8 - Pandas-inspect_clean

October 14, 2021

Table of Contents

1 Ingest

[55]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib as mpl
import seaborn as sns
from numpy.random import randn


#import os
#for dirname, _, filenames in os.walk('/kaggle/input'):
#    for filename in filenames:
```

```
#         print(os.path.join(dirname, filename))
```

[56]:
```
df = pd.read_csv('/Users/jimcody/Documents/2021Python/intropython/data/
↪diabetes_inspect.csv')
df.head()
```

[56]:

	encounter_id	patient_nbr	race	gender	age	weight	\
0	2278392	8222157	Caucasian	Female	xyz	?	
1	149190	55629189	Caucasian	Female	NaN	?	
2	64410	86047875	AfricanAmerican	female	[20-30)	?	
3	500364	82442376	Caucasian	Mle	[30-40)	?	
4	16680	42519267	Caucasian	M	[40-50)	?	

	admission_type_id	discharge_disposition_id	admission_source_id	\
0	6	25	1	
1	1	1	7	
2	1	1	7	
3	1	1	7	
4	1	1	7	

	time_in_hospital	…	glipizide	glyburide	tolbutamide	miglitol	insulin	\
0	1	…	No	No	No	No	No	
1	3	…	No	No	No	No	Up	
2	2	…	Steady	No	No	No	No	
3	2	…	No	No	No	No	Up	
4	1	…	Steady	No	No	No	Steady	

	glyburide-metformin	glipizide-metformin	glimepiride-pioglitazone	\
0	No	No	No	
1	No	No	No	
2	No	No	No	
3	No	No	No	
4	No	No	No	

	diabetesMed	readmitted
0	No	NO
1	Yes	>30
2	Yes	NO
3	Yes	NO
4	Yes	NO

[5 rows x 33 columns]

2 Inspect and Clean

2.1 Examine the diabetes_inspect data (df)

Things to look for:

- Is there inconsistent data?
  - Are there values spelled differently that are really the same?
  - Are there values that need to be modified?
  - Is the data consistently coded for a variable?
- Do data types need to change?
- Are there any columns with missing values? Can we impute missing values?
- Are there any rows that are duplicated?
- Can distribution plots help identify any 'oddities'?
- Are there outliers? Are they legitimate data points?
- Is there unnecessary data?

[57]:
```
# A cursory look at the data
df.shape
```

[57]:
```
(101766, 33)
```

[58]:
```
# Are we ok with the data types?
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101766 entries, 0 to 101765
Data columns (total 33 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   encounter_id              101766 non-null  int64
 1   patient_nbr               101766 non-null  int64
 2   race                      101766 non-null  object
 3   gender                    101766 non-null  object
 4   age                       101765 non-null  object
 5   weight                    101766 non-null  object
 6   admission_type_id         101766 non-null  int64
 7   discharge_disposition_id  101766 non-null  int64
 8   admission_source_id       101766 non-null  int64
 9   time_in_hospital          101766 non-null  int64
 10  payer_code                101766 non-null  object
 11  medical_specialty         101766 non-null  object
 12  num_lab_procedures        101766 non-null  int64
 13  num_procedures            101766 non-null  int64
 14  num_medications           101757 non-null  float64
 15  number_outpatient         101766 non-null  int64
 16  number_emergency          101766 non-null  int64
 17  number_inpatient          101766 non-null  int64
 18  diag_1                    101766 non-null  object
 19  max_glu_serum             101766 non-null  object
 20  A1Cresult                 101766 non-null  object
```

```
 21  metformin                 101766 non-null  object
 22  glimepiride               101766 non-null  object
 23  glipizide                 101766 non-null  object
 24  glyburide                 101766 non-null  object
 25  tolbutamide               101766 non-null  object
 26  miglitol                  101766 non-null  object
 27  insulin                   101766 non-null  object
 28  glyburide-metformin       101766 non-null  object
 29  glipizide-metformin       101766 non-null  object
 30  glimepiride-pioglitazone  101766 non-null  object
 31  diabetesMed               101766 non-null  object
 32  readmitted                101766 non-null  object
dtypes: float64(1), int64(11), object(21)
memory usage: 25.6+ MB
```

NOTES: - change encounterid, patient id to objects - change admin type id, discharge, admin source id to objects

[59]:
```
# Change/Fix some of the data values

df.loc[df.admission_type_id == 1,'admission_type_id']=100
df.loc[df.gender == 'M','gender']='Male'
df.head()
```

[59]:

	encounter_id	patient_nbr	race	gender	age	weight	\
0	2278392	8222157	Caucasian	Female	xyz	?	
1	149190	55629189	Caucasian	Female	NaN	?	
2	64410	86047875	AfricanAmerican	female	[20-30)	?	
3	500364	82442376	Caucasian	Mle	[30-40)	?	
4	16680	42519267	Caucasian	Male	[40-50)	?	

	admission_type_id	discharge_disposition_id	admission_source_id	\
0	6	25	1	
1	100	1	7	
2	100	1	7	
3	100	1	7	
4	100	1	7	

	time_in_hospital	…	glipizide	glyburide	tolbutamide	miglitol	insulin	\
0	1	…	No	No	No	No	No	
1	3	…	No	No	No	No	Up	
2	2	…	Steady	No	No	No	No	
3	2	…	No	No	No	No	Up	
4	1	…	Steady	No	No	No	Steady	

	glyburide-metformin	glipizide-metformin	glimepiride-pioglitazone	\
0	No	No	No	
1	No	No	No	

```
2                   No                   No                         No
3                   No                   No                         No
4                   No                   No                         No

  diabetesMed readmitted
0          No         NO
1         Yes        >30
2         Yes         NO
3         Yes         NO
4         Yes         NO

[5 rows x 33 columns]
```

[60]:
```
# Change/Fix some of the data values

df['gender'] = df['gender'].replace({'M':'Male', 'Mle':'Male', 'F':'Female'})
df.head()
```

[60]:
```
   encounter_id  patient_nbr             race  gender      age weight  \
0       2278392      8222157        Caucasian  Female      xyz      ?
1        149190     55629189        Caucasian  Female      NaN      ?
2         64410     86047875  AfricanAmerican  female  [20-30)      ?
3        500364     82442376        Caucasian    Male  [30-40)      ?
4         16680     42519267        Caucasian    Male  [40-50)      ?

   admission_type_id  discharge_disposition_id  admission_source_id  \
0                  6                        25                    1
1                100                         1                    7
2                100                         1                    7
3                100                         1                    7
4                100                         1                    7

   time_in_hospital  … glipizide glyburide  tolbutamide  miglitol  insulin  \
0                 1  …        No        No           No        No       No
1                 3  …        No        No           No        No       Up
2                 2  …    Steady        No           No        No       No
3                 2  …        No        No           No        No       Up
4                 1  …    Steady        No           No        No   Steady

   glyburide-metformin  glipizide-metformin  glimepiride-pioglitazone  \
0                   No                   No                        No
1                   No                   No                        No
2                   No                   No                        No
3                   No                   No                        No
4                   No                   No                        No

  diabetesMed readmitted
```

```
0          No         NO
1         Yes        >30
2         Yes         NO
3         Yes         NO
4         Yes         NO

[5 rows x 33 columns]
```

[61]:
```
# Inconsistent capitalization

df['gender'] = df['gender'].apply(lambda x:x.lower())
df.head()
```

[61]:
```
   encounter_id  patient_nbr             race  gender      age weight  \
0       2278392      8222157        Caucasian  female      xyz      ?
1        149190     55629189        Caucasian  female      NaN      ?
2         64410     86047875  AfricanAmerican  female  [20-30)      ?
3        500364     82442376        Caucasian    male  [30-40)      ?
4         16680     42519267        Caucasian    male  [40-50)      ?

   admission_type_id  discharge_disposition_id  admission_source_id  \
0                  6                        25                    1
1                100                         1                    7
2                100                         1                    7
3                100                         1                    7
4                100                         1                    7

   time_in_hospital  … glipizide glyburide  tolbutamide  miglitol  insulin  \
0                 1  …        No        No           No        No       No
1                 3  …        No        No           No        No       Up
2                 2  …    Steady        No           No        No       No
3                 2  …        No        No           No        No       Up
4                 1  …    Steady        No           No        No   Steady

   glyburide-metformin  glipizide-metformin  glimepiride-pioglitazone  \
0                   No                   No                        No
1                   No                   No                        No
2                   No                   No                        No
3                   No                   No                        No
4                   No                   No                        No

  diabetesMed readmitted
0          No         NO
1         Yes        >30
2         Yes         NO
3         Yes         NO
4         Yes         NO
```

```
[5 rows x 33 columns]
```

[62]:
```
# Change data type

df = df.astype({'encounter_id': str, 'patient_nbr': str})
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101766 entries, 0 to 101765
Data columns (total 33 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   encounter_id              101766 non-null  object
 1   patient_nbr               101766 non-null  object
 2   race                      101766 non-null  object
 3   gender                    101766 non-null  object
 4   age                       101765 non-null  object
 5   weight                    101766 non-null  object
 6   admission_type_id         101766 non-null  int64
 7   discharge_disposition_id  101766 non-null  int64
 8   admission_source_id       101766 non-null  int64
 9   time_in_hospital          101766 non-null  int64
 10  payer_code                101766 non-null  object
 11  medical_specialty         101766 non-null  object
 12  num_lab_procedures        101766 non-null  int64
 13  num_procedures            101766 non-null  int64
 14  num_medications           101757 non-null  float64
 15  number_outpatient         101766 non-null  int64
 16  number_emergency          101766 non-null  int64
 17  number_inpatient          101766 non-null  int64
 18  diag_1                    101766 non-null  object
 19  max_glu_serum             101766 non-null  object
 20  A1Cresult                 101766 non-null  object
 21  metformin                 101766 non-null  object
 22  glimepiride               101766 non-null  object
 23  glipizide                 101766 non-null  object
 24  glyburide                 101766 non-null  object
 25  tolbutamide               101766 non-null  object
 26  miglitol                  101766 non-null  object
 27  insulin                   101766 non-null  object
 28  glyburide-metformin       101766 non-null  object
 29  glipizide-metformin       101766 non-null  object
 30  glimepiride-pioglitazone  101766 non-null  object
 31  diabetesMed               101766 non-null  object
 32  readmitted                101766 non-null  object
dtypes: float64(1), int64(9), object(23)
memory usage: 25.6+ MB
```

[64]:
```
# Change data type

df['admission_type_id'] = df['admission_type_id'].astype(str)
df['discharge_disposition_id'] = df['discharge_disposition_id'].astype(str)
df['admission_source_id'] = df['admission_source_id'].astype(str)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101766 entries, 0 to 101765
Data columns (total 33 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   encounter_id              101766 non-null  object
 1   patient_nbr               101766 non-null  object
 2   race                      101766 non-null  object
 3   gender                    101766 non-null  object
 4   age                       101765 non-null  object
 5   weight                    101766 non-null  object
 6   admission_type_id         101766 non-null  object
 7   discharge_disposition_id  101766 non-null  object
 8   admission_source_id       101766 non-null  object
 9   time_in_hospital          101766 non-null  int64
 10  payer_code                101766 non-null  object
 11  medical_specialty         101766 non-null  object
 12  num_lab_procedures        101766 non-null  int64
 13  num_procedures            101766 non-null  int64
 14  num_medications           101757 non-null  float64
 15  number_outpatient         101766 non-null  int64
 16  number_emergency          101766 non-null  int64
 17  number_inpatient          101766 non-null  int64
 18  diag_1                    101766 non-null  object
 19  max_glu_serum             101766 non-null  object
 20  A1Cresult                 101766 non-null  object
 21  metformin                 101766 non-null  object
 22  glimepiride               101766 non-null  object
 23  glipizide                 101766 non-null  object
 24  glyburide                 101766 non-null  object
 25  tolbutamide               101766 non-null  object
 26  miglitol                  101766 non-null  object
 27  insulin                   101766 non-null  object
 28  glyburide-metformin       101766 non-null  object
 29  glipizide-metformin       101766 non-null  object
 30  glimepiride-pioglitazone  101766 non-null  object
 31  diabetesMed               101766 non-null  object
 32  readmitted                101766 non-null  object
dtypes: float64(1), int64(6), object(26)
memory usage: 25.6+ MB
```

[65]:
```
# Rename a few columns

short_names = {'admission_type_id':'admin_type', # creating a dict of the names␣
 ↪to be changed
               'discharge_disposition_id':'discharge_dispo',
               'admission_source_id':'admin_source',
               'num_lab_procedures':'lab_procedures',
               'num_procedures':'procedures'}

df.rename(columns=short_names, inplace=True) # passing the dict to the rename␣
 ↪method
                                            # inplace=True
df.info()

######## Appendix A has an explanation of 'inplace'
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101766 entries, 0 to 101765
Data columns (total 33 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   encounter_id              101766 non-null  object
 1   patient_nbr               101766 non-null  object
 2   race                      101766 non-null  object
 3   gender                    101766 non-null  object
 4   age                       101765 non-null  object
 5   weight                    101766 non-null  object
 6   admin_type                101766 non-null  object
 7   discharge_dispo           101766 non-null  object
 8   admin_source              101766 non-null  object
 9   time_in_hospital          101766 non-null  int64
 10  payer_code                101766 non-null  object
 11  medical_specialty         101766 non-null  object
 12  lab_procedures            101766 non-null  int64
 13  procedures                101766 non-null  int64
 14  num_medications           101757 non-null  float64
 15  number_outpatient         101766 non-null  int64
 16  number_emergency          101766 non-null  int64
 17  number_inpatient          101766 non-null  int64
 18  diag_1                    101766 non-null  object
 19  max_glu_serum             101766 non-null  object
 20  A1Cresult                 101766 non-null  object
 21  metformin                 101766 non-null  object
 22  glimepiride               101766 non-null  object
 23  glipizide                 101766 non-null  object
 24  glyburide                 101766 non-null  object
 25  tolbutamide               101766 non-null  object
 26  miglitol                  101766 non-null  object
```

```
 27  insulin                   101766 non-null  object
 28  glyburide-metformin       101766 non-null  object
 29  glipizide-metformin       101766 non-null  object
 30  glimepiride-pioglitazone  101766 non-null  object
 31  diabetesMed               101766 non-null  object
 32  readmitted                101766 non-null  object
dtypes: float64(1), int64(6), object(26)
memory usage: 25.6+ MB
```

[14]:
```
df['payer_code'].nunique()
```

[14]:
```
18
```

[15]:
```
df['payer_code'].unique()
```

[15]:
```
array(['?', 'MC', 'MD', 'HM', 'UN', 'BC', 'SP', 'CP', 'SI', 'DM', 'CM',
       'CH', 'PO', 'WC', 'OT', 'OG', 'MP', 'FR'], dtype=object)
```

[16]:
```
df['payer_code'].value_counts()
```

[16]:
```
?     40256
MC    32439
HM     6274
SP     5007
BC     4655
MD     3532
CP     2533
UN     2448
CM     1937
OG     1033
PO      592
DM      549
CH      146
WC      135
OT       95
MP       79
SI       55
FR        1
Name: payer_code, dtype: int64
```

[ ]:
```

```

2.1.1 Looking for duplicates

[66]:
```
df.shape
```

[66]:
```
(101766, 33)
```

[67]:
```
# checking for duplicates
df.loc[df.duplicated()]
```

[67]:
```
Empty DataFrame
Columns: [encounter_id, patient_nbr, race, gender, age, weight, admin_type,
discharge_dispo, admin_source, time_in_hospital, payer_code, medical_specialty,
lab_procedures, procedures, num_medications, number_outpatient,
number_emergency, number_inpatient, diag_1, max_glu_serum, A1Cresult, metformin,
glimepiride, glipizide, glyburide, tolbutamide, miglitol, insulin, glyburide-
metformin, glipizide-metformin, glimepiride-pioglitazone, diabetesMed,
readmitted]
Index: []

[0 rows x 33 columns]
```

[68]:
```
# This will drop all duplicate rows

df.drop_duplicates(keep = 'first', inplace = True)

# keep - which duplicate to keep, default is none!

df.loc[df.duplicated()]
```

[68]:
```
Empty DataFrame
Columns: [encounter_id, patient_nbr, race, gender, age, weight, admin_type,
discharge_dispo, admin_source, time_in_hospital, payer_code, medical_specialty,
lab_procedures, procedures, num_medications, number_outpatient,
number_emergency, number_inpatient, diag_1, max_glu_serum, A1Cresult, metformin,
glimepiride, glipizide, glyburide, tolbutamide, miglitol, insulin, glyburide-
metformin, glipizide-metformin, glimepiride-pioglitazone, diabetesMed,
readmitted]
Index: []

[0 rows x 33 columns]
```

[69]:
```
df.shape
```

[69]:
```
(101766, 33)
```

2.1.2 Looking for missing values

[70]:
```
# Just listing the columns and how many rows
# for each have a missing value.

df.isnull().sum()
```

[70]:
```
encounter_id                0
patient_nbr                 0
race                        0
gender                      0
age                         1
weight                      0
admin_type                  0
discharge_dispo             0
admin_source                0
time_in_hospital            0
payer_code                  0
medical_specialty           0
lab_procedures              0
procedures                  0
num_medications             9
number_outpatient           0
number_emergency            0
number_inpatient            0
diag_1                      0
max_glu_serum               0
A1Cresult                   0
metformin                   0
glimepiride                 0
glipizide                   0
glyburide                   0
tolbutamide                 0
miglitol                    0
insulin                     0
glyburide-metformin         0
glipizide-metformin         0
glimepiride-pioglitazone    0
diabetesMed                 0
readmitted                  0
dtype: int64
```

[71]:
```
df_null = df.isna().mean().round(4) * 100

df_null.sort_values(ascending=False).head()
```

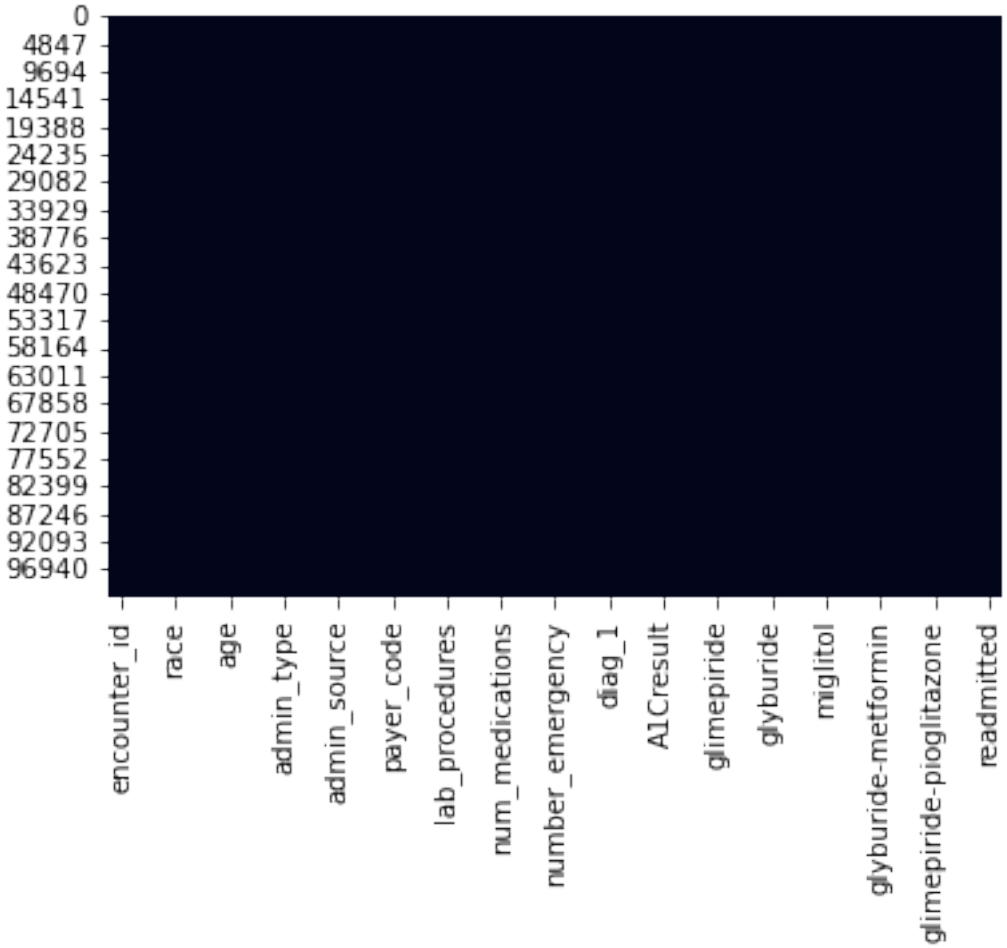
[71]:
```
num_medications    0.01
encounter_id       0.00
glyburide          0.00
max_glu_serum      0.00
A1Cresult          0.00
dtype: float64
```

Note: num_medications is numeric with missing values. This will need to be fixed before using it.

[23]:
```
# Plotting missing values

sns.heatmap(df.isnull(), cbar=False)
```

[23]: <AxesSubplot:>



2.1.3 Imputing missing values

[24]:
```
df['num_medications'].describe()
```

[24]:
```
count    101757.000000
mean         16.021964
std           8.127864
min           1.000000
25%          10.000000
50%          15.000000
75%          20.000000
```

```
max          81.000000
Name: num_medications, dtype: float64
```

[25]:
```
df['num_medications'].median()
```

[25]:
```
15.0
```

[26]:
```
df['num_medications'].mode()
```

[26]:
```
0    13.0
dtype: float64
```

[27]:
```
# Fill missing values of num_medications with the average of num_medications␣
↪(mean)

#df[ 'num_medications' ] = df.num_medications.fillna( df.num_medications.mean()␣
↪)

df.num_medications.fillna( df.num_medications.mean(),inplace=True )
df_null = df.isna().mean().round(4) * 100

df_null.sort_values(ascending=False).head()

# Can be filled with an arbitrary number
# df.num_medications.fillna( 101,inplace=True )

# backward, forward ->  df.fillna(method='bfill') , df.fillna(method='ffill')
```

[27]:
```
encounter_id                0.0
number_inpatient            0.0
diabetesMed                 0.0
glimepiride-pioglitazone    0.0
glipizide-metformin         0.0
dtype: float64
```

[28]:
```
df[ 'num_medications' ]
```

[28]:
```
0          1.000000
1         16.021964
2         13.000000
3         16.021964
4          8.000000
             …
101761    16.000000
101762    18.000000
101763     9.000000
101764    21.000000
```

```
101765     3.000000
Name: num_medications, Length: 101766, dtype: float64
```

2.1.4 Using visuals to get a sense of the data

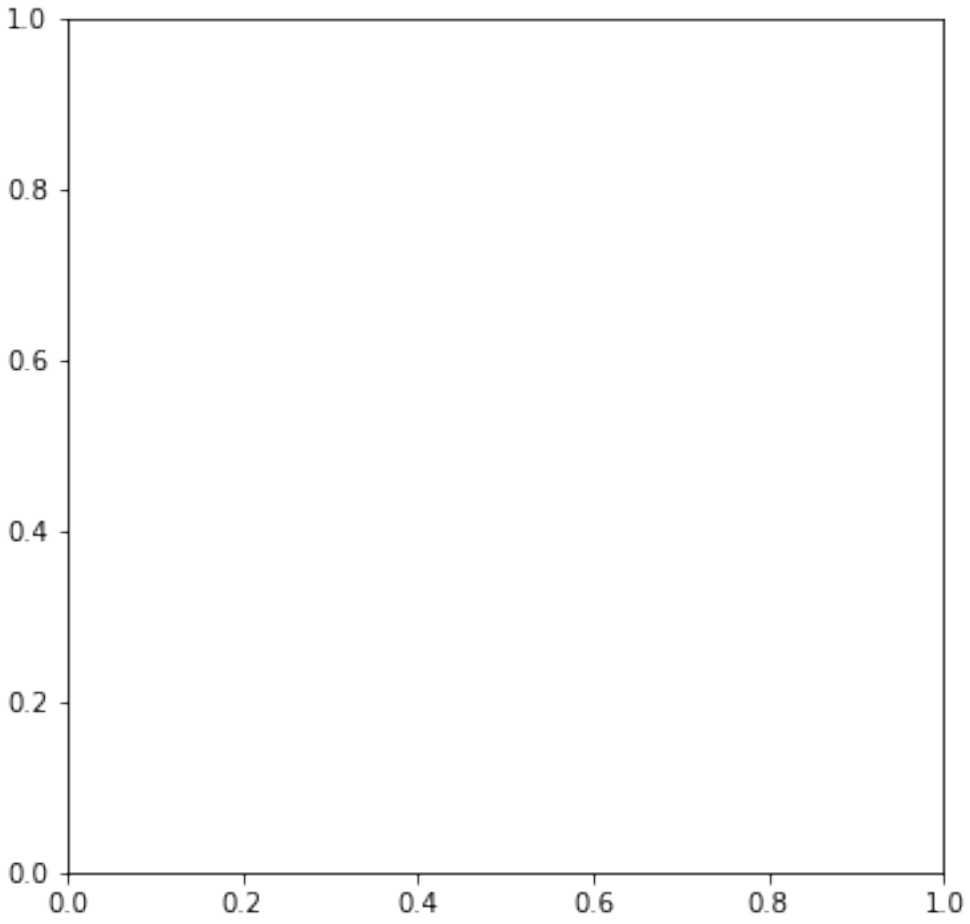
[29]:
```
df.info()
```

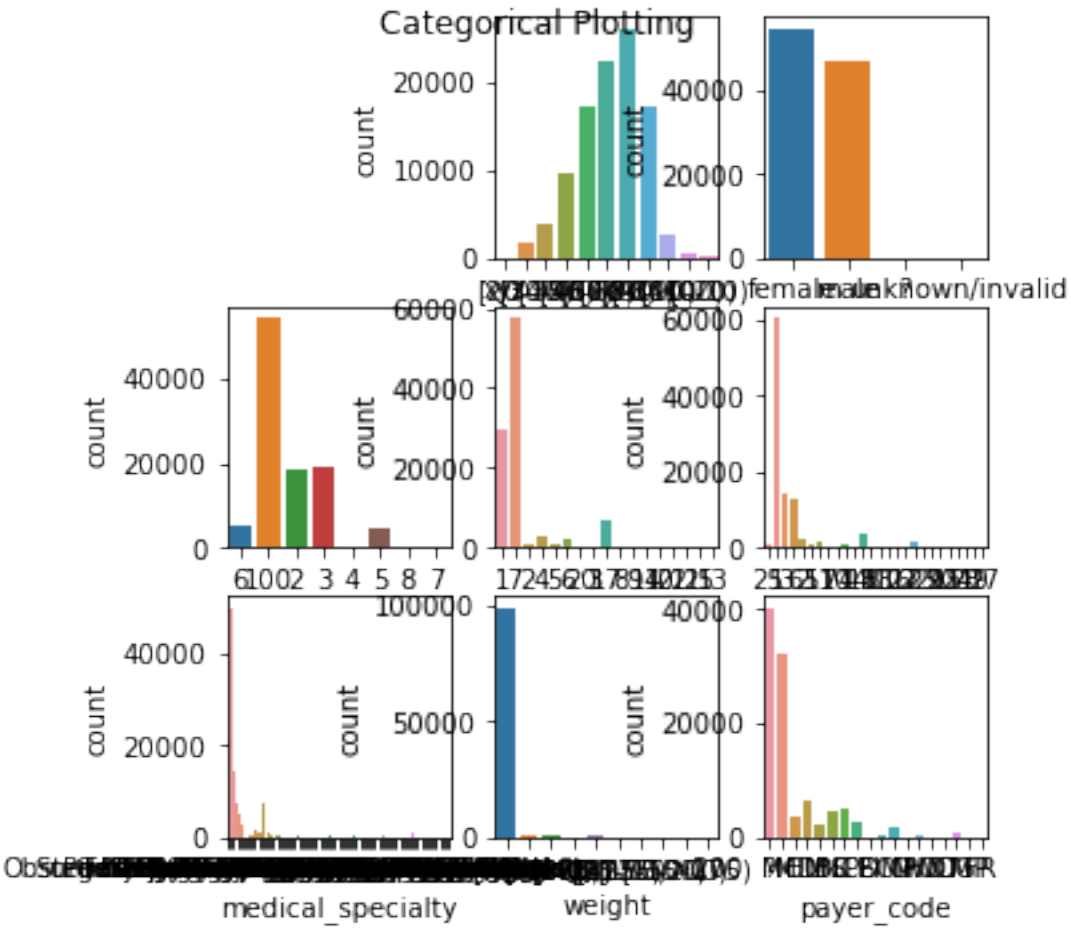
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 101766 entries, 0 to 101765
Data columns (total 33 columns):
 #   Column                    Non-Null Count   Dtype
---  ------                    --------------   -----
 0   encounter_id              101766 non-null  object
 1   patient_nbr               101766 non-null  object
 2   race                      101766 non-null  object
 3   gender                    101766 non-null  object
 4   age                       101765 non-null  object
 5   weight                    101766 non-null  object
 6   admin_type                101766 non-null  object
 7   discharge_dispo           101766 non-null  object
 8   admin_source              101766 non-null  object
 9   time_in_hospital          101766 non-null  int64
 10  payer_code                101766 non-null  object
 11  medical_specialty         101766 non-null  object
 12  lab_procedures            101766 non-null  int64
 13  procedures                101766 non-null  int64
 14  num_medications           101766 non-null  float64
 15  number_outpatient         101766 non-null  int64
 16  number_emergency          101766 non-null  int64
 17  number_inpatient          101766 non-null  int64
 18  diag_1                    101766 non-null  object
 19  max_glu_serum             101766 non-null  object
 20  A1Cresult                 101766 non-null  object
 21  metformin                 101766 non-null  object
 22  glimepiride               101766 non-null  object
 23  glipizide                 101766 non-null  object
 24  glyburide                 101766 non-null  object
 25  tolbutamide               101766 non-null  object
 26  miglitol                  101766 non-null  object
 27  insulin                   101766 non-null  object
 28  glyburide-metformin       101766 non-null  object
 29  glipizide-metformin       101766 non-null  object
 30  glimepiride-pioglitazone  101766 non-null  object
 31  diabetesMed               101766 non-null  object
 32  readmitted                101766 non-null  object
dtypes: float64(1), int64(6), object(26)
memory usage: 26.4+ MB
```

Categorical data

[30]:
```
# Create a bar chart for each categorical variables to see the distribution of␣
 ↪the data
# Cannot use catplot w/ kind=count in a sublplot
plt.figure(figsize = (20,20))
plt.subplot(3,3,1)
sns.catplot(x="race", kind="count", palette="ch:.25", data=df)
#sns.countplot(x="race", data=df)
plt.subplot(3,3,2)
sns.countplot(x="age", data=df)
plt.subplot(3,3,3)
sns.countplot(x="gender", data=df)
plt.subplot(3,3,4)
sns.countplot(x="admin_type", data=df)
plt.subplot(3,3,5)
sns.countplot(x="admin_source", data=df)
plt.subplot(3,3,6)
sns.countplot(x="discharge_dispo", data=df)
plt.subplot(3,3,7)
sns.countplot(x="medical_specialty", data=df)
plt.subplot(3,3,8)
sns.countplot(x="weight", data=df)
plt.subplot(3,3,9)
sns.countplot(x="payer_code", data=df)

plt.suptitle('Categorical Plotting')
plt.show()
```

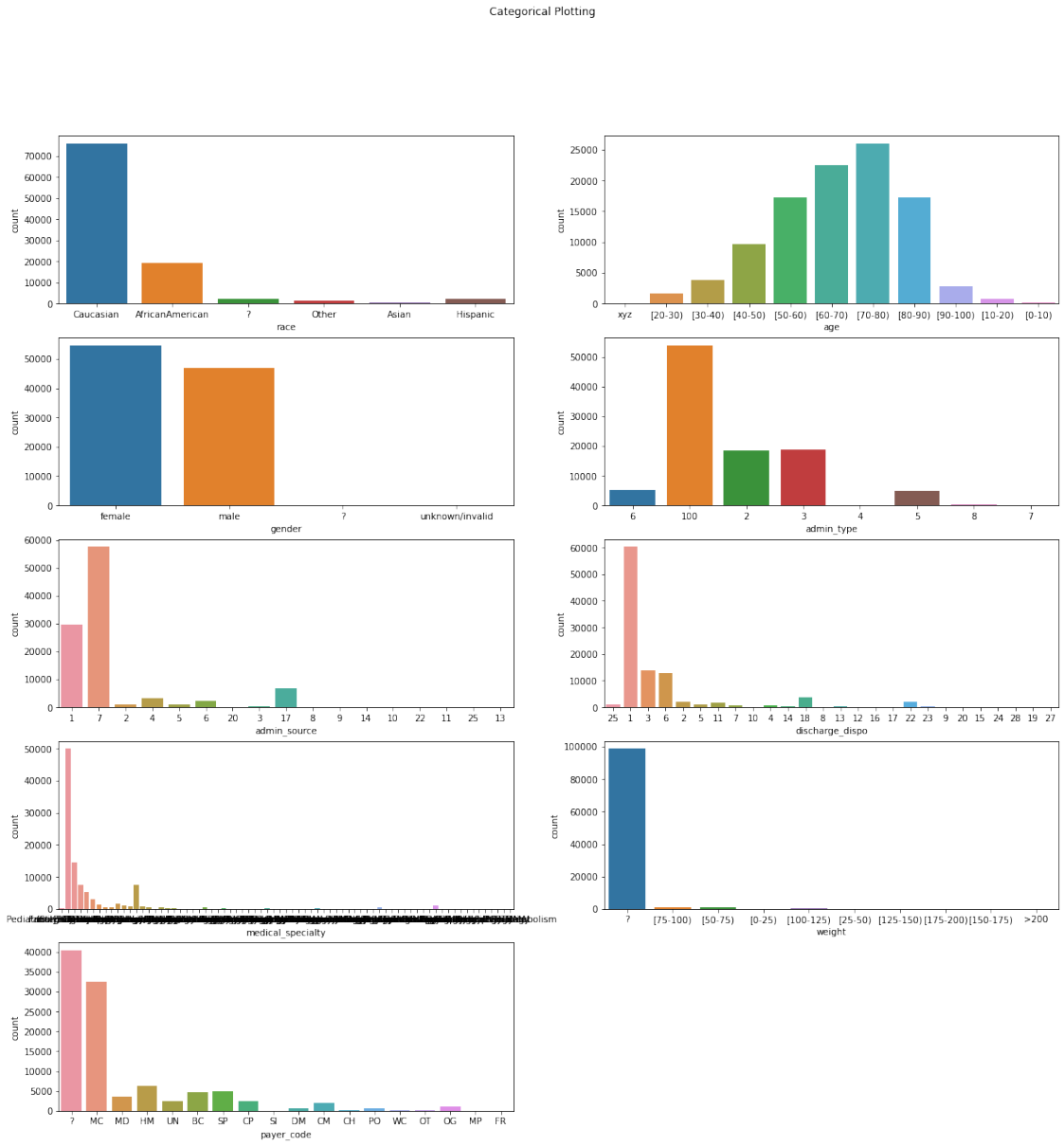

1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0



[31]:
```
# Create a bar chart for each categorical variables to see the distribution of␣
 ↪the data
plt.figure(figsize = (20,20))
plt.subplot(521)
sns.countplot(x="race", data=df)
plt.subplot(522)
sns.countplot(x="age", data=df)
plt.subplot(523)
sns.countplot(x="gender", data=df)
plt.subplot(524)
sns.countplot(x="admin_type", data=df)
plt.subplot(525)
sns.countplot(x="admin_source", data=df)
plt.subplot(526)
sns.countplot(x="discharge_dispo", data=df)
plt.subplot(527)
sns.countplot(x="medical_specialty", data=df)
```

```
plt.subplot(528)
sns.countplot(x="weight", data=df)
plt.subplot(529)
sns.countplot(x="payer_code", data=df)

plt.suptitle('Categorical Plotting')
plt.show()
```



2.1.5 Examine categorical data a little more closely

[32]:
```
for column in df.columns:          # df.columns is a data frame attribute
    print(f"{column}: Number of unique values {df[column].nunique()}")
    print("==========================================================")
```

```
encounter_id: Number of unique values 101766
==========================================================
patient_nbr: Number of unique values 71518
==========================================================
race: Number of unique values 6
==========================================================
gender: Number of unique values 4
==========================================================
age: Number of unique values 11
==========================================================
weight: Number of unique values 10
==========================================================
admin_type: Number of unique values 8
==========================================================
discharge_dispo: Number of unique values 26
==========================================================
admin_source: Number of unique values 17
==========================================================
time_in_hospital: Number of unique values 14
==========================================================
payer_code: Number of unique values 18
==========================================================
medical_specialty: Number of unique values 73
==========================================================
lab_procedures: Number of unique values 118
==========================================================
procedures: Number of unique values 7
==========================================================
num_medications: Number of unique values 76
==========================================================
number_outpatient: Number of unique values 39
==========================================================
number_emergency: Number of unique values 33
==========================================================
number_inpatient: Number of unique values 21
==========================================================
diag_1: Number of unique values 717
==========================================================
max_glu_serum: Number of unique values 4
==========================================================
A1Cresult: Number of unique values 4
==========================================================
```

```
metformin: Number of unique values 4
============================================================
glimepiride: Number of unique values 4
============================================================
glipizide: Number of unique values 4
============================================================
glyburide: Number of unique values 4
============================================================
tolbutamide: Number of unique values 2
============================================================
miglitol: Number of unique values 4
============================================================
insulin: Number of unique values 4
============================================================
glyburide-metformin: Number of unique values 4
============================================================
glipizide-metformin: Number of unique values 2
============================================================
glimepiride-pioglitazone: Number of unique values 2
============================================================
diabetesMed: Number of unique values 2
============================================================
readmitted: Number of unique values 3
============================================================
```

[ ]:
```
object_col = []
for column in df.columns:
    if df[column].dtype == object and len(df[column].unique()) <= 30:
        object_col.append(column)
        print(f"{column} : {df[column].unique()}")
        print(df[column].value_counts())
        print("====================================")
```

[34]:
```
df['payer_code'].nunique()
```

[34]:
```
18
```

[35]:
```
df['payer_code'].value_counts()
```

[35]:
```
?     40256
MC    32439
HM     6274
SP     5007
BC     4655
MD     3532
CP     2533
UN     2448
CM     1937
```

```
OG     1033
PO      592
DM      549
CH      146
WC      135
OT       95
MP       79
SI       55
FR        1
Name: payer_code, dtype: int64
```

[36]:
```
df['medical_specialty'].nunique()
```

[36]:
```
73
```

[37]:
```
df['medical_specialty'].value_counts()
```

[37]:
```
?                               49949
InternalMedicine                14635
Emergency/Trauma                 7565
Family/GeneralPractice           7440
Cardiology                       5352
                                …
SportsMedicine                      1
Speech                              1
Perinatology                        1
Neurophysiology                     1
Pediatrics-InfectiousDiseases       1
Name: medical_specialty, Length: 73, dtype: int64
```

[38]:
```
df['weight'].nunique()
```

[38]:
```
10
```

[39]:
```
df['weight'].value_counts()
```

[39]:
```
?            98569
[75-100)      1336
[50-75)        897
[100-125)      625
[125-150)      145
[25-50)         97
[0-25)          48
[150-175)       35
[175-200)       11
>200             3
Name: weight, dtype: int64
```

2.1.6 Dropping columns and rows

[40]:
```
df.shape
```

[40]:
```
(101766, 33)
```

[41]:
```
# Remove a single column
df = df.drop('payer_code',axis=1)  # Axis=1 means drop the column
df = df.drop('weight',axis=1)

# inplace=True not used so columns still exist. Just not in this instance.
# Fix that.
```

[42]:
```
# Remove multiple columns

# glyburide-metformin
# glipizide-metformin
# glimepiride-pioglitazone

drop_columns = {'medical_specialty','glyburide-metformin','glipizide-metformin',
                'glimepiride-pioglitazone'}
df = df.drop(columns = drop_columns) # inplace=True not used so columns still␣
 ↪exist.
                                     # Just not in this instance.
#df.head()
```

[43]:
```
type
```

[43]:
```
type
```

[44]:
```
# Delete by selecting rows not equal to the condition
df = df.loc[df['age']!= 'xyz']
df = df.loc[df.gender != '?']
#df = df.loc[df['gender']!='?']
#df.shape
```

[45]:
```
no_age = df[df['age'].isnull()].index
#no_age
df = df.drop(no_age, axis = 0)   # axis = 0 means drop the row
df.shape
```

[45]:
```
(101763, 27)
```

Quantitative data

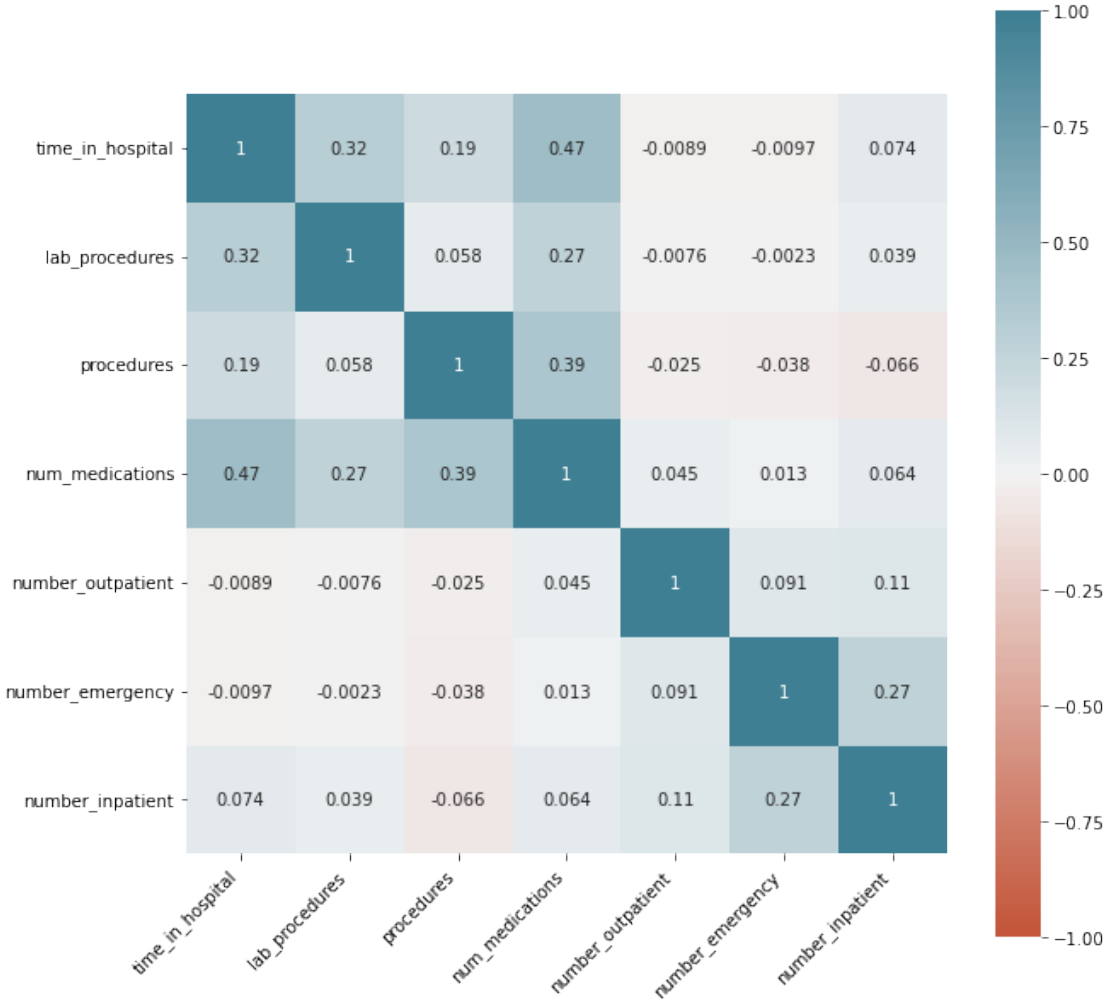
[46]:
```
# Pairplot to see the big picture
# sns.pairplot(df)
```

[47]:
```
# Correlations
df2 = df.corr()
df2
```

[47]:

	time_in_hospital	lab_procedures	procedures	num_medications	number_outpatient	number_emergency	number_inpatient
time_in_hospital	1.000000	0.318450	0.191470	0.466129	-0.008918	-0.009683	0.073620
lab_procedures	0.318450	1.000000	0.058079	0.268160	-0.007598	-0.002276	0.039240
procedures	0.191470	0.058079	1.000000	0.385762	-0.024826	-0.038185	-0.066249
num_medications	0.466129	0.268160	0.385762	1.000000	0.045187	0.013173	0.064177
number_outpatient	-0.008918	-0.007598	-0.024826	0.045187	1.000000	0.091457	0.107334
number_emergency	-0.009683	-0.002276	-0.038185	0.013173	0.091457	1.000000	0.266557
number_inpatient	0.073620	0.039240	-0.066249	0.064177	0.107334	0.266557	1.000000

[48]:
```
plt.figure(figsize=(10,10))
corr = df2.corr()
ax = sns.heatmap(
    df2,
    vmin=-1, vmax=1, center=0,
    cmap=sns.diverging_palette(20, 220, n=200),
    square=True,
    annot=True, annot_kws={"size":10}
)
ax.set_xticklabels(
    ax.get_xticklabels(),
    rotation=45,
    horizontalalignment='right')
plt.show()
```

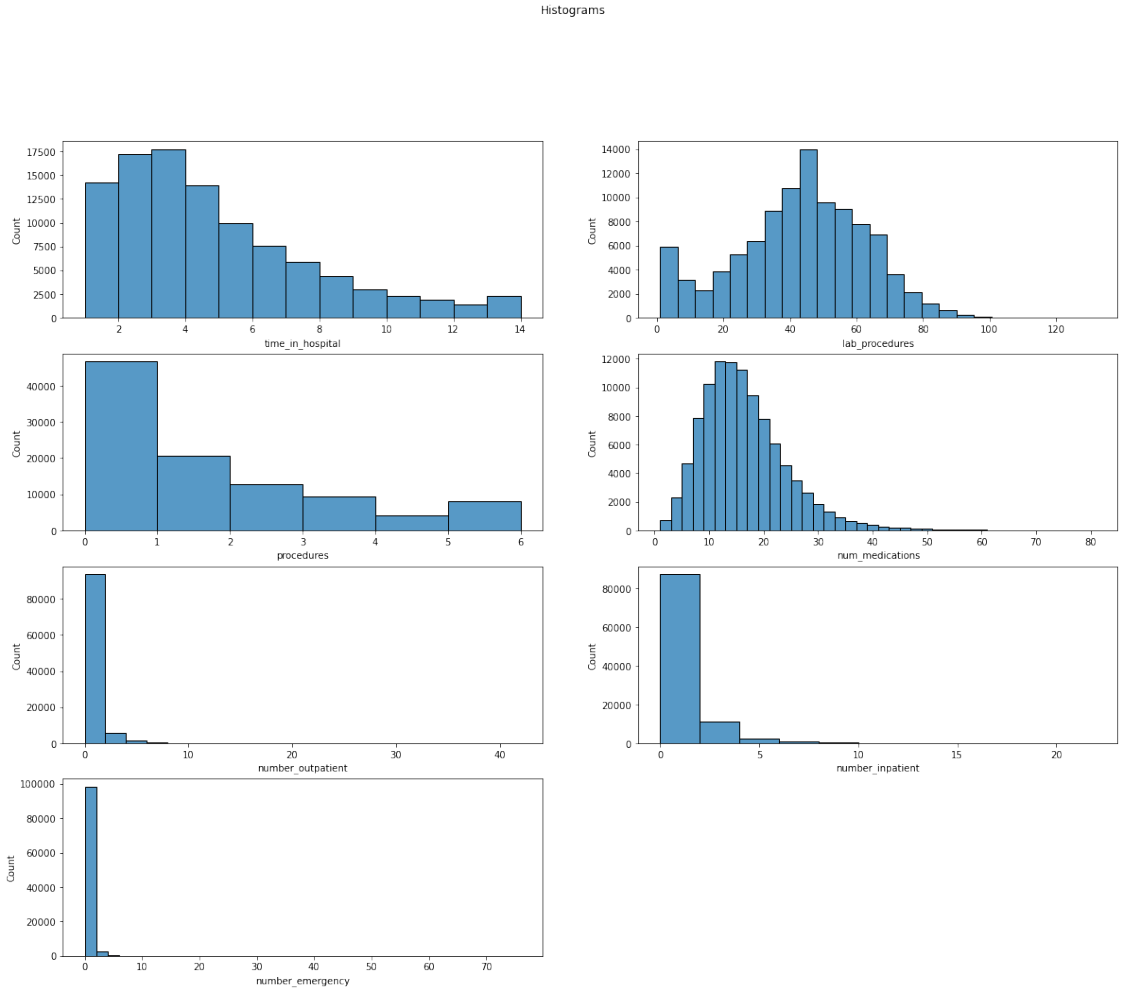


[49]:
```
# Histograms

plt.figure(figsize = (20,20))
plt.subplot(521)
sns.histplot(data=df, x='time_in_hospital', binwidth = 1)
plt.subplot(522)
sns.histplot(data=df, x='lab_procedures', bins=25)
plt.subplot(523)
sns.histplot(data=df, x='procedures', binwidth = 1)
plt.subplot(524)
sns.histplot(data=df, x='num_medications', binwidth = 2)
plt.subplot(525)
sns.histplot(data=df, x='number_outpatient', binwidth = 2)
plt.subplot(526)
sns.histplot(data=df, x='number_inpatient', binwidth = 2)
```

```
plt.subplot(527)
sns.histplot(data=df, x='number_emergency', binwidth = 2)


plt.suptitle('Histograms')
plt.show()
```

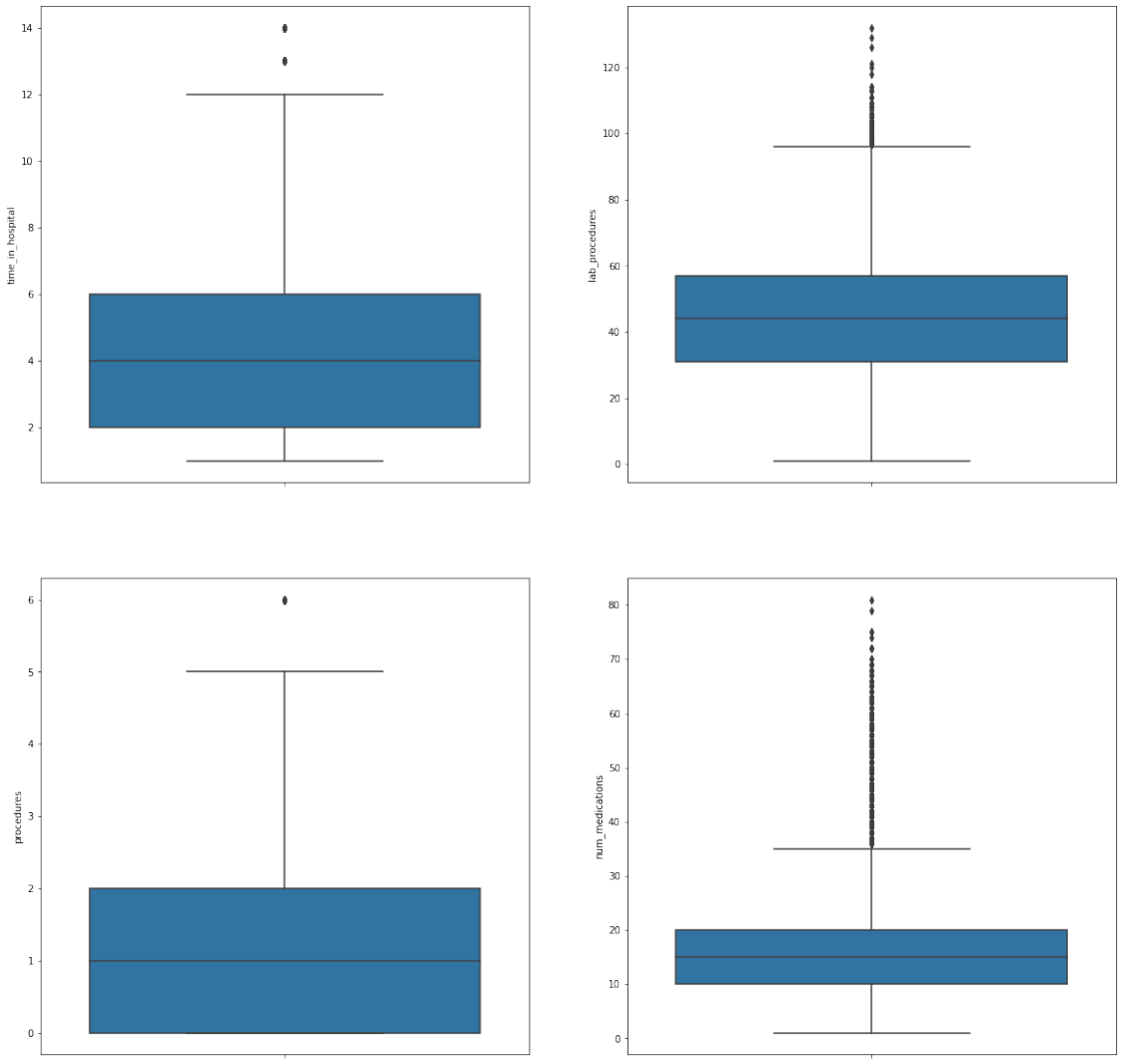


[50]:
```
# Focusing on a few variables

plt.figure(figsize = (20,20))
plt.subplot(221)
sns.boxplot(data=df, y="time_in_hospital")
plt.subplot(222)
sns.boxplot(data=df, y="lab_procedures")
plt.subplot(223)
sns.boxplot(data=df, y="procedures")
```

```
plt.subplot(224)
sns.boxplot(data=df, y="num_medications")
```

[50]: <AxesSubplot:ylabel='num_medications'>

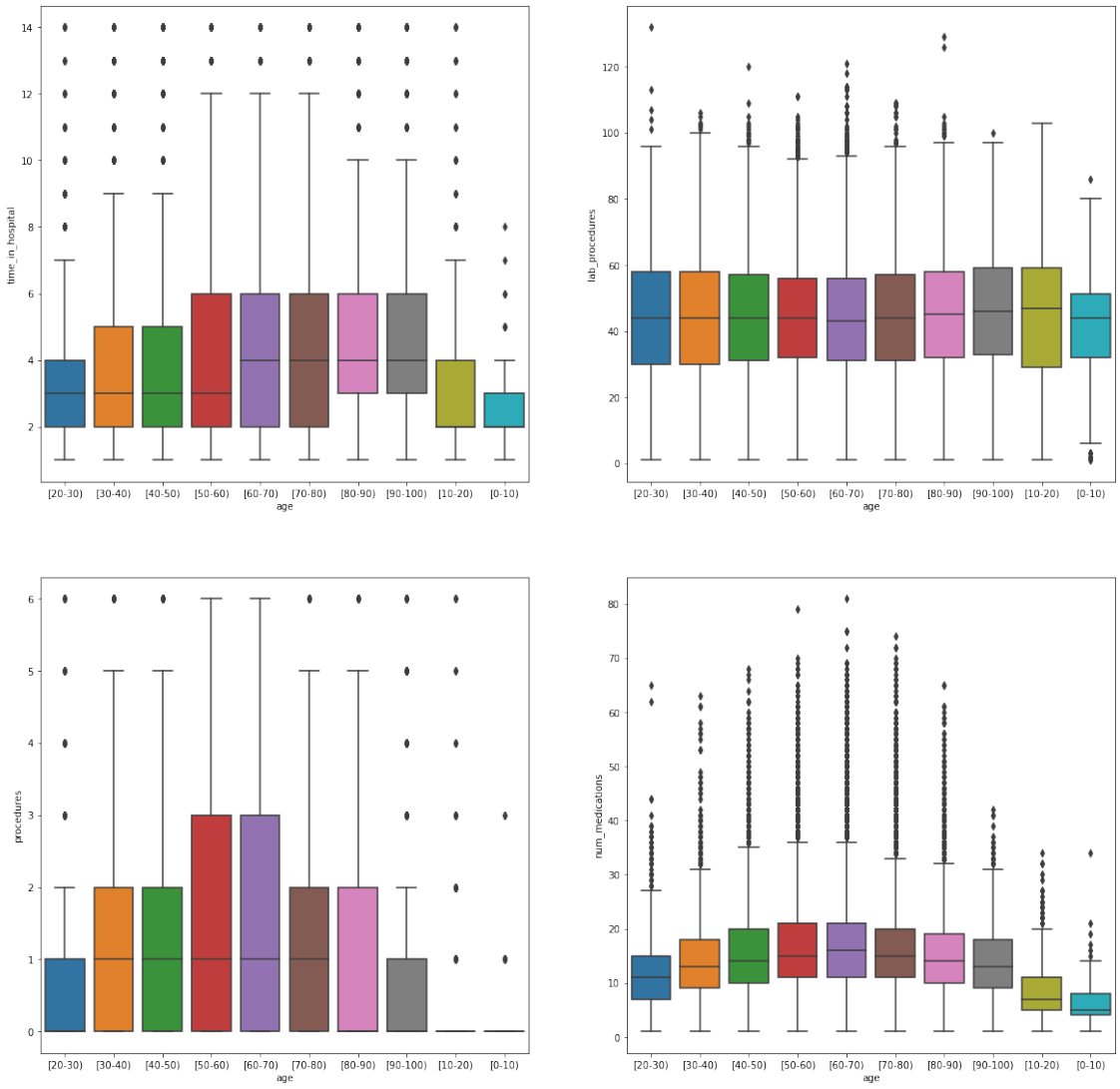


[51]:
```
# Focusing on a few variables

plt.figure(figsize = (20,20))
plt.subplot(221)
sns.boxplot(data=df, x='age', y="time_in_hospital")
plt.subplot(222)
sns.boxplot(data=df, x='age', y="lab_procedures")
plt.subplot(223)
sns.boxplot(data=df, x='age', y="procedures")
```

```
plt.subplot(224)
sns.boxplot(data=df, x='age', y="num_medications")
```

[51]: <AxesSubplot:xlabel='age', ylabel='num_medications'>



2.1.7 Removing outliers

[52]:
```
#outliers
dfoutliers = df[(df['num_medications']>70)]
dfoutliers.shape
#filtering outliers out
#df_movie = df_movie[(df_movie['minute']>43) & (df_movie['minute']<158)]
```

[52]: (8, 27)

3 Exercise - 30 minutes

3.0.1 See Beer Notebook - Part 1

4 Appendix A: 'inplace'

[53]:
```
#import pandas as pd
#import numpy as np
client_dictionary = {'lastname': ['Smith','Chu','White','Patel','Borgini'],
                     'firstname': [None, 'Jenny', 'Ben', 'Akshay', 'Elisa'],
                     'city': ['Chicago', 'New York', 'Atlanta', 'New York',␣
 ↪'Rome'],
                     'age': [27, 35, 56, None, 28],
                     'wins': [0, None, 9, 1, 12]}
df1 = pd.DataFrame(client_dictionary)
df2 = pd.DataFrame(client_dictionary)


df1.head()

df2.head()


df1.dropna(inplace=True) # When inplace is True, no output

# Confirm it worked
df1.head()

df2.dropna(inplace=False) # When inplace = False there is output.

# Confirm it worked
df2.head()

# inplace = False returns a dataframe with the Nans dropped but...
# This is a new object.  It is not the original df2 dataframe.

# inplace = True re-assigns the new object to the original.
# It is the same as writing - df2 = df2.dropna(inplace=False)
```

[53]:

	lastname	firstname	city	age	wins
0	Smith	None	Chicago	27.0	0.0
1	Chu	Jenny	New York	35.0	NaN
2	White	Ben	Atlanta	56.0	9.0
3	Patel	Akshay	New York	NaN	1.0
4	Borgini	Elisa	Rome	28.0	12.0