Babyhealth analysis using CTU-CHB Intrapartum Cardiotocography Database

link to github repo : https://github.com/Ali19971/Final-report-DSCI235 (https://github.com/Ali19971/Final-report-DSCI235)

The Idea of figuring out the overall health of a newborn baby without any hardcore test is very fascinating in itself. It's a major issue in the medical field since the babies are so fragile we can not perform a lot of tests on them. SO the idea of this project is to come with a solution that can be used to conclude some important insights about the health of a newborn without doing a lot of tests on them.

We are going to address some very interesting and important analytical questions which will end up giving us very brief knowledge about the nature of the data, the hidden patterns in the dataset, and some really important relationships between various attributes and baby health.

To come up with some useful insights, I will be answering the following questions:

- **What is the average age of mothers for male and female babies?**
- **What is the value under which 95% of the gestweeks would lie?**
- **Identify the parity for most of the mothers?**
- **Compare the mean and trimmed mean of age and comment on the presence of outliers. Back your answer up with a boxplot.**
- **Find the percentage variation in age for the diabetic and non-diabetic patients.**
- **Define a function to plot the categories of the rectype and find the category with the highest frequency**
- **What is the probability of a male baby being healthy, find the same for the female baby.**
- **Given that the mother is diabetic and the rectype is ultrasound find out if the health of the baby is good or not.**
- **What are the chances for a baby to be healthy if the mother is not very young? Compare the result with the same of a young mother.**

In [6]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

%matplotlib inline
```

In [7]:
```
1 data = pd.read_csv('train.csv')
2 data.head()
```

Out[7]:

	recordID	babyhealth	gestweeks	sex	age	gravidity	parity	diabetes	hypertension	preecla
0	5086	1	38.0	1.0	27.0	1.0	0.0	0.0	0.0	
1	5491	0	40.0	1.0	32.0	1.0	1.0	1.0	0.0	
2	5097	0	41.0	1.0	29.0	2.0	0.0	0.0	0.0	
3	5108	1	39.0	1.0	29.0	1.0	0.0	0.0	0.0	
4	5264	0	39.0	1.0	28.0	1.0	0.0	0.0	1.0	

Database description

The data contains the following fields:

- **recordtD**: Unique hash ID of partum record.
- **babyhealth**: (Outcome) indicates the overall health status of the baby at birth as measured according to the pH levels after birth. babyhealth is '0' if baby's health is considered normal, or '1', if there is a health issue.
- **gestweeks**: is the number of weeks of gestation where 39-40 weeks represent a normal term delivery.
- **sex**: this is '1' for female, '2' for male.
- **age**: the age in years of the mother.
- **gravidity**: is the number of times the mother has been pregnant In her life.
- **parity**: is the number of times the woman has been pregnant for more than 24 weeks (in a single pregnancy).
- **diabetes**: binary value for diabetes: '1': yes.
- **hypertension**: binary value for hypertension: '1': yes.
- **preeclampsia**: binary value for preeclampsia: '1': yes.
- **pyrexia**: binary value that indicates the presence of pyrexia (high temperature) in the mother ('1') or not ('0').
- **meconium**: binary value that indicates the presence ('r) or not ('O') of rneconiurn.
- **noprogress**: binary value that indicates if there has been an abort of the pregnancy ('1') or not ('0').
- **rectype**: codes the type of measurement device. It could be '1': ultrasound, '2': scalp, '12': both.

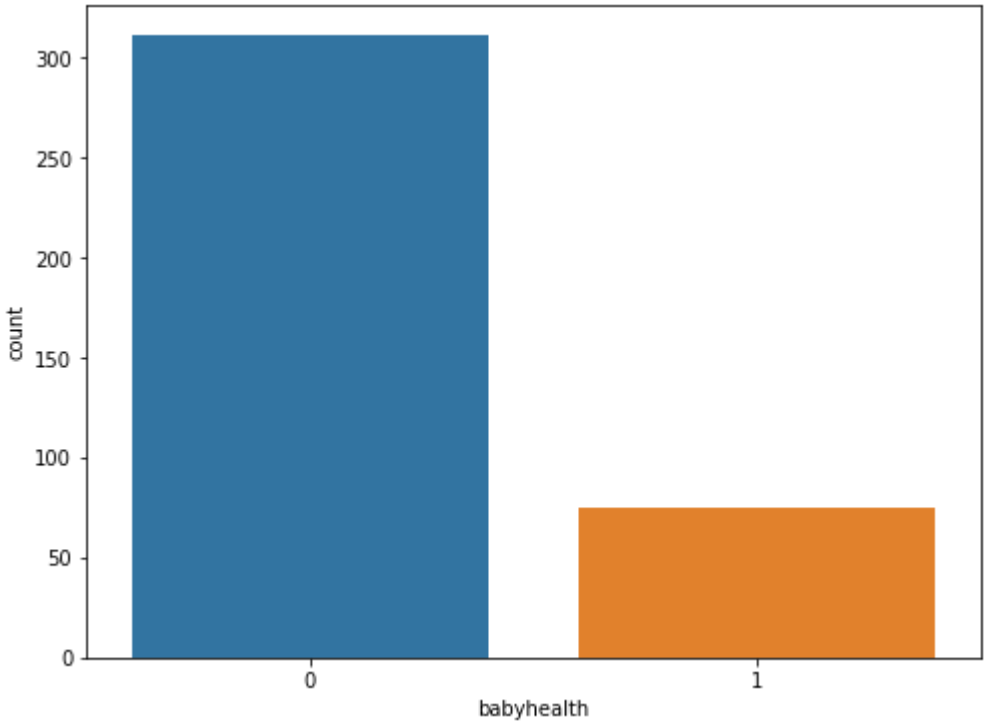
In [10]:
```
# Now we will look at the basic info about the data.
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 386 entries, 0 to 385
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   recordID      386 non-null    int64  
 1   babyhealth    386 non-null    int64  
 2   gestweeks     386 non-null    float64
 3   sex           386 non-null    float64
 4   age           386 non-null    float64
 5   gravidity     384 non-null    float64
 6   parity        386 non-null    float64
 7   diabetes      386 non-null    float64
 8   hypertension  386 non-null    float64
 9   preeclampsia  386 non-null    float64
 10  pyrexia       386 non-null    float64
 11  meconium      386 non-null    float64
 12  noprogress    386 non-null    float64
 13  rectype       384 non-null    float64
dtypes: float64(12), int64(2)
memory usage: 42.3 KB
```

In [84]:
```
# From the above table we can see that the data does not contain any
# but still for lets find out the number of null values in each col
print('Number of null values in each columns is shown below:\n')
print(data.isna().sum())

# Now lets fill the null values with the mode of the corresponding
for col in data.columns:
    data.fillna(data[col].mode()[0])
print('\nThe list after filling the missing values is shown below:\
print(data.isna().sum())
```

```
Number of null values in each columns is shown below:

recordID        0
babyhealth      0
gestweeks       0
sex             0
age             0
gravidity       2
parity          0
diabetes        0
hypertension    0
preeclampsia    0
pyrexia         0
meconium        0
noprogress      0
rectype         2
dtype: int64

The list after filling the missing values is shown below:

recordID        0
babyhealth      0
gestweeks       0
sex             0
age             0
gravidity       2
parity          0
diabetes        0
hypertension    0
preeclampsia    0
pyrexia         0
meconium        0
noprogress      0
rectype         2
dtype: int64
```

In [15]:
```
# Proportion of different categories of the Baby health is shown be

plt.figure(figsize = (8, 6))
sns.countplot(data['babyhealth'])
plt.show()
```



? 1. What is the average age of mothers for male and female babies?

In [22]:
```
temp = data.groupby('sex')['age'].mean()

print(f'The average age of mothers for male babies is {round(temp[2
print(f'The average age of mothers for female babies is {round(temp

```

```
The average age of mothers for male babies is 29.61
The average age of mothers for female babies is 29.78
```

The above result shows that there is no significant relationship between the age of the mother and the gender of the baby, since the average age of mother for male babies and for female babies is almost the same.

2. What is the value under which 95% of the gestweeks would lie?

In [27]:
```
temp = data['gestweeks'].quantile(0.95)
print(f'The value under which 95% of the gestweeks would lie is {te
```

```
The value under which 95% of the gestweeks would lie is 41.0
```

3. Identify the parity for most of the mothers?

In [32]:
```
parity_count = data.groupby('parity').count()
parity_count.reset_index(inplace=True)
max_parity = parity_count[parity_count['babyhealth'] == parity_coun
print(f'The parity of most of the mothers is {max_parity}')

```

```
The parity of most of the mothers is 0.0
```

Parity is defined as the number of times that a woman has given birth to a fetus with a gestational age of 24 weeks or more, regardless of whether the child was born alive or was stillborn.

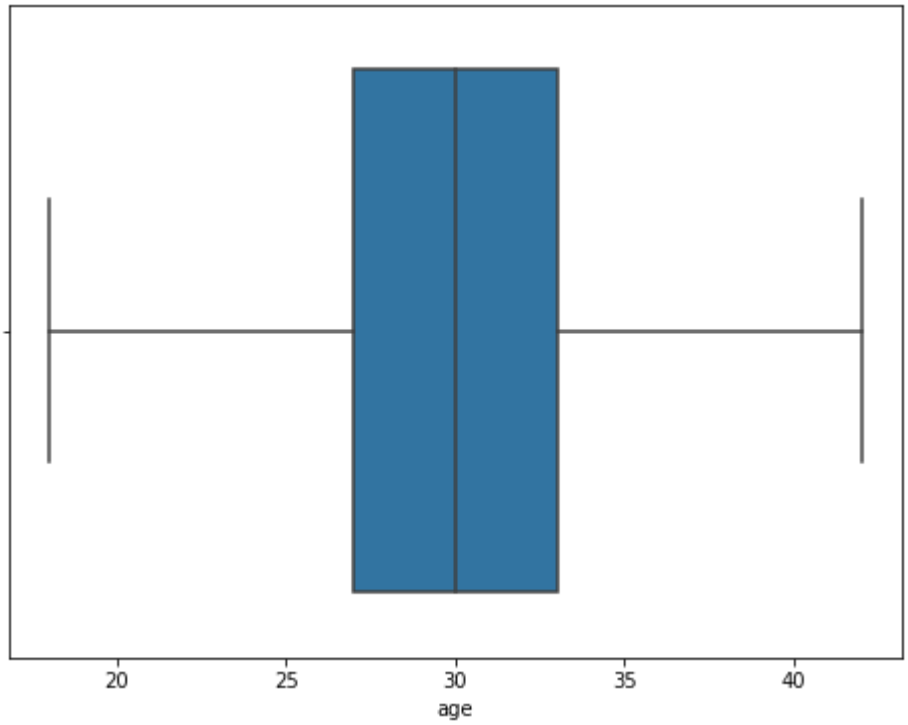
The above result shows that the parity of most of the women is 0, which means that most of the women have never given birth to a fetus with a gestational age of 24 weeks or more.

4. Compare the mean and trimmed mean of age and comment on the presence of outliers. Back your answer up with a boxplot.

In [35]:
```
trimmed_age = data[data['age'] != data['age'].min()]
trimmed_age = trimmed_age[trimmed_age['age'] != trimmed_age['age'].m

print(f'Trimmed min: {trimmed_age.mean()}')
print(f"Normal min {data['age'].mean()}")
```

```
Trimmed min: 29.671916010498688
Normal min 29.707253886010363
```

In [36]:
```
plt.figure(figsize = (8, 6))
sns.boxplot(data['age'])
plt.show()
```



If the trimmed mean and the normal mean differ significantly then we can say that the data contains some outliers, but in this case, the difference between the trimmed mean and the normal mean is very small, which clearly concludes that there is no outlier present in the age column.

From the boxplot, we can see that the data does not contain any outliers too.

5. Find the percentage variation in age for the diabetic and non-diabetic patients.

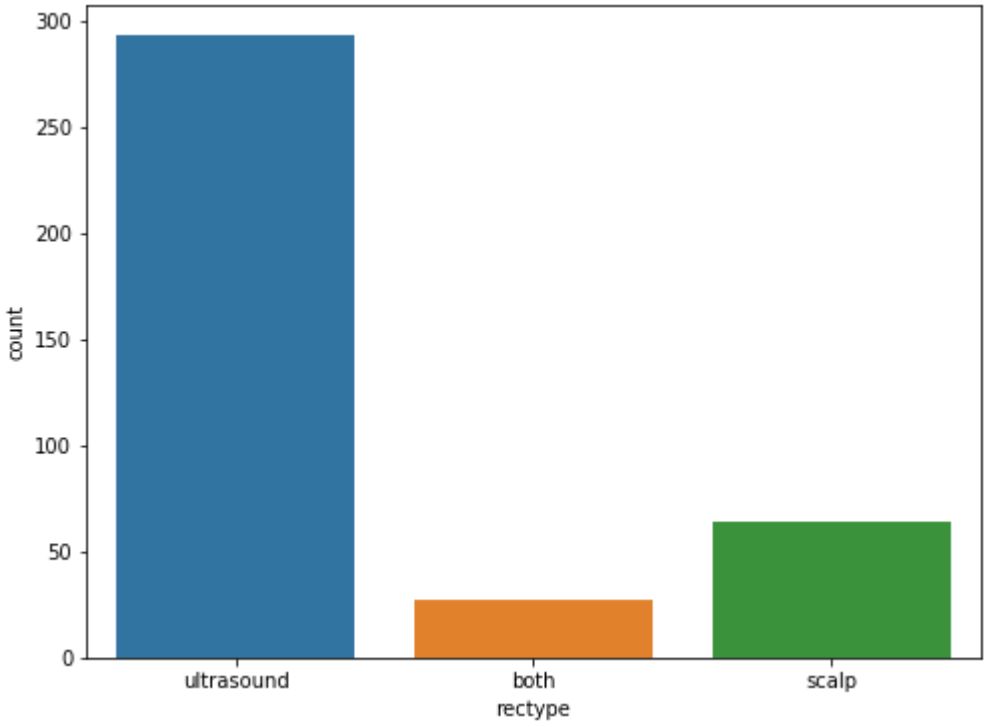
In [58]:
```
temp = data.groupby('diabetes')['age'].var()
perc_var = temp / data.groupby('diabetes')['age'].var().sum()*100
print(f"percentage variation in age for diabetic and non diabetic pa
for diab in perc_var.index:
    if diab == 1:
        print(f'Diabetic    : {perc_var[diab]}')
    else:
        print(f'Non-diabetic: {perc_var[diab]}')
```

```
percentage variation in age for diabetic and non diabetic patients is
shown below

Non-diabetic: 58.10333793233483
Diabetic    : 41.89666206766516
```

6. Define a function to plot the categories of the rectype and find the category with the highest frequency

In [66]:
```
rectype = {1: 'ultrasound', 2: 'scalp', 12: 'both'}
def plot(df, col):
    def change_rec_name(rec):
        return rectype[rec]
    plt.figure(figsize = (8, 6))
    sns.countplot(df[col].dropna().apply(change_rec_name))
    freq = df.groupby(col).count()
    freq = freq.reset_index()
    return freq[freq[freq.columns[-1]] == freq[freq.columns[-1]].ma

res = plot(data, 'rectype')
print(f'The category with highest frequency is {rectype[res]}')
```

```
The category with highest frequency is ultrasound
```



7. What is the probability of a male baby to be health, find the same for the female baby.

In [68]:
```
male = data[data['sex'] == 2]
healthy_baby = male[male['babyhealth'] == 0]['recordID'].count()
un_healthy_baby = male[male['babyhealth'] == 1]['recordID'].count()

prob = healthy_baby/(un_healthy_baby + healthy_baby)*100

print(f'The posibility of a baby of bieng healthy if it is boy is :


# Lets check the same for female babies as well

female = data[data['sex'] == 1]
healthy_baby = female[female['babyhealth'] == 0]['recordID'].count(
un_healthy_baby = female[female['babyhealth'] == 1]['recordID'].cou

prob = healthy_baby/(un_healthy_baby + healthy_baby)*100

print(f'The posibility of a baby of bieng healthy if it is girl is
```

```
The posibility of a baby of bieng healthy if it is boy is : 81.5028901
734104
The posibility of a baby of bieng healthy if it is girl is : 79.812206
57276995
```

From the above result, we can clearly conclude that the probability for a male baby to be healthy is more than the probability of a female baby to be healthy.

8. Given that the mother is diabetic and the rectype is ultrasound find out if the health of the baby is good or not.

In [77]:
```
rectypeinv = {'ultrasound': 1, 'scalp': 2, 'both': 3}

temp = data[(data['diabetes'] == 1) & (data['rectype'] == rectypeinv
cnt_healthy = temp.groupby('babyhealth').count()['recordID'][0]
cnt_total = temp['recordID'].count()

prob_healthy = cnt_healthy / cnt_total * 100

if prob_healthy >= 75:
    print(f'''Given that the mother is diabetic and the rectype is
    the baby will be healthy with a probability of {prob_healthy}''
else:
    print(f'''Given that the mother is diabetic and the rectype is
    the baby will be healthy''')
```

```
Given that the mother is diabetic and the rectype is ultrasound
    the baby will be healthy with a probability of 81.25
```

From the above result, we can see that the probability for a child to be healthy is more than 80% even if the mother suffers from a medical condition like diabetes. So, we can clearly conclude that the diabetes of

the mother has not a lot to do with the health of the baby.

9. What are the chances for a baby to be healthy if the mother is not very young? Compare the result with the same of a young mother.

Note: Consider a woman to be not very young if she is of age more than that of the mean of the age column.

In [80]:
```
temp = data[data['age'] > data['age'].mean()]
cnt_healthy = temp.groupby('babyhealth').count()['recordID'][0]
cnt_total = temp['recordID'].count()

prob_healthy = cnt_healthy / cnt_total * 100

print(f'There is a {prob_healthy}% probability that the baby will b

# Now lets do the same for the young mothers
temp = data[data['age'] <= data['age'].mean()]
cnt_healthy = temp.groupby('babyhealth').count()['recordID'][0]
cnt_total = temp['recordID'].count()

prob_healthy = cnt_healthy / cnt_total * 100

print(f'There is a {prob_healthy}% probability that the baby will b

```

```
There is a 83.74384236453201% probability that the baby will be health
y if the mother is not very young.
There is a 77.04918032786885% probability that the baby will be health
y if the mother is young.
```

The above result shows that the probability of a baby to be healthy is 83.74% if the mother is not very young and the probability for the baby to be healthy if the mother is young is 74.04%. From the result, we can conclude that it's better for any woman to be not very young if the baby is supposed to be surely healthy.