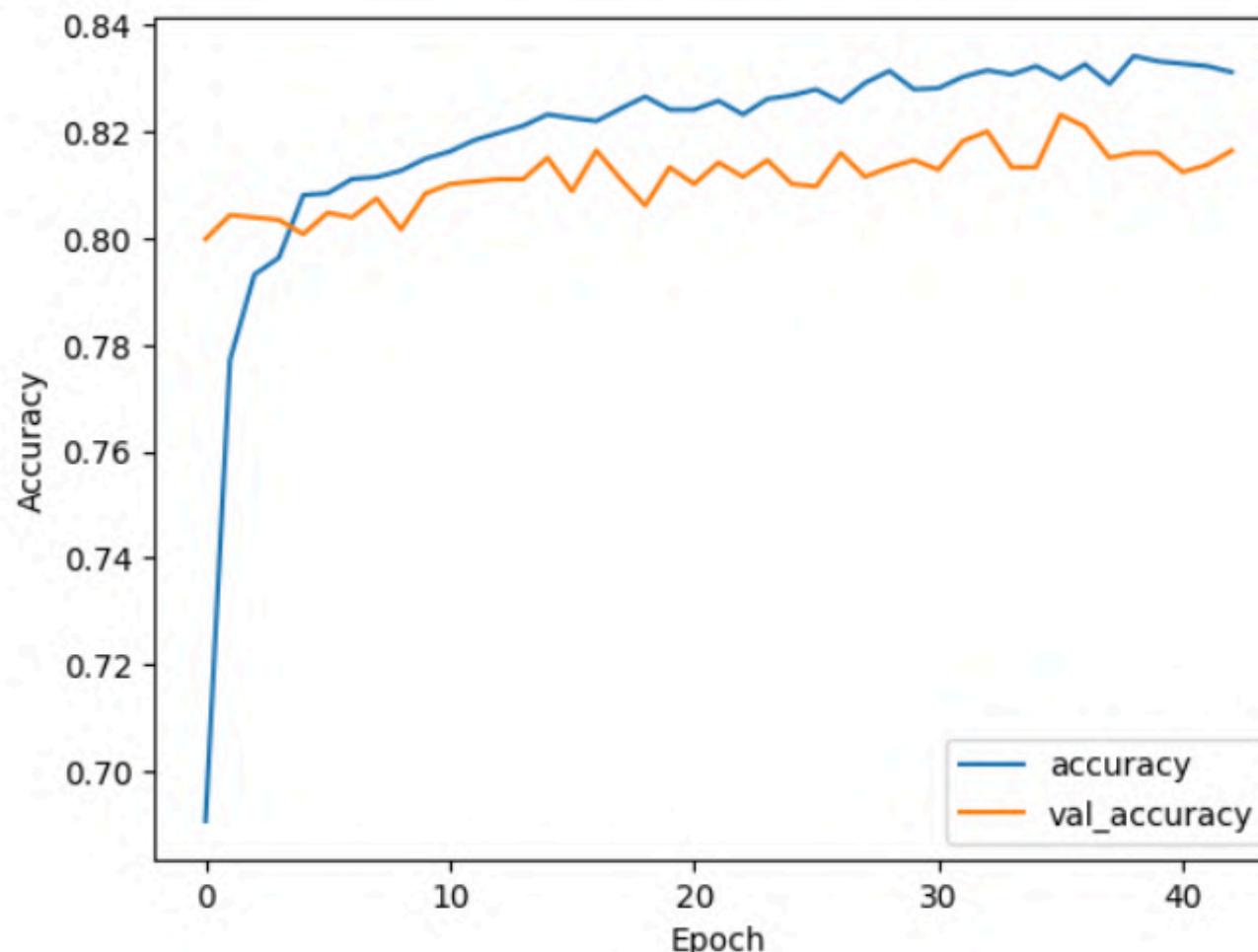
conf_matrix = confusion_matrix(Y_test, Y_pred)

plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix')
plt.show()
```

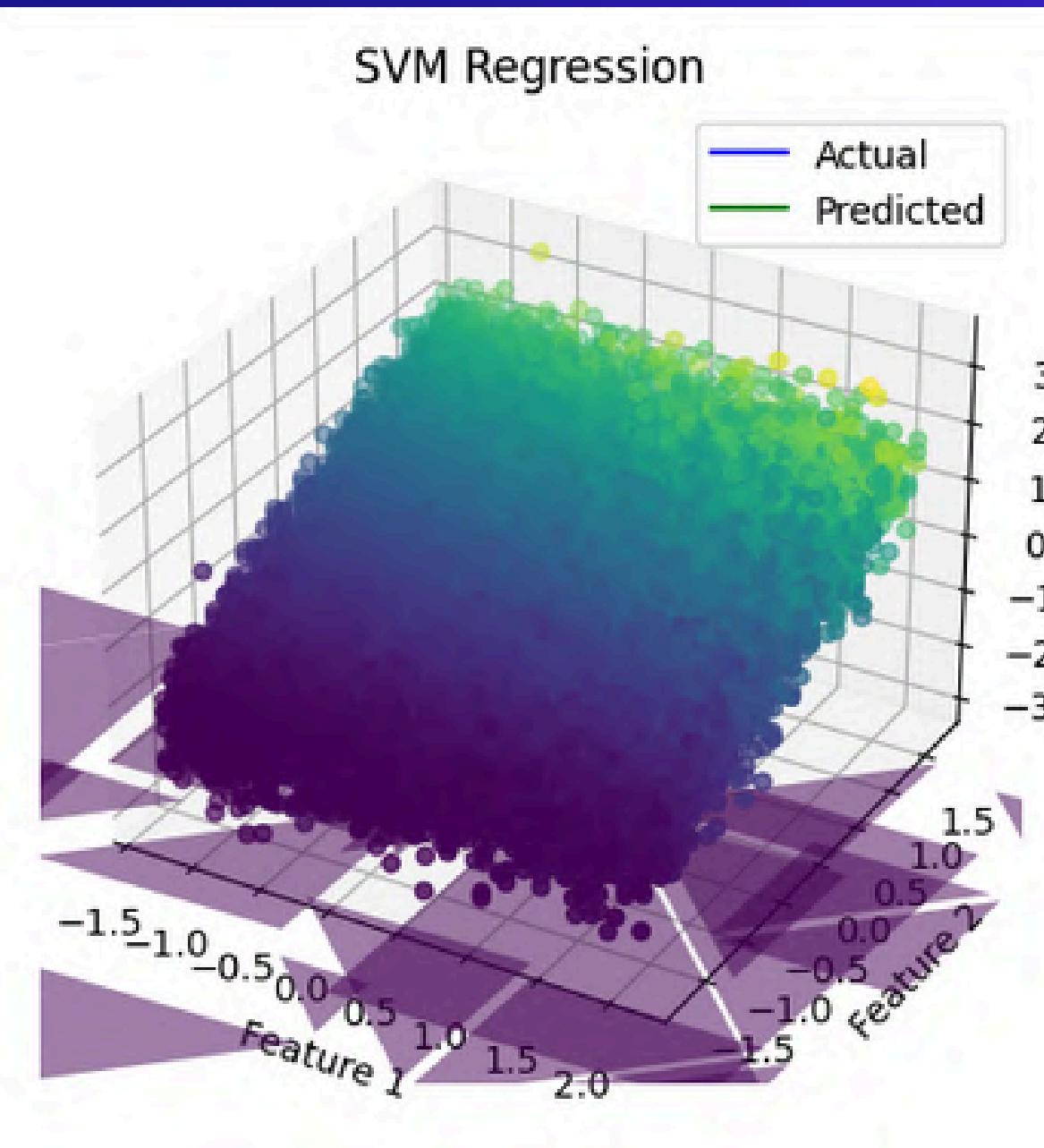


```
56]: import matplotlib.pyplot as plt

plt.plot(history.history['accuracy'], label='accuracy')
plt.plot(history.history['val_accuracy'], label='val_accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Model : SVM



```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

mae = mean_absolute_error(Y_test, test_data_prediction,multioutput='uniform_average')
mse = mean_squared_error(Y_test, test_data_prediction)
r2 = r2_score(Y_test, test_data_prediction)

print("Mean Absolute Error:", mae)
print("Mean Squared Error:", mse)
print("R-squared:", r2)
```

```
Mean Absolute Error: 3.2603316678133396
Mean Squared Error: 44.1262269753927
R-squared: 0.988550270502633
```

```
...
penalizes misclassification linearly by imposing a plenalty on the margin violation
...

def hinge_loss(y_true,y_pred):

    n_samples = len(y_true)
    loss = np.maximum(0, 1 - y_true * y_pred)
    return np.sum(loss) / n_samples

y_true = Y_test
y_pred = test_data_prediction

print("Hinge loss:", hinge_loss(y_true,y_pred))
```

Hinge loss: 0.0

ANN & DT GUI

(Term Deposit Prediction)

| | | | | |
|------------|--|-----------|-----------------------------------|--|
| Job: | <input type="text" value="admin. -"/> | Age: | <input type="text" value="49"/> | Client will likely subscribe to a term deposit |
| Marital: | <input type="text" value="married -"/> | Balance: | <input type="text" value="2343"/> | <input type="button" value="Predict ANN"/> |
| Education: | <input type="text" value="secondary -"/> | Day: | <input type="text" value="5"/> | |
| Default: | <input type="text" value="no -"/> | Duration: | <input type="text" value="1042"/> | Client will likely subscribe to a term deposit |
| Housing: | <input type="text" value="yes -"/> | Campaign: | <input type="text" value="1"/> | <input type="button" value="Predict DT"/> |
| Loan: | <input type="text" value="no -"/> | Pdays: | <input type="text" value="-1"/> | |
| Contact: | <input type="text" value="unknown -"/> | Previous: | <input type="text" value="0"/> | |
| Month: | <input type="text" value="may -"/> | | | |
| Poutcome: | <input type="text" value="unknown -"/> | | | |

ANN & DT GUI

(Term Deposit Prediction)

| | | | | |
|------------|--|-----------|----------------------------------|--|
| Job: | <input type="button" value="housemaid ▾"/> | Age: | <input type="text" value="41"/> | Client may not subscribe to a term deposit |
| Marital: | <input type="button" value="divorced ▾"/> | Balance: | <input type="text" value="22"/> | <input type="button" value="Predict ANN"/> |
| Education: | <input type="button" value="primary ▾"/> | Day: | <input type="text" value="18"/> | |
| Default: | <input type="button" value="no ▾"/> | Duration: | <input type="text" value="238"/> | Client may not subscribe to a term deposit |
| Housing: | <input type="button" value="yes ▾"/> | Campaign: | <input type="text" value="3"/> | <input type="button" value="Predict DT"/> |
| Loan: | <input type="button" value="no ▾"/> | Pdays: | <input type="text" value="126"/> | |
| Contact: | <input type="button" value="cellular ▾"/> | Previous: | <input type="text" value="4"/> | |
| Month: | <input type="button" value="nov ▾"/> | | | |
| Poutcome: | <input type="button" value="other ▾"/> | | | |

SVM GUI

1)Age:

2)Height:

3)Weight:

4)Duration:

5)Heart_Rate:

6)Body_Temp:

7)Gender (male|female):

Calories_Burnt

predicted Calories Burnt: 65.14393749324438

1)Age:

2)Height:

3)Weight:

4)Duration:

5)Heart_Rate:

6)Body_Temp:

7)Gender (male|female):

Calories_Burnt

predicted Calories Burnt: 216.5178273099675

RESULTS AND ACHIEVEMENTS

01

Model : ANN

- Accuracy: $\approx 0.8240035772323608$
-

02

Model : DT

- Accuracy: 0.7917599641737573
-

03

Model : SVM

- Accuracy: 0.988550270502633

COVER SHEET



Advanced Machine Learning Cover Sheet

Row Number (in PDF): 24

Time: 10:40

| # | Member Name (printed in Arabic) | Member ID (printed) | Handwritten signature |
|---|---------------------------------|---------------------|-----------------------|
| 1 | مريم عماد الدين عباس سعيد | 20210920 | |
| 2 | مريم أحمد محمود محمد | 20210906 | |
| 3 | منه الله هرميس مصطفى محمد | 20210964 | |
| 4 | نورهان فؤاد محمد ابوزريع | 20211017 | |
| 5 | زينب تامر اسماعيل وهبه | 20210385 | |
| 6 | زينب هاشم عبدالله الخولي | 20210388 | |
| 7 | | | |

Project Requirements(grades)

| | |
|-----------------------------|--|
| Documentation | |
| Preprocessing | |
| Preprocessing Understanding | |
| Models | |
| Models Understanding | |
| GUI | |

30

THANK YOU!

