36. Naive Bayes (Practical)

```
In [4]: import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        from mlxtend.plotting import plot_decision_regions
```

```
In [6]: dataset = pd.read_csv(r'Data/placement_3.csv')
        dataset.head(3)
```

Out[6]:

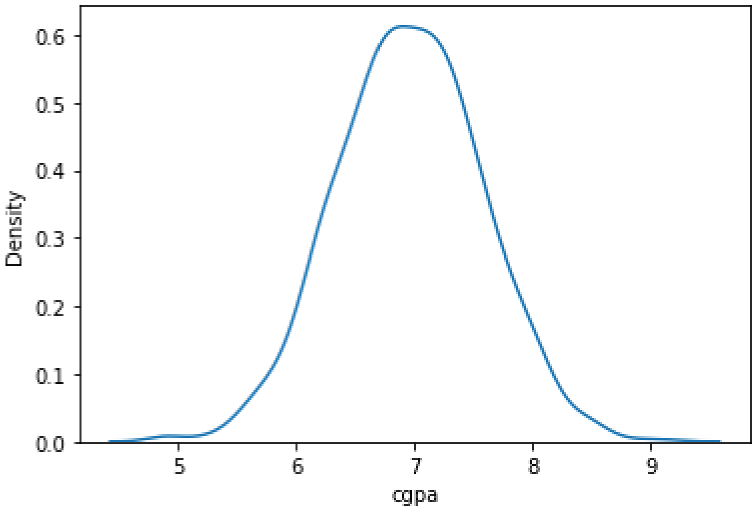
	cgpa	score	placed
0	7.19	26	1
1	7.46	38	1
2	7.54	40	1

```
In [7]: dataset.isnull().sum()
```

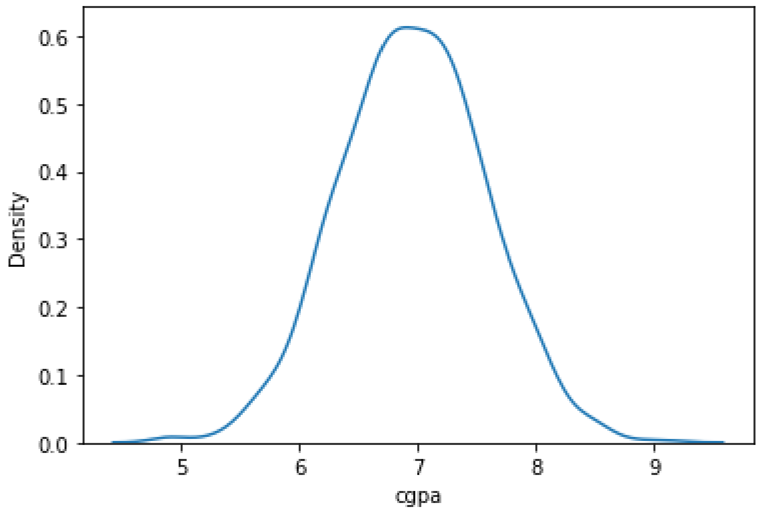
```
Out[7]: cgpa      0
        score     0
        placed    0
        dtype: int64
```

To check if the data is normally distributed or not, we will use disribution plot to check this

```
In [17]: sns.kdeplot(data=dataset["cgpa"])
         plt.show()
```



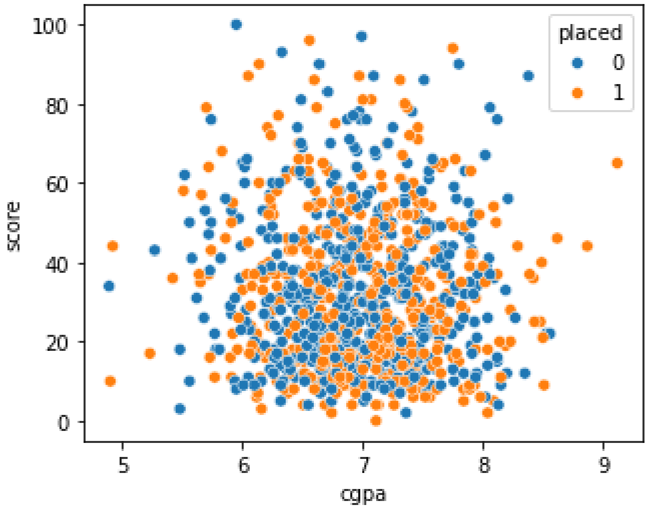
```
In [18]: sns.kdeplot(data=dataset["cgpa"])
         plt.show()
```



So will apply Gaussian Naive Bayes, b/c data is normally distributed.

In [10]:

```
plt.figure(figsize=(5,4))
sns.scatterplot(x="cgpa", y="score", data=dataset, hue="placed")
plt.show()
```



In [11]:

```
x = dataset.iloc[:,:-1]
x
```

Out[11]:

	cgpa	score
0	7.19	26
1	7.46	38
2	7.54	40
3	6.42	8
4	7.23	17
...	...	...
995	8.87	44
996	9.12	65
997	4.89	34
998	8.62	46
999	4.90	10

1000 rows × 2 columns

In [13]:
```
y = dataset["placed"]
y
```

Out[13]:
```
0      1
1      1
2      1
3      1
4      0
      ..
995    1
996    1
997    0
998    1
999    1
Name: placed, Length: 1000, dtype: int64
```

In [14]:
```
from sklearn.model_selection import train_test_split
```

In [15]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.20, random_st
```

In [ ]:
```

```

In [19]:
```
from sklearn.naive_bayes import GaussianNB, MultinomialNB, BernoulliNB
```

In [20]:
```
gnb = GaussianNB()
gnb.fit(x_train, y_train)
```

Out[20]:

```
▾ GaussianNB
GaussianNB()
```

In [23]:

```python
gnb.score(x_test, y_test)*100, gnb.score(x_train, y_train)*100
```

Out[23]:

```
(53.0, 53.5)
```

In [ ]:

```python

```

In [24]:

```python
mnb = MultinomialNB()
mnb.fit(x_train, y_train)
```

Out[24]:

```
▾ MultinomialNB
MultinomialNB()
```

In [25]:

```python
gnb.score(x_test, y_test)*100, gnb.score(x_train, y_train)*100
```

Out[25]:

```
(53.0, 53.5)
```

In [ ]:

```python

```

In [ ]:

```python
mnb = MultinomialNB()
mnb.fit(x_train, y_train)
```

In [ ]:

```python

```

In [26]:

```python
bnb = BernoulliNB()
bnb.fit(x_train, y_train)
```

Out[26]:

```
▾ BernoulliNB
BernoulliNB()
```

In [27]:

```python
gnb.score(x_test, y_test)*100, gnb.score(x_train, y_train)*100
```

Out[27]:

```
(53.0, 53.5)
```

In [ ]:

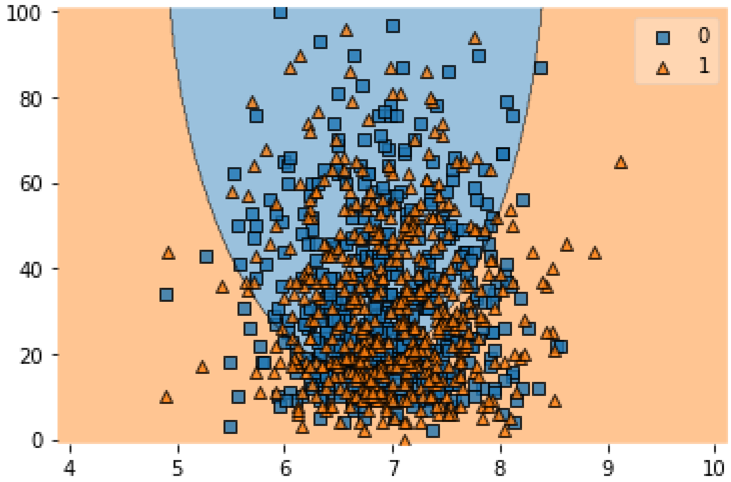
```python

```

In [28]:

```python
plot_decision_regions(x.to_numpy(), y.to_numpy(), clf=gnb)
plt.show()
```

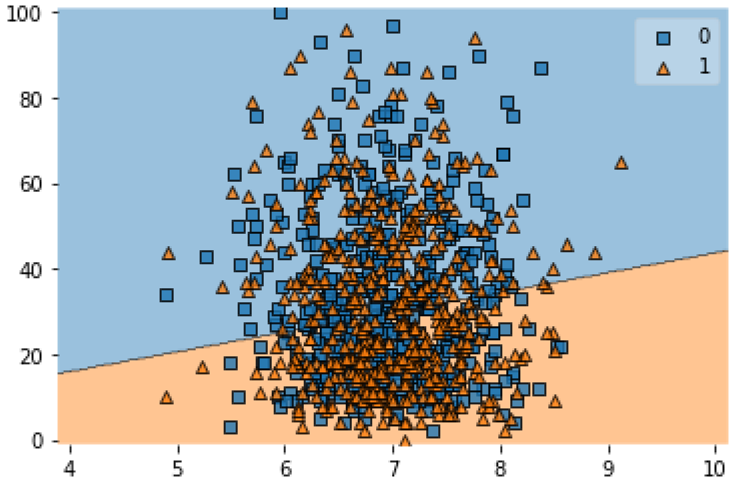
```
C:\Users\rashi\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\bas
e.py:450: UserWarning: X does not have valid feature names, but GaussianNB was fitte
d with feature names
  warnings.warn(
```



In [29]:

```
plot_decision_regions(x.to_numpy(), y.to_numpy(), clf=mnb)
plt.show()
```

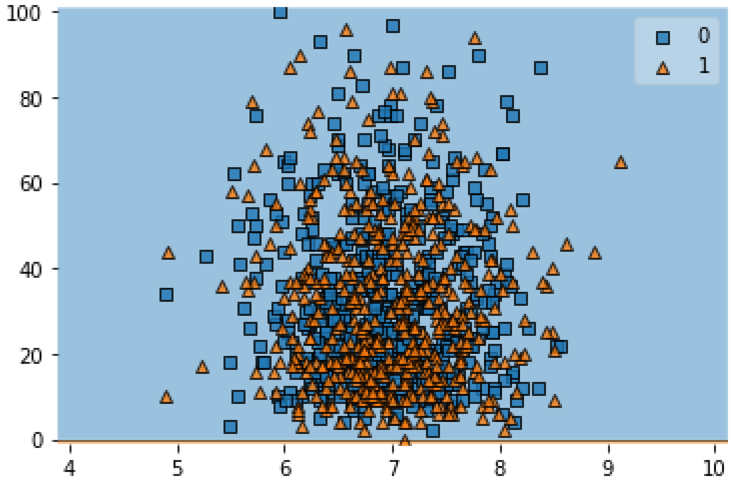
```
C:\Users\rashi\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\bas
e.py:450: UserWarning: X does not have valid feature names, but MultinomialNB was fi
tted with feature names
  warnings.warn(
```



In [30]:

```
plot_decision_regions(x.to_numpy(), y.to_numpy(), clf=bnb)
plt.show()
```

```
C:\Users\rashi\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\bas
e.py:450: UserWarning: X does not have valid feature names, but BernoulliNB was fitt
ed with feature names
  warnings.warn(
```



```
In [ ]:
```

```
In [31]: gnb.predict([[6.17, 5.17]])
```

```
C:\Users\rashi\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\bas
e.py:450: UserWarning: X does not have valid feature names, but GaussianNB was fitte
d with feature names
  warnings.warn(
```

```
Out[31]: array([1], dtype=int64)
```

```
In [ ]:
```