"""

Title: Final Project

Author: Stephen Marth

Date: 15 March 2025

Version: Task 1: Data Exploration

"""

# https://www.kaggle.com/datasets/shantanugarg274/lung-cancer-prediction-dataset  << Data Set

import numpy as np

import os

import matplotlib.pyplot as plt

file\_path = "C:\\Users\\steph\\OneDrive - UNC-Wilmington\\3. ISE\_221\\ISE\_221\_Final\\Lung Cancer Dataset.csv"

file\_name = os.path.basename(file\_path)

feature\_threshold = 0.15  #threshold to dertimine which features to use

#function to help determine which features to use Pearson Correlation Formula

def feature\_correlation(X, y, feature\_name):

    correlations = []

    for i in range(X.shape[1]):  # Loop through features

        cor = np.corrcoef(X[:, i], y)[0, 1]  #(Feature[],target)[]

        correlations.append((feature\_name[i], cor))  #store feature name and correlation

    correlations.sort(key=lambda x: abs(x[1]), reverse=True)#sort

    print(f"\nFeature Correlation with {target}:")

    for feature, cor in correlations:

        print(f"{feature}: {cor:.4f}")

    return correlations

# Load Data and pre process

#name = numpygenfrmtxt("path",delimiter csv, skip header = 1 row)

data = np.genfromtxt(file\_path, delimiter=',', skip\_header=1, dtype=str, encoding="utf-8")

#print(name[rows])

#print(data[0:5]) #print rows to make sure it loaded

# change words to number, still a string

data[data == "YES"] = "1"

data[data == "NO"] = "0"

#change data type

data = data.astype(float)

y = data[:, -1]

X = data[:, :-1]

#print to make sure it was done right

#print("this is after conversion of yes and no to 1 and 0",data[:5])

with open(file\_path, "r") as file:

    header = file.readline().strip().split(",")

target = header[-1]

#print(data.shape)

print (f"{file\_name} has\n{data.shape[1]} Features\n{data.shape[0]} Samples") #use f" embed variables in statement

#get a features list take hader seperate by ,

#print feature list loop with index

print("Feature List with Index:")

index = 0

for feature in header:

    print(f"[{index}] {feature}")

    index += 1

#check for missing data with numpy.isnan(name).sum()

missing\_values = np.isnan(data).sum()

print(f"There are {missing\_values} missing values.") # 5000 are the string yes or no

unique, counts = np.unique(y, return\_counts=True)#finds values of 1 and 0 and total

plt.figure(figsize=(6, 4))

plt.bar(unique, counts, color=['blue', 'red'])

plt.xticks([0, 1], labels=["Negative", "Positive"])

plt.xlabel(target)

plt.ylabel("Number of Samples")

plt.title(f"Target Variable in {file\_name}")

plt.show()

correlations = feature\_correlation(X, y, header[:-1])

#Choosing features to use. based on r value interpritiation I dont have anything with a srong linear corliation. so we are going to set the bar low. might be best to use all.

selected\_features = [feature for feature, corr in correlations if abs(corr) > feature\_threshold]

# Print selected features

print(f"\n These Features are above a correlation threshold of {feature\_threshold}:")

print(selected\_features)

#okay i got six whats next

predictive\_feature\_indices = [header.index(feature) for feature, corr in correlations if abs(corr) > feature\_threshold]

X\_selected = X[:, predictive\_feature\_indices]

print(f"Predictive Features update shape: {X\_selected.shape}")

#sick

 #end with visual bar chart of selected figures

predictive\_features = [feature for feature, corr in correlations if abs(corr) > feature\_threshold]

r\_values = [corr for feature, corr in correlations if abs(corr) > feature\_threshold]

# Plot bar chart for selected features

plt.figure(figsize=(10, 5))

plt.bar(predictive\_features, r\_values, color='blue', alpha=1)

plt.xlabel("Features")

plt.ylabel("Correlation")

plt.title(f"Feature Correlation with {target} above {feature\_threshold} threshold")

plt.xticks(rotation=45)

plt.show()

#create a new csv file with only predictive features and target

# should normalize data next