Problem Statement: - Social Network Ad

How to develop an AI solution to personalize advertisements for users based on historical data, and predict whether a user is likely to purchase the advertised product. The AI system should display advertisements only to users with a high probability of making a purchase, in order to target the right customers effectively.

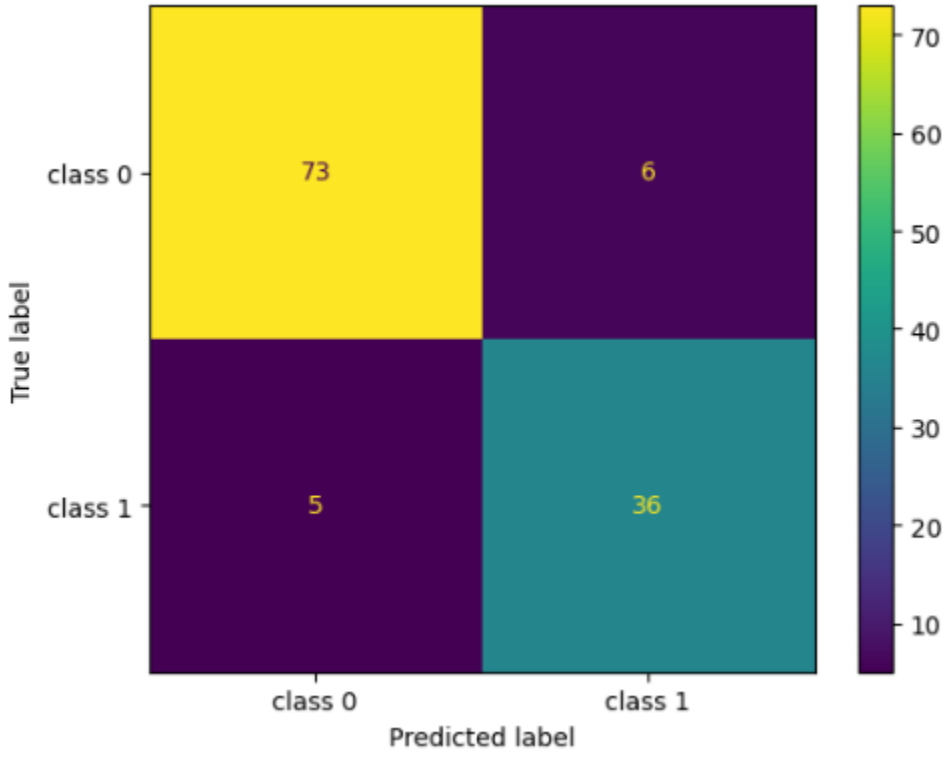
Class 0 – User purchased

Class 1 – User did not purchase

Support – Actual No. of. Samples in Each class

Random Forest Classifier

Code link: https://github.com/krthiksha/Machine-Learning-Classification_module/blob/main/1.RandomForest_classification.ipynb



	precision	recall	f1-score	support
0	0.94	0.92	0.93	79
1	0.86	0.88	0.87	41
accuracy			0.91	120
macro avg	0.90	0.90	0.90	120
weighted avg	0.91	0.91	0.91	120

Classification report for random forest classifier

1) What is the overall performance of the model?
   Accuracy : 0.91
2) What is the percentage of correctly classified class 0?
   Recall of class 0 : 0.92
3) What is the percentage of correctly classified class 1?
   Recall of class 1 : 0.88
4) What is the percentage of correctly and wrongly classified class 0?
   Precision of class 0 : 0.94
5) What is the percentage of correctly and wrongly classified class 1?
   Precision of class 1 : 0.86
6) Measure the balance between precision and recall for class 0?
   F1 score of class 0 : 0.93
7) Measure the balance between precision and recall for class 1?
   F1 score of class 1 : 0.87
8) What is the macro average of precision?
   macro average of precision : 0.90
9) What is the macro average of recall?
   macro average of recall : 0.90
10) What is the macro average of f1 score?
    macro average of f1 score : 0.90
11) What is the weighted average of precision?
    weighted average of precision : 0.91
12) What is the weighted average of recall?
    weighted average of recall : 0.91
13) What is the weighted average of f1 score?
    weighted average of f1 score : 0.91

Algorithm : RandomForestClassifier

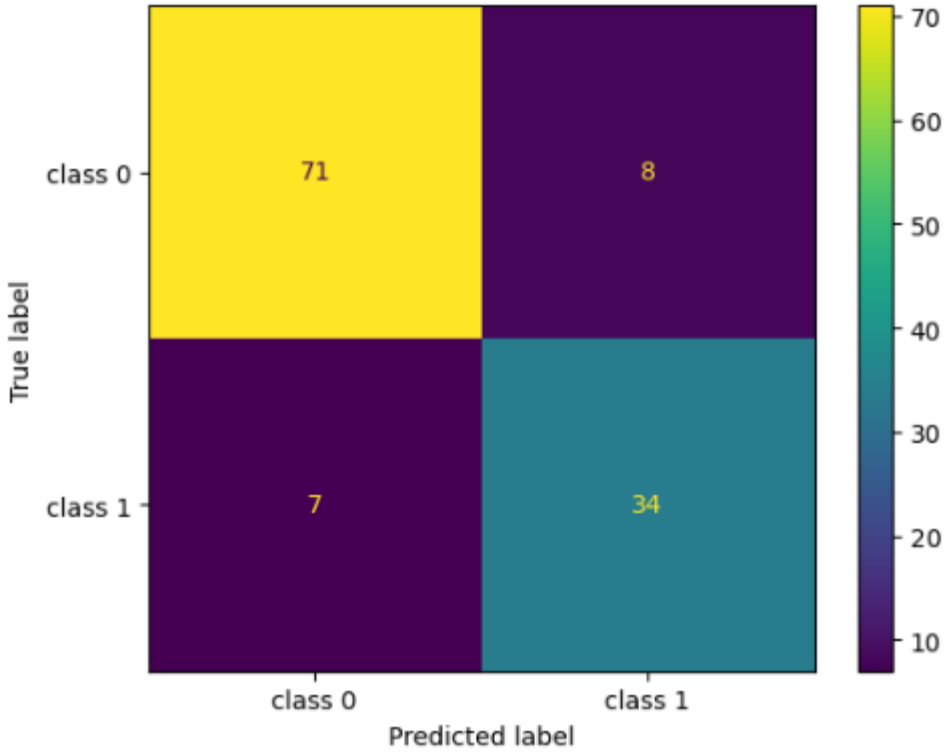
Accuracy (overall performance of the model) = 0.91

overall metrics performance (precision,recall,f1-score) = good

Result : **Good Model**

Decision Tree Classifier

Code link: https://github.com/krthiksha/Machine-Learning-Classification_module/blob/main/2.DecisionTree_classification.ipynb



	precision	recall	f1-score	support
0	0.91	0.90	0.90	79
1	0.81	0.83	0.82	41
accuracy			0.88	120
macro avg	0.86	0.86	0.86	120
weighted avg	0.88	0.88	0.88	120

Classification report for random forest classifier

1) What is the percentage of correct classification of both the classes?

   Accuracy : 0.88

2) How many actual positives did I find for class 0?

   Recall of class 0 : 0.90

3) How many actual positives did I find class 1?

   Recall of class 1 : 0.85

4) How correct my positive predictions for class 0?

   Precision of class 0 : 0.92

5) How correct my positive predictions for class 1?
   Precision of class 1 : 0.81
6) What is the overall performance of class 0?
   F1 score of class 0 : 0.93
7) What is the overall performance of class 1?
   F1 score of class 1 : 0.87
8) What is the macro precision?
   macro average of precision : 0.87
9) What is the macro recall?
   macro average of recall : 0.88
10) What is the macro f1 measure?
    macro average of f1 score : 0.87
11) What is the weighted precision?
    weighted average of precision : 0.89
12) What is the weighted recall?
    weighted average of recall : 0.88
13) What is the weighted f1 score?
    weighted average of f1 score : 0.88

Algorithm : DecisionTreeClassifier

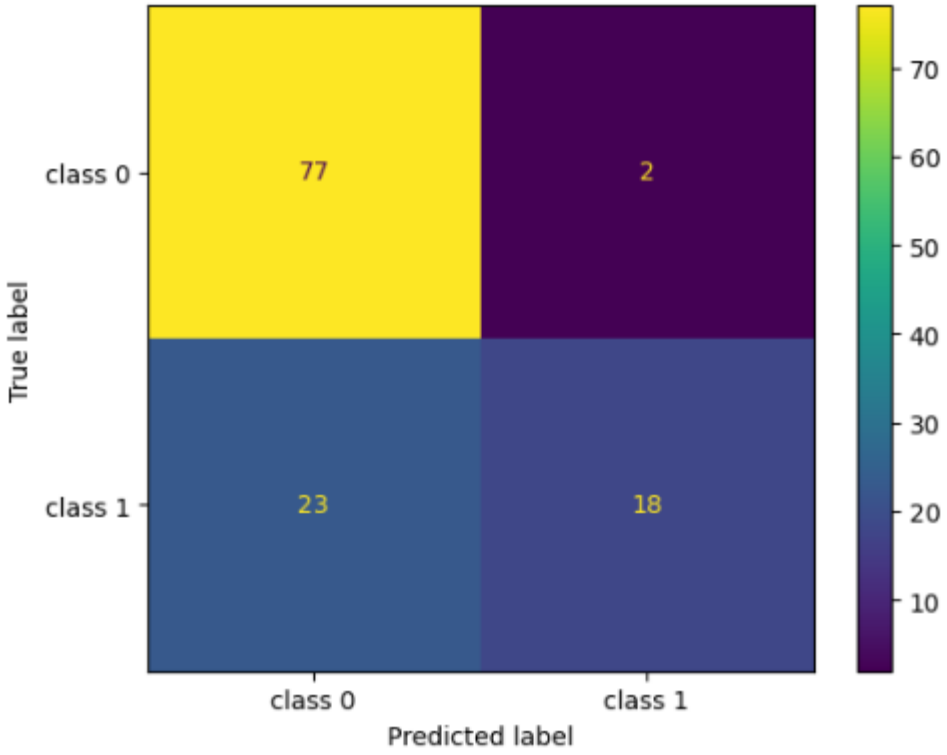
Accuracy (overall performance of the model) = 0.88

overall metrics performance (precision,recall,f1-score) = good

Result **: Good Model but not better than Randomforestclassifier**

SVC (support vector classifier)

Code: https://github.com/krthiksha/Machine-Learning-Classification_module/blob/main/3.SVM_classification.ipynb



	precision	recall	f1-score	support
0	0.77	0.97	0.86	79
1	0.90	0.44	0.59	41
accuracy			0.79	120
macro avg	0.83	0.71	0.73	120
weighted avg	0.81	0.79	0.77	120

Classification report for random forest classifier

1) Overall how many predictions where correct?

   Accuracy : 0.79

2) Of all actual users for class 0 (Not purchases), How many did the model correctly Identified?

   Recall of class 0 : 0.97

3) Of all actual users for class 1 (purchased), how many did model correctly identified?

   Recall of class 1 : 0.44

4) Of all actual users for class 0 (not purchased), how many were actually correct?

   Precision of class 0 : 0.77

5) Of all actual users for class 1 (purchased), how many were actually correct?

Precision of class 1 : 0.90

6) What is F1 measure of class 0?
   F1 score of class 0 : 0.86
7) What is F1 measure of class 1?
   F1 score of class 1 : 0.59
8) What is the average performance of precision for the model?
   macro average of precision : 0.83
9) What is the average performance of recall for the model?
   macro average of recall : 0.71
10) What is the average performance of f1 score for the model?
    macro average of f1 score : 0.73
11) What is the sum of product of proportion rate of each class in precision?
    weighted average of precision : 0.81
12) What is the sum of product of proportion rate of each class in recall?
    weighted average of recall : 0.79
13) What is the sum of product of proportion rate of each class in f1 score?
    weighted average of f1 score : 0.77

Algorithm : SVC

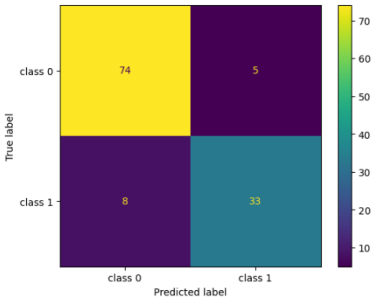
Accuracy (overall performance of the model) = 0.79

overall metrics performance (precision,recall,f1-score) = poor

Result : **poor model**

LOGISTIC REGRESSION (CLASSIFICATION ALGORITHM)

Code: https://github.com/krthiksha/Machine-Learning-Classification_module/blob/main/4.Logistic_Regression_classification.ipynb



	precision	recall	f1-score	support
0	0.90	0.94	0.92	79
1	0.87	0.80	0.84	41
accuracy			0.89	120
macro avg	0.89	0.87	0.88	120
weighted avg	0.89	0.89	0.89	120

Algorithm : Logistic Regression

Accuracy (overall performance of the model) = 0.89

overall metrics performance (precision,recall,f1-score) = good

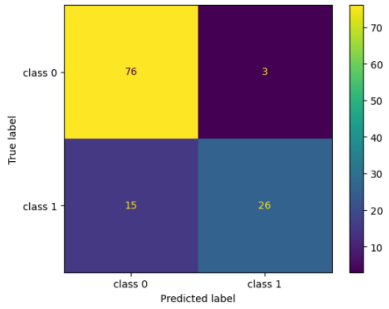
Result : **good model but not better than Randomforestclassifier**

PROBLEM STATEMENT:- Social network Ad (User Purchase Prediciton)
Algorithm: Logistic regression

SL.NO	solver	max_iter	C	Confusion matrix	model accuracy	Remark
1	lbfgs	100	1.0	[[74 5] [8 33]]	0.89 (ConvergenceWarning)	Warning
2	lbfgs	120	1.0	[[74 5] [8 33]]	0.89	Good Model
3	lbfgs	200	1.0	[[74 5] [8 33]]	0.89	Good Model
4	liblinear	120	1.0	[[79 0] [41 0]]	0.66	poor
5	newton-cg	120	1.0	[[74 5] [8 33]]	0.89	Good Model
6	newton-cholesky	120	1.0	[[74 5] [8 33]]	0.89	Good Model
7	sag	120	1.0	[[79 0] [41 0]]	0.66	poor
8	saga	120	1.0	[[79 0] [41 0]]	0.66	poor
9	lbfgs	120	0.01	[[74 5] [11 30]]	0.87	Good Model

KNN classifier:

Code: https://github.com/krthiksha/Machine-Learning-Classification_module/blob/main/4.KNN_classification.ipynb



	precision	recall	f1-score	support
0	0.84	0.96	0.89	79
1	0.90	0.63	0.74	41
accuracy			0.85	120
macro avg	0.87	0.80	0.82	120
weighted avg	0.86	0.85	0.84	120

Algorithm : K nearest neighbors classifier

Accuracy (overall performance of the model) = 0.85

overall metrics performance (precision,recall,f1-score) = good

Result**: good model but not better than Randomforestclassifier for the problem statement (social network ad)**

PROBLEM STATEMENT:- Social network Ad (User Purchase Prediciton) Algorithm: K nearest neighbors								
SL.NO	**n_neighbors**	**metric**	**P (power parameter)**	**algorithm**	**weights**	**Confusion matrix**	**model accuracy**	**Remark**
1	5	minkowski	2	auto	uniform	[[69 10] [11 30]]	0.82	poor
2	7	minkowski	2	auto	uniform	[[72 7] [13 28]]	0.83	poor
3	8	minkowski	2	auto	uniform	[[76 3] [16 25]]	0.84	poor
4	11	minkowski	2	auto	uniform	[[74 5] [13 28]]	0.85	Good Model
5	21	minkowski	2	auto	uniform	[[76 3] [15 26]]	0.85	Good Model
6	21	minkowski	2	auto	distance	[[58 21] [10 31]]	0.74	poor
7	21	minkowski	2	ball_tree	uniform	[[76 3] [15 26]]	0.85	Good Model
8	21	minkowski	2	kd_tree	uniform	[[76 3] [15 26]]	0.85	Good Model
9	21	minkowski	2	brute	uniform	[[76 3] [15 26]]	0.85	Good Model

Naïve Bayes

Code: https://github.com/krthiksha/Machine-Learning-Classification_module/blob/main/4.NB_classification.ipynb

PROBLEM STATEMENT:- Social network Ad (User Purchase Prediciton) Algorithm: Naïve Bayes				
SL.NO	**NB types**	**Confusion matrix**	**model accuracy**	**Remark**
1	**GaussianNB**	[[74 5] [8 33]]	0.89	Good Model
2	MultinomialNB	[[68 11] [28 13]]	0.68	poor
3	ComplementNB	[[42 37] [20 21]]	0.53	poor
4	BernoulliNB	[[79 0] [41 0]]	0.66	poor
5	**CategoricalNB**	[[76 3] [9 32]]	0.90	Good Model

Algorithm : Naïve bayes - GaussianNB

Accuracy (overall performance of the model) = 0.89

overall metrics performance (precision,recall,f1-score) = good

Result**: good model but the Randomforestclassifier model performance (0.91) for the problem statement (social network ad)**

Best Model for social network ad problem is *Random forest classifier* with accuracy => 0.91

🔷 Random Forest

```lua
[[73,  6],
 [ 5, 36]]
```

- FP = 6 → predicted buy, actually no
- FN = 5 → missed buyers ❌
- TP = 36 → correctly caught buyers

🔷 Categorical Naive Bayes

```lua
[[76,  3],
 [ 9, 32]]
```

- FP = 3 → fewer false ads ✅
- FN = 9 → **missed more buyers** ❌
- TP = 32 → fewer buyers caught

S

Evaluation Metrics using Confusion Matrix

Accuracy

$$\text{Accuracy} = \frac{\text{Correct predictions}}{\text{All predictions}} \qquad \text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Accuracy Scenarios:

- Overall performance of the model?
- What is the Percentage of correct classification of both the classes?
- Overall how many predictions were correct?

Calculation of Random Forest classifier:

Accuracy = (73+36) / (73+36+5+6)

=109/120

=0.90833 ~ 0.91

Recall

$$\text{Recall (or Sensitivity)} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

Recall Scenarios:

- Percentage of correctly classified of a specific class?
- How many actual positives did I find?
- Of all actual users for a specific class, how many did the model correctly identified?

Calculation of Random Forest classifier:

Recall (class 0) = 73 / (73+6)

= 73/79 = 0.92

Precision

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Precision Scenarios:

- Percentage of correctly and wrongly classified of a specific class?
- How correct my positive predictions for a specific class?
- Of all users for a specific class, how many were actually correct?

Calculation of Random Forest classifier:

Precision (class 0) = 73 / (73+5) = 73/78

= 0.9358 ~ 0.94

F1 score / F1 measure

$$F1\ score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} = 2 \cdot \frac{Precision * Recall}{Precision + Recall}$$

$$\Rightarrow F1\ score = 2 \cdot \frac{Precision * Recall}{Precision + Recall}$$

Scenarios:

- Balance between precision and recall for classes?
- Overall performance of a specific class?
- What is F1 measure of a specific class?

Calculation of Random Forest classifier:

F1 score (class 0) = 2 ((0.94 * 0.92) / (0.94 + 0.92))

= 2(0.8648/1.86)

= 2(0.464)

=0.9298 ~ 0.93

Macro average

Macro precision = (precision1 + precision2) / 2

Macro recall = (recall1 + recall2) / 2

Macro f1score = (f1score 1 + f1score 2) / 2

Calculation of Random Forest classifier:

Macro precision = (0.94 + 0.86) / 2 = 1.8/2 =0.90

Scenarios:

Metrics – (precision, recall, f1 score)

- Macro average of metrics
- Macro metrics
- Average performance for the metrics for the model

Weighted average

Weighted average = Sum (metrics * proportion rate)

Scenarios:

Metrics – (precision, recall, f1 score)

- Weighted average of metrics
- Weighted metrics
- Sum of product of proportional rate of each classes in metrics

Calculation of Random Forest classifier:

Weighted precision = (0.94 * (79/120)) + (0.86 * (41/120))

= (0.94 * (0.658)) + (0.86 * (0.34))

= 0.618 + 0.29

= 0.9085 ~ 0.91

Grid Search CV classification – algorithm comparison report

Random forest classifier

Grid search cv – parameter grid:

```
param_grid = {
"criterion": ["gini", "entropy", "log_loss"],
"max_features": ["sqrt", "log2", None],
"class_weight": ["balanced", "balanced_subsample"]
}
model = RandomForestClassifier()
```

Model Accuracy: 0.93

Confusion matrix:

```
[[54  4]
 [ 2 20]]
```

Classification report:

	precision	recall	f1-score	support
0	0.96	0.93	0.95	58
1	0.83	0.91	0.87	22
accuracy			0.93	80
macro avg	0.90	0.92	0.91	80
weighted avg	0.93	0.93	0.93	80

Result:

In the random forest classifier, the accuracy of the model is 0.93. And also in the confusion metric, the false negative is less, which is 2. And the recall of the model for purchased is 91, with precision of 83. It's overall performance is also good, with accuracy of 93 and also false negative is less, so that leads to the minimized missing of the buyers. So it is actually a very good model.

Support Vector classifier SVC

Grid search cv – parameter grid:

```
param_grid = {
"kernel": ["linear", "poly", "rbf", "sigmoid"],  #precomputed - need x to be a square matrix
"gamma": ["scale", "auto"]
}
model = SVC(probability=True)
```

Model Accuracy: 0.95

Confusion matrix:

```
[[55  3]
 [ 1 21]]
```

Classification report:

	precision	recall	f1-score	support
0	0.98	0.95	0.96	58
1	0.88	0.95	0.91	22
accuracy			0.95	80
macro avg	0.93	0.95	0.94	80
weighted avg	0.95	0.95	0.95	80

Result:

Support vector classifier for the same problem, accuracy of 95% and the confusion matrix false negative is actually 1 and the recall for both the classes is 95% with the precision of 83 for class 1 and 98 for class 0 and the overall performance is really very good and comparing to the random forest, support vector classifier is really **performing much better than random forest** so it is a very good model.

Decision tree classifier

Grid search cv – parameter grid:

```
param_grid = {
"criterion": ["gini", "entropy", "log_loss"],
"splitter" : ["best", "random"],
"max_features" : ["sqrt", "log2"],
}
model = DecisionTreeClassifier()
```

Model Accuracy: 0.91

Confusion matrix:

```
[[55  3]
 [ 4 18]]
```

Classification report:

	precision	recall	f1-score	support
0	0.93	0.95	0.94	58
1	0.86	0.82	0.84	22
accuracy			0.91	80
macro avg	0.89	0.88	0.89	80
weighted avg	0.91	0.91	0.91	80

Result:

Decision tree classifier for the same problem, accuracy of the model is 91 percent, but the confusion matrix false negative is 4, which is greater number comparatively with support vector classifier and the random forest. The recall for the purchase is 82 and precision is 86, so that is also less. So I conclude that **Decision Tree classifier model is performing good** but it is comparatively less performance when comparing with the Random Forest and Support Vector classifier.

Logistic regression

Grid search cv – parameter grid:

```
param_grid = [
  # L2 penalties
  {
    "solver": ["lbfgs", "newton-cg", "newton-cholesky", "sag"],
    "penalty": ["l2", None]
  },
  # L1 penalties
  {
    "solver": ["liblinear", "saga"],
    "penalty": ["l1"]
  },
  # ElasticNet
```

```
    {
        "solver": ["saga"],
        "penalty": ["elasticnet"],
        "l1_ratio": [0.1, 0.5, 0.9]
    }
]
model = LogisticRegression()
```

Model Accuracy: 0.93

Confusion matrix:

```
[[56  2]
 [ 4 18]]
```

Classification report:

	precision	recall	f1-score	support
0	0.93	0.97	0.95	58
1	0.90	0.82	0.86	22
accuracy			0.93	80
macro avg	0.92	0.89	0.90	80
weighted avg	0.92	0.93	0.92	80

Result:

Logistic regression model accuracy 93%, but the confusion matrix false negative is 4, and recall for class 1 is 82 (purchased). So it is comparatively **good model**, but when comparing with the support vector classifier, which is with the highest accuracy and a good performing model, this logistic regression is performing less.

KNN

Grid search cv – parameter grid:

```
param_grid ={
"weights":["uniform", "distance"],
"algorithm":["auto", "ball_tree", "kd_tree", "brute"]
}
model = KNeighborsClassifier()
```

Model Accuracy: 0.95

Confusion matrix:

```
[[55  3]
 [ 1 21]]
```

Classification report:

	precision	recall	f1-score	support
0	0.98	0.95	0.96	58
1	0.88	0.95	0.91	22
accuracy			0.95	80
macro avg	0.93	0.95	0.94	80
weighted avg	0.95	0.95	0.95	80

Result:

k-neighbor classifier algorithm, accuracy with a 95% and also confusion matrix for false negative is 1, recall is 95, precision is 88 for the class 1, and overall performing also it is good, and also all the metrics, accuracy scores are similar to the support vector classifier, which is one of the highest performing model as of now, as per the analysis. So I conclude that k-nearest neighbor is also performing same as the support vector classifier, so both the model is the highest performing model for the social network ad as of now, as per the analysis.

Naïve bayes

Grid search cv – parameter grid:

```
param_grid ={
"alpha" : [0.1, 0.5, 1.0],
"binarize" : [0.0, 0.5, 1.0]
}
model = BernoulliNB()
```

Model Accuracy: 0.90

Confusion matrix:

```
[[51  7]
 [ 1 21]]
```

Classification report:

clf_report:	precision	recall	f1-score	support
0	0.98	0.88	0.93	58
1	0.75	0.95	0.84	22
accuracy			0.90	80
macro avg	0.87	0.92	0.88	80
weighted avg	0.92	0.90	0.90	80

Result:

finalizing bernoulliNB – As minimizing missed purchases (false negatives) is more important

Naive bayes, Bernoulli NB, Model Accuracy 90%, Confusion Matrix False Negative is 1, but the Recall for Class 1 is 95, Precision for Class 1 is 75, so overall it is **good**, but **comparing with the highest performing models KNN and the Support Vector Classifier,** this model is performing less.

FINALIZED MODEL FOR SOCIAL NETWORK AD – problem statement

Algorithm : KNN classifier and Support vector classifier (SVC)

Model Accuracy: 0.95

Confusion matrix:

```
[[55  3]
 [ 1 21]]
```

Classification report:

	precision	recall	f1-score	support
0	0.98	0.95	0.96	58
1	0.88	0.95	0.91	22
accuracy			0.95	80
macro avg	0.93	0.95	0.94	80
weighted avg	0.95	0.95	0.95	80

Analysis Report:

So, I finalize the model for **Social Network Ad Problem Statement in order to find user will purchase or not purchase the product.** K-nearest neighbor classifier algorithm and support vector classifier algorithm with accuracy of 95%. Both are performing well overall with minimized false negatives. So, the number of missing the buyers will be less. So, I conclude this both algorithm is performing well with accuracy 95% and I finalize this both at my final model.