cervical-cancer-prediction

February 7, 2024

1 Installing Libraries

[1]:
```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

2 Importing the Dataset

[2]:
```
from google.colab import drive
drive.mount("/content/gdrive")
data = pd.read_csv('/content/gdrive/My Drive/Final Year Project/cervicalcancer.
  ↪csv')
print(data.shape)
data
```

```
Mounted at /content/gdrive
(835, 36)
```

[2]:
```
     Age  Number of sexual partners  First sexual intercourse  \
0     18                        4.0                      15.0
1     15                        1.0                      14.0
2     34                        1.0                       NaN
3     52                        5.0                      16.0
4     46                        3.0                      21.0
..   …                         …                        …
830   34                        3.0                      18.0
831   32                        2.0                      19.0
832   25                        2.0                      17.0
833   33                        2.0                      24.0
```

```
834   29                      2.0                      20.0

     Num of pregnancies  Smokes  Smokes (years)  Smokes (packs/year)  \
0                   1.0     0.0             0.0                  0.0
1                   1.0     0.0             0.0                  0.0
2                   1.0     0.0             0.0                  0.0
3                   4.0     1.0            37.0                 37.0
4                   4.0     0.0             0.0                  0.0
..                  …     …             …                  …
830                 0.0     0.0             0.0                  0.0
831                 1.0     0.0             0.0                  0.0
832                 0.0     0.0             0.0                  0.0
833                 2.0     0.0             0.0                  0.0
834                 1.0     0.0             0.0                  0.0

     Hormonal Contraceptives  Hormonal Contraceptives (years)  IUD  …  \
0                        0.0                             0.00  0.0  …
1                        0.0                             0.00  0.0  …
2                        0.0                             0.00  0.0  …
3                        1.0                             3.00  0.0  …
4                        1.0                            15.00  0.0  …
..                       …                             …  …  …
830                      0.0                             0.00  0.0  …
831                      1.0                             8.00  0.0  …
832                      1.0                             0.08  0.0  …
833                      1.0                             0.08  0.0  …
834                      1.0                             0.50  0.0  …

     STDs: Time since first diagnosis  STDs: Time since last diagnosis  \
0                                 NaN                              NaN
1                                 NaN                              NaN
2                                 NaN                              NaN
3                                 NaN                              NaN
4                                 NaN                              NaN
..                                …                              …
830                               NaN                              NaN
831                               NaN                              NaN
832                               NaN                              NaN
833                               NaN                              NaN
834                               NaN                              NaN

     Dx:Cancer  Dx:CIN  Dx:HPV  Dx  Hinselmann  Schiller  Citology  Biopsy
0            0       0       0   0           0         0         0       0
1            0       0       0   0           0         0         0       0
2            0       0       0   0           0         0         0       0
3            1       0       1   0           0         0         0       0
4            0       0       0   0           0         0         0       0
```

```
..          …     …     …  ..        …        …        …     …
830          0       0       0   0         0        0        0        0
831          0       0       0   0         0        0        0        0
832          0       0       0   0         0        0        1        0
833          0       0       0   0         0        0        0        0
834          0       0       0   0         0        0        0        0

[835 rows x 36 columns]
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 835 entries, 0 to 834
Data columns (total 36 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   Age                                 835 non-null    int64
 1   Number of sexual partners           810 non-null    float64
 2   First sexual intercourse            828 non-null    float64
 3   Num of pregnancies                  779 non-null    float64
 4   Smokes                              822 non-null    float64
 5   Smokes (years)                      822 non-null    float64
 6   Smokes (packs/year)                 822 non-null    float64
 7   Hormonal Contraceptives             732 non-null    float64
 8   Hormonal Contraceptives (years)     732 non-null    float64
 9   IUD                                 723 non-null    float64
 10  IUD (years)                         723 non-null    float64
 11  STDs                                735 non-null    float64
 12  STDs (number)                       735 non-null    float64
 13  STDs:condylomatosis                 735 non-null    float64
 14  STDs:cervical condylomatosis        735 non-null    float64
 15  STDs:vaginal condylomatosis         735 non-null    float64
 16  STDs:vulvo-perineal condylomatosis  735 non-null    float64
 17  STDs:syphilis                       735 non-null    float64
 18  STDs:pelvic inflammatory disease    735 non-null    float64
 19  STDs:genital herpes                 735 non-null    float64
 20  STDs:molluscum contagiosum          735 non-null    float64
 21  STDs:AIDS                           735 non-null    float64
 22  STDs:HIV                            735 non-null    float64
 23  STDs:Hepatitis B                    735 non-null    float64
 24  STDs:HPV                            735 non-null    float64
 25  STDs: Number of diagnosis           835 non-null    int64
 26  STDs: Time since first diagnosis    71 non-null     float64
 27  STDs: Time since last diagnosis     71 non-null     float64
 28  Dx:Cancer                           835 non-null    int64
 29  Dx:CIN                              835 non-null    int64
 30  Dx:HPV                              835 non-null    int64
 31  Dx                                  835 non-null    int64
 32  Hinselmann                          835 non-null    int64
 33  Schiller                            835 non-null    int64
 34  Citology                            835 non-null    int64
 35  Biopsy                              835 non-null    int64
dtypes: float64(26), int64(10)
memory usage: 235.0 KB
```

[4]:
```
data.describe()
```

[4]:
```
              Age  Number of sexual partners  First sexual intercourse  \
count  835.000000                 810.000000                828.000000
mean    27.023952                   2.551852                 17.020531
std      8.482986                   1.676686                  2.817000
min     13.000000                   1.000000                 10.000000
25%     21.000000                   2.000000                 15.000000
50%     26.000000                   2.000000                 17.000000
75%     32.000000                   3.000000                 18.000000
max     84.000000                  28.000000                 32.000000

       Num of pregnancies      Smokes  Smokes (years)  Smokes (packs/year)  \
count          779.000000  822.000000      822.000000           822.000000
mean             2.304236    0.149635        1.253850             0.465823
std              1.455817    0.356930        4.140727             2.256273
min              0.000000    0.000000        0.000000             0.000000
25%              1.000000    0.000000        0.000000             0.000000
50%              2.000000    0.000000        0.000000             0.000000
75%              3.000000    0.000000        0.000000             0.000000
max             11.000000    1.000000       37.000000            37.000000

       Hormonal Contraceptives  Hormonal Contraceptives (years)         IUD  \
count               732.000000                       732.000000  723.000000
mean                  0.651639                         2.302916    0.114799
std                   0.476777                         3.794180    0.319000
min                   0.000000                         0.000000    0.000000
25%                   0.000000                         0.000000    0.000000
50%                   1.000000                         0.500000    0.000000
75%                   1.000000                         3.000000    0.000000
max                   1.000000                        30.000000    1.000000

       …  STDs: Time since first diagnosis  STDs: Time since last diagnosis  \
count  …                         71.000000                        71.000000
mean   …                          6.140845                         5.816901
std    …                          5.895024                         5.755271
min    …                          1.000000                         1.000000
25%    …                          2.000000                         2.000000
50%    …                          4.000000                         3.000000
```

```
75%    …                     8.000000                          7.500000
max    …                    22.000000                         22.000000

        Dx:Cancer      Dx:CIN      Dx:HPV          Dx  Hinselmann    Schiller  \
count  835.000000  835.000000  835.000000  835.000000  835.000000  835.000000
mean     0.021557    0.010778    0.021557    0.028743    0.041916    0.087425
std      0.145319    0.103320    0.145319    0.167182    0.200518    0.282626
min      0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
25%      0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
50%      0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
75%      0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
max      1.000000    1.000000    1.000000    1.000000    1.000000    1.000000

         Citology      Biopsy
count  835.000000  835.000000
mean     0.051497    0.064671
std      0.221142    0.246091
min      0.000000    0.000000
25%      0.000000    0.000000
50%      0.000000    0.000000
75%      0.000000    0.000000
max      1.000000    1.000000

[8 rows x 36 columns]
```

[5]:
```
# Determining the null values in each column
data.isnull().sum()
```

[5]:
```
Age                                     0
Number of sexual partners              25
First sexual intercourse                7
Num of pregnancies                     56
Smokes                                 13
Smokes (years)                         13
Smokes (packs/year)                    13
Hormonal Contraceptives               103
Hormonal Contraceptives (years)       103
IUD                                   112
IUD (years)                           112
STDs                                  100
STDs (number)                         100
STDs:condylomatosis                   100
STDs:cervical condylomatosis          100
STDs:vaginal condylomatosis           100
STDs:vulvo-perineal condylomatosis    100
STDs:syphilis                         100
STDs:pelvic inflammatory disease      100
```

```
STDs:genital herpes                   100
STDs:molluscum contagiosum            100
STDs:AIDS                             100
STDs:HIV                              100
STDs:Hepatitis B                      100
STDs:HPV                              100
STDs: Number of diagnosis               0
STDs: Time since first diagnosis      764
STDs: Time since last diagnosis       764
Dx:Cancer                               0
Dx:CIN                                  0
Dx:HPV                                  0
Dx                                      0
Hinselmann                              0
Schiller                                0
Citology                                0
Biopsy                                  0
dtype: int64
```

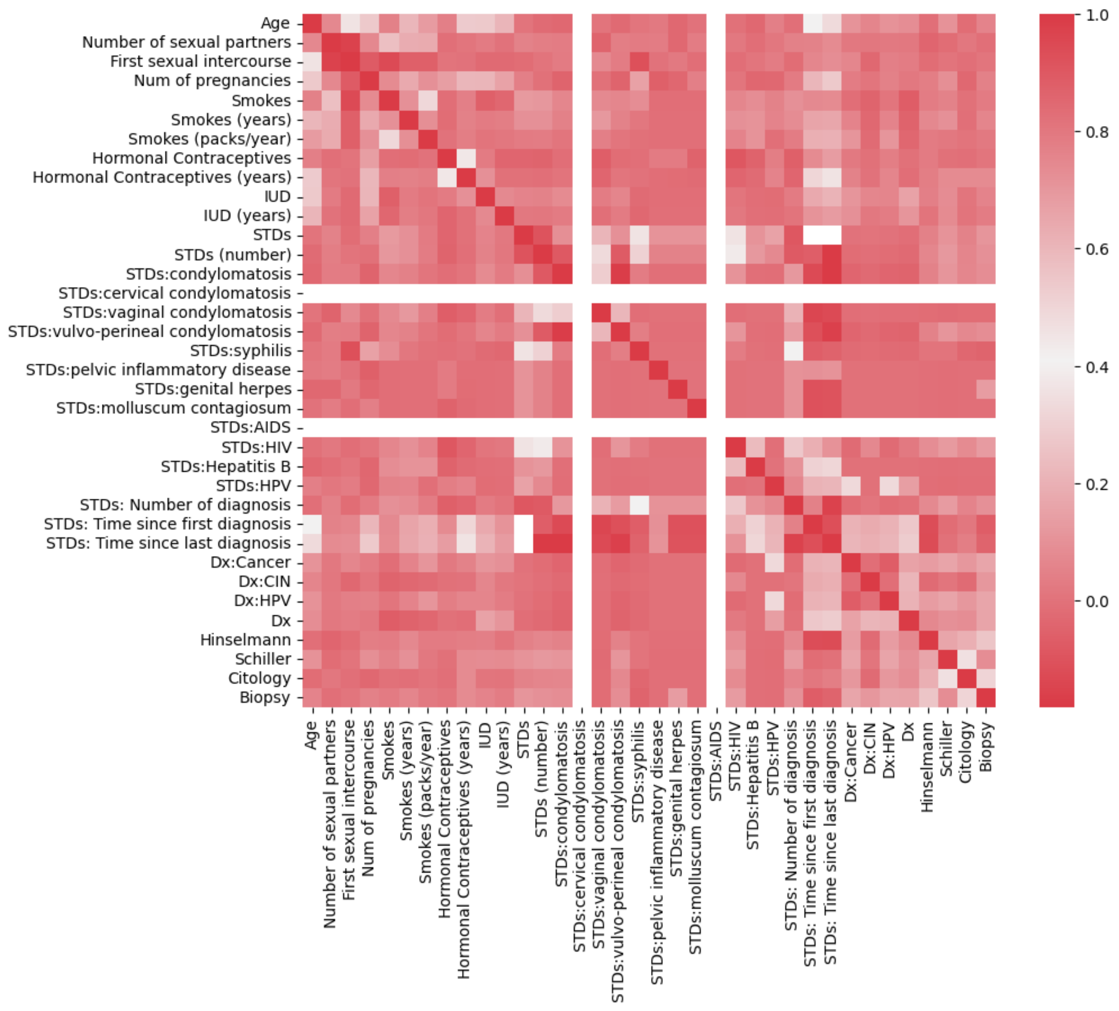
[6]:
```
data.columns
```

[6]:
```
Index(['Age', 'Number of sexual partners', 'First sexual intercourse',
       'Num of pregnancies', 'Smokes', 'Smokes (years)', 'Smokes (packs/year)',
       'Hormonal Contraceptives', 'Hormonal Contraceptives (years)', 'IUD',
       'IUD (years)', 'STDs', 'STDs (number)', 'STDs:condylomatosis',
       'STDs:cervical condylomatosis', 'STDs:vaginal condylomatosis',
       'STDs:vulvo-perineal condylomatosis', 'STDs:syphilis',
       'STDs:pelvic inflammatory disease', 'STDs:genital herpes',
       'STDs:molluscum contagiosum', 'STDs:AIDS', 'STDs:HIV',
       'STDs:Hepatitis B', 'STDs:HPV', 'STDs: Number of diagnosis',
       'STDs: Time since first diagnosis', 'STDs: Time since last diagnosis',
       'Dx:Cancer', 'Dx:CIN', 'Dx:HPV', 'Dx', 'Hinselmann', 'Schiller',
       'Citology', 'Biopsy'],
      dtype='object')
```

3 Correlation Plot

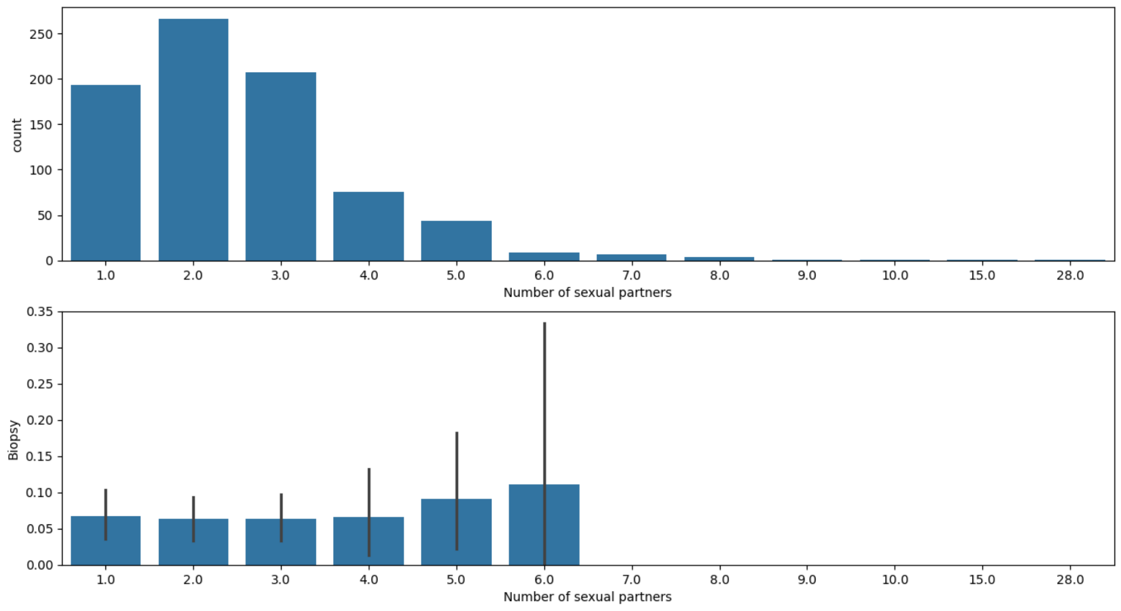
[7]:
```
# correlation plot

f, ax = plt.subplots(figsize = (10, 8))

corr = data.corr()
sns.heatmap(corr, mask = np.zeros_like(corr, dtype = np.bool),
            cmap = sns.diverging_palette(10, 10, as_cmap = True), square =␣
 ↪True, ax = ax)
```

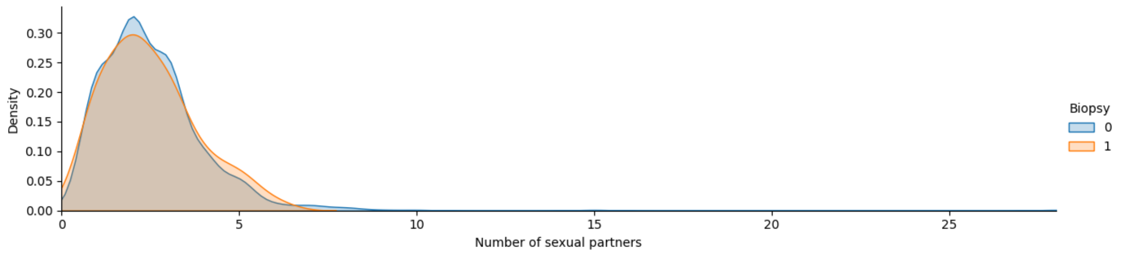
[7]: <Axes: >



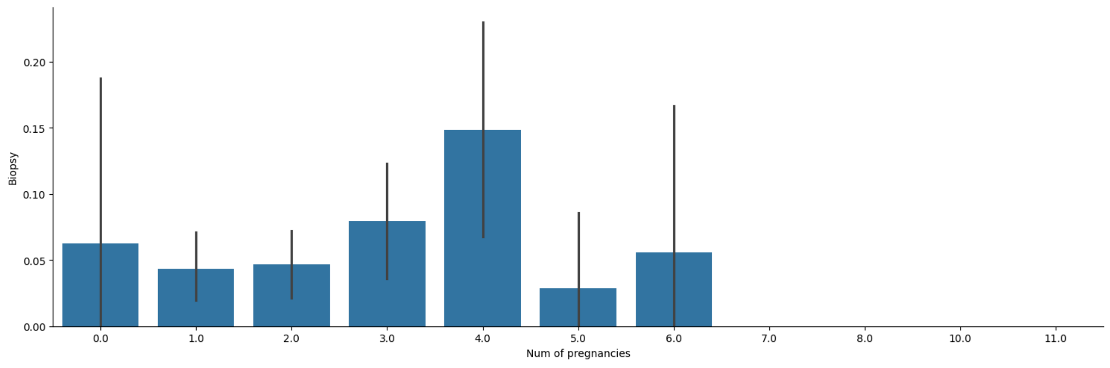
[8]:
```
# Biopsy vs no. of sexual partners

#categorical to categorical
fig, (ax1,ax2) = plt.subplots(2, 1, figsize = (15, 8))
sns.countplot(x = 'Number of sexual partners', data = data, ax=ax1)
sns.barplot(x = 'Number of sexual partners', y = 'Biopsy', data = data, ax=ax2)

#continuous to categorical
facet = sns.FacetGrid(data, hue='Biopsy',aspect=4)
facet.map(sns.kdeplot,'Number of sexual partners',shade= True)
facet.set(xlim=(0, data['Number of sexual partners'].max()))
facet.add_legend()
```

[8]: <seaborn.axisgrid.FacetGrid at 0x7cedb5df5e70>



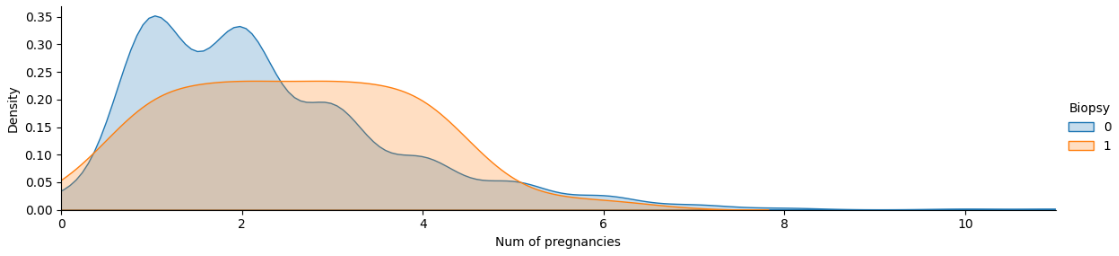


[11]:
```
# Biopsy vs no. of pregnancies

sns.catplot(x='Num of pregnancies', y='Biopsy', data=data, kind='bar',␣
 ↪height=5, aspect=3)
plt.show()
```



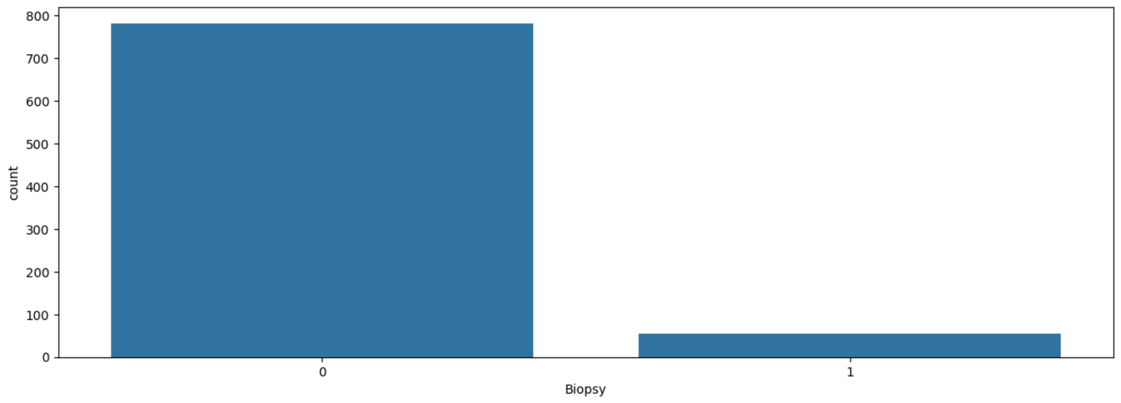
```
[12]: #continuous to categorical
      facet = sns.FacetGrid(data, hue='Biopsy', aspect=4)
      facet.map(sns.kdeplot,'Num of pregnancies', shade= True)
      facet.set(xlim=(0, data['Num of pregnancies'].max()))
      facet.add_legend()
```

```
[12]: <seaborn.axisgrid.FacetGrid at 0x7cedb40a3550>
```



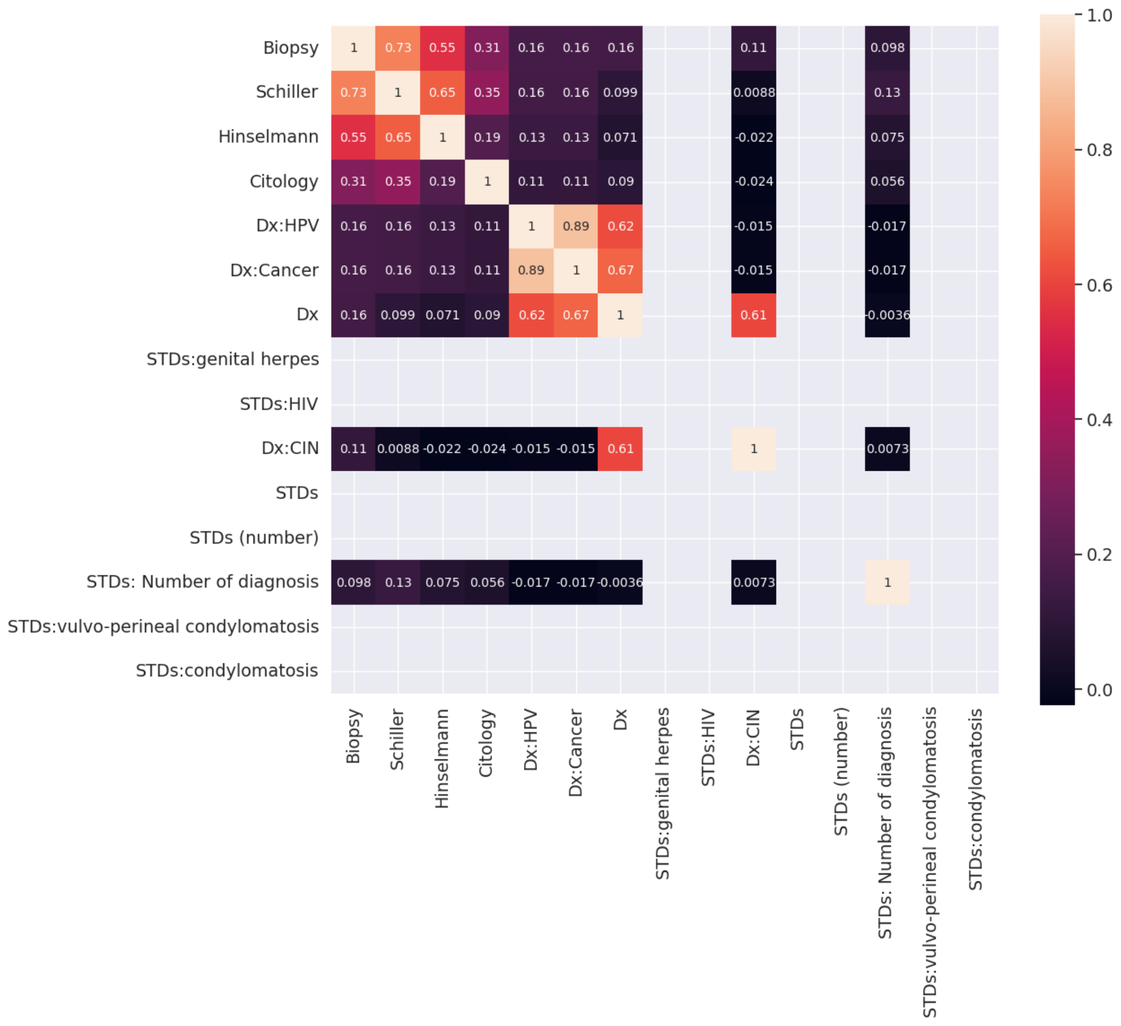
```
[13]: fig, (axis1) = plt.subplots(1, 1, figsize = (15, 5))
      sns.countplot(x = 'Biopsy', data=data, ax = axis1)
```

```
[13]: <Axes: xlabel='Biopsy', ylabel='count'>
```



```
[14]: # list the heatmap of top correlation

      corr = data.corr()

      # number of variables for heatmap
      k = 15
```

```
cols = corr.nlargest(k, 'Biopsy')['Biopsy'].index
cm = np.corrcoef(data[cols].values.T)

plt.figure(figsize=(12, 10))

sns.set(font_scale=1.25)
sns.heatmap(cm, cbar = True, annot = True, square = True, annot_kws = {'size':␣
 ↪10},
                 yticklabels = cols.values, xticklabels = cols.values)
plt.show()
```



	Biopsy	Schiller	Hinselmann	Citology	Dx:HPV	Dx:Cancer	Dx	STDs:genital herpes	STDs:HIV	Dx:CIN	STDs	STDs (number)	STDs: Number of diagnosis	STDs:vulvo-perineal condylomatosis	STDs:condylomatosis
Biopsy	1	0.73	0.55	0.31	0.16	0.16	0.16			0.11			0.098		
Schiller	0.73	1	0.65	0.35	0.16	0.16	0.099			0.0088			0.13		
Hinselmann	0.55	0.65	1	0.19	0.13	0.13	0.071			-0.022			0.075		
Citology	0.31	0.35	0.19	1	0.11	0.11	0.09			-0.024			0.056		
Dx:HPV	0.16	0.16	0.13	0.11	1	0.89	0.62			-0.015			-0.017		
Dx:Cancer	0.16	0.16	0.13	0.11	0.89	1	0.67			-0.015			-0.017		
Dx	0.16	0.099	0.071	0.09	0.62	0.67	1			0.61			-0.0036		
STDs:genital herpes															
STDs:HIV															
Dx:CIN	0.11	0.0088	-0.022	-0.024	-0.015	-0.015	0.61			1			0.0073		
STDs															
STDs (number)															
STDs: Number of diagnosis	0.098	0.13	0.075	0.056	-0.017	-0.017	-0.0036			0.0073			1		
STDs:vulvo-perineal condylomatosis															
STDs:condylomatosis															

4 Data Preprocessing

[17]:
```
# Inputing the missing values from the given dataset
# we will impute the categorical variables with 0 or 1 and continuous variables␣
↪with median value

data['Number of sexual partners'] = data['Number of sexual partners'].
↪fillna(data['Number of sexual partners'].median())
data['Number of sexual partners'].isnull().any()
```

[17]:
```
False
```

[18]:
```
# Inputing the missing values from First sexual intercourse

data['First sexual intercourse'] = data['First sexual intercourse'].
↪fillna(data['First sexual intercourse'].median())
data['First sexual intercourse'].isnull().any()
```

[18]:
```
False
```

[19]:
```
# Inputing the missing values from Num of pregnancies

data['Num of pregnancies'] = data['Num of pregnancies'].fillna(data['Num of␣
↪pregnancies'].median())
data['Num of pregnancies'].isnull().any()
```

[19]:
```
False
```

[20]:
```
# Inputing the missing values from Smokes

data['Smokes'] = data['Smokes'].fillna(data['Smokes'].median())
data['Smokes'].isnull().any()
```

[20]:
```
False
```

[21]:
```
# Inputing the missing values from Smokes (years)

data['Smokes (years)'] = data['Smokes (years)'].fillna(1)
data['Smokes (years)'].isnull().any()
```

[21]:
```
False
```

[22]:
```
# Inputing the missing values from Smokes (packs/year)

data['Smokes (packs/year)'] = data['Smokes (packs/year)'].fillna(data['Smokes␣
↪(packs/year)'].median())
data['Smokes (packs/year)'].isnull().any()
```

[22]: False

[23]:
```
# Inputing the missing values from Hormonal Contraceptives

data['Hormonal Contraceptives'] = data['Hormonal Contraceptives'].
 ↪fillna(data['Hormonal Contraceptives'].median())
data['Hormonal Contraceptives'].isnull().any()
```

[23]: False

[24]:
```
# Inputing the missing values from Hormonal Contraceptives (years)

data['Hormonal Contraceptives (years)'] = data['Hormonal Contraceptives␣
 ↪(years)'].fillna(data['Hormonal Contraceptives (years)'].median())
data['Hormonal Contraceptives (years)'].isnull().any()
```

[24]: False

[25]:
```
# Inputing the missing values from IUD

data['IUD'] = data['IUD'].fillna(0)
data['IUD'].isnull().any()
```

[25]: False

[26]:
```
# Inputing the missing values from IUD (years)

data['IUD (years)'] = data['IUD (years)'].fillna(0)
data['IUD (years)'].isnull().any()
```

[26]: False

[27]:
```
# Inputing the missing values from STDs

data['STDs'] = data['STDs'].fillna(1)
data['STDs'].isnull().any()
```

[27]: False

[28]:
```
# Inputing the missing values from STDs (number)

data['STDs (number)'] = data['STDs (number)'].fillna(data['STDs (number)'].
 ↪median())
data['STDs (number)'].isnull().any()
```

[28]: False

[29]:
```
# Inputing the missing values from STDs:condylomatosis

data['STDs:condylomatosis'] = data['STDs:condylomatosis'].fillna(data['STDs:
↪condylomatosis'].median())
data['STDs:condylomatosis'].isnull().any()
```

[29]:
```
False
```

[30]:
```
# Inputing the missing values from STDs:cervical condylomatosis

data['STDs:cervical condylomatosis'] = data['STDs:cervical condylomatosis'].
↪fillna(data['STDs:cervical condylomatosis'].median())
data['STDs:cervical condylomatosis'].isnull().any()
```

[30]:
```
False
```

[31]:
```
# Inputing the missing values from STDs:vaginal condylomatosis

data['STDs:vaginal condylomatosis'] = data['STDs:vaginal condylomatosis'].
↪fillna(data['STDs:vaginal condylomatosis'].median())
data['STDs:vaginal condylomatosis'].isnull().any()
```

[31]:
```
False
```

[32]:
```
# Inputing the missing values from STDs:vulvo-perineal condylomatosis

data['STDs:vulvo-perineal condylomatosis'] = data['STDs:vulvo-perineal␣
↪condylomatosis'].fillna(data['STDs:vulvo-perineal condylomatosis'].median())
data['STDs:vulvo-perineal condylomatosis'].isnull().any()
```

[32]:
```
False
```

[33]:
```
# Inputing the missing values from STDs:syphilis

data['STDs:syphilis'] = data['STDs:syphilis'].fillna(data['STDs:syphilis'].
↪median())
data['STDs:syphilis'].isnull().any()
```

[33]:
```
False
```

[34]:
```
# Inputing the missing values from STDs:pelvic inflammatory diseases

data['STDs:pelvic inflammatory disease'] = data['STDs:pelvic inflammatory␣
↪disease'].fillna(data['STDs:pelvic inflammatory disease'].median())
data['STDs:pelvic inflammatory disease'].isnull().any()
```

[34]:
```
False
```

[35]:
```
# Inputing the missing values from STDs:genital herpes

data['STDs:genital herpes'] = data['STDs:genital herpes'].fillna(data['STDs:
 ↪genital herpes'].median())
data['STDs:genital herpes'].isnull().any()
```

[35]:
```
False
```

[ ]:
```
# Inputing the missing values from STDs:molluscum contagiosum

data['STDs:molluscum contagiosum'] = data['STDs:molluscum contagiosum'].
 ↪fillna(data['STDs:molluscum contagiosum'].median())
data['STDs:molluscum contagiosum'].isnull().any()
```

[ ]:
```
False
```

[36]:
```
# Inputing the missing values from STDs:AIDS

data['STDs:AIDS'] = data['STDs:AIDS'].fillna(data['STDs:AIDS'].median())
data['STDs:AIDS'].isnull().any()
```

[36]:
```
False
```

[37]:
```
# Inputing the missing values from STDs:HIV

data['STDs:HIV'] = data['STDs:HIV'].fillna(data['STDs:HIV'].median())
data['STDs:HIV'].isnull().any()
```

[37]:
```
False
```

[38]:
```
# Inputing the missing values from STDs:Hepatitis B

data['STDs:Hepatitis B'] = data['STDs:Hepatitis B'].fillna(data['STDs:Hepatitis␣
 ↪B'].median())
data['STDs:Hepatitis B'].isnull().any()
```

[38]:
```
False
```

[39]:
```
# Inputing the missing values from STDs:HPV

data['STDs:HPV'] = data['STDs:HPV'].fillna(data['STDs:HPV'].median())
data['STDs:HPV'].isnull().any()
```

[39]:
```
False
```

[40]:
```
# Inputing the missing values from STDs: Time since first diagnosis
```

```
data['STDs: Time since first diagnosis'] = data['STDs: Time since first␣
↪diagnosis'].fillna(data['STDs: Time since first diagnosis'].median())
data['STDs: Time since first diagnosis'].isnull().any()
```

[40]: False

[41]:
```
# Inputing the missing values from STDs: Time since last diagnosis

data['STDs: Time since last diagnosis'] = data['STDs: Time since last␣
↪diagnosis'].fillna(data['STDs: Time since last diagnosis'].median())
data['STDs: Time since last diagnosis'].isnull().any()
```

[41]: False

[49]:
```
#STDs:molluscum contagiosum STDs: Time since last diagnosis

data['STDs:molluscum contagiosum'] = data['STDs:molluscum contagiosum'].
↪fillna(data['STDs:molluscum contagiosum'].median())
data['STDs:molluscum contagiosum'].isnull().any()
```

[49]: False

[50]:
```
# Determining the null values in each column
data.isnull().sum()
```

[50]:
```
Age                                    0
Number of sexual partners              0
First sexual intercourse               0
Num of pregnancies                     0
Smokes                                 0
Smokes (years)                         0
Smokes (packs/year)                    0
Hormonal Contraceptives                0
Hormonal Contraceptives (years)        0
IUD                                    0
IUD (years)                            0
STDs                                   0
STDs (number)                          0
STDs:condylomatosis                    0
STDs:cervical condylomatosis           0
STDs:vaginal condylomatosis            0
STDs:vulvo-perineal condylomatosis     0
STDs:syphilis                          0
STDs:pelvic inflammatory disease       0
STDs:genital herpes                    0
STDs:molluscum contagiosum             0
STDs:AIDS                              0
```

```
STDs:HIV                              0
STDs:Hepatitis B                      0
STDs:HPV                              0
STDs: Number of diagnosis             0
STDs: Time since first diagnosis      0
STDs: Time since last diagnosis       0
Dx:Cancer                             0
Dx:CIN                                0
Dx:HPV                                0
Dx                                    0
Hinselmann                            0
Schiller                              0
Citology                              0
Biopsy                                0
dtype: int64
```

[51]:
```
data.describe()
```

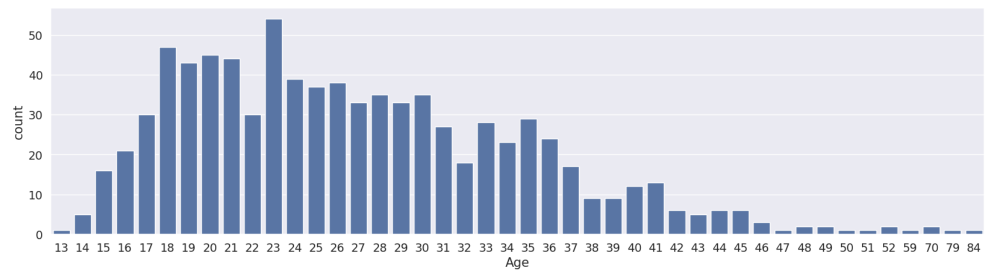
[51]:
```
              Age  Number of sexual partners  First sexual intercourse  \
count  835.000000                 835.000000                835.000000
mean    27.023952                   2.535329                 17.020359
std      8.482986                   1.654044                  2.805154
min     13.000000                   1.000000                 10.000000
25%     21.000000                   2.000000                 15.000000
50%     26.000000                   2.000000                 17.000000
75%     32.000000                   3.000000                 18.000000
max     84.000000                  28.000000                 32.000000

       Num of pregnancies      Smokes  Smokes (years)  Smokes (packs/year)  \
count          835.000000  835.000000      835.000000           835.000000
mean             2.283832    0.147305        1.249898             0.458571
std              1.408152    0.354623        4.108449             2.239363
min              0.000000    0.000000        0.000000             0.000000
25%              1.000000    0.000000        0.000000             0.000000
50%              2.000000    0.000000        0.000000             0.000000
75%              3.000000    0.000000        0.000000             0.000000
max             11.000000    1.000000       37.000000            37.000000

       Hormonal Contraceptives  Hormonal Contraceptives (years)         IUD  \
count               835.000000                       835.000000  835.000000
mean                  0.694611                         2.080520    0.099401
std                   0.460848                         3.601364    0.299379
min                   0.000000                         0.000000    0.000000
25%                   0.000000                         0.000000    0.000000
50%                   1.000000                         0.500000    0.000000
75%                   1.000000                         3.000000    0.000000
max                   1.000000                        30.000000    1.000000
```

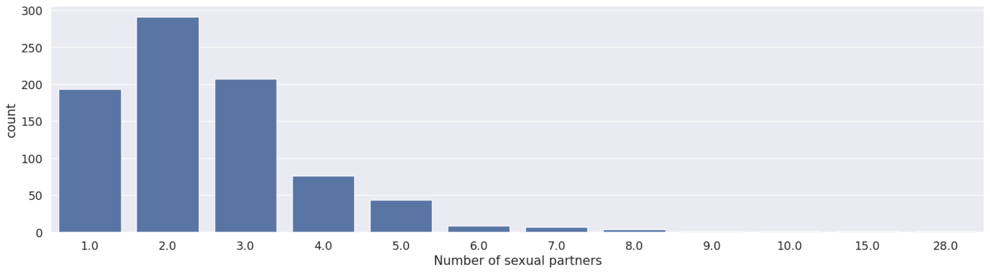
```
       …  STDs: Time since first diagnosis  STDs: Time since last diagnosis  \
count  …                        835.000000                       835.000000   
mean   …                          4.182036                         3.239521   
std    …                          1.809358                         1.843420   
min    …                          1.000000                         1.000000   
25%    …                          4.000000                         3.000000   
50%    …                          4.000000                         3.000000   
75%    …                          4.000000                         3.000000   
max    …                         22.000000                        22.000000   

        Dx:Cancer      Dx:CIN      Dx:HPV          Dx  Hinselmann    Schiller  \
count  835.000000  835.000000  835.000000  835.000000  835.000000  835.000000   
mean     0.021557    0.010778    0.021557    0.028743    0.041916    0.087425   
std      0.145319    0.103320    0.145319    0.167182    0.200518    0.282626   
min      0.000000    0.000000    0.000000    0.000000    0.000000    0.000000   
25%      0.000000    0.000000    0.000000    0.000000    0.000000    0.000000   
50%      0.000000    0.000000    0.000000    0.000000    0.000000    0.000000   
75%      0.000000    0.000000    0.000000    0.000000    0.000000    0.000000   
max      1.000000    1.000000    1.000000    1.000000    1.000000    1.000000   

         Citology      Biopsy  
count  835.000000  835.000000  
mean     0.051497    0.064671  
std      0.221142    0.246091  
min      0.000000    0.000000  
25%      0.000000    0.000000  
50%      0.000000    0.000000  
75%      0.000000    0.000000  
max      1.000000    1.000000  

[8 rows x 36 columns]
```

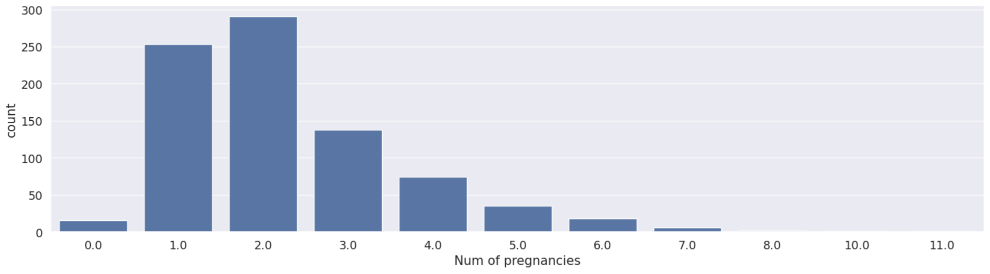
5 Data Visualization

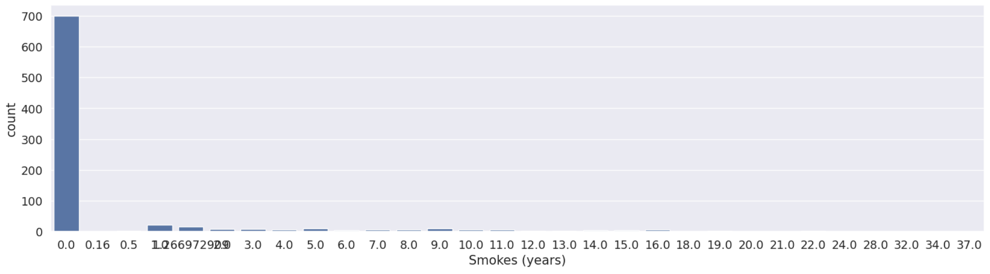
[52]:
```
fig, (ax1,ax2,ax3,ax4,ax5,ax6,ax7) = plt.subplots(7, 1, figsize = (20,40))
sns.countplot(x='Age', data=data, ax=ax1)
sns.countplot(x='Number of sexual partners', data=data, ax=ax2)
sns.countplot(x='Num of pregnancies', data=data, ax=ax3)
sns.countplot(x='Smokes (years)', data=data, ax=ax4)
sns.countplot(x='Hormonal Contraceptives (years)', data=data, ax=ax5)
sns.countplot(x='IUD (years)', data=data, ax=ax6)
sns.countplot(x='STDs (number)', data=data, ax=ax7)
```

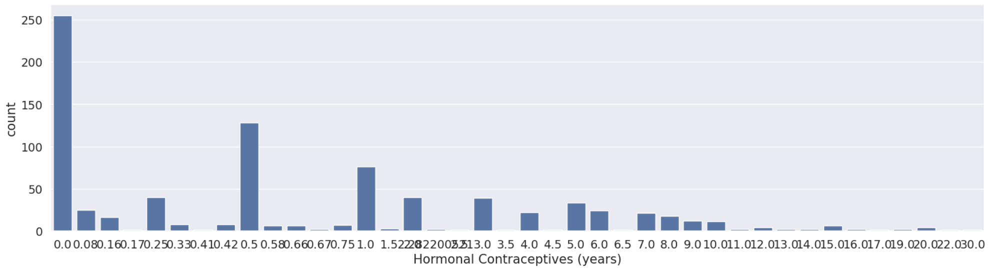
[52]:
```
<Axes: xlabel='STDs (number)', ylabel='count'>
```

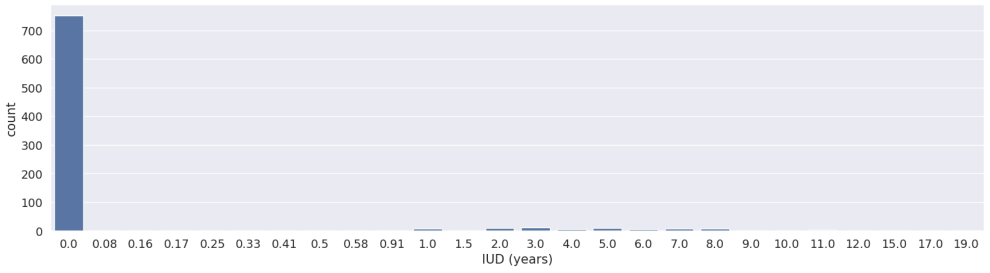

count
Age
13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 59 70 79 84

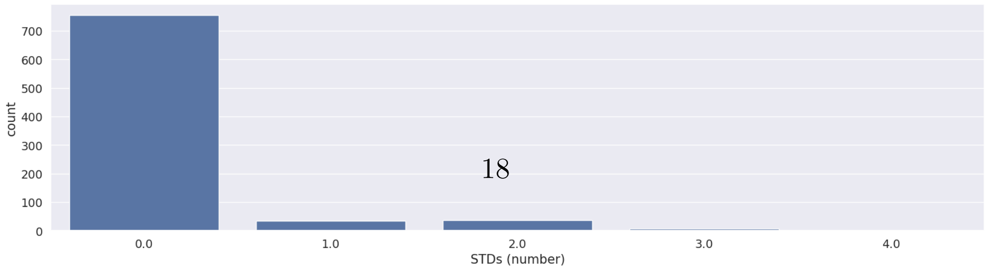

count
Number of sexual partners
1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0 15.0 28.0


count
Num of pregnancies
0.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 10.0 11.0


count
Smokes (years)


count
Hormonal Contraceptives (years)


count
IUD (years)


count
STDs (number)
0.0 1.0 2.0 3.0 4.0

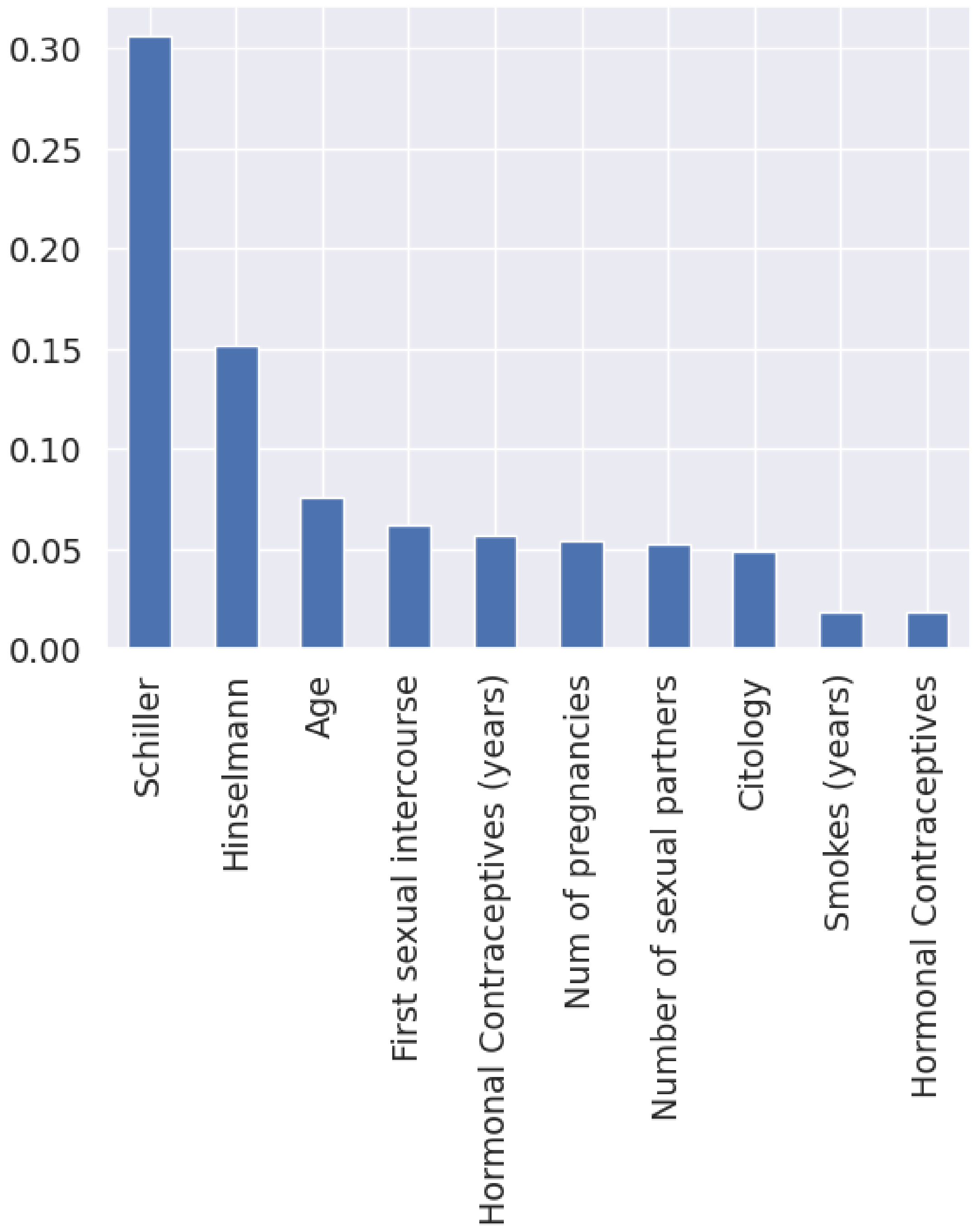
```
[53]: x=data[['Age', 'Number of sexual partners', 'First sexual intercourse','Num of␣
      ↪pregnancies', 'Smokes', 'Smokes (years)', 'Smokes (packs/year)','Hormonal␣
      ↪Contraceptives', 'Hormonal Contraceptives (years)', 'IUD','IUD (years)',␣
      ↪'STDs', 'STDs (number)', 'STDs:condylomatosis','STDs:cervical␣
      ↪condylomatosis', 'STDs:vaginal condylomatosis','STDs:vulvo-perineal␣
      ↪condylomatosis', 'STDs:syphilis','STDs:pelvic inflammatory disease', 'STDs:
      ↪genital herpes','STDs:molluscum contagiosum', 'STDs:AIDS', 'STDs:HIV','STDs:
      ↪Hepatitis B', 'STDs:HPV', 'STDs: Number of diagnosis','STDs: Time since␣
      ↪first diagnosis', 'STDs: Time since last diagnosis','Dx:Cancer', 'Dx:CIN',␣
      ↪'Dx:HPV', 'Dx', 'Hinselmann', 'Schiller', 'Citology']]
      y=data[['Biopsy']]
```

```
[54]: print(x.shape)
      print(y.shape)
```

```
(835, 35)
(835, 1)
```

```
[55]: from sklearn.ensemble import ExtraTreesClassifier
      # Building the model
      model = ExtraTreesClassifier()

      # Training the model
      model.fit(x, y)
      col=x.columns
      imp=pd.Series(model.feature_importances_,index=col)
      imp
      imp.nlargest(10).plot(kind='bar')
```

```
[55]: <Axes: >
```



[56]:
```
#Only considering the most important features for prediction

xnew=data[['Schiller', 'Hinselmann', 'Age', 'First sexual intercourse',␣
 ↪'Hormonal Contraceptives (years)', 'Num of pregnancies', 'Number of sexual␣
 ↪partners', 'Citology', 'Smokes (years)', 'Hormonal Contraceptives']]
```

[57]:
```
# splitting the dataset into  training and test set

from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(xnew, y, test_size = 0.4,␣
 ↪random_state = 45)

print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
(501, 10)
(501, 1)
(334, 10)
(334, 1)
```

[58]:
```
# MinMaxScaling

from sklearn.preprocessing import MinMaxScaler

# creating a minmax scaler
mm = MinMaxScaler()

# feeding the independent data into the scaler
x_train = mm.fit_transform(x_train)
x_test = mm.fit_transform(x_test)
```

MODELLING

#Logistic Regression

[59]:
```
from sklearn.linear_model import LogisticRegression

# creating the model
model = LogisticRegression()

# feeding the training data into the model
model.fit(x_train, y_train)

# predicting the test set results
y_pred = model.predict(x_test)

# Calculating the accuracies
print("Training accuracy :", model.score(x_train, y_train))
print("Testing accuracy :", model.score(x_test, y_test))

# classification report
```

```
print(classification_report(y_test, y_pred))

# confusion matrix
print(confusion_matrix(y_test, y_pred))
```

```
Training accuracy : 0.9640718562874252
Testing accuracy : 0.9550898203592815
              precision    recall  f1-score   support

           0       0.96      0.99      0.98       313
           1       0.75      0.43      0.55        21

    accuracy                           0.96       334
   macro avg       0.86      0.71      0.76       334
weighted avg       0.95      0.96      0.95       334

[[310   3]
 [ 12   9]]
```

#Support Vector Machine

[60]:
```
from sklearn.svm import SVC

# creating the model
model = SVC()

# feeding the training data into the model
model.fit(x_train, y_train)

# predicting the test set results
y_pred = model.predict(x_test)

# Calculating the accuracies
print("Training accuracy :", model.score(x_train, y_train))
print("Testing accuracy :", model.score(x_test, y_test))

# classification report
print(classification_report(y_test, y_pred))

# confusion matrix
print(confusion_matrix(y_test, y_pred))
```

```
Training accuracy : 0.9680638722554891
Testing accuracy : 0.9580838323353293
              precision    recall  f1-score   support

           0       0.97      0.98      0.98       313
           1       0.68      0.62      0.65        21
```

```
    accuracy                           0.96       334
   macro avg       0.83      0.80      0.81       334
weighted avg       0.96      0.96      0.96       334

[[307   6]
 [  8  13]]
```

#Decision Tree

```
[61]: from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier

# creating the model
model = DecisionTreeClassifier()

# feeding the training data into the model
model.fit(x_train, y_train)

# predicting the test set results
yb_pred = model.predict(x_test)

# Calculating the accuracies
print("Training accuracy :", model.score(x_train, y_train))
print("Testing accuracy :", model.score(x_test, y_test))

# classification report
print(classification_report(y_test, y_pred))

# confusion matrix
print(confusion_matrix(y_test, y_pred))
```

```
Training accuracy : 1.0
Testing accuracy : 0.9101796407185628
              precision    recall  f1-score   support

           0       0.97      0.98      0.98       313
           1       0.68      0.62      0.65        21

    accuracy                           0.96       334
   macro avg       0.83      0.80      0.81       334
weighted avg       0.96      0.96      0.96       334

[[307   6]
 [  8  13]]
```

#Random Forest

[62]:
```python
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report

# creating the model
model = LogisticRegression()

# feeding the training data into the model
model.fit(x_train, y_train)

# predicting the test set results
y_pred = model.predict(x_test)

# Calculating the accuracies
print("Training accuracy :", model.score(x_train, y_train))
print("Testing accuracy :", model.score(x_test, y_test))

# classification report
print(classification_report(y_test, y_pred))

# confusion matrix
print(confusion_matrix(y_test, y_pred))
```

```
Training accuracy : 0.9640718562874252
Testing accuracy : 0.9550898203592815
              precision    recall  f1-score   support

           0       0.96      0.99      0.98       313
           1       0.75      0.43      0.55        21

    accuracy                           0.96       334
   macro avg       0.86      0.71      0.76       334
weighted avg       0.95      0.96      0.95       334

[[310   3]
 [ 12   9]]
```