problem staement:predict and analyze

In [1]:
```
import numpy as np
import pandas as pd
from sklearn import preprocessing
import matplotlib.pyplot as plt
# plt.rc("font", size=14)
import seaborn as sns
sns.set(style="white") #white background style for seaborn plots
sns.set(style="whitegrid", color_codes=True)
import warnings
warnings.simplefilter(action='ignore')
```

In [2]:
```
df = pd.read_csv(r"C:\Users\anu\Downloads\framingham.csv")
df
```

Out[2]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose
0	1	39	4.0	0	0.0	0.0	0	0	0	195.0	106.0	70.0	26.97	80.0	77.0
1	0	46	2.0	0	0.0	0.0	0	0	0	250.0	121.0	81.0	28.73	95.0	76.0
2	1	48	1.0	1	20.0	0.0	0	0	0	245.0	127.5	80.0	25.34	75.0	70.0
3	0	61	3.0	1	30.0	0.0	0	1	0	225.0	150.0	95.0	28.58	65.0	103.0
4	0	46	3.0	1	23.0	0.0	0	0	0	285.0	130.0	84.0	23.10	85.0	85.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4235	0	48	2.0	1	20.0	NaN	0	0	0	248.0	131.0	72.0	22.00	84.0	86.0
4236	0	44	1.0	1	15.0	0.0	0	0	0	210.0	126.5	87.0	19.16	86.0	NaN
4237	0	52	2.0	0	0.0	0.0	0	0	0	269.0	133.5	83.0	21.47	80.0	107.0
4238	1	40	3.0	0	0.0	0.0	0	1	0	185.0	141.0	98.0	25.60	67.0	72.0
4239	0	39	3.0	1	30.0	0.0	0	0	0	196.0	133.0	86.0	20.91	85.0	80.0

4240 rows × 16 columns

In [4]:
```
df.head()
```

Out[4]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose	Te
0	1	39	4.0	0	0.0	0.0	0	0	0	195.0	106.0	70.0	26.97	80.0	77.0	
1	0	46	2.0	0	0.0	0.0	0	0	0	250.0	121.0	81.0	28.73	95.0	76.0	
2	1	48	1.0	1	20.0	0.0	0	0	0	245.0	127.5	80.0	25.34	75.0	70.0	
3	0	61	3.0	1	30.0	0.0	0	1	0	225.0	150.0	95.0	28.58	65.0	103.0	
4	0	46	3.0	1	23.0	0.0	0	0	0	285.0	130.0	84.0	23.10	85.0	85.0	

In [5]:
```
df.tail()
```

Out[5]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose
4235	0	48	2.0	1	20.0	NaN	0	0	0	248.0	131.0	72.0	22.00	84.0	86.0
4236	0	44	1.0	1	15.0	0.0	0	0	0	210.0	126.5	87.0	19.16	86.0	NaN
4237	0	52	2.0	0	0.0	0.0	0	0	0	269.0	133.5	83.0	21.47	80.0	107.0
4238	1	40	3.0	0	0.0	0.0	0	1	0	185.0	141.0	98.0	25.60	67.0	72.0
4239	0	39	3.0	1	30.0	0.0	0	0	0	196.0	133.0	86.0	20.91	85.0	80.0

In [7]:
```
df.shape
```

Out[7]:
```
(4240, 16)
```

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4240 entries, 0 to 4239
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   male             4240 non-null   int64  
 1   age              4240 non-null   int64  
 2   education        4135 non-null   float64
 3   currentSmoker    4240 non-null   int64  
 4   cigsPerDay       4211 non-null   float64
 5   BPMeds           4187 non-null   float64
 6   prevalentStroke  4240 non-null   int64  
 7   prevalentHyp     4240 non-null   int64  
 8   diabetes         4240 non-null   int64  
 9   totChol          4190 non-null   float64
 10  sysBP            4240 non-null   float64
 11  diaBP            4240 non-null   float64
 12  BMI              4221 non-null   float64
 13  heartRate        4239 non-null   float64
 14  glucose          3852 non-null   float64
 15  TenYearCHD       4240 non-null   int64  
dtypes: float64(9), int64(7)
memory usage: 530.1 KB
```

In [9]:
```
df.describe()
```

Out[9]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP
count	4240.000000	4240.000000	4135.000000	4240.000000	4211.000000	4187.000000	4240.000000	4240.000000	4240.000000	4190.000000	4240.000000
mean	0.429245	49.580189	1.979444	0.494104	9.005937	0.029615	0.005896	0.310613	0.025708	236.699523	132.354599
std	0.495027	8.572942	1.019791	0.500024	11.922462	0.169544	0.076569	0.462799	0.158280	44.591284	22.033300
min	0.000000	32.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	107.000000	83.500000
25%	0.000000	42.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	206.000000	117.000000
50%	0.000000	49.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	234.000000	128.000000
75%	1.000000	56.000000	3.000000	1.000000	20.000000	0.000000	0.000000	1.000000	0.000000	263.000000	144.000000
max	1.000000	70.000000	4.000000	1.000000	70.000000	1.000000	1.000000	1.000000	1.000000	696.000000	295.000000

In [10]:
```
df.isnull().sum()
```

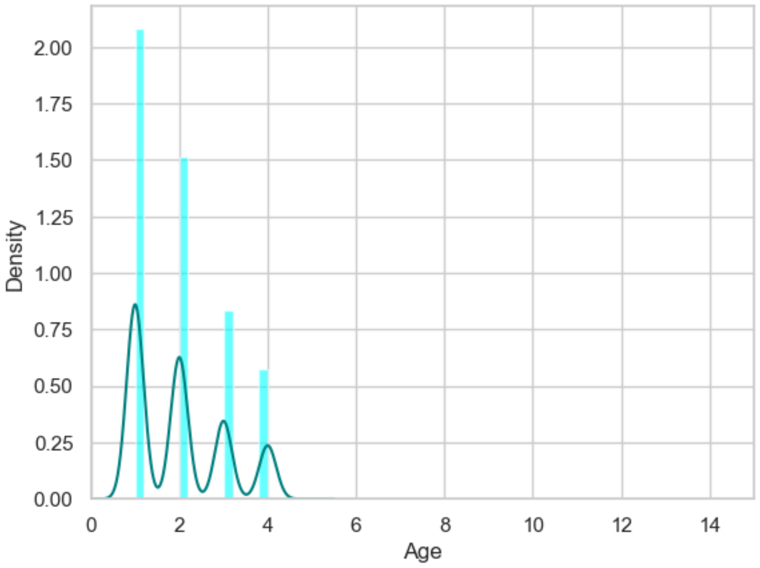
Out[10]:
```
male                 0
age                  0
education          105
currentSmoker        0
cigsPerDay          29
BPMeds              53
prevalentStroke      0
prevalentHyp         0
diabetes             0
totChol             50
sysBP                0
diaBP                0
BMI                 19
heartRate            1
glucose            388
TenYearCHD           0
dtype: int64
```

In [11]:
```
df.describe().any()
```

Out[11]:
```
male               True
age                True
education          True
currentSmoker      True
cigsPerDay         True
BPMeds             True
prevalentStroke    True
prevalentHyp       True
diabetes           True
totChol            True
sysBP              True
diaBP              True
BMI                True
heartRate          True
glucose            True
TenYearCHD         True
dtype: bool
```

In [12]:

```
ax = df["education"].hist(bins=15, density=True, stacked=True, color='cyan', alpha=0.6)
df["education"].plot(kind='density', color='teal')
ax.set(xlabel='Age')
plt.xlim(-0,15)
plt.show()
```



In [13]:

```
print(df["education"].mean(skipna=True))
print(df["education"].median(skipna=True))
```

```
1.9794437726723095
2.0
```

In [14]:

```
print((df['glucose'].isnull().sum()/df.shape[0]*100))
```

```
9.150943396226415
```

In [15]:

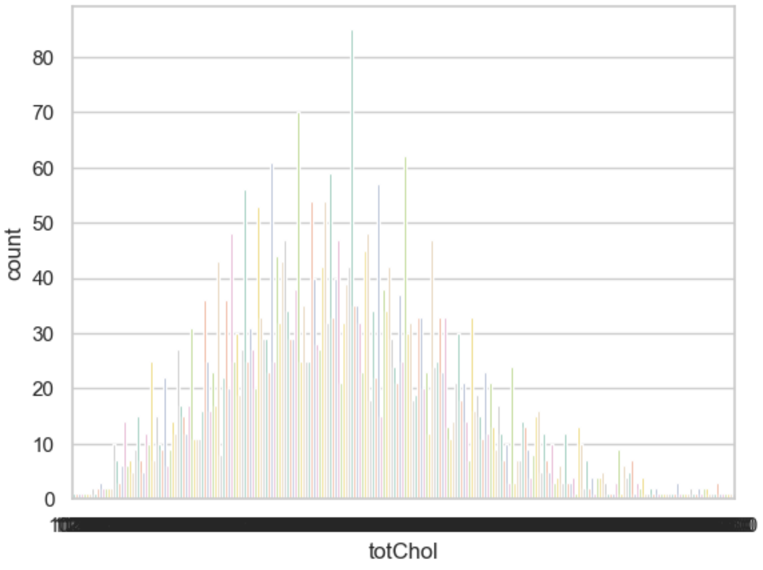
```
print((df['totChol'].isnull().sum()/df.shape[0]*100))
```

```
1.179245283018868
```

In [16]:

```
print(df['totChol'].value_counts())
sns.countplot(x='totChol', data=df, palette='Set2')
plt.show()
```

```
totChol
240.0    85
220.0    70
260.0    62
210.0    61
232.0    59
         ..
392.0     1
405.0     1
359.0     1
398.0     1
119.0     1
Name: count, Length: 248, dtype: int64
```



In [17]:

```
print(df['totChol'].value_counts().idxmax())
```

```
240.0
```

In [18]:

```
data = df.copy()
data["education"].fillna(df["education"].median(skipna=True), inplace=True)
data["totChol"].fillna(df['totChol'].value_counts().idxmax(), inplace=True)
data.drop('glucose', axis=1, inplace=True)
```

In [19]:

```
data.isnull().sum()
```

Out[19]:

```
male                0
age                 0
education           0
currentSmoker       0
cigsPerDay         29
BPMeds             53
prevalentStroke     0
prevalentHyp        0
diabetes            0
totChol             0
sysBP               0
diaBP               0
BMI                19
heartRate           1
TenYearCHD          0
dtype: int64
```

In [20]:

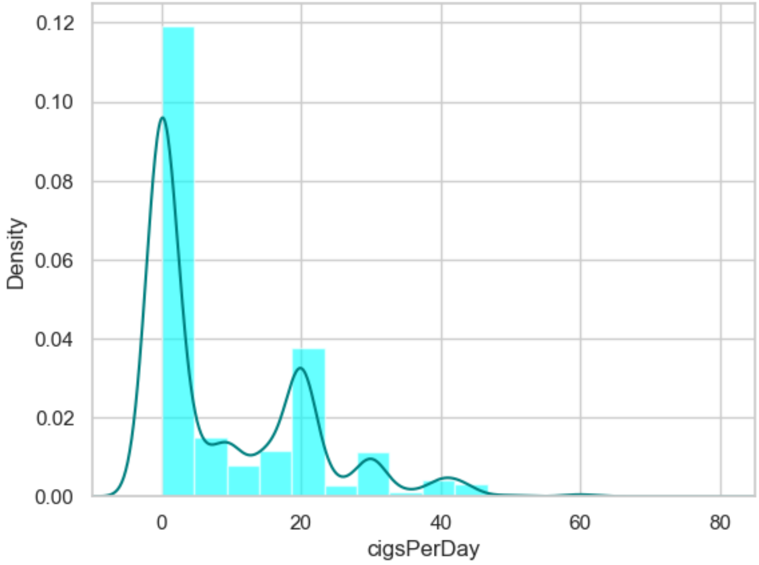
```
data.head()
```

Out[20]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	TenYearCHD
0	1	39	4.0	0	0.0	0.0	0	0	0	195.0	106.0	70.0	26.97	80.0	0
1	0	46	2.0	0	0.0	0.0	0	0	0	250.0	121.0	81.0	28.73	95.0	0
2	1	48	1.0	1	20.0	0.0	0	0	0	245.0	127.5	80.0	25.34	75.0	0
3	0	61	3.0	1	30.0	0.0	0	1	0	225.0	150.0	95.0	28.58	65.0	1
4	0	46	3.0	1	23.0	0.0	0	0	0	285.0	130.0	84.0	23.10	85.0	0

In [21]:

```
ax = df["cigsPerDay"].hist(bins=15, density=True, stacked=True, color='cyan', alpha=0.6)
df["cigsPerDay"].plot(kind='density', color='teal')
ax.set(xlabel='cigsPerDay')
plt.xlim(-10,85)
plt.show()
```



In [23]:

```
print(df["cigsPerDay"].mean(skipna=True))
print(df["cigsPerDay"].median(skipna=True))
```

```
9.005936832106388
0.0
```

In [24]:

```
print((df['BPMeds'].isnull().sum()/df.shape[0]*100))
```

```
1.25
```

In [25]:

```
print((df['BMI'].isnull().sum()/df.shape[0]*100))
```

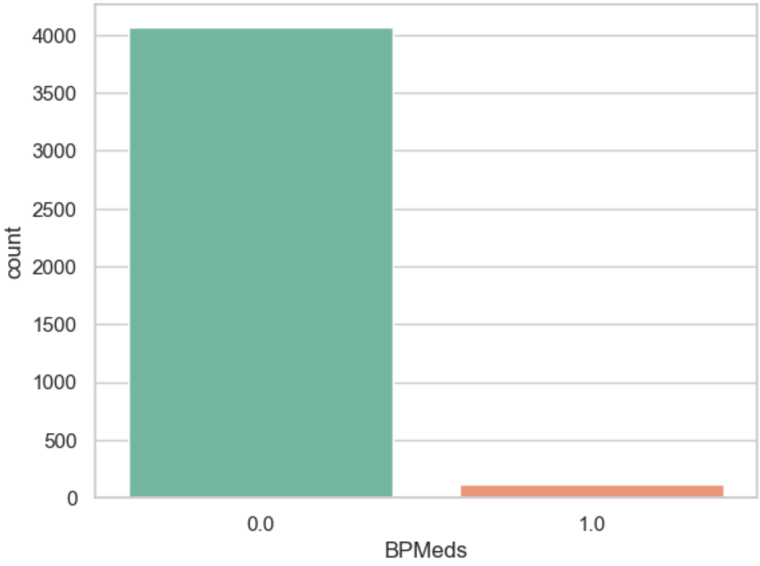
```
0.4481132075471698
```

In [26]:
```
print((df['heartRate'].isnull().sum()/df.shape[0]*100))
```

```
0.02358490566037736
```

In [27]:
```
print(df['BPMeds'].value_counts())
sns.countplot(x='BPMeds', data=df, palette='Set2')
plt.show()
```

```
BPMeds
0.0    4063
1.0     124
Name: count, dtype: int64
```



In [28]:
```
print(df['heartRate'].value_counts().idxmax())
```

```
75.0
```

In [29]:
```
data = df.copy()
data["cigsPerDay"].fillna(df["cigsPerDay"].median(skipna=True), inplace=True)
data["BPMeds"].fillna(df['BPMeds'].value_counts().idxmax(), inplace=True)
data["education"].fillna(df["education"].median(skipna=True), inplace=True)
data["totChol"].fillna(df['totChol'].value_counts().idxmax(), inplace=True)
data.drop('glucose', axis=1, inplace=True)
data.drop('BMI', axis=1, inplace=True)
data.drop('heartRate', axis=1, inplace=True)
```

In [30]:
```
data.isnull().sum()
```

Out[30]:
```
male               0
age                0
education          0
currentSmoker      0
cigsPerDay         0
BPMeds             0
prevalentStroke    0
prevalentHyp       0
diabetes           0
totChol            0
sysBP              0
diaBP              0
TenYearCHD         0
dtype: int64
```

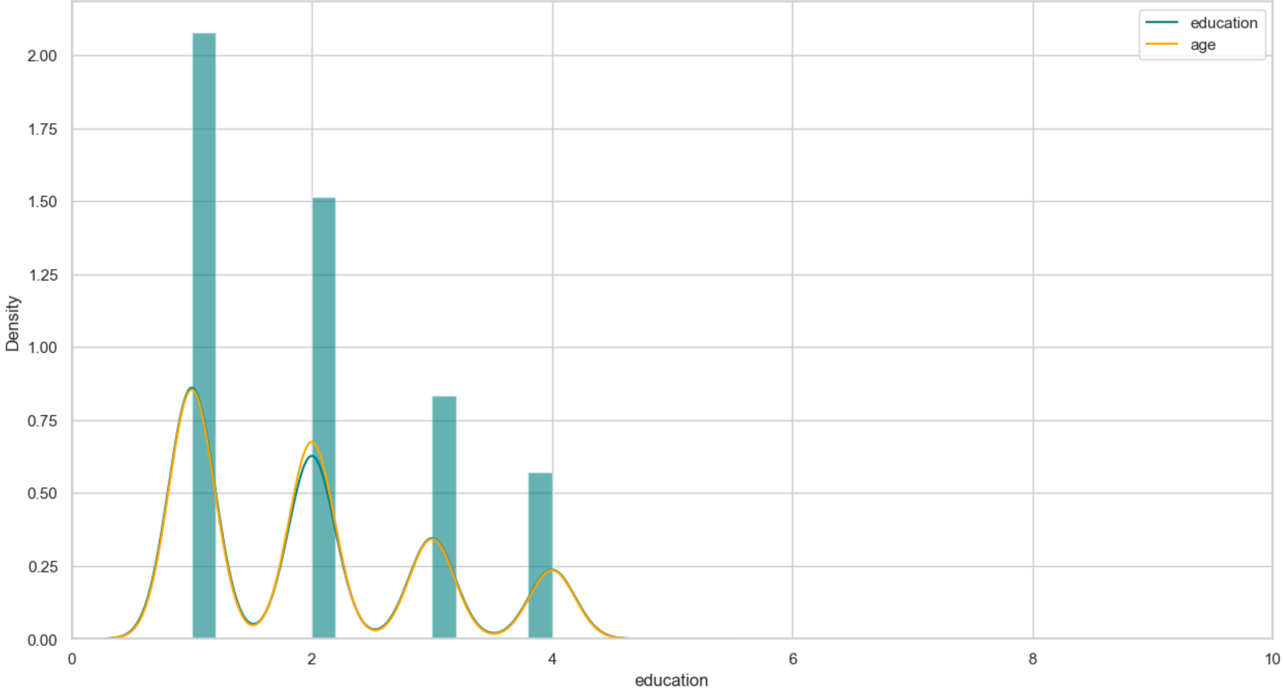
In [31]:
```
data.head()
```

Out[31]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	TenYearCHD
0	1	39	4.0	0	0.0	0.0	0	0	0	195.0	106.0	70.0	0
1	0	46	2.0	0	0.0	0.0	0	0	0	250.0	121.0	81.0	0
2	1	48	1.0	1	20.0	0.0	0	0	0	245.0	127.5	80.0	0
3	0	61	3.0	1	30.0	0.0	0	1	0	225.0	150.0	95.0	1
4	0	46	3.0	1	23.0	0.0	0	0	0	285.0	130.0	84.0	0

In [32]:

```
plt.figure(figsize=(15,8))
ax = df["education"].hist(bins=15, density=True, stacked=True, color='teal', alpha=0.6)
df["education"].plot(kind='density', color='teal')
ax =data["education"].hist(bins=15, density=True, stacked=True, color='orange', alpha=0)
data["education"].plot(kind='density', color='orange')
ax.legend(['education', 'age'])
ax.set(xlabel='education')
plt.xlim(-0,10)
plt.show()
```



In [33]:

```
data['Disease']=np.where((data["prevalentHyp"]+data["prevalentStroke"])>0, 0, 1)
data.drop('prevalentHyp', axis=1, inplace=True)
data.drop('prevalentStroke', axis=1, inplace=True)
```

In [34]:

```
#create categorical variables and drop some variables
training=pd.get_dummies(data, columns=["currentSmoker","totChol","sysBP"])
training.drop('TenYearCHD', axis=1, inplace=True)
training.drop('male', axis=1, inplace=True)
training.drop('diaBP', axis=1, inplace=True)
final_train = training
final_train.head()
```

Out[34]:

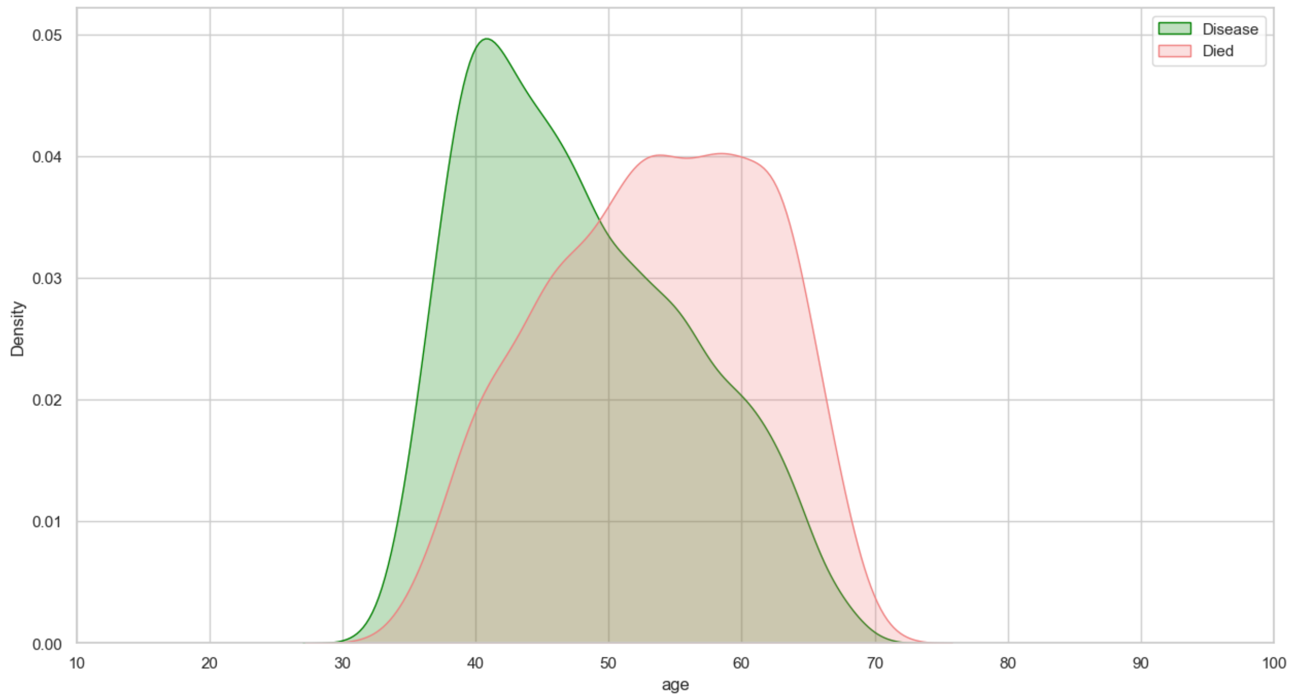
	age	education	cigsPerDay	BPMeds	diabetes	Disease	currentSmoker_0	currentSmoker_1	totChol_107.0	totChol_113.0	...	sysBP_215.0	sysBP_217.0
0	39	4.0	0.0	0.0	0	1	True	False	False	False	...	False	False
1	46	2.0	0.0	0.0	0	1	True	False	False	False	...	False	False
2	48	1.0	20.0	0.0	0	1	False	True	False	False	...	False	False
3	61	3.0	30.0	0.0	0	0	False	True	False	False	...	False	False
4	46	3.0	23.0	0.0	0	1	False	True	False	False	...	False	False

5 rows × 490 columns

Exploratory Data Analysis

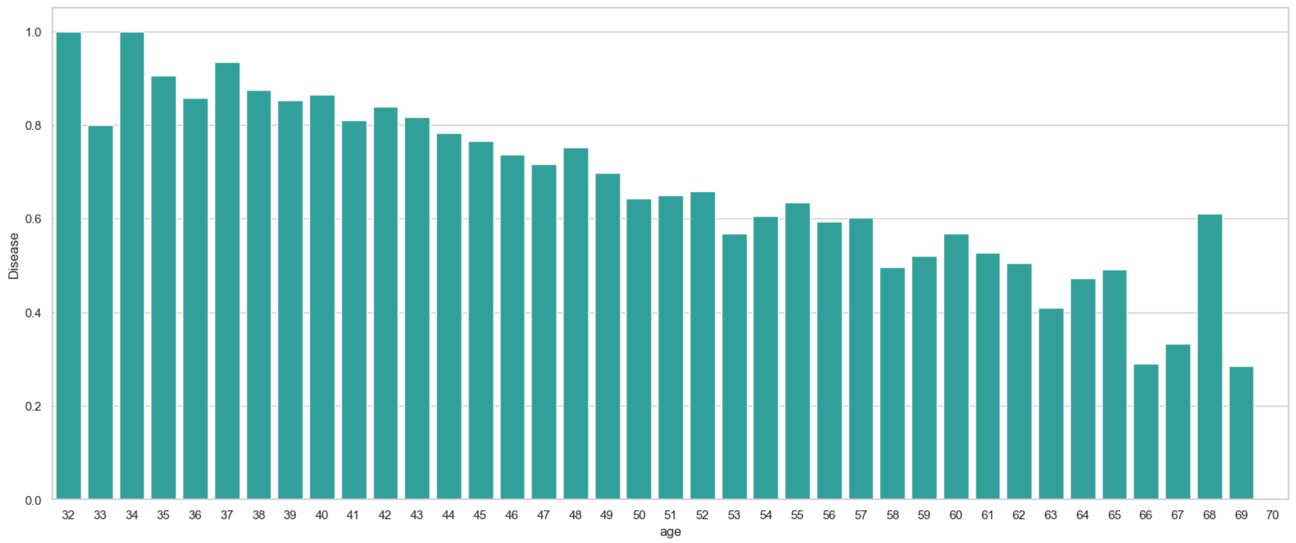
In [38]:

```
plt.figure(figsize=(15,8))
ax=sns.kdeplot(final_train["age"][final_train.Disease == 1], color="green", shade=True)
sns.kdeplot(final_train["age"][final_train.Disease == 0], color="lightcoral", shade=True)
plt.legend(['Disease', 'Died'])
ax.set(xlabel='age')
plt.xlim(10,100)
plt.show()
```



In [40]:

```
plt.figure(figsize=(20,8))
avg_survival_byage = final_train[["age", "Disease"]].groupby(['age'], as_index=False).mean()
g = sns.barplot(x='age', y='Disease', data=avg_survival_byage, color="LightSeaGreen")
plt.show()
```



In [42]:

```
final_train['IsMinor']=np.where(final_train['age']<=16, 1, 0)
print(final_train['IsMinor'])
```

```
0       0
1       0
2       0
3       0
4       0
       ..
4235    0
4236    0
4237    0
4238    0
4239    0
Name: IsMinor, Length: 4240, dtype: int32
```

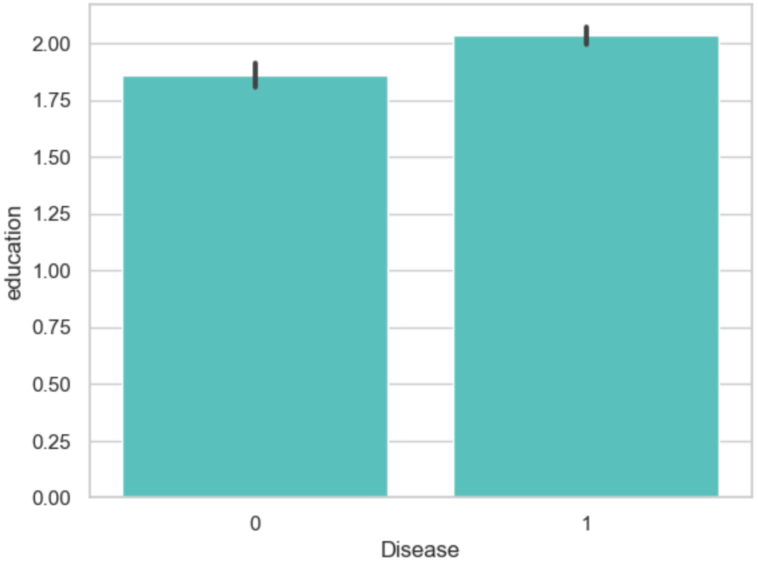
In [43]:

```
final_train['IsMinor']=np.where(final_train['age']<=16, 1, 0)
print(final_train['IsMinor'])
```

```
0       0
1       0
2       0
3       0
4       0
       ..
4235    0
4236    0
4237    0
4238    0
4239    0
Name: IsMinor, Length: 4240, dtype: int32
```

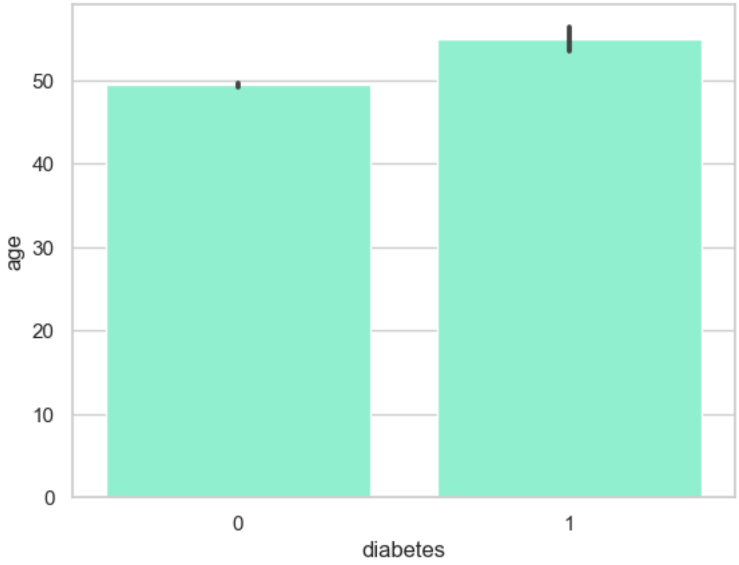
In [44]:

```
sns.barplot(x='Disease', y='education', data=final_train, color="mediumturquoise")
plt.show()
```



In [45]:

```
import seaborn as sns
import matplotlib.pyplot as plt
# Assuming 'train_df' is your DataFrame containing the data
sns.barplot(x='diabetes', y='age', data=df, color='aquamarine')
plt.show()
```



In [ ]: