```
"""
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
```