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)
```