NAME : Shashwat Shah

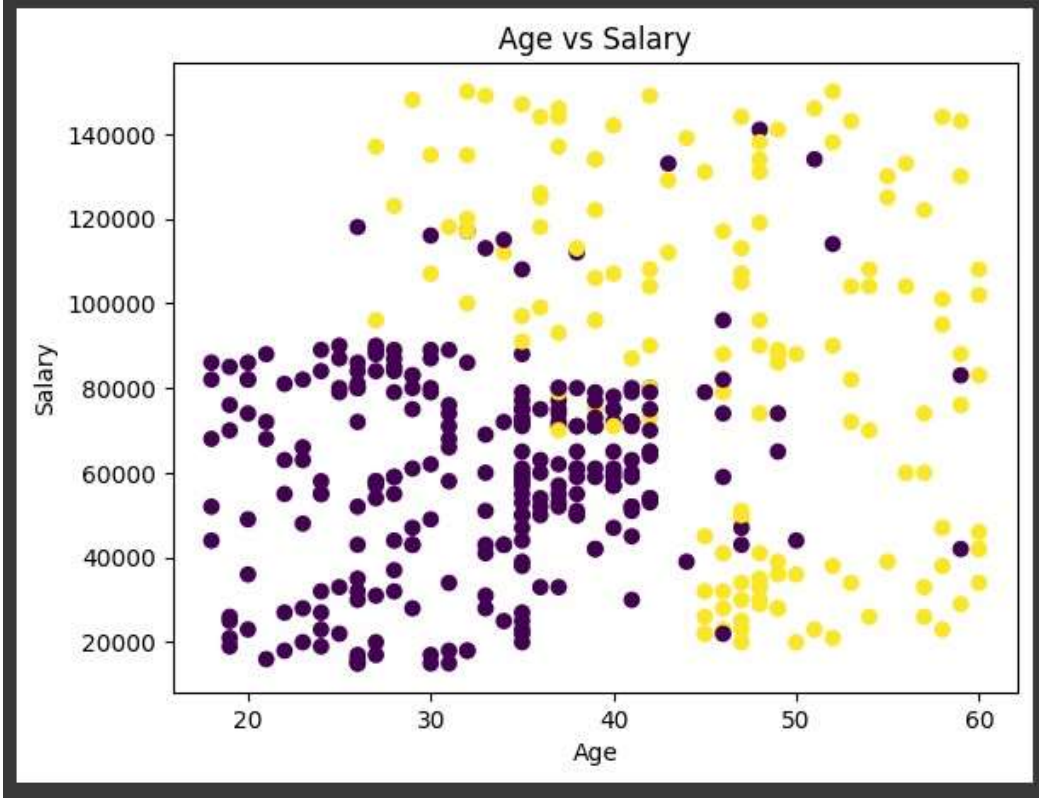
SAPID:60004220126

C22

B DIV

LINEAR DATASET (USER DATA)

```
import pandas as pd
import matplotlib.pyplot as plt

# Read the CSV file
df = pd.read_csv('User_Data.csv')

# Create a scatter plot of estimated age and salary, colored by purchased
plt.scatter(df['Age'], df['EstimatedSalary'], c=df['Purchased'])
plt.xlabel('Age')
plt.ylabel('Salary')
plt.title('Age vs Salary')
plt.show()
```



```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
df = df.dropna()
# Assuming the target variable is in the last column
X = df.iloc[:, :-1]  # Features
y = df.iloc[:, -1]   # Target variable
X = pd.get_dummies(X)

# Split data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Define kernel types
kernel_types = ['linear', 'poly', 'rbf', 'sigmoid']

for kernel in kernel_types:
    # Train SVM model
    svm_model = SVC(kernel=kernel)
    svm_model.fit(X_train, y_train)

    # Make predictions
    y_pred = svm_model.predict(X_test)

    # Calculate metrics
    accuracy = accuracy_score(y_test, y_pred)
    confusion_mat = confusion_matrix(y_test, y_pred)
    class_report = classification_report(y_test, y_pred)

    # Print results
    print(f"\nKernel: {kernel}")
    print("Accuracy:", accuracy)
    print("Confusion Matrix:")
    print(confusion_mat)
    print("Classification Report:")
    print(class_report)
```

```
Kernel: linear
Accuracy: 0.7375
Confusion Matrix:
[[48  4]
 [17 11]]
Classification Report:
              precision    recall  f1-score   support

           0       0.74      0.92      0.82        52
           1       0.73      0.39      0.51        28

    accuracy                           0.74        80
   macro avg       0.74      0.66      0.67        80
weighted avg       0.74      0.74      0.71        80
```

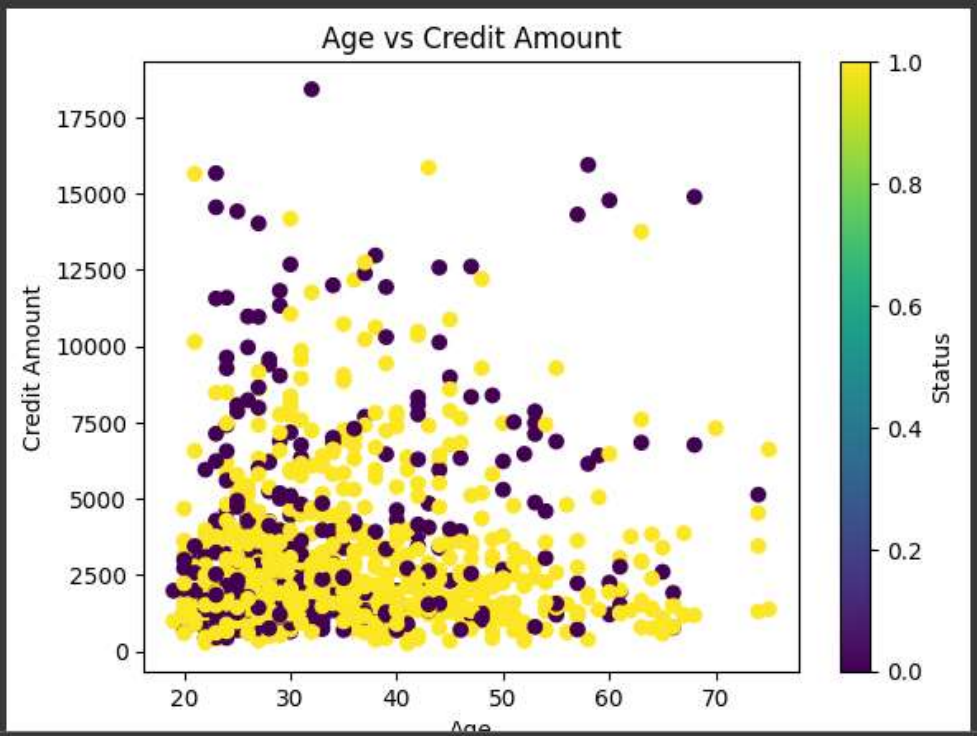
```
Kernel: poly
Accuracy: 0.65
Confusion Matrix:
[[52  0]
 [28  0]]
Classification Report:
              precision    recall  f1-score   support

           0       0.65      1.00      0.79        52
           1       0.00      0.00      0.00        28

    accuracy                           0.65        80
   macro avg       0.33      0.50      0.39        80
weighted avg       0.42      0.65      0.51        80
```

```
Kernel: rbf
Accuracy: 0.65
Confusion Matrix:
[[52  0]
 [28  0]]
Classification Report:
              precision    recall  f1-score   support

           0       0.65      1.00      0.79        52
           1       0.00      0.00      0.00        28

    accuracy                           0.65        80
   macro avg       0.33      0.50      0.39        80
weighted avg       0.42      0.65      0.51        80
```

```
Kernel: sigmoid
Accuracy: 0.65
Confusion Matrix:
[[52  0]
 [28  0]]
Classification Report:
              precision    recall  f1-score   support

           0       0.65      1.00      0.79        52
           1       0.00      0.00      0.00        28

    accuracy                           0.65        80
   macro avg       0.33      0.50      0.39        80
weighted avg       0.42      0.65      0.51        80
```

Non linear dataset(german_csv_file)

```
[9] import pandas as pd
    import matplotlib.pyplot as plt

    # Read the CSV file
    df = pd.read_csv('/content/GermanCredit.csv')

    # Create a scatter plot of age and credit amount, colored by status
    plt.scatter(df['age'], df['amount'], c=df['credit_risk'])
    plt.xlabel('Age')
    plt.ylabel('Credit Amount')
    plt.title('Age vs Credit Amount')
    plt.colorbar(label='Status')
    plt.show()
```



```python
import pandas as pd
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# Read the CSV file
df = pd.read_csv('/content/GermanCredit.csv')

# Splitting the data into features (X) and target variable (y)
X = df[['age', 'amount']]
y = df['credit_risk']

# Splitting the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# List of kernel types
kernels = ['linear', 'poly', 'rbf', 'sigmoid']

# Loop through each kernel type
for kernel in kernels:
    print(f"Kernel: {kernel}")

    # Create a support vector machine classifier
    clf = SVC(kernel=kernel)

    # Fit the classifier to the training data
    clf.fit(X_train, y_train)

    # Predict the labels of the test data
    y_pred = clf.predict(X_test)

    # Print accuracy
    accuracy = accuracy_score(y_test, y_pred)
    print(f"Accuracy: {accuracy:.2f}")
```

```
# Splitting the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# List of kernel types
kernels = ['linear', 'poly', 'rbf', 'sigmoid']

# Loop through each kernel type
for kernel in kernels:
    print(f"Kernel: {kernel}")

    # Create a support vector machine classifier
    clf = SVC(kernel=kernel)

    # Fit the classifier to the training data
    clf.fit(X_train, y_train)

    # Predict the labels of the test data
    y_pred = clf.predict(X_test)

    # Print accuracy
    accuracy = accuracy_score(y_test, y_pred)
    print(f"Accuracy: {accuracy:.2f}")

    # Print confusion matrix
    print("Confusion Matrix:")
    print(confusion_matrix(y_test, y_pred))

    # Print classification report
    print("Classification Report:")
    print(classification_report(y_test, y_pred))
    print()
```

```
Kernel: linear
Accuracy: 0.67
Confusion Matrix:
[[  3  56]
 [ 10 131]]
Classification Report:
              precision    recall  f1-score   support

           0       0.23      0.05      0.08        59
           1       0.70      0.93      0.80       141

    accuracy                           0.67       200
   macro avg       0.47      0.49      0.44       200
weighted avg       0.56      0.67      0.59       200
```

```
Kernel: poly
Accuracy: 0.72
Confusion Matrix:
[[  3  56]
 [  0 141]]
Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.05      0.10        59
           1       0.72      1.00      0.83       141

    accuracy                           0.72       200
   macro avg       0.86      0.53      0.47       200
weighted avg       0.80      0.72      0.62       200
```

```
Kernel: rbf
Accuracy: 0.71
Confusion Matrix:
[[  3  56]
 [  1 140]]
Classification Report:
              precision    recall  f1-score   support

           0       0.75      0.05      0.10        59
           1       0.71      0.99      0.83       141

    accuracy                           0.71       200
   macro avg       0.73      0.52      0.46       200
weighted avg       0.72      0.71      0.61       200
```

```
Kernel: sigmoid
Accuracy: 0.66
Confusion Matrix:
[[ 22  37]
 [ 32 109]]
Classification Report:
              precision    recall  f1-score   support

           0       0.41      0.37      0.39        59
           1       0.75      0.77      0.76       141

    accuracy                           0.66       200
   macro avg       0.58      0.57      0.57       200
weighted avg       0.65      0.66      0.65       200
```