In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import pearsonr
import seaborn as sns

data = pd.read_csv(r"C:/Users/Nishi/Downloads/diabetic_data-1.csv")
```

In [2]:
```
data.shape
```

Out[2]:
```
(101766, 50)
```

In [3]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101766 entries, 0 to 101765
Data columns (total 50 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                       101766 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           101766 non-null  int64 
 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  diag_2                    101766 non-null  object
 20  diag_3                    101766 non-null  object
 21  number_diagnoses          101766 non-null  int64 
 22  max_glu_serum             5346 non-null    object
 23  A1Cresult                 17018 non-null   object
 24  metformin                 101766 non-null  object
 25  repaglinide               101766 non-null  object
 26  nateglinide               101766 non-null  object
 27  chlorpropamide            101766 non-null  object
 28  glimepiride               101766 non-null  object
 29  acetohexamide             101766 non-null  object
 30  glipizide                 101766 non-null  object
 31  glyburide                 101766 non-null  object
 32  tolbutamide               101766 non-null  object
 33  pioglitazone              101766 non-null  object
 34  rosiglitazone             101766 non-null  object
 35  acarbose                  101766 non-null  object
 36  miglitol                  101766 non-null  object
 37  troglitazone              101766 non-null  object
 38  tolazamide                101766 non-null  object
 39  examide                   101766 non-null  object
 40  citoglipton               101766 non-null  object
 41  insulin                   101766 non-null  object
 42  glyburide-metformin       101766 non-null  object
 43  glipizide-metformin       101766 non-null  object
 44  glimepiride-pioglitazone  101766 non-null  object
 45  metformin-rosiglitazone   101766 non-null  object
 46  metformin-pioglitazone    101766 non-null  object
 47  change                    101766 non-null  object
 48  diabetesMed               101766 non-null  object
 49  readmitted                101766 non-null  object
```

```
dtypes: int64(13), object(37)
memory usage: 38.8+ MB
```

In [4]:
```
nan_counts = data.isna().sum()
print(nan_counts)
```

```
encounter_id                    0
patient_nbr                     0
race                            0
gender                          0
age                             0
weight                          0
admission_type_id               0
discharge_disposition_id        0
admission_source_id             0
time_in_hospital                0
payer_code                      0
medical_specialty               0
num_lab_procedures              0
num_procedures                  0
num_medications                 0
number_outpatient               0
number_emergency                0
number_inpatient                0
diag_1                          0
diag_2                          0
diag_3                          0
number_diagnoses                0
max_glu_serum               96420
A1Cresult                   84748
metformin                       0
repaglinide                     0
nateglinide                     0
chlorpropamide                  0
glimepiride                     0
acetohexamide                   0
glipizide                       0
glyburide                       0
tolbutamide                     0
pioglitazone                    0
rosiglitazone                   0
acarbose                        0
miglitol                        0
troglitazone                    0
tolazamide                      0
examide                         0
citoglipton                     0
insulin                         0
glyburide-metformin             0
glipizide-metformin             0
glimepiride-pioglitazone        0
metformin-rosiglitazone         0
metformin-pioglitazone          0
change                          0
diabetesMed                     0
readmitted                      0
dtype: int64
```

In [5]:
```
data.max_glu_serum.value_counts()
```

```
Out[5]: max_glu_serum
        Norm    2597
        >200    1485
        >300    1264
        Name: count, dtype: int64
```

```
In [6]: data.max_glu_serum.mode()
```

```
Out[6]: 0    Norm
        Name: max_glu_serum, dtype: object
```

```
In [7]: data.A1Cresult.value_counts()
```

```
Out[7]: A1Cresult
        >8      8216
        Norm    4990
        >7      3812
        Name: count, dtype: int64
```

```
In [8]: data.head(20)
```

```
Out[8]:
```

	encounter_id	patient_nbr	race	gender	age	weight	admission_type_id	d
0	2278392	8222157	Caucasian	Female	[0-10)	?	6	
1	149190	55629189	Caucasian	Female	[10-20)	?	1	
2	64410	86047875	AfricanAmerican	Female	[20-30)	?	1	
3	500364	82442376	Caucasian	Male	[30-40)	?	1	
4	16680	42519267	Caucasian	Male	[40-50)	?	1	
5	35754	82637451	Caucasian	Male	[50-60)	?	2	
6	55842	84259809	Caucasian	Male	[60-70)	?	3	
7	63768	114882984	Caucasian	Male	[70-80)	?	1	
8	12522	48330783	Caucasian	Female	[80-90)	?	2	
9	15738	63555939	Caucasian	Female	[90-100)	?	3	
10	28236	89869032	AfricanAmerican	Female	[40-50)	?	1	
11	36900	77391171	AfricanAmerican	Male	[60-70)	?	2	
12	40926	85504905	Caucasian	Female	[40-50)	?	1	
13	42570	77586282	Caucasian	Male	[80-90)	?	1	
14	62256	49726791	AfricanAmerican	Female	[60-70)	?	3	
15	73578	86328819	AfricanAmerican	Male	[60-70)	?	1	
16	77076	92519352	AfricanAmerican	Male	[50-60)	?	1	
17	84222	108662661	Caucasian	Female	[50-60)	?	1	
18	89682	107389323	AfricanAmerican	Male	[70-80)	?	1	

	encounter_id	patient_nbr	race	gender	age	weight	admission_type_id	d
19	148530	69422211	?	Male	[70-80)	?	3	

20 rows × 50 columns

In [9]:
```
data.weight.value_counts()
```

Out[9]:
```
weight
?            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: count, dtype: int64
```

In [10]:
```
data.drop('max_glu_serum', axis=1, inplace=True)
data.drop('A1Cresult', axis=1, inplace=True)
data.drop('weight', axis=1, inplace=True)
```

In [11]:
```
data.age.value_counts()
```

Out[11]:
```
age
[70-80)     26068
[60-70)     22483
[50-60)     17256
[80-90)     17197
[40-50)      9685
[30-40)      3775
[90-100)     2793
[20-30)      1657
[10-20)       691
[0-10)        161
Name: count, dtype: int64
```

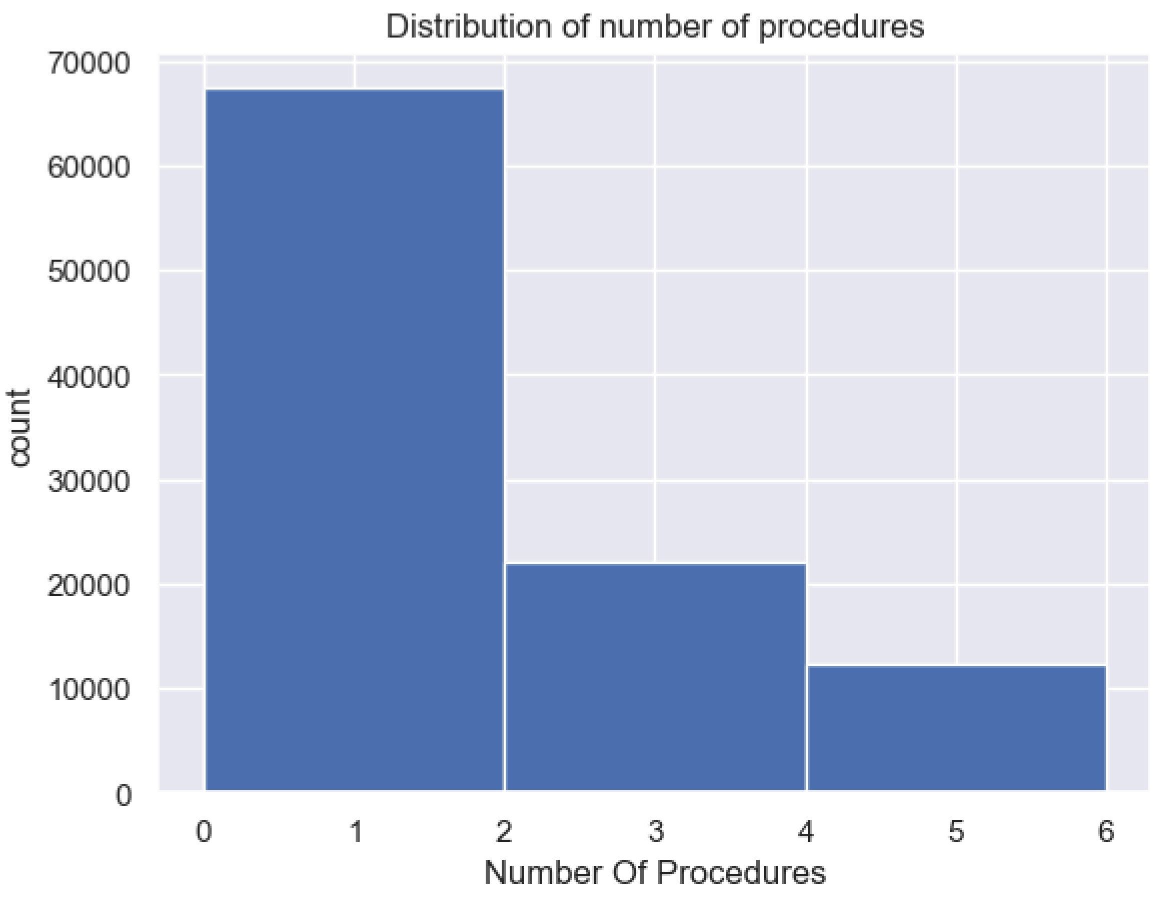
In [12]:
```
data.time_in_hospital.value_counts()
```

```
Out[12]:  time_in_hospital
          3     17756
          2     17224
          1     14208
          4     13924
          5      9966
          6      7539
          7      5859
          8      4391
          9      3002
          10     2342
          11     1855
          12     1448
          13     1210
          14     1042
          Name: count, dtype: int64
```

```
In [13]:  data.num_procedures.value_counts()
```

```
Out[13]:  num_procedures
          0    46652
          1    20742
          2    12717
          3     9443
          6     4954
          4     4180
          5     3078
          Name: count, dtype: int64
```

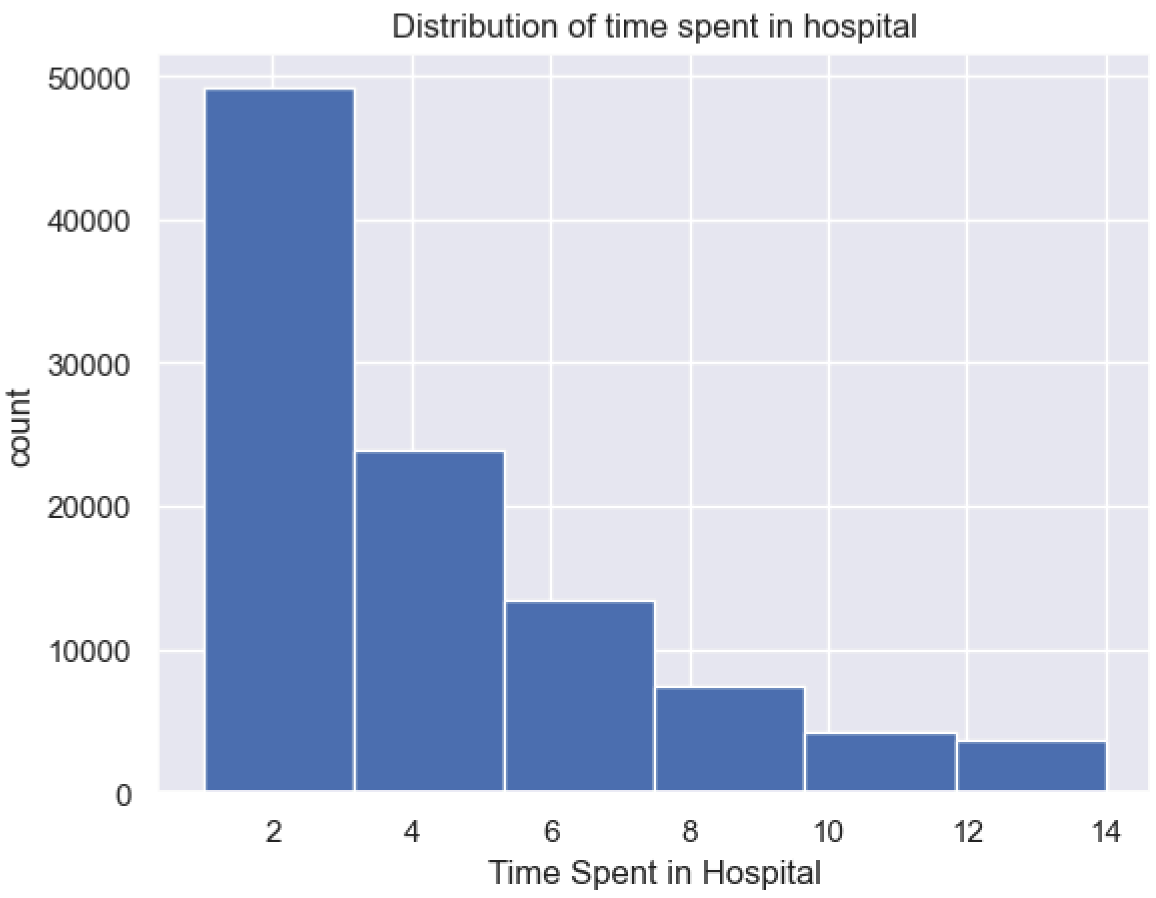
```
In [14]:  sns.set()
          num_of_procs = data.num_procedures
          plt.hist(num_of_procs, bins=3)
          # Label axes
          plt.xlabel('Number Of Procedures')
          plt.ylabel('count')
          plt.title('Distribution of number of procedures')
          # Show histogram
          plt.show()
```



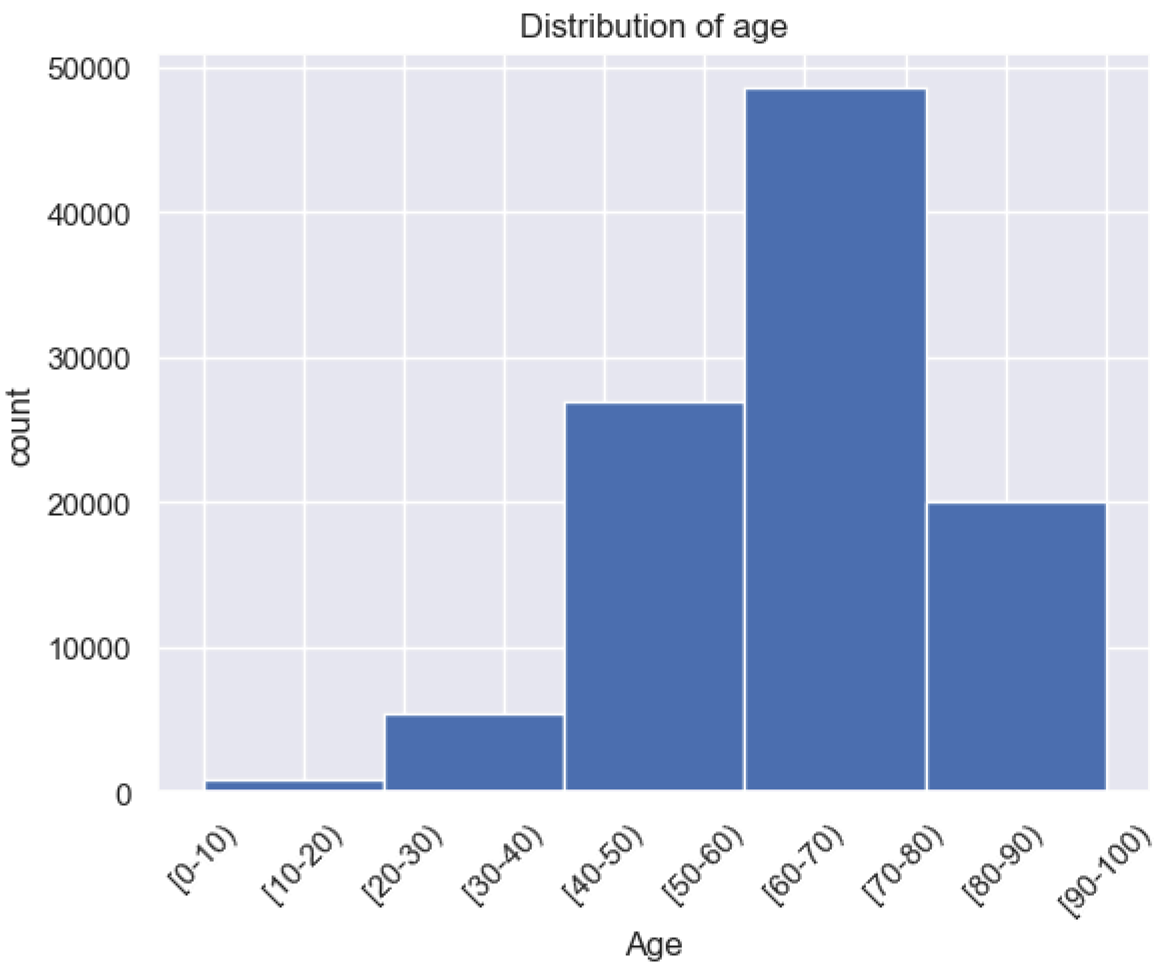
In [15]:
```
data.time_in_hospital.value_counts()
```

Out[15]:
```
time_in_hospital
3     17756
2     17224
1     14208
4     13924
5      9966
6      7539
7      5859
8      4391
9      3002
10     2342
11     1855
12     1448
13     1210
14     1042
Name: count, dtype: int64
```

In [16]:
```
sns.set()
time_in_hosp = data.time_in_hospital
plt.hist(time_in_hosp, bins=6)
# Label axes
plt.xlabel('Time Spent in Hospital')
plt.ylabel('count')
plt.title('Distribution of time spent in hospital')
# Show histogram
plt.show()
```



In [17]:
```
sns.set()
Age = data.age
plt.hist(Age, bins=5)
# Label axes
plt.xlabel('Age')
plt.ylabel('count')
plt.title('Distribution of age')
plt.xticks(rotation=45)
# Show histogram
plt.show()
```



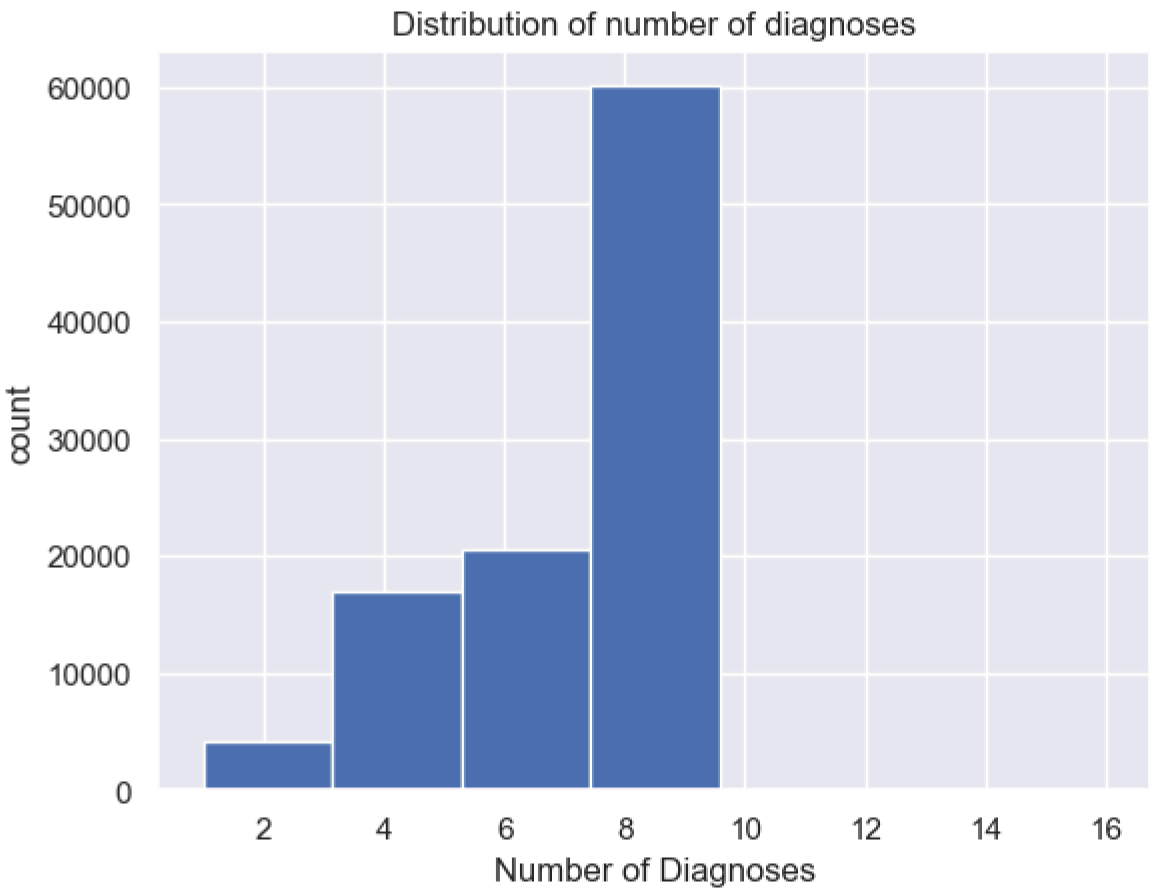
In [18]:
```
data.age.value_counts()
```

Out[18]:
```
age
[70-80)     26068
[60-70)     22483
[50-60)     17256
[80-90)     17197
[40-50)      9685
[30-40)      3775
[90-100)     2793
[20-30)      1657
[10-20)       691
[0-10)        161
Name: count, dtype: int64
```

In [19]:
```
data.number_diagnoses.value_counts()
```

Out[19]:
```
number_diagnoses
9     49474
5     11393
8     10616
7     10393
6     10161
4      5537
3      2835
2      1023
1       219
16       45
10       17
13       16
11       11
15       10
12        9
14        7
Name: count, dtype: int64
```

In [20]:
```python
sns.set()
diagnoses = data.number_diagnoses
plt.hist(diagnoses, bins=7)
# Label axes
plt.xlabel('Number of Diagnoses')
plt.ylabel('count')
plt.title('Distribution of number of diagnoses')
# Show histogram
plt.show()
```

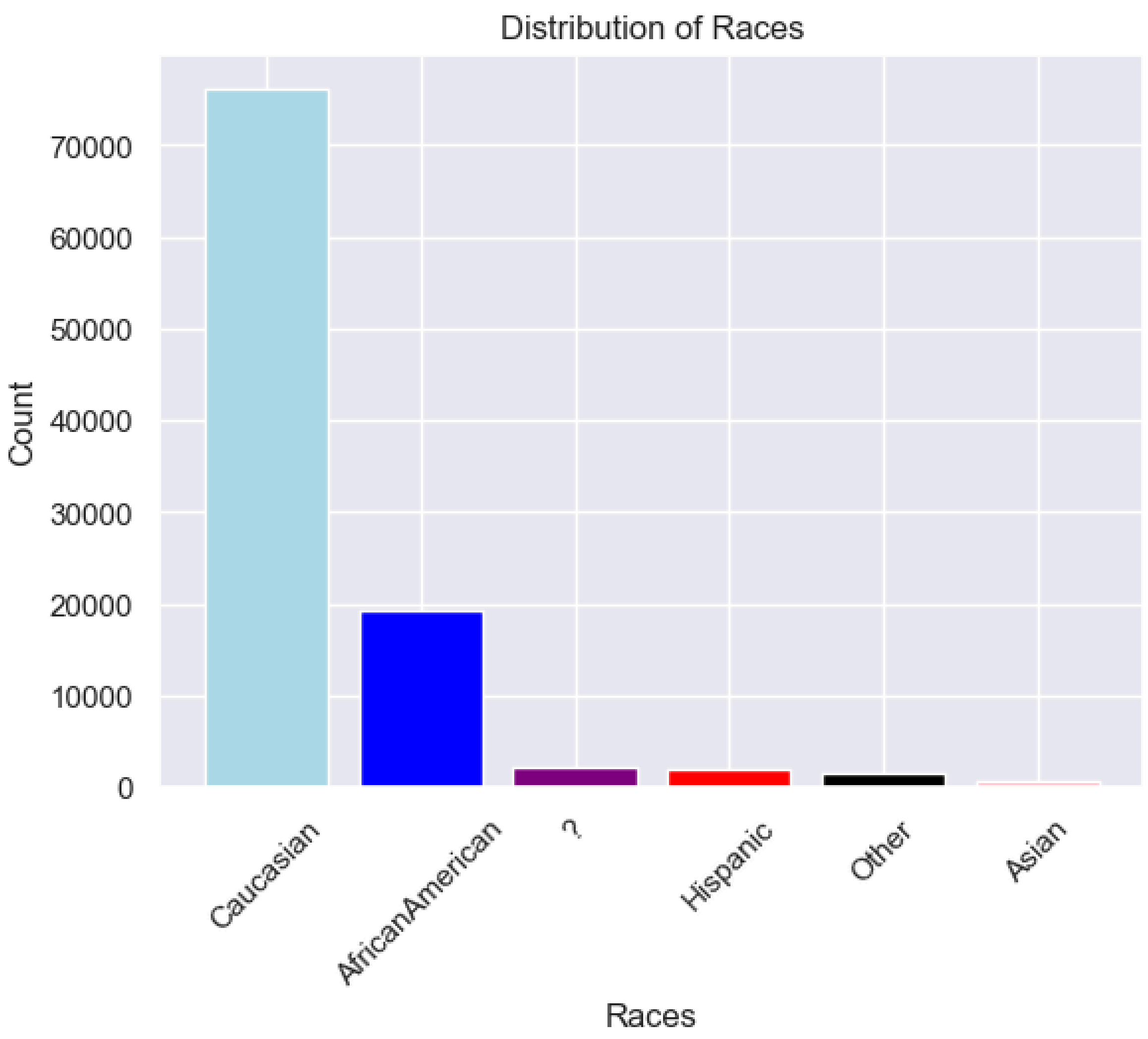


In [21]:
```
data.readmitted.value_counts()
```

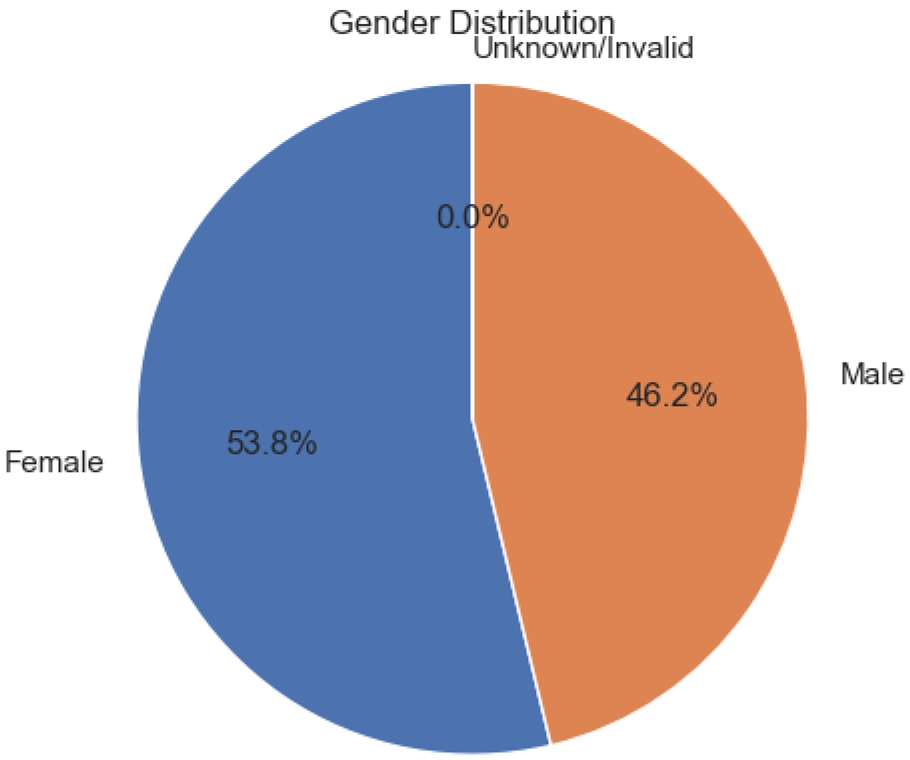
Out[21]:
```
readmitted
NO     54864
>30    35545
<30    11357
Name: count, dtype: int64
```

In [22]:
```
race_counts = data.race.value_counts()
gender_counts = data.gender.value_counts()
```

In [23]:
```
# creating vertical bar chart
fig, ax = plt.subplots()
color = ['lightblue', 'blue', 'purple', 'red', 'black', 'pink']
ax.bar(race_counts.keys(), race_counts.values, color=color)

plt.xticks(rotation=45)
ax.set_ylabel('Count')
ax.set_xlabel('Races')
ax.set_title('Distribution of Races')
plt.show()
```



In [24]:
```
fig, ax = plt.subplots()
ax.pie(gender_counts.values, labels=gender_counts.keys(), autopct='%1.1f%%', starta
ax.axis('equal')
plt.title('Gender Distribution')
plt.show()
```



In [25]:
```
data.diabetesMed.value_counts()
```

Out[25]:
```
diabetesMed
Yes    78363
No     23403
Name: count, dtype: int64
```

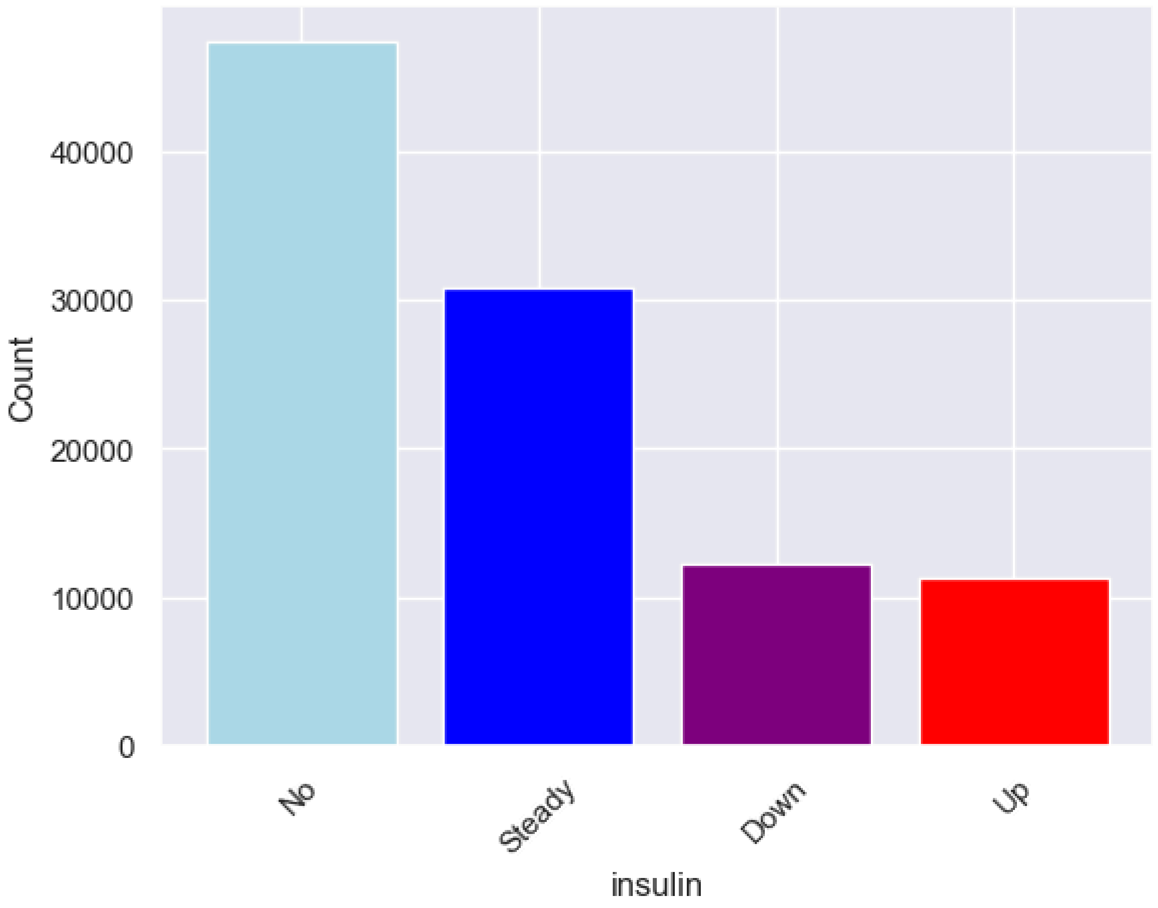
In [26]:
```
data.insulin.value_counts()
```

Out[26]:
```
insulin
No        47383
Steady    30849
Down      12218
Up        11316
Name: count, dtype: int64
```

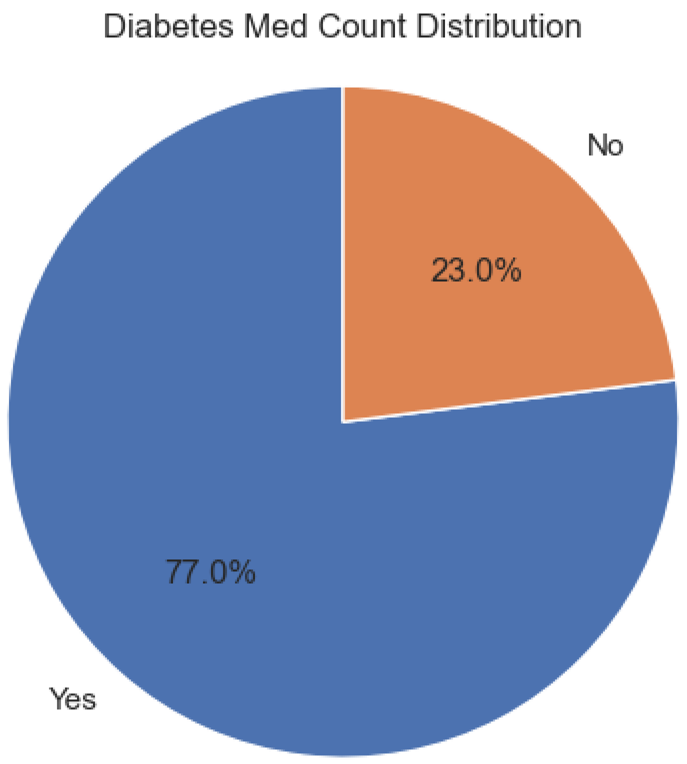
In [27]:
```
diabetesMedCount = data.diabetesMed.value_counts()
insulinCount = data.insulin.value_counts()
```

In [28]:
```
# creating vertical bar chart
fig, ax = plt.subplots()
color = ['lightblue', 'blue', 'purple', 'red', 'black', 'pink']
ax.bar(insulinCount.keys(), insulinCount.values, color=color)
```

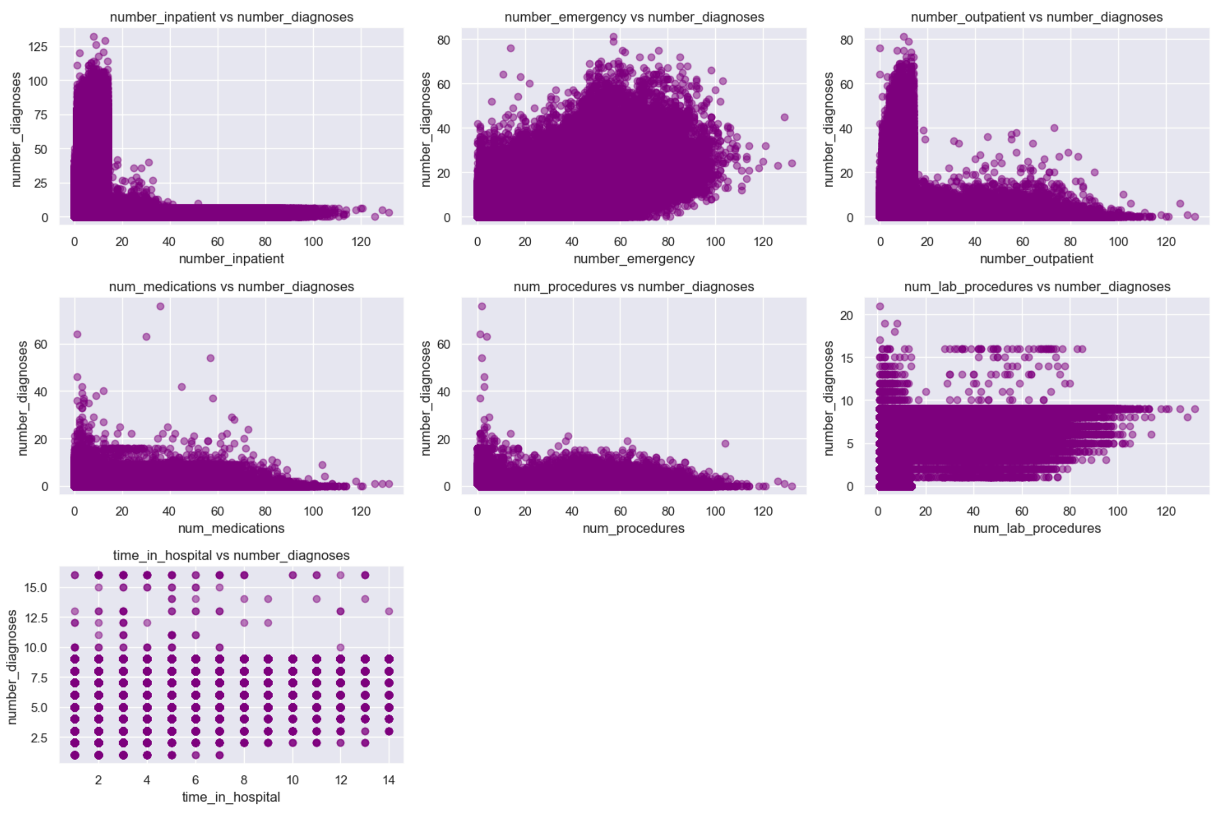
```
plt.xticks(rotation=45)
ax.set_ylabel('Count')
ax.set_xlabel('insulin')

plt.show()
```



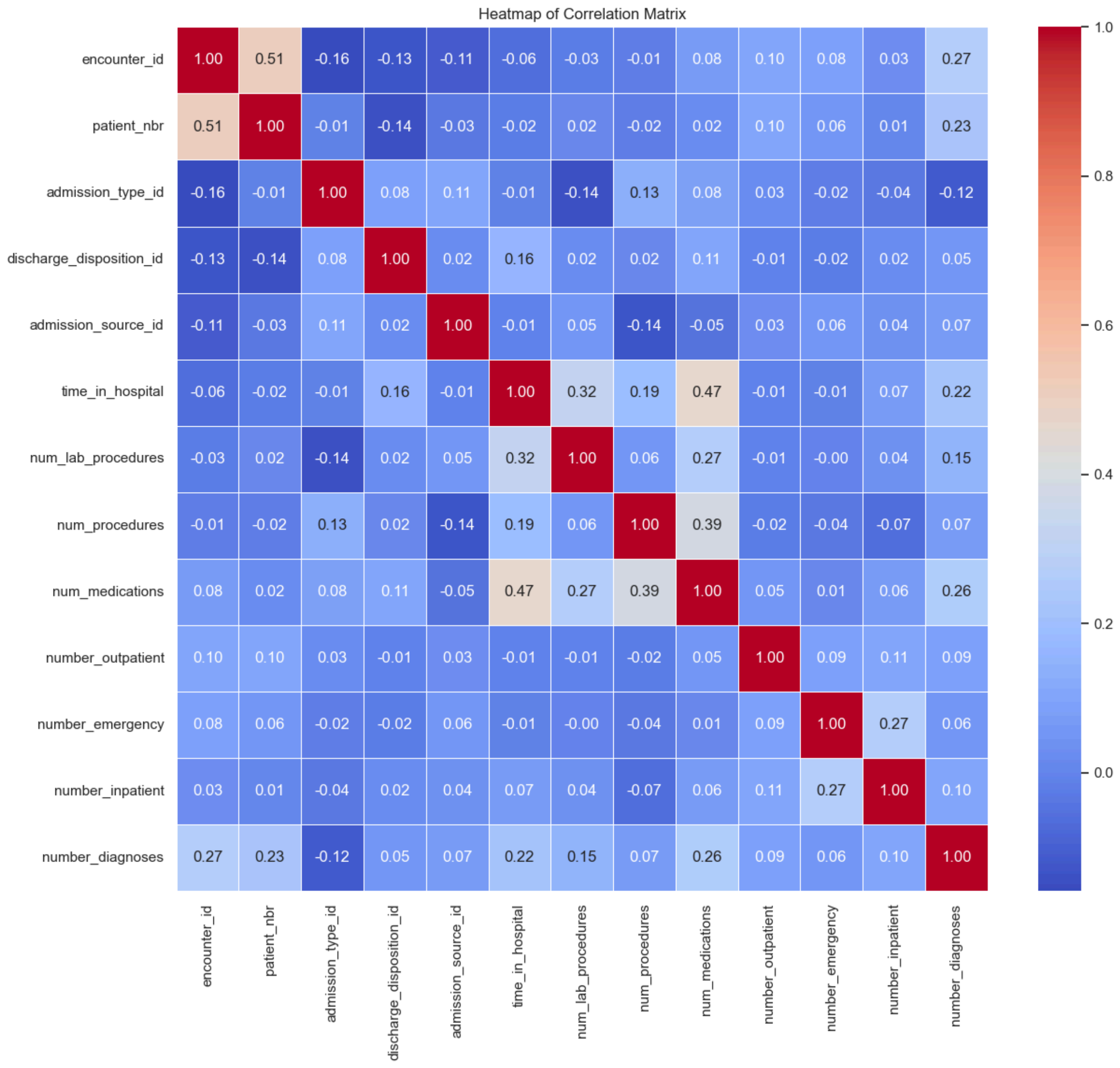
In [29]:
```
fig, ax = plt.subplots()
ax.pie(diabetesMedCount.values, labels=diabetesMedCount.keys(), autopct='%1.1f%%',
ax.axis('equal')
plt.title('Diabetes Med Count Distribution')
plt.show()
```



In [30]:
```
import matplotlib.pyplot as plt

continuous_vars = ['time_in_hospital', 'num_lab_procedures', 'num_procedures',
                   'num_medications', 'number_outpatient', 'number_emergency',
                   'number_inpatient', 'number_diagnoses']

# Generate scatter plots for each pair of continuous variables
plt.figure(figsize=(15, 10))
for i in range(len(continuous_vars)):
    for j in range(i + 1, len(continuous_vars)):
        plt.subplot(3, 3, j - i)
        plt.scatter(data[continuous_vars[i]], data[continuous_vars[j]], color='purp
        plt.xlabel(continuous_vars[i])
        plt.ylabel(continuous_vars[j])
        plt.title(f'{continuous_vars[i]} vs {continuous_vars[j]}')
plt.tight_layout()
plt.show()
```

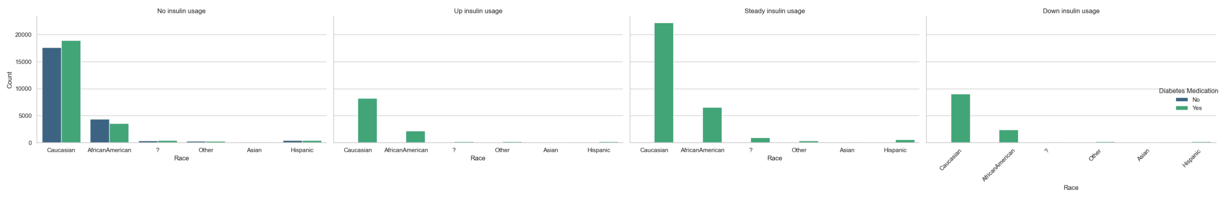


In [31]:
```
numerical_data = data.select_dtypes(include=[np.number])

# Calculating the correlation matrix
full_correlation_matrix = numerical_data.corr()

# Creating a heatmap to visualize the full correlation matrix
plt.figure(figsize=(14, 12))
full_heatmap = sns.heatmap(full_correlation_matrix, annot=True, cmap='coolwarm', fm
plt.title('Heatmap of Correlation Matrix')
plt.show()
```

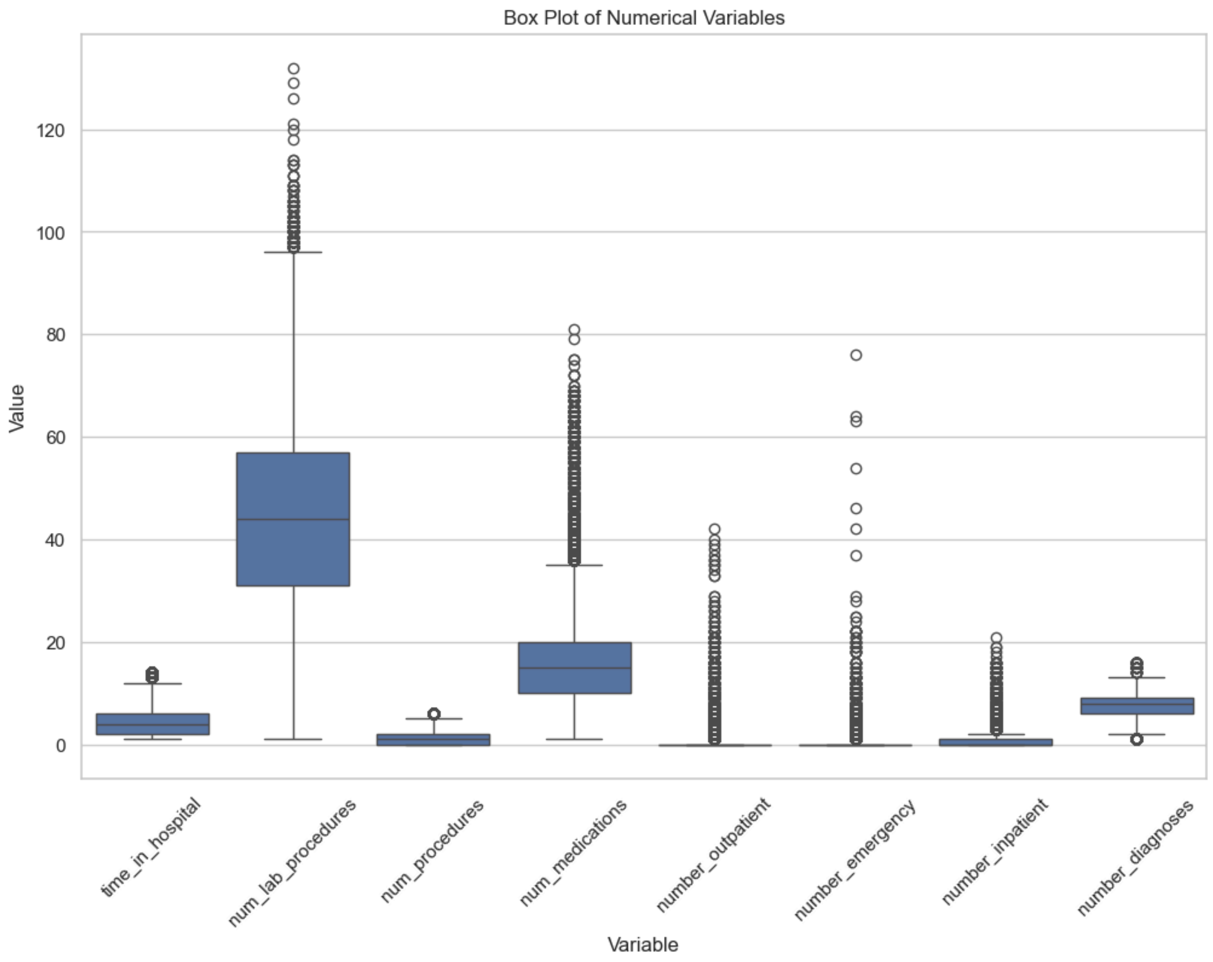


	encounter_id	patient_nbr	admission_type_id	discharge_disposition_id	admission_source_id	time_in_hospital	num_lab_procedures	num_procedures	num_medications	number_outpatient	number_emergency	number_inpatient	number_diagnoses
encounter_id	1.00	0.51	-0.16	-0.13	-0.11	-0.06	-0.03	-0.01	0.08	0.10	0.08	0.03	0.27
patient_nbr	0.51	1.00	-0.01	-0.14	-0.03	-0.02	0.02	-0.02	0.02	0.10	0.06	0.01	0.23
admission_type_id	-0.16	-0.01	1.00	0.08	0.11	-0.01	-0.14	0.13	0.08	0.03	-0.02	-0.04	-0.12
discharge_disposition_id	-0.13	-0.14	0.08	1.00	0.02	0.16	0.02	0.02	0.11	-0.01	-0.02	0.02	0.05
admission_source_id	-0.11	-0.03	0.11	0.02	1.00	-0.01	0.05	-0.14	-0.05	0.03	0.06	0.04	0.07
time_in_hospital	-0.06	-0.02	-0.01	0.16	-0.01	1.00	0.32	0.19	0.47	-0.01	-0.01	0.07	0.22
num_lab_procedures	-0.03	0.02	-0.14	0.02	0.05	0.32	1.00	0.06	0.27	-0.01	-0.00	0.04	0.15
num_procedures	-0.01	-0.02	0.13	0.02	-0.14	0.19	0.06	1.00	0.39	-0.02	-0.04	-0.07	0.07
num_medications	0.08	0.02	0.08	0.11	-0.05	0.47	0.27	0.39	1.00	0.05	0.01	0.06	0.26
number_outpatient	0.10	0.10	0.03	-0.01	0.03	-0.01	-0.01	-0.02	0.05	1.00	0.09	0.11	0.09
number_emergency	0.08	0.06	-0.02	-0.02	0.06	-0.01	-0.00	-0.04	0.01	0.09	1.00	0.27	0.06
number_inpatient	0.03	0.01	-0.04	0.02	0.04	0.07	0.04	-0.07	0.06	0.11	0.27	1.00	0.10
number_diagnoses	0.27	0.23	-0.12	0.05	0.07	0.22	0.15	0.07	0.26	0.09	0.06	0.10	1.00

In [32]:
```python
# Create a catplot to visualize the relationship between race, insulin usage, and d
sns.set(style="whitegrid")

# Create the plot
g = sns.catplot(
    data=data,
    x="race",
    hue="diabetesMed",
    col="insulin",
    kind="count",
    height=5,
    aspect=1.5,
    palette="viridis"
)

# Set the titles and labels
g.set_titles(col_template="{col_name} insulin usage")
g.set_axis_labels("Race", "Count")
g._legend.set_title('Diabetes Medication')

plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



In [33]:
```
plt.figure(figsize=(12, 8))
box_plot = sns.boxplot(x='variable', y='value', data=pd.melt(data[['time_in_hospita
box_plot.set_title('Box Plot of Numerical Variables')
box_plot.set_xlabel('Variable')
box_plot.set_ylabel('Value')
plt.xticks(rotation=45)  # Rotate labels for better readability
plt.show()
```



In [35]:
```
# Columns to be dropped
columns_to_drop = [
    'encounter_id',
    'patient_nbr',
    'admission_type_id',
    'discharge_disposition_id',
    'admission_source_id',
    'payer_code'
]
```

```
data2 = data.drop(columns=columns_to_drop)
```

In [36]:
```
#Remove outliers

# Define a function to remove outliers using the IQR method
def remove_outliers(df, columns):
    for column in columns:
        Q1 = df[column].quantile(0.25)
        Q3 = df[column].quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR
        df = df[(df[column] >= lower_bound) & (df[column] <= upper_bound)]
    return df

# List of numerical columns to check for outliers
numerical_columns = [
    'time_in_hospital', 'num_lab_procedures', 'num_procedures',
    'num_medications', 'number_outpatient', 'number_emergency',
    'number_inpatient', 'number_diagnoses'
]

# Remove outliers from the dataset
data2 = remove_outliers(data2, numerical_columns)

print(data2.head())
```

```
       race  gender      age  time_in_hospital medical_specialty  \
1  Caucasian  Female  [10-20)                 3                 ?   
3  Caucasian    Male  [30-40)                 2                 ?   
4  Caucasian    Male  [40-50)                 1                 ?   
6  Caucasian    Male  [60-70)                 4                 ?   
7  Caucasian    Male  [70-80)                 5                 ?   

   num_lab_procedures  num_procedures  num_medications  number_outpatient  \
1                  59               0               18                  0   
3                  44               1               16                  0   
4                  51               0                8                  0   
6                  70               1               21                  0   
7                  73               0               12                  0   

   number_emergency  ...  citoglipton insulin glyburide-metformin  \
1                 0  ...           No      Up                  No   
3                 0  ...           No      Up                  No   
4                 0  ...           No  Steady                  No   
6                 0  ...           No  Steady                  No   
7                 0  ...           No      No                  No   

  glipizide-metformin  glimepiride-pioglitazone metformin-rosiglitazone  \
1                  No                        No                      No   
3                  No                        No                      No   
4                  No                        No                      No   
6                  No                        No                      No   
7                  No                        No                      No   

  metformin-pioglitazone change diabetesMed readmitted  
1                     No     Ch         Yes        >30  
3                     No     Ch         Yes         NO  
4                     No     Ch         Yes         NO  
6                     No     Ch         Yes         NO  
7                     No     No         Yes        >30  

[5 rows x 41 columns]
```

In [37]:
```python
data2.readmitted.value_counts()
```

Out[37]:
```
readmitted
NO     39280
>30    20943
<30     6369
Name: count, dtype: int64
```

In [38]:
```python
data2['readmitted'] = data2['readmitted'].apply(lambda x: 'YES' if x == '<30' else
data2.readmitted.value_counts()
```

Out[38]:
```
readmitted
NO     60223
YES     6369
Name: count, dtype: int64
```

In [40]:
```python
from sklearn.preprocessing import MinMaxScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from imblearn.over_sampling import SMOTE

# Normalize continuous variables
scaler = MinMaxScaler()
continuous_vars = ['time_in_hospital', 'num_lab_procedures', 'num_procedures',
                   'num_medications', 'number_outpatient', 'number_emergency',
                   'number_inpatient', 'number_diagnoses']
data2[continuous_vars] = scaler.fit_transform(data2[continuous_vars])

# Check for and handle missing values
data2.fillna(method='ffill', inplace=True)

# Encode categorical variables and target variable
categorical_cols = data2.select_dtypes(include=['object']).columns.tolist()
# Remove the target column 'readmitted' from categorical columns
categorical_cols.remove('readmitted')

# One-hot encoding categorical variables
data_encoded2 = pd.get_dummies(data2, columns=categorical_cols, drop_first=True)

# Encoding the target variable
label_encoder = LabelEncoder()
data_encoded2['readmitted_encoded'] = label_encoder.fit_transform(data2['readmitted


# Splitting the data
X = data_encoded2.drop(['readmitted', 'readmitted_encoded'], axis=1)  # Features
y = data_encoded2['readmitted_encoded']  # Target variable

# Apply SMOTE to balance the dataset
smote = SMOTE(random_state=42)
X_balanced, y_balanced = smote.fit_resample(X, y)

X_train, X_test, y_train, y_test = train_test_split(X_balanced, y_balanced, test_si
```

```
C:\Users\Nishi\AppData\Local\Temp\ipykernel_12368\1867024009.py:16: FutureWarning: D
ataFrame.fillna with 'method' is deprecated and will raise in a future version. Use
obj.ffill() or obj.bfill() instead.
  data2.fillna(method='ffill', inplace=True)
```

In [44]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, precision_score, recall_score, f1_sco

# Fitting the DecisionTreeClassifier
DecisionTree = DecisionTreeClassifier(random_state=0, max_depth=9, min_samples_leaf
DecisionTree.fit(X_train, y_train)

# Evaluate the DecisionTreeClassifier model
y_pred = DecisionTree.predict(X_test)
conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix (Decision Tree):")
print(conf_matrix)

# Calculate metrics
```

```
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, pos_label=1)
recall = recall_score(y_test, y_pred, pos_label=1)
f1 = f1_score(y_test, y_pred, pos_label=1)

# Print out the metrics
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision (yes): {precision:.4f}")
print(f"Recall (yes): {recall:.4f}")
print(f"F1-score (yes): {f1:.4f}")
```

```
Confusion Matrix (Decision Tree):
[[11757   190]
 [ 3569  8574]]
Accuracy: 0.8440
Precision (yes): 0.9783
Recall (yes): 0.7061
F1-score (yes): 0.8202
```

In [45]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, precision_score, recall_score, f1_sco

# Fitting the RandomForestClassifier
rf_classifier = RandomForestClassifier(random_state=0)
rf_classifier.fit(X_train, y_train)

# Evaluate the RandomForestClassifier model
y_pred = rf_classifier.predict(X_test)
conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix (Random Forest):")
print(conf_matrix)

# Calculate metrics
accuracy = accuracy_score(y_test, y_pred)
precision_yes = precision_score(y_test, y_pred, pos_label=1)
recall_yes = recall_score(y_test, y_pred, pos_label=1)
f1_yes = f1_score(y_test, y_pred, pos_label=1)

# Print out the metrics
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision (yes): {precision_yes:.4f}")
print(f"Recall (yes): {recall_yes:.4f}")
print(f"F1-score (yes): {f1_yes:.4f}")
```

```
Confusion Matrix (Random Forest):
[[11910    37]
 [  684 11459]]
Accuracy: 0.9701
Precision (yes): 0.9968
Recall (yes): 0.9437
F1-score (yes): 0.9695
```

In [46]:
```
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import confusion_matrix, precision_score, recall_score, f1_sco

# Fitting the GradientBoostingClassifier
gb_classifier = GradientBoostingClassifier(random_state=0)
```

```
gb_classifier.fit(X_train, y_train)

# Evaluate the GradientBoostingClassifier model
y_pred = gb_classifier.predict(X_test)
conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix (Gradient Boosting):")
print(conf_matrix)

# Calculate metrics
accuracy = accuracy_score(y_test, y_pred)
precision_yes = precision_score(y_test, y_pred, pos_label=1)
recall_yes = recall_score(y_test, y_pred, pos_label=1)
f1_yes = f1_score(y_test, y_pred, pos_label=1)

# Print out the metrics
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision (yes): {precision_yes:.4f}")
print(f"Recall (yes): {recall_yes:.4f}")
print(f"F1-score (yes): {f1_yes:.4f}")
```

```
Confusion Matrix (Gradient Boosting):
[[11945     2]
 [ 1966 10177]]
Accuracy: 0.9183
Precision (yes): 0.9998
Recall (yes): 0.8381
F1-score (yes): 0.9118
```

```
In [ ]:
```