time_study_plot

May 26, 2021

1 Plotting Time Periods

This notebook visualizes the notes over time plotted against important periods during the adulterated heparin crisis. + python = 3.7.x

[ ]:
```
import pickle
import time
import os
import re

import numpy as np
import pandas as pd
from datetime import datetime
import sys

from importlib_metadata import version

%matplotlib inline
import matplotlib.pyplot as plt

import psycopg2
from sqlalchemy import create_engine

import seaborn as sns
```

[ ]:
```
PASSWORD = os.environ.get("PASSWORD")
USERNAME = os.environ.get("USERNAME")
POSTGRES_CONNECT = os.environ.get("POSTGRES_CONNECT")
POSTGRES_ENGINE = os.environ.get("POSTGRES_ENGINE")

conn = psycopg2.connect(POSTGRES_CONNECT)

cur = conn.cursor();
cur.execute("""SET search_path = mimiciii;""")

engine = create_engine(POSTGRES_ENGINE)#'postgresql://postgres:
↪postgres@localhost/MIMIC')
```

[ ]:
```
libraries =␣
 ↪['pandas','sqlalchemy','psycopg2','tqdm','scipy','numpy','matplotlib','scikit-learn']
print('last ran: ',datetime.now() )
print("Python Version:", sys.version[0:7])
print( "operating system:", sys.platform)

for lib in libraries:
    print(lib + ' version: ' + version(lib))
```

[ ]:
```
df = pd.read_sql("""select * from mimiciii.time_study_notes""", engine)
```

1.1 Plotting Time Periods

Plotting total number of admissions over time against key dates during the adulterated heparin crisis.

[ ]:
```
times = df.set_index('true_admittime').resample('M').count()
times.plot(style='-', figsize=(16,6), c='k', legend=None)
plt.axvline(x='2006-06', c='r') # pig disease discovered
plt.axvline(x='2007-11', c='r') # CDC study of patients getting sick (earliest␣
 ↪cases)
plt.axvline(x='2008-01', c='b', linestyle=':') # Baxter issues first heparin␣
 ↪recall
plt.axvline(x='2008-05', c='b', linestyle=':') # final heparin recall completed
plt.title('Admissions Over Period of Interest');
```

[ ]:
```
# Proportion of Time Periods
labels = 'Period A', 'Period B', 'Period C'
pre = df[(df['true_admittime'] < '2006-06-01')].shape[0]
during = df[(df['true_admittime'] > '2006-06-01') & (df['true_admittime'] <␣
 ↪'2007-11-01')].shape[0]
post = df[(df['true_admittime'] > '2007-11-01')].shape[0]
sizes = [pre, during, post]

fig1, ax1 = plt.subplots()
ax1.pie(sizes, labels=labels, autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')
ax1.set_title('Proportion of Classes')
plt.show();
```

[ ]:
```
times = df.set_index('true_admittime').resample('Q').count()
times.plot(style='-', figsize=(16,6), c='k', legend=None)
plt.axvline(x='2006-06', c='r') # pig disease discovered
plt.axvline(x='2007-11', c='r') # CDC study of patients getting sick (earliest␣
 ↪cases)
```

```
plt.axvline(x='2008-01', c='b', linestyle=':') # Baxter issues first heparin␣
 ↪recall
plt.axvline(x='2008-05', c='b', linestyle=':') # final heparin recall completed
plt.title('Admissions Over Period of Interest');
```

1.2 plots

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

[ ]:
```
df.head()
```

[ ]:
```
pd.to_datetime('2008-07-01T00')
```

[ ]:
```
periods = []
for i in df['true_admittime']:
    if i < pd.to_datetime('2008-07-01T00') and i >= pd.
 ↪to_datetime('2007-07-01T00'):
        periods.append('2007_2008')
    elif i < pd.to_datetime('2007-07-01T00') and i >= pd.
 ↪to_datetime('2006-07-01T00'):
        periods.append('2006_2007')
    elif i < pd.to_datetime('2006-07-01T00') and i >= pd.
 ↪to_datetime('2005-07-01T00'):
        periods.append('2005_2006')
    elif i < pd.to_datetime('2005-07-01T00') and i >= pd.
 ↪to_datetime('2004-07-01T00'):
        periods.append('2004_2005')
    elif i < pd.to_datetime('2004-07-01T00') and i >= pd.
 ↪to_datetime('2003-07-01T00'):
        periods.append('2003_2004')
    elif i < pd.to_datetime('2003-07-01T00') and i >= pd.
 ↪to_datetime('2002-07-01T00'):
        periods.append('2002_2003')
    elif i < pd.to_datetime('2002-07-01T00') and i >= pd.
 ↪to_datetime('2001-07-01T00'):
        periods.append('2001_2002')
    elif i < pd.to_datetime('2001-07-01T00') and i >= pd.
 ↪to_datetime('2000-07-01T00'):
        periods.append('2000_2001')
    elif i < pd.to_datetime('2009-07-01T00') and i >= pd.
 ↪to_datetime('2008-07-01T00'):
        periods.append('2008_2009')
```

[ ]:
```
df['time_period'] = periods
```

[ ]:
```
chart = sns.countplot(df['time_period'],
                      color = 'gray')
chart.set_xticklabels(chart.get_xticklabels(), rotation=45)
chart.set_title('Admissions Per Time Period')
chart.set(xlabel='Time Period (July 1 - June 30 Cycle)', ylabel='Number␣
↪Admissions');
```

[ ]:
```
df['time_period'].value_counts()
```