

# Trial 1

## Loading Modules and functions

```
In [ ]: from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
from sklearn.model_selection import train_test_split
import numpy as np
import pandas as pd
```

## Loading Data

```
In [ ]: df = pd.read_csv("../processed.csv")
df
```

Out[ ]:

	Initial_Price	Final_Price	Win_Flag	Mac_Flag	Linux_Flag	Positive_Reviews	Negative_Reviews	Memory_MB	Storage_MB	target
0	52.0	52.0	True	True	False	57.0	7.0	1024	50	1
1	0.0	0.0	True	True	False	53.0	6.0	2048	3072	1
2	0.0	0.0	True	False	False	133.0	69.0	2048	100	0
3	530.0	530.0	True	False	False	22.0	9.0	2048	500	0
4	229.0	229.0	True	True	True	226.0	44.0	2048	1500	1
...	...	...	...	...	...	...	...	...	...	...
57467	85.0	85.0	True	False	False	0.0	4.0	4096	200	-1
57468	349.0	349.0	True	True	False	2.0	1.0	1024	1024	1
57469	164.0	164.0	True	False	False	8.0	1.0	4096	20480	1
57470	610.0	610.0	True	False	False	1.0	0.0	4096	3072	1
57471	570.0	285.0	True	False	False	0.0	1.0	1024	2048	-1

57472 rows × 10 columns

## Splitting Data 33% test, 66% train

```
In [ ]: y = df["target"]
X = df.drop(labels=["target"],axis=1)

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.33, random_state=100,shuffle=True)
```

## Fitting model

```
In [ ]: bayes = GaussianNB()
bayes.fit(X_train, y_train)
```

Out[ ]:

GaussianNB ⓘ ?

GaussianNB()

## Predicting Likelihood of test cases

```
In [ ]: bayes.predict_proba(X_test)
```

```
Out[ ]: array([[3.77478260e-01, 6.22521738e-01, 1.64695980e-09],
               [3.77478260e-01, 6.22521738e-01, 1.64695980e-09],
               [3.77478260e-01, 6.22521738e-01, 1.64695980e-09],
               ...,
               [3.77478260e-01, 6.22521738e-01, 1.64695980e-09],
               [3.77478260e-01, 6.22521738e-01, 1.64695980e-09],
               [3.77478260e-01, 6.22521738e-01, 1.64695980e-09]])
```

## Model evaluation

```
In [ ]: preds = bayes.predict(X_test)
print(f"Accuracy : {accuracy_score(y_test, preds)}")
```

Accuracy : 0.18786249077296213

## Important metrics

```
In [ ]: print(classification_report(y_test, preds))
```

	precision	recall	f1-score	support
-1	0.00	0.00	0.00	2123
0	0.19	1.00	0.32	3563
1	0.00	0.00	0.00	13280
accuracy			0.19	18966
macro avg	0.06	0.33	0.11	18966
weighted avg	0.04	0.19	0.06	18966

c:\Users\user\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\metrics\\_classification.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero\_division` parameter to control this behavior.  
\_warn\_prf(average, modifier, f"{metric.capitalize()} is", len(result))  
c:\Users\user\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\metrics\\_classification.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero\_division` parameter to control this behavior.  
\_warn\_prf(average, modifier, f"{metric.capitalize()} is", len(result))  
c:\Users\user\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\metrics\\_classification.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero\_division` parameter to control this behavior.  
\_warn\_prf(average, modifier, f"{metric.capitalize()} is", len(result))

## Confusion Matrix Display

```
In [ ]: confusion_matrix(y_test,preds)
```

Out[ ]: array([[ 0, 2123, 0],  
[ 0, 3563, 0],  
[ 0, 13280, 0]], dtype=int64)

## 1st trial Conclusion

- 1. Bayes Classifier gave *worst* performance of 18% accuracy
- 2. Its confusion matrix suggests classifier does no prediction of +1 and -1 class labels