Appendix\_Codes document

**Cleaned Audit\_Dataset (removing rows). This is Initial accuracy**

import numpy as np  
import pandas as pd  
from sklearn.decomposition import PCA  
from sklearn.impute import SimpleImputer  
from sklearn.preprocessing import StandardScaler  
from sklearn.svm import SVC  
from sklearn.model\_selection import cross\_val\_score

file\_path = r'C:\Users\tebib\OneDrive\Desktop\J2\Topics in intelligent systems\Midterm\Audit\_Trial4.csv'  
data = pd.read\_csv(file\_path)

# Handle missing values  
imputer = SimpleImputer(missing\_values=np.nan, strategy='mean')  
data\_imputed = pd.DataFrame(imputer.fit\_transform(data), columns=data.columns)

# Standardize the data  
scaler = StandardScaler()  
X\_normalized = scaler.fit\_transform(data\_imputed.drop('Risk', axis=1))

# PCA transformation  
pca = PCA(n\_components=4)  
X\_pca = pca.fit\_transform(X\_normalized)

# Prepare the target variable  
y = data\_imputed['Risk']

# SVM Model  
svm\_classifier = SVC()

# 10-fold Cross-Validation  
accuracy = cross\_val\_score(svm\_classifier, X\_pca, y, cv=10)

# Calculate Mean Accuracy and Standard Deviation  
mean\_accuracy = accuracy.mean()  
std\_deviation = accuracy.std()

print(f"Mean Accuracy: {mean\_accuracy}")  
print(f"Standard Deviation: {std\_deviation}")

**Cleaned Audit\_Dataset (replacing missing values with means )**

import pandas as pd  
from sklearn.impute import SimpleImputer  
from sklearn.model\_selection import cross\_val\_score  
from sklearn.svm import SVC

# Load dataset  
file\_path = 'C:/Users/tebib/OneDrive/Desktop/J2/Topics in intelligent systems/Midterm/Audit\_Trial4.csv'  
data = pd.read\_csv(file\_path)

data = data.replace('0', pd.NA)

imputer = SimpleImputer(missing\_values=pd.NA, strategy='mean')  
data\_imputed = imputer.fit\_transform(data.drop('Risk', axis=1))  
data\_imputed = pd.DataFrame(data\_imputed, columns=data.drop('Risk', axis=1).columns)

# Add the target variable 'Risk' back into the DataFrame  
data\_imputed['Risk'] = data['Risk'].values

svm\_classifier = SVC()

# Separate features and target  
X = data\_imputed.drop('Risk', axis=1)  
y = data\_imputed['Risk']

# Perform 10-fold cross-validation  
accuracy = cross\_val\_score(svm\_classifier, X, y, cv=10)

# Calculate mean accuracy and standard deviation  
mean\_accuracy = accuracy.mean()  
std\_deviation = accuracy.std()

# Print the results  
print("Mean Accuracy: ", mean\_accuracy)  
print("Standard Deviation: ", std\_deviation)

**Normalized Audit\_Dataset**

import numpy as np  
import pandas as pd  
from sklearn.impute import SimpleImputer  
from sklearn.preprocessing import StandardScaler  
from sklearn.svm import SVC  
from sklearn.model\_selection import cross\_val\_score

# Load dataset  
file\_path = r'C:\Users\tebib\OneDrive\Desktop\J2\Topics in intelligent systems\Midterm\Audit\_Trial4.csv'  
data = pd.read\_csv(file\_path)

# Separate features and target  
X = data.drop('Risk', axis=1)  # Replace 'Risk' with the actual name of your target column  
y = data['Risk']  # Replace 'Risk' with the actual name of your target column

# Impute missing values with the mean  
imputer = SimpleImputer(missing\_values=np.nan, strategy='mean')  
X\_imputed = imputer.fit\_transform(X)

# Normalize features  
scaler = StandardScaler()  
X\_normalized = scaler.fit\_transform(X\_imputed)

# Initialize SVM classifier  
svm\_classifier = SVC()

# Perform 10-fold cross-validation  
accuracies = cross\_val\_score(svm\_classifier, X\_normalized, y, cv=10)

# Calculate mean accuracy and standard deviation  
mean\_accuracy = np.mean(accuracies)  
std\_deviation = np.std(accuracies)

# Print the results  
print("Normalized Audit\_Dataset - SVM Accuracies:")  
print("10-fold cross-validation mean average:", mean\_accuracy)  
print("Standard deviation of accuracies:", std\_deviation)

**Reduced feature Dataset to only half i.e 8 attributes, pick your best to show highest accuracy.**

import pandas as pd  
from sklearn.model\_selection import train\_test\_split, cross\_val\_score  
from sklearn.impute import SimpleImputer  
from sklearn.preprocessing import StandardScaler  
from sklearn.svm import SVC

# Load dataset  
file\_path = r'C:\Users\tebib\OneDrive\Desktop\J2\Topics in intelligent systems\Midterm\Audit\_Trial4.csv'  
data = pd.read\_csv(file\_path)

# Define the top 8 features  
top\_features = ['Score', 'PARA\_A', 'TOTAL', 'SCORE\_A', 'SCORE\_B', 'District', 'PARA\_B', 'MONEY\_Marks']

# Select top features and target variable  
X = data[top\_features]  
y = data['Risk']  # Replace 'Risk' with the actual name of your target column

# Handle missing values  
imputer = SimpleImputer(missing\_values=np.nan, strategy='mean')  
X\_imputed = imputer.fit\_transform(X)

# Normalize features  
scaler = StandardScaler()  
X\_normalized = scaler.fit\_transform(X\_imputed)

# Initialize SVM classifier  
svm\_classifier = SVC()

# Perform 10-fold cross-validation  
accuracies = cross\_val\_score(svm\_classifier, X\_normalized, y, cv=10)

# Calculate mean accuracy and standard deviation  
mean\_accuracy = accuracies.mean()  
std\_deviation = accuracies.std()

# Print the results  
print("Reduced feature dataset (8 features) - SVM Accuracies:")  
print("10-fold cross-validation mean average:", mean\_accuracy)  
print("Standard deviation of accuracies:", std\_deviation)

**Apply PCA on the Dataset and use only 4 features**

import pandas as pd

import numpy as np

from sklearn.impute import SimpleImputer

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA

from sklearn.svm import SVC

from sklearn.model\_selection import cross\_val\_score

# Load dataset

file\_path = r'C:\Users\tebib\OneDrive\Desktop\J2\Topics in intelligent systems\Midterm\Audit\_Trial4.csv'

data = pd.read\_csv(file\_path)

# Separate features and target

X = data.drop('Risk', axis=1)  # Replace 'Risk' with the actual name of your target column

y = data['Risk']  # Replace 'Risk' with the actual name of your target column

# Impute missing values and normalize features

imputer = SimpleImputer(missing\_values=np.nan, strategy='mean')

X\_imputed = imputer.fit\_transform(X)

scaler = StandardScaler()

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

# Apply PCA

pca = PCA(n\_components=4)

X\_pca = pca.fit\_transform(X\_scaled)

# Initialize SVM classifier

svm\_classifier = SVC()

# Perform 10-fold cross-validation

accuracies = cross\_val\_score(svm\_classifier, X\_pca, y, cv=10)

# Calculate mean accuracy and standard deviation

mean\_accuracy = np.mean(accuracies)

std\_deviation = np.std(accuracies)

# Print the results

print("PCA with 4 features - SVM Accuracies:")

print("10-fold cross-validation mean average:", mean\_accuracy)

print("Standard deviation of accuracies:", std\_deviation)