ETL_Project

August 14, 2024

1 Data Patterns and Representations

1.0.1 Jonhs Hopkins University

1.0.2 Xiang Feng Liang

Source: https://www.kaggle.com/datasets/akashnath29/lung-cancer-dataset

1.0.3 (a) Automate the process of loading your dataset from its source. This could involve fetching data from an API, reading from a file system, or scraping a web page.

[1]:
```python
import os
import pandas as pd
import subprocess

dataset = 'akashnath29/lung-cancer-dataset'
download_path = 'lung_cancer_data'

def download_dataset(dataset, path):
    if not os.path.exists(path):
        os.makedirs(path)
    command = f'kaggle datasets download -d {dataset} -p {path} --unzip'
    subprocess.run(command, shell=True, check=True)

def read_file(file_path):
    return pd.read_csv(file_path)

def load_data():
    download_dataset(dataset, download_path)
    file_path = os.path.join(download_path, 'dataset.csv')
    data = read_file(file_path)
    return data

load_data()
```

```
Dataset URL: https://www.kaggle.com/datasets/akashnath29/lung-cancer-dataset
License(s): ODbL-1.0
Downloading lung-cancer-dataset.zip to lung_cancer_data
```

```
100%|          | 68.8k/68.8k [00:00<00:00, 2.46MB/s]
```

[1]:
```
     GENDER  AGE  SMOKING  YELLOW_FINGERS  ANXIETY  PEER_PRESSURE  \
0         M   65        1               1        1              2
1         F   55        1               2        2              1
2         F   78        2               2        1              1
3         M   60        2               1        1              1
4         F   80        1               1        2              1
…       …   …      …             …       …            …
2995      F   71        2               1        1              2
2996      F   75        1               2        1              1
2997      F   62        2               2        2              1
2998      M   30        1               1        2              2
2999      M   40        1               2        2              1

      CHRONIC_DISEASE  FATIGUE  ALLERGY  WHEEZING  ALCOHOL_CONSUMING  \
0                   2        1        2         2                  2
1                   1        2        2         2                  1
2                   1        2        1         2                  1
3                   2        1        2         1                  1
4                   1        2        1         2                  1
…                 …      …      …       …                …
2995                2        1        1         1                  1
2996                1        2        2         2                  2
2997                2        2        2         2                  1
2998                2        2        2         2                  2
2999                1        1        2         2                  2

      COUGHING  SHORTNESS_OF_BREATH  SWALLOWING_DIFFICULTY  CHEST_PAIN  \
0            2                    2                      2           1
1            1                    1                      2           2
2            1                    2                      1           1
3            2                    1                      2           2
4            1                    1                      1           2
…          …                  …                    …         …
2995         2                    1                      1           2
2996         1                    1                      2           1
2997         1                    2                      2           2
2998         1                    2                      1           2
2999         1                    1                      1           1

     LUNG_CANCER
0             NO
1             NO
2            YES
3            YES
```

```
4              NO
…            …
2995           NO
2996           NO
2997          YES
2998          YES
2999          YES

[3000 rows x 16 columns]
```

1.0.4 (b) Implement scripts or functions that clean the data by handling missing values, removing duplicates, correcting errors, and dealing with outliers.

[2]:
```
def binary_feature_check(df):
    df_no_age = df.drop('AGE', axis = 1)
    non_binary_features = []
    for feature in df_no_age.columns:
        unique_values = df_no_age[feature].nunique()
        if unique_values != 2:
            non_binary_features.append(feature)
    if len(non_binary_features) > 0:
        print(f"The non-binary features are '{non_binary_features}'.")

    return non_binary_features



def clean_data(df):
    try:
        binary_features = binary_feature_check(df)

        df.drop_duplicates(inplace=True)

        print('Data Cleaned Successfully.')
        return df

    except Exception as e:
        print(f"Data cleansed failed due to: {e}")
        return 0


def check_outliers_in_age(df):

    q1 = df['AGE'].quantile(0.25)
    q3 = df['AGE'].quantile(0.75)
    iqr = q3 - q1
    lower_bound = q1 - 1.5 * iqr
```

```
    upper_bound = q3 + 1.5 * iqr

    outliers = (df['AGE'] < lower_bound) | (df['AGE'] > upper_bound)

    if outliers.any():
        print("Outliers detected in the 'AGE' column.")
    else:
        print("No outliers detected in the 'AGE' column.")
```

1.0.5 (c) Transform the dataset into a format that's suitable for analysis. This may include normalizingor standardizing numerical data, encoding categorical variables, and creating new features from existing ones.

[3]:
```
from sklearn.model_selection import train_test_split



def transform_data(df):

    df['LUNG_CANCER'] = df['LUNG_CANCER'].map({'YES': 1, 'NO': 0})

    for i in df.iloc[:,2:-1]:
        df[i] = df[i].map({1:0, 2:1})

    df['is_male'] = df['GENDER'].map({'F': 0, 'M':1})
    df['is_female'] = df['GENDER'].map({'F': 1, 'M':0})

    df.drop('GENDER', axis = 1)

    x_features = df.drop('LUNG_CANCER', axis = 1)
    y_feature = df['LUNG_CANCER']


    X_train, X_test, y_train, y_test = train_test_split(x_features, y_feature,␣
↪test_size=0.2, random_state=42)

    print('The shape of X_train is ', X_train.shape)
    print('The shape of y_train is ', y_train.shape)
    print('The shape of X_test is ', X_test.shape)
    print('The shape of y_test is ', y_test.shape)

    return X_train, X_test, y_train, y_test
```

1.0.6 (d) Store the processed data in a structured format that is accessible for further analysis and modeling. We are requiring that you store your data in a database. However, you may choose which one (i.e. MySQL, PostgreSQL, SQLite, etc)

[4]:
```python
import sqlite3

def store_data(df, db_name, dataset_name):
    try:
        conn = sqlite3.connect(db_name)
        df.to_sql(dataset_name, conn, if_exists='replace', index=False)
        conn.close()
    except Exception as e:
        print(f"Failed to store data into database due to: {e}")


def run_data_store_pipeline():
    download_dataset(dataset, download_path)
    file_path = os.path.join(download_path, 'dataset.csv')
    data = read_file(file_path)
    clean_data(data)
    X_train, y_train, X_test, y_test = transform_data(data)

    db_name = 'lung_cancer.db'
    store_data(X_train, db_name, 'X_train')
    store_data(y_train, db_name, 'y_train')
    store_data(X_test, db_name, 'X_test')
    store_data(y_test, db_name, 'y_test')

    return X_train, y_train, X_test, y_test

run_data_store_pipeline()
```

```
Dataset URL: https://www.kaggle.com/datasets/akashnath29/lung-cancer-dataset
License(s): ODbL-1.0
Downloading lung-cancer-dataset.zip to lung_cancer_data

Data Cleaned Successfully.
The shape of X_train is  (2398, 17)
The shape of y_train is  (2398,)
The shape of X_test is  (600, 17)
The shape of y_test is  (600,)

100%|          | 68.8k/68.8k [00:00<00:00, 2.72MB/s]
```

[4]:
```
(      GENDER  AGE  SMOKING  YELLOW_FINGERS  ANXIETY  PEER_PRESSURE  \
 1570       M   54        1               0        0              0
 2230       M   55        1               1        1              0
 2297       F   75        1               0        0              0
```

```
1801      M   61        1              1        1             0
1273      F   49        0              0        1             1
…      …  …       …            …      …            …
1639      F   42        0              1        1             0
1095      M   50        1              1        0             0
1130      M   68        0              1        1             0
1294      M   50        0              0        1             0
860       M   78        0              1        1             0

      CHRONIC_DISEASE  FATIGUE  ALLERGY  WHEEZING  ALCOHOL_CONSUMING  \
1570                0        0        0         1                  1
2230                0        0        1         1                  1
2297                1        1        0         1                  1
1801                1        0        1         0                  0
1273                1        1        1         1                  1
…                …      …      …       …                …
1639                1        0        0         1                  1
1095                0        1        1         1                  1
1130                1        0        0         1                  1
1294                1        1        1         0                  1
860                 0        0        0         0                  1

      COUGHING  SHORTNESS_OF_BREATH  SWALLOWING_DIFFICULTY  CHEST_PAIN  \
1570         1                    1                      0           0
2230         0                    0                      0           1
2297         0                    1                      0           0
1801         1                    0                      0           0
1273         1                    0                      1           0
…          …                  …                    …         …
1639         1                    0                      0           0
1095         0                    1                      1           0
1130         0                    1                      1           1
1294         1                    1                      1           1
860          1                    1                      1           1

      is_male  is_female
1570        1          0
2230        1          0
2297        0          1
1801        1          0
1273        0          1
…         …        …
1639        0          1
1095        1          0
1130        1          0
1294        1          0
860         1          0
```

```
[2398 rows x 17 columns],
     GENDER  AGE  SMOKING  YELLOW_FINGERS  ANXIETY  PEER_PRESSURE  \
1376      F   67        0               0        0              1
932       F   42        0               0        1              0
144       F   30        0               0        1              0
1753      M   65        1               1        1              1
51        M   69        1               0        0              0
…       …   …        …             …      …            …
637       M   54        0               0        1              1
695       F   34        0               1        0              1
226       F   67        1               0        0              1
2262      M   75        1               0        1              0
1103      F   60        0               1        0              1

      CHRONIC_DISEASE  FATIGUE  ALLERGY  WHEEZING  ALCOHOL_CONSUMING  \
1376                0        0        0         0                  1
932                 0        0        1         0                  1
144                 0        1        0         1                  1
1753                1        0        0         0                  0
51                  1        1        0         0                  0
…                 …      …      …       …                …
637                 1        1        1         1                  0
695                 1        1        0         0                  0
226                 1        1        0         0                  1
2262                0        1        0         0                  0
1103                0        0        1         1                  1

      COUGHING  SHORTNESS_OF_BREATH  SWALLOWING_DIFFICULTY  CHEST_PAIN  \
1376         1                    1                      0           0
932          1                    1                      1           1
144          0                    1                      0           1
1753         1                    0                      1           0
51           1                    1                      1           0
…          …                  …                    …         …
637          0                    0                      0           0
695          1                    1                      0           0
226          1                    1                      1           0
2262         1                    1                      1           0
1103         1                    0                      1           0

      is_male  is_female
1376        0          1
932         0          1
144         0          1
1753        1          0
51          1          0
```

```
…        …         …
637         1          0
695         0          1
226         0          1
2262        1          0
1103        0          1

[600 rows x 17 columns],
1570    0
2230    0
2297    0
1801    1
1273    1
       ..
1639    1
1095    0
1130    0
1294    1
860     1
Name: LUNG_CANCER, Length: 2398, dtype: int64,
1376    0
932     1
144     1
1753    0
51      1
       ..
637     0
695     1
226     1
2262    0
1103    0
Name: LUNG_CANCER, Length: 600, dtype: int64)
```

1.0.7 (e) Create a workflow that orchestrates the execution of your data pipeline tasks in the correct order and monitors their success. Consider how you can schedule the pipeline to run automatically or trigger it based on specific events.

[5]:
```python
def download_dataset(dataset, path):
    if not os.path.exists(path):
        os.makedirs(path)
    try:
        command = f'kaggle datasets download -d {dataset} -p {path} --unzip'
        subprocess.run(command, shell=True, check=True)
        return True

    except Exception as e:
        print(f"Dataset download failed due to: {e}")
```

```
        return False


def read_file(file_path):
    return pd.read_csv(file_path)


def binary_feature_check(df):
    df_no_age = df.drop('AGE', axis = 1)
    non_binary_features = []
    for feature in df_no_age.columns:
        unique_values = df_no_age[feature].nunique()
        if unique_values != 2:
            non_binary_features.append(feature)
    if len(non_binary_features) > 0:
        print(f"The non-binary features are '{non_binary_features}'.")

    return non_binary_features



def clean_data(df):
    try:
        binary_features = binary_feature_check(df)

        df.drop_duplicates(inplace=True)

        print('Data Cleaned Successfully.')
        return df

    except Exception as e:
        print(f"Data cleansed failed due to: {e}")
        return 0



def check_outliers_in_age(df):

    q1 = df['AGE'].quantile(0.25)
    q3 = df['AGE'].quantile(0.75)
    iqr = q3 - q1
    lower_bound = q1 - 1.5 * iqr
    upper_bound = q3 + 1.5 * iqr

    outliers = (df['AGE'] < lower_bound) | (df['AGE'] > upper_bound)

    if outliers.any():
```

```
        print("Outliers detected in the 'AGE' column.")
    else:
        print("No outliers detected in the 'AGE' column.")


def transform_data(df):

    df['LUNG_CANCER'] = df['LUNG_CANCER'].map({'YES': 1, 'NO': 0})

    for i in df.iloc[:,2:-1]:
        df[i] = df[i].map({1:0, 2:1})

    df['is_male'] = df['GENDER'].map({'F': 0, 'M':1})
    df['is_female'] = df['GENDER'].map({'F': 1, 'M':0})

    df.drop('GENDER', axis = 1)

    x_features = df.drop('LUNG_CANCER', axis = 1)
    y_feature = df['LUNG_CANCER']


    X_train, X_test, y_train, y_test = train_test_split(x_features, y_feature,␣
 ↪test_size=0.2, random_state=42)

    print('The shape of X_train is ', X_train.shape)
    print('The shape of y_train is ', y_train.shape)
    print('The shape of X_test is ', X_test.shape)
    print('The shape of y_test is ', y_test.shape)

    return X_train, X_test, y_train, y_test




def store_data(df, db_name, dataset_name):
    try:
        conn = sqlite3.connect(db_name)
        df.to_sql(dataset_name, conn, if_exists='replace', index=False)
        conn.close()
    except Exception as e:
        print(f"Failed to store data into database due to: {e}")




def lung_cancer_data_pipeline():
```

```python
    dataset = 'akashnath29/lung-cancer-dataset'
    download_path = 'lung_cancer_data'

    is_downloaded = download_dataset(dataset, download_path)
    if not is_downloaded:
        return False

    file_path = os.path.join(download_path, 'dataset.csv')
    data = read_file(file_path)
    non_binary_features = clean_data(data)
    if not non_binary_features.empty:
        data.drop(columns=non_binary_features, axis=1)


    check_outliers_in_age(data)


    try:
        X_train, y_train, X_test, y_test = transform_data(data)

    except Exception as e:
        print(f"An error occurred when transforming the data: {e}")
        return False

    db_name = 'lung_cancer.db'
    store_data(X_train, db_name, 'X_train')
    store_data(y_train, db_name, 'y_train')
    store_data(X_test, db_name, 'X_test')
    store_data(y_test, db_name, 'y_test')

    return X_train, y_train, X_test, y_test

lung_cancer_data_pipeline()
```

```
Dataset URL: https://www.kaggle.com/datasets/akashnath29/lung-cancer-dataset
License(s): ODbL-1.0
Downloading lung-cancer-dataset.zip to lung_cancer_data

Data Cleaned Successfully.
No outliers detected in the 'AGE' column.
The shape of X_train is  (2398, 17)
The shape of y_train is  (2398,)
The shape of X_test is  (600, 17)
The shape of y_test is  (600,)

100%|      | 68.8k/68.8k [00:00<00:00, 2.30MB/s]
```

[5]:

```
(      GENDER  AGE  SMOKING  YELLOW_FINGERS  ANXIETY  PEER_PRESSURE  \
 1570      M   54        1               0        0              0
 2230      M   55        1               1        1              0
 2297      F   75        1               0        0              0
 1801      M   61        1               1        1              0
 1273      F   49        0               0        1              1
 …      …   …    …            …       …            …
 1639      F   42        0               1        1              0
 1095      M   50        1               1        0              0
 1130      M   68        0               1        1              0
 1294      M   50        0               0        1              0
 860       M   78        0               1        1              0

       CHRONIC_DISEASE  FATIGUE  ALLERGY  WHEEZING  ALCOHOL_CONSUMING  \
 1570                0        0        0         1                  1
 2230                0        0        1         1                  1
 2297                1        1        0         1                  1
 1801                1        0        1         0                  0
 1273                1        1        1         1                  1
 …                 …      …      …       …                 …
 1639                1        0        0         1                  1
 1095                0        1        1         1                  1
 1130                1        0        0         1                  1
 1294                1        1        1         0                  1
 860                 0        0        0         0                  1

       COUGHING  SHORTNESS_OF_BREATH  SWALLOWING_DIFFICULTY  CHEST_PAIN  \
 1570         1                    1                      0           0
 2230         0                    0                      0           1
 2297         0                    1                      0           0
 1801         1                    0                      0           0
 1273         1                    0                      1           0
 …          …                    …                    …          …
 1639         1                    0                      0           0
 1095         0                    1                      1           0
 1130         0                    1                      1           1
 1294         1                    1                      1           1
 860          1                    1                      1           1

       is_male  is_female
 1570        1          0
 2230        1          0
 2297        0          1
 1801        1          0
 1273        0          1
 …        …        …
 1639        0          1
```

```
1095        1          0
1130        1          0
1294        1          0
860         1          0

[2398 rows x 17 columns],
     GENDER  AGE  SMOKING  YELLOW_FINGERS  ANXIETY  PEER_PRESSURE  \
1376      F   67        0               0        0              1
932       F   42        0               0        1              0
144       F   30        0               0        1              0
1753      M   65        1               1        1              1
51        M   69        1               0        0              0
…      …  …       …             …      …            …
637       M   54        0               0        1              1
695       F   34        0               1        0              1
226       F   67        1               0        0              1
2262      M   75        1               0        1              0
1103      F   60        0               1        0              1

      CHRONIC_DISEASE  FATIGUE  ALLERGY  WHEEZING  ALCOHOL_CONSUMING  \
1376                0        0        0         0                  1
932                 0        0        1         0                  1
144                 0        1        0         1                  1
1753                1        0        0         0                  0
51                  1        1        0         0                  0
…               …      …      …       …                …
637                 1        1        1         1                  0
695                 1        1        0         0                  0
226                 1        1        0         0                  1
2262                0        1        0         0                  0
1103                0        0        1         1                  1

      COUGHING  SHORTNESS_OF_BREATH  SWALLOWING_DIFFICULTY  CHEST_PAIN  \
1376         1                    1                      0           0
932          1                    1                      1           1
144          0                    1                      0           1
1753         1                    0                      1           0
51           1                    1                      1           0
…       …                …                    …         …
637          0                    0                      0           0
695          1                    1                      0           0
226          1                    1                      1           0
2262         1                    1                      1           0
1103         1                    0                      1           0

      is_male  is_female
1376        0          1
```

```
932         0          1
144         0          1
1753        1          0
51          1          0
…       …        …
637         1          0
695         0          1
226         0          1
2262        1          0
1103        0          1

[600 rows x 17 columns],
1570    0
2230    0
2297    0
1801    1
1273    1
       ..
1639    1
1095    0
1130    0
1294    1
860     1
Name: LUNG_CANCER, Length: 2398, dtype: int64,
1376    0
932     1
144     1
1753    0
51      1
       ..
637     0
695     1
226     1
2262    0
1103    0
Name: LUNG_CANCER, Length: 600, dtype: int64)
```

1.1 Scheduling method for consideration

1) Using a Timer or Condition Within the Notebook You can use Python's time module to run a cell automatically after a certain period or based on a condition. Here's an example using a simple timer:
2) Using papermill for Parameterized Execution papermill is a tool that allows you to run Jupyter Notebooks programmatically and can be used to parameterize and execute notebooks. This is particularly useful for scheduling tasks.
3) Using Scheduled Tasks (External Tools) If you need to run a Jupyter Notebook at specific intervals (e.g., daily, hourly), you can use scheduling tools like cron (Linux/Mac) or Task Scheduler (Windows) along with a script to run the notebook using papermill.

1.1.1 (f) Document your data pipeline design, including each component's role and how they interact. Write tests for your data cleaning and transformation logic to ensure accuracy and reliability.

1.1.2 Functions:

1) Download the Dataset In the download_dataset function we download our lung cancer dataset directly from Kaggle and return true in the case of success and false in the case of failure. The parameters are the dataset which is the name of the Kaggle dataset we want to download and the path which is the location where we want the dataset downloaded to. The method first checks if the specified path exists and makes the directory if it does not. We then use a sub process command to download the specified Kaggle dataset and download it into the specified path. In the case that the download is unsuccesful for whatever reason we print the resulting exception and the function returns false. If the download is successful the function returns true. These boolean values are later used in the lung_cancer_data_pipeline function to determine if the pipeline should continue or not. If the pipeline continues it will read the newly created csv file and convert it into a dataframe for cleansing.

2) Cleaning the Dataset In the clean_data function we perform multiple data cleaning techniques. The parameter data is the result of downloading and converting the lung cancer dataset into a dataframe. The first stage of the data cleanse involves checking that the binary features are actually binary. The is_binary_feature function takes in the dataframe and removes the age column (the only non-binary feature). It then checks that all other features have a unique value count of 2 (conditions for a binary feature). In the case that one feature is not binary we record the non binary feature name into a list and send it to the console. If there is a non binary feature in the dataframe the is_binary_feature function returns the list of non binary features and they will be dropped from the dataframe. After checking that the features are binary we drop all duplicate records from the dataframe. Next is to check for outliers in the age data. We use the standard IQR outlier calculation to determine if there are any outliers in the data. If there are outliers then we remove them from the dataframe and return the newly updated datframe.

3) Transforming the Dataset In the transform_data function we transform the data in our dataset for future modelling. First we ensure that all binary features are mapping values from (0:1). This includes mapping the lung cancer column's values from "no":0 and "yes":1. Next we notice that all other features are on a binary scalse of (1:2) so we map them to (0:1) respectively. Finally we create two new features for gender, is_male and is_female. These new binary features are also mapped to a (0:1) scale. Finally we drop the gender column and split the data into test and train datasets using an 80-20 test to train ratio. The test and train datasets are then returned back to the lung_cancer_data_pipeline for loading into the database.

4) Loading the Dataset into a Database We used an SQLite3 database implementation for our data which gets implemented by the store_data function. Store_data creates a connection to the database which is used to input the test and train dataframes into our SQLite database. Exception handling was added to this method to ensure that the data was loaded into the database successfully.

Testing for cleaning data

[6]:
```python
test_data = {
    'GENDER': ['F', 'M', 'F', 'M', 'F', 'M', 'M', 'F', 'F'],
    'LUNG_CANCER': ['YES', 'NO', 'YES', 'NO', 'YES', 'NO', 'NO', 'YES','YES'],
    'SMOKING': [1, 0, 1, 0, 1, 0, 0, 1, 1],
    'AGE': [23, 45, 23, 45, 23, 45, 45, 23, 23],  # 'AGE' will not be dropped,␣
↪but it should also not be included in non-binary features
    'WEIGHT': [70, 80, 60, 75, 65, 85, 78, 72, 72]  # 'WEIGHT' is not a binary␣
↪feature
}
df = pd.DataFrame(test_data)

# Test the functions
cleaned_data = clean_data(df)
cleaned_data
```

```
The non-binary features are '['WEIGHT']'.
Data Cleaned Successfully.
```

[6]:

	GENDER	LUNG_CANCER	SMOKING	AGE	WEIGHT
0	F	YES	1	23	70
1	M	NO	0	45	80
2	F	YES	1	23	60
3	M	NO	0	45	75
4	F	YES	1	23	65
5	M	NO	0	45	85
6	M	NO	0	45	78
7	F	YES	1	23	72

Test validated due to weight being included in the non-binary feature list and duplicated record being removed.

Testing for Transforming data

[7]:
```python
def test_gender_columns(df):
    required_columns = ['is_male', 'is_female']

    for column in required_columns:
        if column not in df.columns:
            print(f"Column '{column}' is missing.")
            return False
        else:
            print(f"Column '{column}' is present.")

    for column in required_columns:
        if not df[column].isin([0, 1]).all():
            print(f"Column '{column}' contains values outside the range [0, 1].
↪")
            return False
        else:
```

```
            print(f"All values in column '{column}' are within the range [0, 1].
↪")

    print("Test passed: DataFrame includes required columns with correct value␣
↪ranges.")
    return True

def test_columns_value_range(df, columns):
    for column in columns:
        if column not in df.columns:
            print(f"Column '{column}' is missing.")
            return False
        if not df[column].between(0, 1).all():
            print(f"Column '{column}' contains values outside the range [0, 1].
↪")
            return False
    print("Test passed: All specified columns have values within the range [0,␣
↪1].")
    return True

test_data = {
    'GENDER': ['F', 'M', 'F', 'M', 'F', 'M', 'M', 'F'],
    'AGE': [23, 45, 23, 45, 23, 45, 45, 23],
    'SMOKING': [1, 2, 1, 2, 1, 2, 2, 1],
    'LUNG_CANCER': ['YES', 'NO', 'YES', 'NO', 'YES', 'NO', 'NO', 'YES']
}

df = pd.DataFrame(test_data)

# Test the function
X_train, X_test, y_train, y_test = transform_data(df)

test_gender_columns(X_train)

test_columns_value_range(X_train, ['SMOKING'])
```

```
The shape of X_train is  (6, 5)
The shape of y_train is  (6,)
The shape of X_test is  (2, 5)
The shape of y_test is  (2,)
Column 'is_male' is present.
Column 'is_female' is present.
All values in column 'is_male' are within the range [0, 1].
All values in column 'is_female' are within the range [0, 1].
Test passed: DataFrame includes required columns with correct value ranges.
Test passed: All specified columns have values within the range [0, 1].
```

[7]: True

2 Final Project Part 4

(a) Given your understanding of your data set along with your newfound learning in the space of feature generation, consider the usefulness of both subject matter driven and data driven feature generation. Provide a thorough analysis.

(b) Identify a specific domain within your dataset that will serve as the foundation for a data-driven inquiry. This might involve examining consumer behavior patterns in retail sales data or tracking changes in environmental parameters over time. Following the identification of this domain, pro- ceed with modeling your data to extract actionable insights that will be of value to your audience. (You may choose to communicate the reliability of results of your modeling phase as a means of transparency in the analysis being shown for your final presentatio

[2]:
```
import os
import pandas as pd
import subprocess
df = pd.read_csv('/Users/felixliang/Downloads/lung_cancer_data/dataset.csv')
```

[3]:
```
df = transform_data(df)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[3], line 1
----> 1 df = transform_data(df)

NameError: name 'transform_data' is not defined
```

[128]:
```
X_train, X_test, y_train, y_test = df[0],df[1],df[2],df[3]
X_train, X_test = X_train.drop(columns = 'GENDER'),X_test.drop(columns =␣
 ↪'GENDER')
X_train
```

[128]:

	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC_DISEASE	\
642	62	0	1	0	0	1	
700	69	1	1	1	1	1	
226	67	1	0	0	1	1	
1697	70	0	1	1	0	1	
1010	32	1	1	0	1	0	
…	…	…	…	…	…	…	
1638	51	1	0	1	1	1	
1095	50	1	1	0	0	0	
1130	68	0	1	1	0	1	
1294	50	0	0	1	0	1	

```
860    78        0               1        1              0                0

      FATIGUE  ALLERGY  WHEEZING  ALCOHOL_CONSUMING  COUGHING  \
642         0        1         1                  0         0
700         1        0         1                  1         1
226         1        0         0                  1         1
1697        0        1         1                  1         1
1010        1        0         0                  0         1
…         …      …       …                …       …
1638        0        1         1                  0         1
1095        1        1         1                  1         0
1130        0        0         1                  1         0
1294        1        1         0                  1         1
860         0        0         0                  1         1

      SHORTNESS_OF_BREATH  SWALLOWING_DIFFICULTY  CHEST_PAIN  is_male  \
642                     0                      0           0        1
700                     0                      1           1        0
226                     1                      1           0        0
1697                    0                      1           1        1
1010                    0                      0           1        1
…                    …                    …         …      …
1638                    0                      1           1        0
1095                    1                      1           0        1
1130                    1                      1           1        1
1294                    1                      1           1        1
860                     1                      1           1        1

      is_female
642           0
700           1
226           1
1697          0
1010          0
…         …
1638          1
1095          0
1130          0
1294          0
860           0

[2400 rows x 16 columns]
```

[129]:
```python
from sklearn.feature_selection import SelectKBest, chi2


chi2_selector = SelectKBest(chi2, k='all')
```

```
chi2_selector.fit(X_train, y_train)


chi2_scores = chi2_selector.scores_
chi2_pvalues = chi2_selector.pvalues_

chi2_results = pd.DataFrame({'Feature': X_train.columns, 'Chi2 Score':␣
↪chi2_scores, 'p-value': chi2_pvalues})
chi2_results = chi2_results.sort_values(by='Chi2 Score', ascending=False)
print(chi2_results)
```

```
                  Feature  Chi2 Score   p-value
0                     AGE   15.046326  0.000105
8                WHEEZING    1.922697  0.165560
10               COUGHING    1.373723  0.241173
11    SHORTNESS_OF_BREATH    0.648398  0.420686
15              is_female    0.549946  0.458340
2           YELLOW_FINGERS    0.542254  0.461500
14                is_male    0.527499  0.467660
9       ALCOHOL_CONSUMING    0.397945  0.528153
4           PEER_PRESSURE    0.332870  0.563974
5         CHRONIC_DISEASE    0.218197  0.640417
3                 ANXIETY    0.208594  0.647871
1                 SMOKING    0.129862  0.718576
13             CHEST_PAIN    0.093655  0.759581
6                 FATIGUE    0.076749  0.781753
12  SWALLOWING_DIFFICULTY    0.076340  0.782320
7                 ALLERGY    0.010806  0.917208
```

Chi-Squared test: A higher Chi-Square statistic indicates a greater deviation from the null hypothesis, suggesting a stronger association between the feature and the target variable.

p-value: The p-value is the probability of observing the Chi-Square statistic as extreme as, or more extreme than, the value calculated, assuming the null hypothesis is true.

- p-value < (e.g., 0.05): Reject the null hypothesis, indicating a statistically significant association between the feature and the target variable.
- p-value : Fail to reject the null hypothesis, suggesting no significant association.

The result suggest that taking off the ALLERGY varibale might help to improve the performance for classification model.

[100]:
```
columns_to_drop = ['AGE']
X_train, X_test = X_train.drop(columns = columns_to_drop, axis = 1), X_test.
↪drop(columns = columns_to_drop,axis=1)
```

2.1 Logistic regression

[101]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix,␣
↪accuracy_score


model = LogisticRegression()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

print(classification_report(y_test, y_pred))
print('Confusion Matrix:\n', confusion_matrix(y_test, y_pred))
print('Accuracy:', accuracy_score(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.51      0.44      0.47       302
           1       0.50      0.58      0.54       298

    accuracy                           0.51       600
   macro avg       0.51      0.51      0.51       600
weighted avg       0.51      0.51      0.51       600

Confusion Matrix:
 [[133 169]
 [126 172]]
Accuracy: 0.5083333333333333
```

2.2 Decision Tree

[102]:
```
from sklearn.tree import DecisionTreeClassifier

tree_model = DecisionTreeClassifier(random_state=42)
tree_model.fit(X_train, y_train)

y_pred_tree = tree_model.predict(X_test)
print(classification_report(y_test, y_pred_tree))
```

```
              precision    recall  f1-score   support

           0       0.48      0.51      0.50       302
           1       0.47      0.44      0.46       298

    accuracy                           0.48       600
   macro avg       0.48      0.48      0.48       600
weighted avg       0.48      0.48      0.48       600
```

2.3 Random Forest

[103]:
```
from sklearn.ensemble import RandomForestClassifier

rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

y_pred_rf = rf_model.predict(X_test)
print(classification_report(y_test, y_pred_rf))
```

```
              precision    recall  f1-score   support

           0       0.47      0.46      0.46       302
           1       0.46      0.47      0.46       298

    accuracy                           0.46       600
   macro avg       0.46      0.46      0.46       600
weighted avg       0.46      0.46      0.46       600
```

2.4 SVM

[104]:
```
from sklearn.svm import SVC

svm_model = SVC(kernel='linear', probability=True, random_state=42)
svm_model.fit(X_train, y_train)

y_pred_svm = svm_model.predict(X_test)
print(classification_report(y_test, y_pred_svm))
```

```
              precision    recall  f1-score   support

           0       0.52      0.52      0.52       302
           1       0.51      0.52      0.52       298

    accuracy                           0.52       600
   macro avg       0.52      0.52      0.52       600
weighted avg       0.52      0.52      0.52       600
```

2.5 Gradient Boosting

[105]:
```python
from sklearn.ensemble import GradientBoostingClassifier

gb_model = GradientBoostingClassifier(n_estimators=100, random_state=42)
gb_model.fit(X_train, y_train)

y_pred_gb = gb_model.predict(X_test)
print(classification_report(y_test, y_pred_gb))
```

```
              precision    recall  f1-score   support

           0       0.52      0.50      0.51       302
           1       0.51      0.52      0.52       298

    accuracy                           0.51       600
   macro avg       0.51      0.51      0.51       600
weighted avg       0.51      0.51      0.51       600
```

2.6 Naive Bayes Classifier

[106]:
```python
from sklearn.naive_bayes import BernoulliNB

nb_model = BernoulliNB()
nb_model.fit(X_train, y_train)

y_pred_nb = nb_model.predict(X_test)
print(classification_report(y_test, y_pred_nb))
```

```
              precision    recall  f1-score   support

           0       0.53      0.47      0.49       302
           1       0.52      0.57      0.54       298

    accuracy                           0.52       600
   macro avg       0.52      0.52      0.52       600
weighted avg       0.52      0.52      0.52       600
```

[ ]:
```python

```