

18. Compare Bayes Classifier and KNN . . .

- Step 1: Study [Color Classification using Bayes Theorem](#)
- Step 2: Apply [Bayes Theorem](#) to re-calculate the sample data shown on the leftest table of the [web page](#) (assuming K=7)
- Step 3: Apply [Naive Bayes Classifier](#) to process the sample data shown on the leftest table of of the [web page](#) (assuming K=7)
- Step 4: Apply [KNN](#) to process the sample data shown on the leftest table of the [web page](#) (assuming K=7)
- Step 5: Compare the result of [Bayes Classifier](#) and [KNN](#)
- References
  - [Compare\\_KNN\\_NaivesBayes.ipynb](#) - Santhi Sree Nagalla, 2021 Summer . . .
  - [Color Classification using Naive Bayes](#)
  - [Naive Bayes Classifier Introductory Overview](#)
  - [KNN](#)
  - [Detailed explanation of Naive Bayes Classification](#)

**Solution:**

Using KNN, K = 7

X1 =Years of working experience	X2 =Number of kids	Y =Credit card is approved or not	Distance to each neighbor = (Target <sub>X1</sub> -Data <sub>X1</sub> ) <sup>2</sup> +(Target <sub>X2</sub> -Data <sub>X2</sub> ) <sup>2</sup> = (6-X1) <sup>2</sup> +(6-X2) <sup>2</sup>	K = 7
3	4	+	(6-3) <sup>2</sup> +(6-4) <sup>2</sup> = 9+4 = 13	+
2	5	+	(6-2) <sup>2</sup> +(6-5) <sup>2</sup> = 16+1 = 17	+
7	3	-	(6-7) <sup>2</sup> +(6-3) <sup>2</sup> = 1+9 = 10	-
5	3	-	(6-5) <sup>2</sup> +(6-3) <sup>2</sup> = 1+9 = 10	-
1	5	+	(6-1) <sup>2</sup> +(6-5) <sup>2</sup> = 25+1 = 26	
2	6	+	(6-2) <sup>2</sup> +(6-6) <sup>2</sup> = 16+0 = 16	+
3	2	-	(6-3) <sup>2</sup> +(6-2) <sup>2</sup> = 9+16 = 25	-
5	1	-	(6-5) <sup>2</sup> +(6-1) <sup>2</sup> = 1+25 = 26	
2	6	+	(6-2) <sup>2</sup> +(6-6) <sup>2</sup> = 16+0 = 16	+
6	6			+

For K = 7 , for some one with 6 years of work experience and 6 kids , credit card approval is positive (+)

**Applying Bayes classifier ( k = 7) :**

Calculate Prior Probability

Prior probability for ["+"]:

$$P (+) = \text{Number of ["+" sample} / \text{Total number of samples} = 5/9 = 0.56$$

Prior probability for ["-"]:

$$P (-) = \text{Number of ["-" sample} / \text{Total number of samples} = 4/9 = 0.44$$

### Calculate Conditional probability :

Likelihood of X given ["+" ]

Probability of X given ["+" ]

count (+, X) / count (+)  $\Rightarrow$  Number of ["+" ] in the vicinity  
(Nearest neighbors) of X

$$P(X | P(X | +)) = \frac{\text{Total number of ["+" ] case}}{\text{Total number of ["+" ] case}} = 4/5$$

Likelihood of X given ["-" ]

Probability of X given ["-" ]

count (-, X) / count (-)  $\Rightarrow$  Number of ["-" ] in the vicinity  
(Nearest neighbors) of X

$$P(X | P(X | -)) = \frac{\text{Total number of ["-" ] case}}{\text{Total number of ["-" ] case}} = 3/4$$

Final probability of X being "+" =  $P(+)*P(X | P(X | +)) = 0.56*0.8 = 0.448$

Final probability of X being "-" =  $P(-)*P(X | P(X | -)) = 0.44*0.75 = 0.33$

The probability of X being "+" is greater than the probability of X being "-", which agrees with the output result obtained using KNN. Hence, Both KNN and Bayes classifier achieved the same result.

Coding:

KNN

Csv file:

	A	B	C
1	X	Y	Z
2	3	4	1
3	2	5	1
4	7	3	0
5	5	3	0
6	1	5	1
7	2	6	1
8	3	2	0
9	5	1	0
10	2	6	1

Note:

The input are represented by x and y values , and the output by z. z = 1 is used for "+" and 0 for "-"



## Using KNN to Predict X

```
[9] import sklearn
    from sklearn.utils import shuffle
    from sklearn.neighbors import KNeighborsClassifier
    from sklearn import linear_model, preprocessing
    import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    from sklearn import datasets, neighbors
    from mlxtend.plotting import plot_decision_regions
    from sklearn.metrics import confusion_matrix, accuracy_score
```


```
[10] from google.colab import files
      uploaded = files.upload()

      Choose Files data.csv
      • data.csv(text/csv) - 140 bytes, last modified: 3/11/2023 - 100% done
      Saving data.csv to data.csv
```

```
[11] data = pd.read_csv("Knnbayes.csv")
      data
```

```
0a  

|   | X | Y | Z |
|---|---|---|---|
| 0 | 3 | 4 | 1 |
| 1 | 2 | 5 | 1 |
| 2 | 7 | 3 | 0 |
| 3 | 5 | 3 | 0 |
| 4 | 1 | 5 | 1 |
| 5 | 2 | 6 | 1 |
| 6 | 3 | 2 | 0 |
| 7 | 5 | 1 | 0 |
| 8 | 2 | 6 | 1 |


```

```
[12] X = data.iloc[:, [0, 1]].values
      print(X)
      y = data.iloc[:, -1].values
      print(y)

      [[3 4]
       [2 5]
       [7 3]
       [5 3]
       [1 5]
       [2 6]
       [3 2]
       [5 1]
       [2 6]]
      [1 1 0 0 1 1 0 0 1]
```


## Predict Output for value:

- k=7

```
[13] x_train, x_test, y_train, y_test = sklearn.model_selection.train_test_split(X, y, test_size=0.1)
      clf = neighbors.KNeighborsClassifier(n_neighbors=7)
      clf.fit(x_train, y_train)
      print("Predict Output for k=7 is", clf.predict([(6,6)]))
```


Predict Output for k=7 is [1]


## ▼ Using Naive Bayes Classification for same Input.

```
✓ 0s  from sklearn.preprocessing import StandardScaler  
sc = StandardScaler()  
X_train = sc.fit_transform(x_train)  
X_test = sc.transform(x_test)
```

```
✓ 0s [21] from sklearn.naive_bayes import GaussianNB  
classifier = GaussianNB()  
classifier.fit(X_train,y_train)
```

▼ GaussianNB  
GaussianNB()

```
✓ 0s  #Predict Output  
predicted= classifier.predict([[6,6]]) # 0:+, 1:-  
print("Predicted Value:", predicted)
```

 Predicted Value: [1]

```
✓ 0s [19] y_pred = classifier.predict(X_test)  
print(y_pred)  
print(y_test)  
cm = confusion_matrix(y_test, y_pred)  
print(cm)  
ac = accuracy_score(y_test,y_pred)  
print("Using Bayes the Accuracy is=",ac)
```

```
[0]  
[0]  
[[1]]  
Using Bayes the Accuracy is= 1.0
```

## Conclusion:

In conclusion, both KNN and Bayes classifiers are effective machine learning algorithms that can be used for classification tasks. KNN is a simple and intuitive algorithm that works well for datasets with few features and large sample sizes. Bayes classifier, on the other hand, is a probabilistic approach that works well for datasets with complex distributions and high-dimensional feature spaces. In general, KNN performs better than Bayes when the dataset is small and the features are simple, while Bayes performs better when the dataset is large and the features are complex. Ultimately, the choice between KNN and Bayes depends on the specific characteristics of the dataset and the classification problem at hand. In this case study, both classification methods obtained the same X result, which is “+”.