İSTANBUL ÜNİVERSİTESİ-CERRAHPAŞA
MÜHENDİSLİK FAKÜLTESİ

Bilgisayar Mühendisliği Bölümü

Introduction to Machine Learning - BIMU4088

Dr. Öğr. Ü. Erdem İSENKUL

LUNG CANCER DETECTION

M. Burak TOKER – 1306220099

Ali Engin SALEPCİ – 1306200132

Murat KUYUMCU - 1306200129

GÜZ - (2023-2024)

1. DATASET

1.1. Problem Description

Lung cancer is one of the most deadly types of cancer worldwide, and early diagnosis is crucial in addressing this disease. This dataset encompasses various parameters such as patients' demographic information, clinical features, imaging results, and treatment details, aiming to provide in-depth analysis and predictions related to lung cancer. Machine learning models leveraging this dataset have the potential to offer valuable insights into the progression of the disease, treatment responses, and survival durations.

The effectiveness of cancer prediction system helps the people to know their cancer risk with low cost and it also helps the people to take the appropriate decision based on their cancer risk status. The data is collected from the website online lung cancer prediction system .

1.2. Dataset Description

Total number of attributes: 16

Number of instances: 309

Attribute information:

1. Gender: M(male), F(female)
2. Age: Age of the patient
3. Smoking: YES=1 , NO=0.
4. Yellow fingers: YES=1 , NO=0.
5. Anxiety: YES=1 , NO=0.
6. Peer pressure: YES=1 , NO=0.
7. Chronic Disease: YES=1 , NO=0.
8. Fatigue: YES=1 , NO=0.
9. Allergy: YES=1 , NO=0.
10. Wheezing: YES=1 , NO=0.
11. Alcohol: YES=1 , NO=0.
12. Coughing: YES=1 , NO=0.
13. Shortness of Breath: YES=1 , NO=0.
14. Swallowing Difficulty: YES=1 , NO=0.
15. Chest pain: YES=1 , NO=0.
16. Lung Cancer: YES , NO.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL CONSUMING	COUGHING	SHORTNESS OF BREATH	SWALLOWING DIFFICULTY	CHEST PAIN	LUNG_CANCER
2	M	69	0	1	1	0	0	1	0	1	1	1	1	1	1	YES
3	M	74	1	0	0	0	1	1	1	0	0	0	1	1	1	YES
4	F	59	0	0	0	1	0	1	0	1	0	1	1	0	1	NO
5	M	63	1	1	1	0	0	0	0	0	1	0	0	1	1	NO
6	F	63	0	1	0	0	0	0	0	1	0	1	1	0	0	NO
7	F	75	0	1	0	0	1	1	1	1	0	1	1	0	0	YES
8	M	52	1	0	0	0	0	1	0	1	1	1	1	0	1	YES
9	F	51	1	1	1	1	0	1	1	0	0	0	1	1	0	YES
10	F	68	1	0	1	0	0	1	0	0	0	0	0	0	0	NO
11	M	53	1	1	1	1	1	0	1	0	1	0	0	1	1	YES
12	F	61	1	1	1	1	1	1	0	1	0	1	1	1	0	YES
13	M	72	0	0	0	0	1	1	1	1	1	1	1	0	1	YES
14	F	60	1	0	0	0	0	1	0	0	0	0	1	0	0	NO
15	M	58	1	0	0	0	0	1	1	1	1	1	1	0	1	YES
16	M	69	1	0	0	0	0	0	1	1	1	1	0	0	1	NO
17	F	48	0	1	1	1	1	1	1	1	0	1	1	1	0	YES
18	M	75	1	0	0	0	1	0	1	1	1	1	1	0	1	YES
19	M	57	1	1	1	1	1	0	0	0	1	0	0	1	1	YES
20	F	68	1	1	1	1	1	1	0	0	0	1	1	0	0	YES

Image 1.1: *Data Examples*

2. CODE

2.1 Libraries

```
1   import numpy as np
2   import matplotlib.pyplot as plt
3   import pandas as pd
4   from sklearn import metrics
5   from sklearn.compose import ColumnTransformer
6   from sklearn.preprocessing import OneHotEncoder
7   from sklearn.preprocessing import LabelEncoder
8   from sklearn.model_selection import train_test_split
9   from sklearn.preprocessing import StandardScaler
10  from sklearn.neighbors import KNeighborsClassifier
11  from sklearn.metrics import confusion_matrix, accuracy_score
```

Image 2.1: *Libraries*

Libraries needed in this project is imported. These libraries are: NumPy, matplotlib, pandas and Skitlearn.

- NumPy library is used for preprocessing and dataset purposes.
- Matplotlib library is used for graph designs.
- Pandas library is used for storage (dataframe) and data manipulation.
- Sklearn is used for one hot encoding, splitting test and training set, scaling data, KNN model, confusion matrix and accuracy score.

2.2 Data Preprocessing

```
13  dataset=pd.read_excel("survey_lung_cancer.xlsx")
14  dataset=dataset.dropna()
15  x=dataset.iloc[:,:-1].values
16  y=dataset.iloc[:,-1].values
```

Image 2.2: *Splitting Dependent and Independent Columns*

Dataset is imported from excel file via pandas library into dataframe. Then, empty rows are getting deleted from this dataset. After that, we split this dataset into independent variables and dependent variables.

```
#One hot encoding the 'GENDER' column
ct= ColumnTransformer(transformers=[("encoder",OneHotEncoder(),[0])],remainder="passthrough")
x=np.array(ct.fit_transform(x))
le=LabelEncoder()
y=le.fit_transform(y)
```

Image 2.3: *Encoding*

'GENDER' column gets one hot encoded using Sklearn library. Before the encoding this column's value consists of 'M' and 'F'. Also, dependent column gets encoded too. Before the encoding the column's value consists of 'YES' and 'NO'.

2.3 Splitting The Training Set and Test Set

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=1)
sc=StandardScaler()
x_train=sc.fit_transform(x_train)
x_test=sc.fit_transform(x_test)
```

Image 2.4: *Training and Test Set*

Dependent and independent columns gets splitted into training and test sets. After that, independent columns get scaled using Sklearn library. Scaling is used because, age column have values far more bigger than 1 and 0. Other columns takes only 1 and 0 as values.

2.4. Classification Process

```
# Finding elbow value for KNN model
error_rate = {}
for i in range(1,40):

    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(x_train,y_train)
    pred_i = knn.predict(x_test)
    error_rate[i]=np.mean(pred_i != y_test)

optimal_neighbor_count=min(error_rate, key=error_rate.get)
```

Image 2.5: *Calculating Optimum KNN Neighbors Count*

First, ideal number of neighbors gets decided with this piece of code. Code loops through values between 1 and 40 and trains the model with these values in each iteration. Stores the error rate in each iteration with 'error_rate' dictionary. Then, finds the one with minimum error rate in this dictionary. In our case, best neighbor count found 27 thanks to this code.

```
40  # Building a KNN model
41  classifier=KNeighborsClassifier(n_neighbors=optimal_neighbor_count, metric="minkowski",p=2)
42  classifier.fit(x_train,y_train)
43  y_pred=classifier.predict(x_test)
```

Image 2.6: *Building KNN Model*

Model gets created with the ideal number of neighbors, Minkowski distance metric and the power of the distance metric (in our case it is 2). First, training set is fitted into the model. After that, model makes predictions about test set. This process is done using Sklearn library.

2.5 Accuracy and Confusion Matrix

```
45  # Finding results
46  cm=metrics.confusion_matrix(y_test,y_pred)
47  acs=accuracy_score(y_test,y_pred)
48  print("Accuracy: ",acs)
```

Image 2.7: *Finding Result*

Confusion matrix is created using result of the model and our test set via Sklearn library. Accuracy score is found using result of the model and our test set via Sklearn library.

Best accuracy of our model:

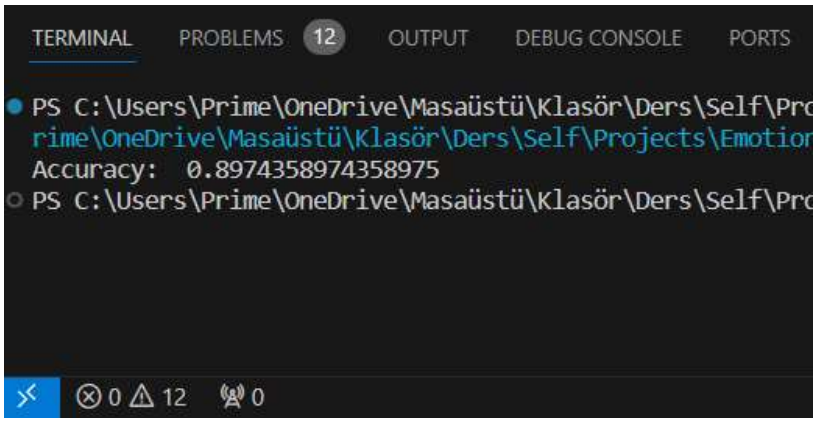

Image 2.8: *Accuracy*

```
# Graph of elbow rule
plt.figure(figsize=(10,6))
plt.plot(range(1,40),error_rate.values(),color='blue', linestyle='dashed', marker='o',
         markerfacecolor='red', markersize=10)
plt.title('Error Rate vs K Value')
plt.xlabel('K')
plt.ylabel('Error Rate')

# Graph of confusion matrix
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = cm, display_labels = [False, True])
cm_display.plot()
plt.show()
```

Image 2.9: *Code of Graphs*

Graph of elbow rule, method finds ideal neighbor count, plotted using matplotlib library. Confusion matrix is displayed using matplotlib library.

Graph of Elbow Rule:

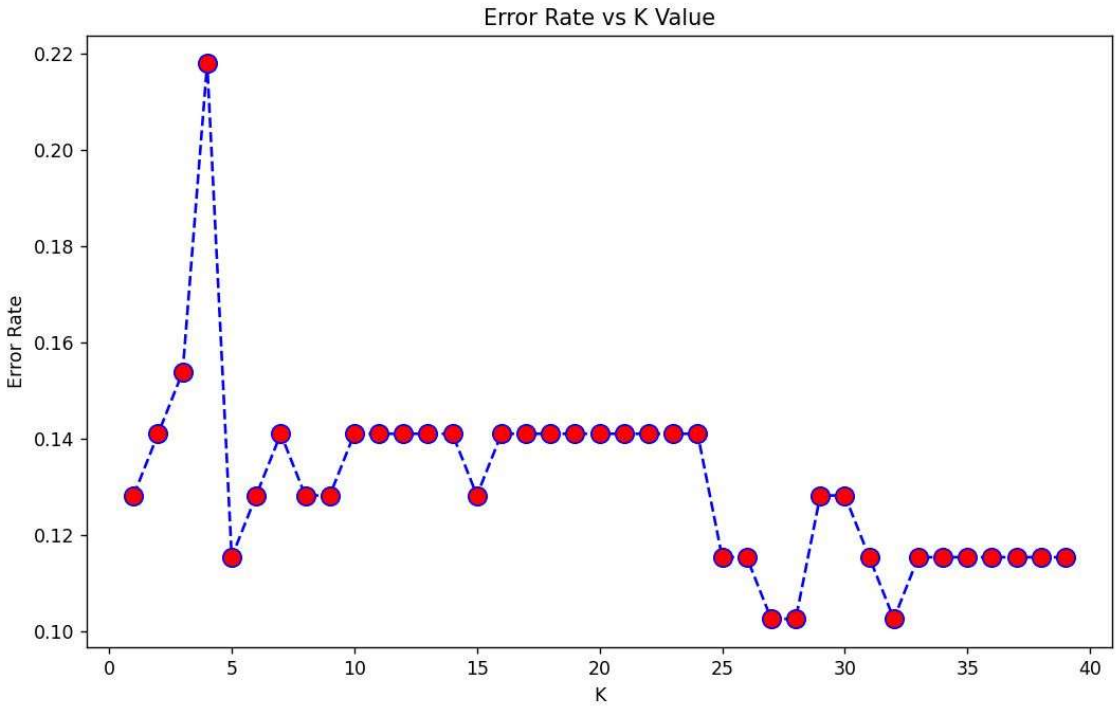

Image 2.10: *Elbow Rule Graph*

Graph of Confusion Matrix:

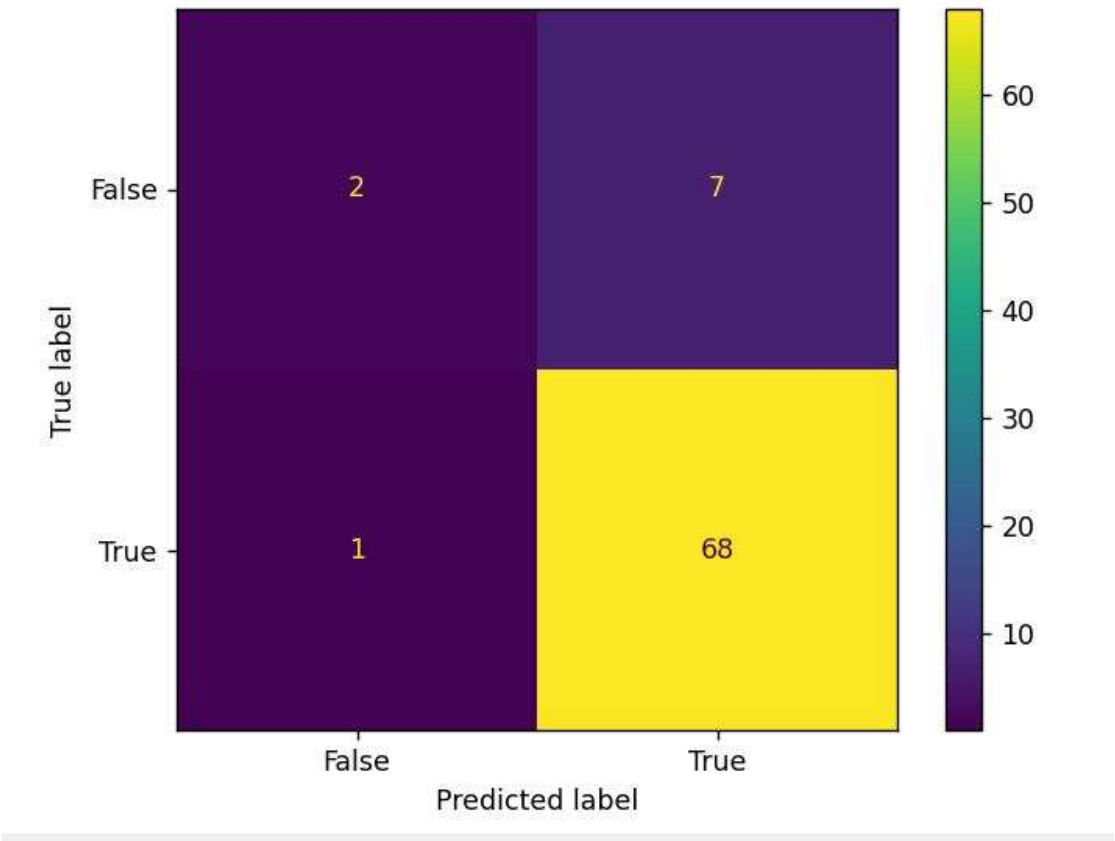

Image 2.11: *Graph of Confusion Matrix*

References

1- BHAT, M.A. “Lung Cancer”. Kaggle. (Dataset) https://www.kaggle.com/datasets/mysarahmadbhat/lung-cancer (2021)

2- National Cancer Institute. “Lung Cancer”. https://www.cancer.gov/types/lung/research/nlst (2014)

3- Doumbia M. “Elbow Method in Supervised Machine Learning(Optimal K Value)”. Medium. https://medium.com/@moussadoumbia_90919/elbow-method-in-supervised-learning-optimal-k-value-99d425f229e7 (2019)

4- Scikit-Learn. “ConfusionMatrixDisplay”. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.ConfusionMatrixDisplay.html

5- Scikit-Learn. “Machine Learning in Python”. https://scikit-learn.org/stable/ (2023)

6- RUTECKI M. “One Hot Encoding - everything you need to know”. Kaggle. https://www.kaggle.com/code/marcinrutecki/one-hot-encoding-everything-you-need-to-know (2023)

7- BAL M.E. “PYTHON İLE VERİ ÖN İŞLEME VE MAKİNE ÖĞRENMESİ”. Youtube. https://www.youtube.com/watch?v=dVed6tskHXY&list=PLz2cqWVPJE62KgRvN7OsU-HAmwxUOPJ4v&index=1 (2020)

8- OJHA Y. “A Beginners guide to Machine Learning — Data Preprocessing” . Medium. https://medium.com/@yogeshojha/data-preprocessing-75485c7188c4 (2019)