Exploratory Data Analysis

IBM-Coursera

Lina Hourieh

Table of Content

IBM-Coursera **0**

Table of Content **1**

About the data 1
Data Dictionary 2
EDA Plan 3
Data Overview 3
Dealing with missing Values 4
Feature Engineering 6
Visualization 8
Categorical Variables Cleaning 9
Remove Duplicates & Unuseful Features 10
Numerical Variables Cleaning 10
Hypothesis Testing 12
Hypothesis 1: 12
Hypothesis 2 13
Hypothesis 3 14
Key Findings 15

About the data

The data is downloaded from Kaggle, from Queen's University Belfast Cancer Research in United Kingdom.

- License CC0: Public Domain
- Visibility: Public
- Date created: 2021-08-05
- Current version: Version 1

Data Dictionary

Patient_ID	string	unique identifier id of a patient
Age	float64	Age at diagnosis (Years)
Gender	string	Male/Female
Protein1, Protein2, Protein3, Protein4	float64	expression levels (undefined units)
Tumour_Stage	string	I, II, III
Histology	string	Infiltrating Ductal Carcinoma, Infiltrating Lobular Carcinoma, Mucinous Carcinoma
ER status	string	Positive/Negative
PR status	string	Positive/Negative
HER2 status	string	Positive/Negative
Surgery_type	string	Lumpectomy, Simple Mastectomy, Modified Radical Mastectomy, Other
Date*of*Surgery	string	Date on which surgery was performed (in DD-MON-YY)
Date*of*Last_Visit	string	Date of last visit (in DD-MON-YY) [can be null, in case the patient didn't visited again after the surgery]
Patient_Status	string	Alive/Dead [can be null, in case the patient didn't visited again after the surgery and there is no information available whether the patient is alive or dead].

EDA Plan

1- Data Overview

Inspect the shape and feature type.

2- Inspect Missing Values

3- Feature Engineering

4- Remove Duplicates & Unuseful Features

5- Cleaning of Categorical Variables

6- Cleaning of Numerical Variables

7- Hypothesis Testing

Data Overview

- **Our target feature** : Patient_Status
- **rows and columns** : 341 , 16

- **features types** : qualitatives : 11 , quantitatives : 5

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 341 entries, 0 to 340
Data columns (total 16 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Patient_ID          334 non-null    object 
 1   Age                 334 non-null    float64
 2   Gender              334 non-null    object 
 3   Protein1            334 non-null    float64
 4   Protein2            334 non-null    float64
 5   Protein3            334 non-null    float64
 6   Protein4            334 non-null    float64
 7   Tumour_Stage        334 non-null    object 
 8   Histology           334 non-null    object 
 9   ER status           334 non-null    object 
 10  PR status           334 non-null    object 
 11  HER2 status         334 non-null    object 
 12  Surgery_type        334 non-null    object 
 13  Date_of_Surgery     334 non-null    object 
 14  Date_of_Last_Visit  317 non-null    object 
 15  Patient_Status      321 non-null    object 
dtypes: float64(5), object(11)
memory usage: 42.8+ KB
```

Dealing with missing Values

```
Total Missing Values:  142
Percentage of Missing Values:  2.602639296187683
```

This is a very low percentage, however, we need to figure out the nature of these missing values and how they are distributed across features.

```
Patient_ID              7
Age                     7
Gender                  7
Protein1                7
Protein2                7
Protein3                7
Protein4                7
Tumour_Stage            7
Histology               7
ER status               7
PR status               7
HER2 status             7
Surgery_type            7
Date_of_Surgery         7
Date_of_Last_Visit     24
Patient_Status         20
dtype: int64
```

Obviously some rows contain all missing values, we can know this by running this line of code:

```
df[df['Patient_ID'].isna() == True]
```

we will find out all features of these rows are NaN. This may be due equipment malfunctions, lost files, or other reasons. we removed them by obtaining their index.

```
df[df['Patient_ID'].isna() == True].index
df.drop(index=[334, 335, 336, 337, 338, 339, 340], inplace=True)
```

Remove rows of null target as well, since they will not help us in the analysis.

```
df[df['Patient_Status'].isna() == True].index
df.drop(index=[7, 22, 99, 111, 182, 196, 206, 219, 221, 285,
286, 305, 321], inplace=True)
```

```
df.isna().sum()
```

```
Patient_ID            0
Age                   0
Gender                0
Protein1              0
Protein2              0
Protein3              0
Protein4              0
Tumour_Stage          0
Histology             0
ER status             0
PR status             0
HER2 status           0
Surgery_type          0
Date_of_Surgery       0
Date_of_Last_Visit    4
Patient_Status        0
dtype: int64
```

We ended up with only 4 missing values.

```
Percentage of Missing Values:  0.0778816199376947
```

We can drop these rows as they have very low percentage or we can fill up the missing values. In order to do that we will **engineer a feature** that will help us.

Feature Engineering

I tried to fill the missing value in the Date_of_Last_Visit using KNN imputer. However, it is not possible to predict its value in this format *09-Nov-18* object. So I followed these steps:

1- Transformed `df['Date_of_Last_Visit']` and `df['Date_of_Surgery']` into datetime.

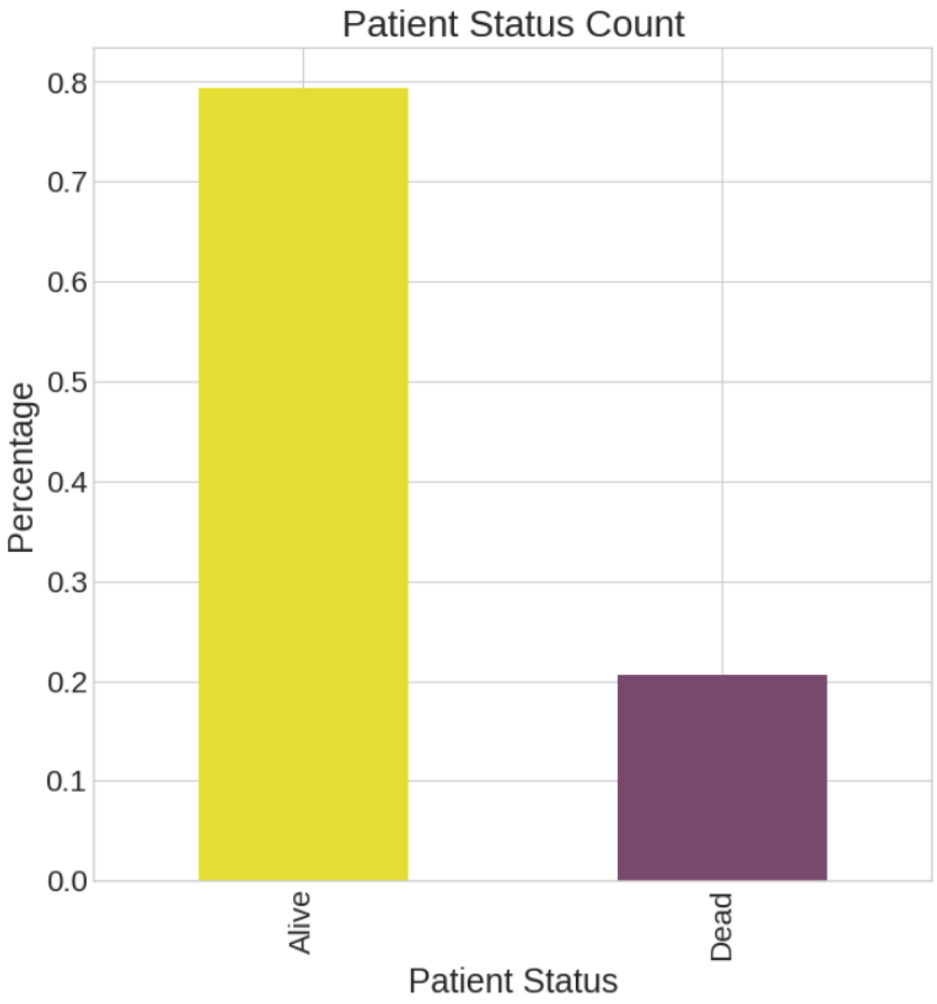
2- Create `df['Recovery_Period']` which is a subtraction of previous pd.series.

3- Use KNN to fill missing values in `df['Recovery_Period']`.

4- Reuse the `df['Recovery_Period'] = df['Date_of_Last_Visit'] - df['Date_of_Surgery']` to fill in missing values in Date of Last Visit

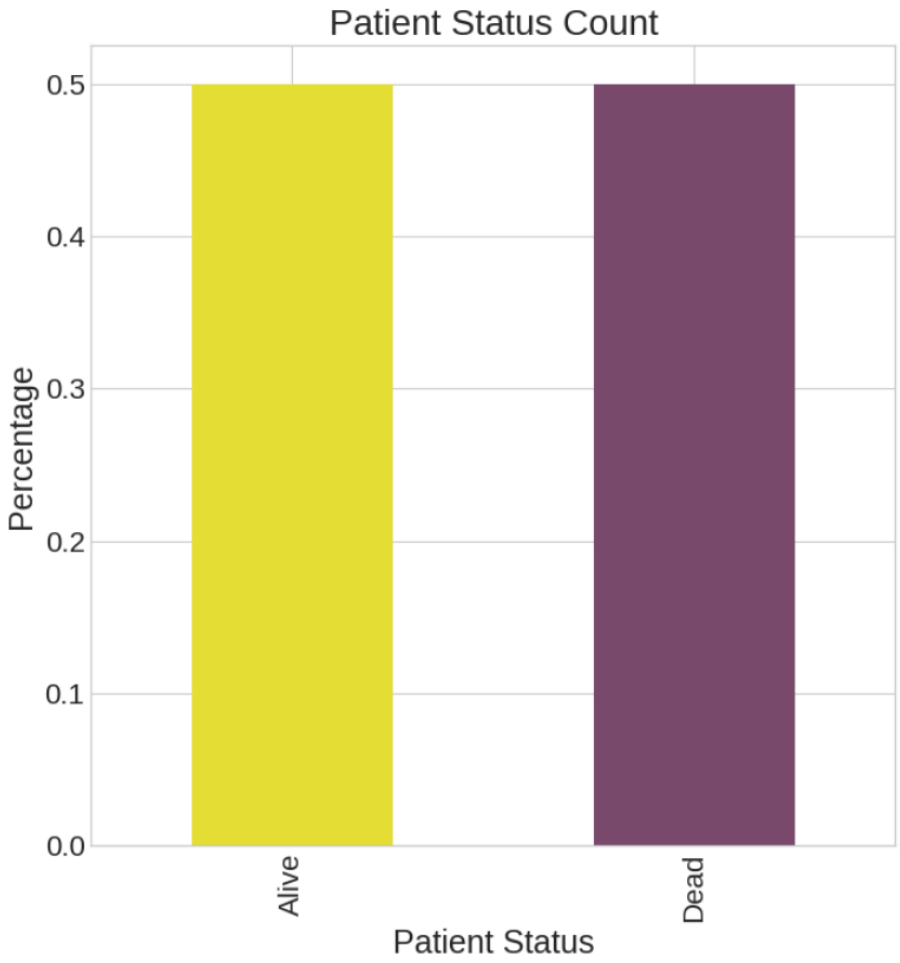
```
Patient_ID            0
Age                   0
Gender                0
Protein1              0
Protein2              0
Protein3              0
Protein4              0
Tumour_Stage          0
Histology             0
ER status             0
PR status             0
HER2 status           0
Surgery_type          0
Date_of_Surgery       0
Date_of_Last_Visit    0
Patient_Status        0
Recovery_Period       0
dtype: int64
```

Visualization



We have a unbalanced outcome. This is usual within medical related data. So undersample.

```
df_alive = df[df['Patient_Status'] ==
'Alive'].sample(df['Patient_Status'].value_counts()[1],random_st
ate=7)
df_dead = df[df['Patient_Status'] == 'Dead']
df = pd.concat([df_alive,df_dead],axis=0)
```



Categorical Variables Cleaning

```
Gender---------------------------------------- ['FEMALE' 'MALE']
Tumour_Stage---------------------------------- ['III' 'II' 'I']
Histology------------------------------------- ['Infiltrating Lobular Carcinoma' 'Infiltrating Ductal Carcinoma'
 'Mucinous Carcinoma']
ER status------------------------------------- ['Positive']
PR status------------------------------------- ['Positive']
HER2 status----------------------------------- ['Negative' 'Positive']
Surgery_type---------------------------------- ['Other' 'Simple Mastectomy' 'Modified Radical Mastectomy' 'Lumpectomy']
Patient_Status-------------------------------- ['Alive' 'Dead']
```

ER and PR status are all positive valuess, so we can remove them along with patient ID as they don't add any value.

Remove Duplicates & Unuseful Features

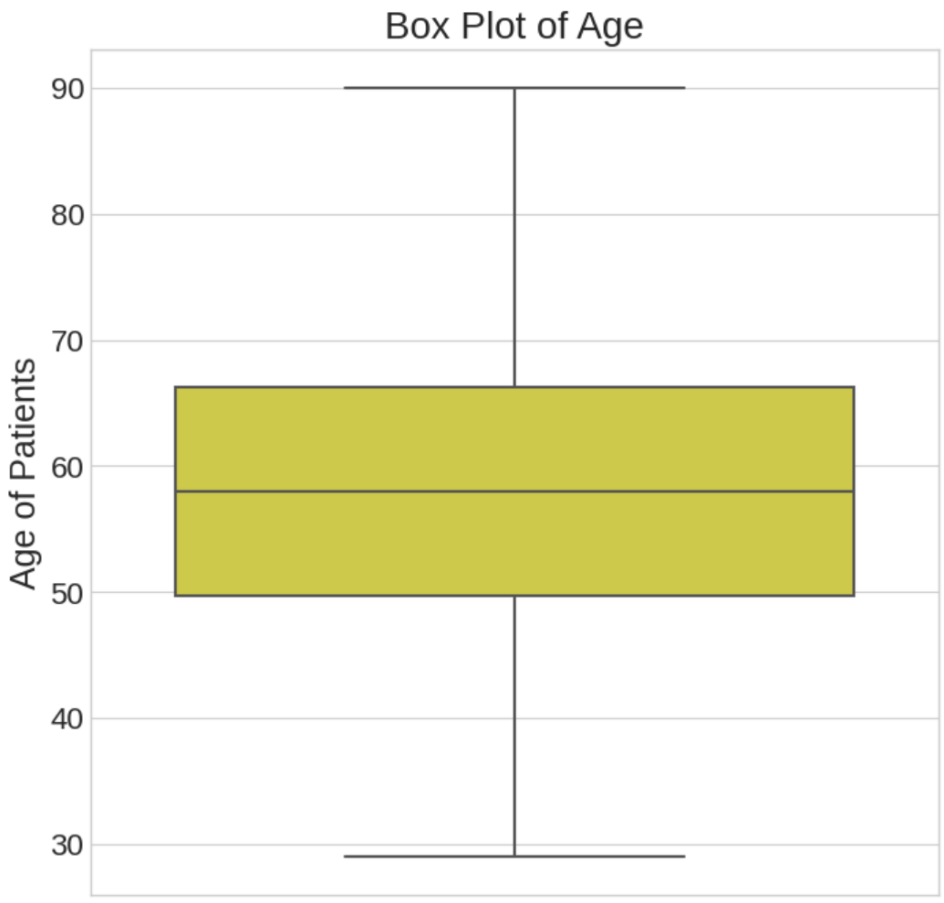
```
df.duplicated().sum()
```

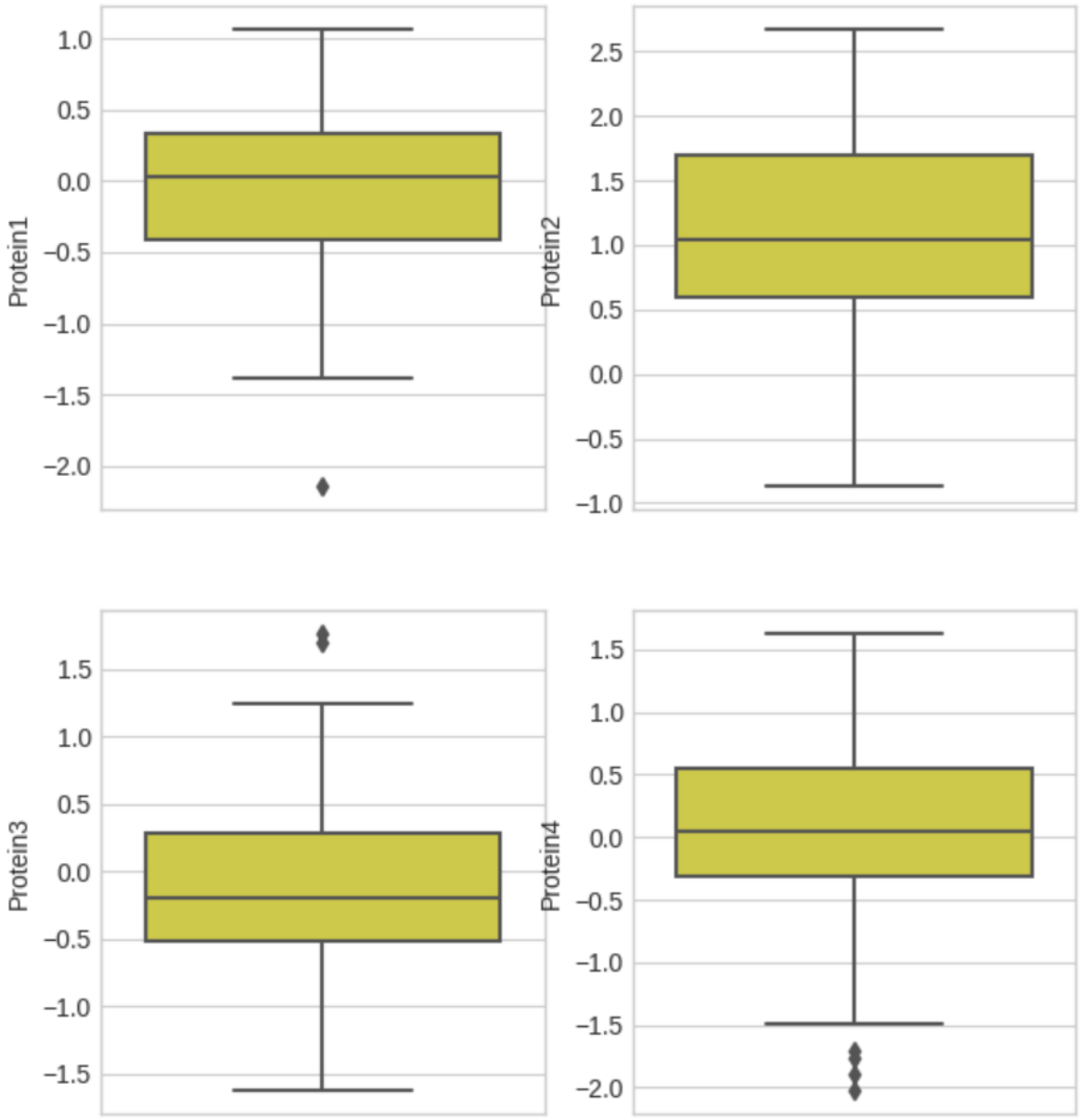
0

```
df.drop(columns=['Patient_ID'],inplace = True)
df.drop(columns=['ER status','PR status'],inplace = True)
```

Numerical Variables Cleaning

```
['Age', 'Protein1', 'Protein2', 'Protein3', 'Protein4']
```






```
def remove_outliers(df,col):
 q_low = df[col].quantile(0.01)
 q_hi  = df[col].quantile(0.99)

 df_filtered = df[(df[col] < q_hi) & (df[col] > q_low)]
 return df_filtered
```

```
df_new = df
for i in num_variables:
 df_filtered = remove_outliers(df_new,i)
 df_new = df_filtered

df = df_new
(112, 14)
```

We removed upper and lower outliers.

Hypothesis Testing

Hypothesis 1:

Surgery_type	Lumpectomy	Modified Radical Mastectomy	Other	Simple Mastectomy
Patient_Status				
Alive	13	18	18	17
Dead	9	20	25	12

H_0 (Null Hypothesis) — the surgery_type and Patient_Status are independent of each other.

H_1 (Alternate Hypothesis) — the surgery_type and Patient_Status are dependent on each other.

And you draw your conclusions based on the following p-value conditions:

$p < 0.05$ — this means the two categorical variables are correlated.

$p > 0.05$ — this means the two categorical variables are not correlated.

```
from scipy.stats import chi2_contingency
```

```
c, p, dof, expected = chi2_contingency(contigency_pct)
print(f'Chi2_score: {c}')
print(f"The p-value is: {p}")
print(f"The degree-of-freedom is: {dof}")
```

```
Chi2_score: 2.834139734405636
The p-value is: 0.4179107006039938
The degree-of-freedom is: 3
```

The p-value is 0.3 > 0.05 which means that we do not reject the null hypothesis at 95% level of confidence. The surgery_type and Patient_Status are independent of each other.

Hypothesis 2

Tumour_Stage	I	II	III
Patient_Status			
Alive	15	33	18
Dead	10	38	18

H_0 (Null Hypothesis) — the Tumour_Stage and Patient_Status are independent of each other.

H_1 (Alternate Hypothesis) — the Tumour_Stage and Patient_Status are dependent on each other.

And you draw your conclusions based on the following p-value conditions:

p < 0.05 — this means the two categorical variables are correlated.

p > 0.05 — this means the two categorical variables are not correlated

```
c, p, dof, expected = chi2_contingency(contigency_pct)
print(f'Chi2_score: {c}')
print(f"The p-value is: {p}")
print(f"The degree-of-freedom is: {dof}")
```

```
Chi2_score: 1.352112676056338
The p-value is: 0.5086188632892799
The degree-of-freedom is: 2
```

The p-value is 0.50 > 0.05 which means that we do not reject the null hypothesis at 95% level of confidence. The Tumour_Stage and Patient_Status are independent of each other.

Hypothesis 3

Histology	Infiltrating Ductal Carcinoma	Infiltrating Lobular Carcinoma	Mucinous Carcinoma
Patient_Status			
Alive	49	15	2
Dead	47	16	3

```
c, p, dof, expected = chi2_contingency(contigency_pct)
print(f'Chi2_score: {c}')
print(f"The p-value is: {p}")
print(f"The degree-of-freedom is: {dof}")
```

```
Chi2_score: 0.2739247311827957
The p-value is: 0.8720030428334034
The degree-of-freedom is: 2
```

The p-value is 0.88 > 0.05 which means that we do not reject the null hypothesis at 95% level of confidence. The Histology and Patient_Status are independent of each other.

Key Findings

- The data contained 142 missing values, some were removed others were filled using KNN.
- The data was unbalanced so we went with undersampling, ending up with 132 sample size.
- No duplications were observed.
- Some outliers were detected and removed