Results:

The report contains outputs, interpretations and comparisons on the following techniques implemented:

1) SVM – Linear Kernel
2) SVM – Gaussian Kernel
3) SVM – Sigmoid Kernel
4) Decision tree (Gini Index, Entropy)
5) Decision tree pruning
6) Decision tree boosting

SVM: Liner Kernel

```
Confusion Matrix using Linear Kernel SVM :
[[4087  276]
[ 950  608]]
--------------------------------------------------------------------
Report using Linear Kernel SVM :

              precision    recall  f1-score   support
    0.0         0.81      0.94      0.87      4363
    1.0         0.69      0.39      0.50      1558
micro avg       0.79      0.79      0.79      5921
macro avg       0.75      0.66      0.68      5921
weighted avg    0.78      0.79      0.77      5921
--------------------------------------------------------------------
 Accuracy using Linear Kernel SVM is :-  79.29403816922817
```

Comment: The accuracy for this is 79% and the false positives are really high as we can see this in the above table. Precision is very good but recall for class 1 is too low as well. But in the end we may find that this is the best alternative.

SVM: Gaussian Kernel

```
Confusion Matrix using Gaussian Kernel SVM :
[[4261  102]
[1160  398]]
--------------------------------------------------------------------
Report using Gaussian Kernel SVM :

              precision    recall  f1-score   support
      0.0         0.79      0.98      0.87      4363
      1.0         0.80      0.26      0.39      1558
micro avg         0.79      0.79      0.79      5921
macro avg         0.79      0.62      0.63      5921
weighted avg      0.79      0.79      0.74      5921
--------------------------------------------------------------------
Accuracy using Gaussian Kernel SVM is :-  78.68603276473569
```

Comment: Looking at the Gaussian Kernel, it seems that it is not doing as good a job as a Linear kernel. Recall is even lower as well as accuracy

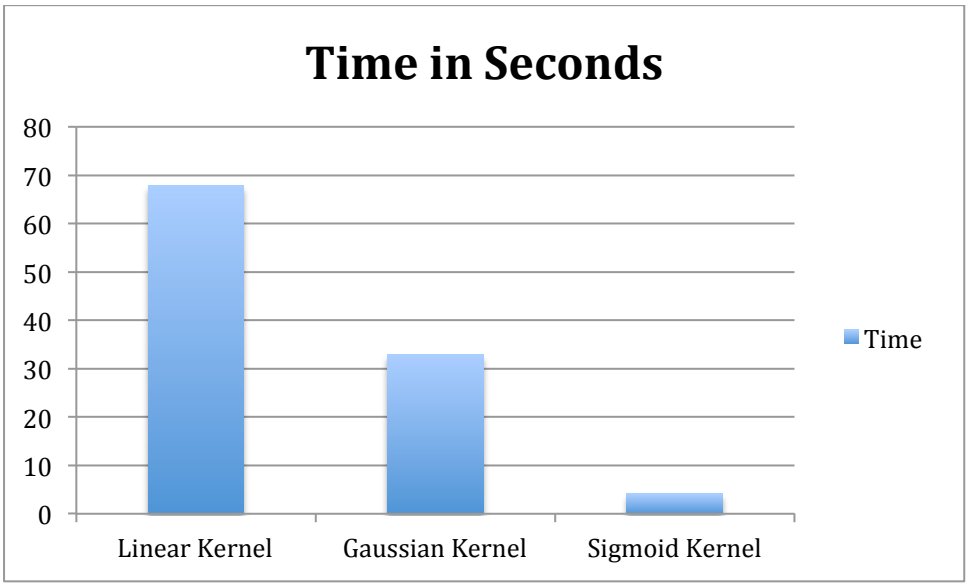
SVM: Sigmoid Kernel

```
Confusion Matrix using Sigmoid Kernel SVM :
[[4363    0]
[1558    0]]

------------------------------------------------------------------
Report using Sigmoid SVM :
               precision    recall  f1-score   support
      0.0       0.74      1.00      0.85      4363
      1.0       0.00      0.00      0.00      1558
micro avg       0.74      0.74      0.74      5921
macro avg       0.37      0.50      0.42      5921
weighted avg       0.54      0.74      0.63      5921
------------------------------------------------------------------
Accuracy using Sigmoid Kernel SVM is :-  73.68687721668637
```

Comment: We can see that sigmoid kernel is not able to classify negative observations and here this can not be used. The accuracy is also the lowest

Final Choice – Linear Kernel works best on the data including accuracy and best classification. However, Gaussian Kernel takes the least amount of time.



DECISION TREE:

Gini Index:

```
----------------------------------------------------------------
Predicted values:
[0. 0. 0. ... 1. 0. 0.]

Confusion Matrix:
[[4273   90]
[1338  220]]
----------------------------------------------------------------
----------------------------------------------------------------
Accuracy :
75.88245228846479
----------------------------------------------------------------
----------------------------------------------------------------
Report:
              precision    recall  f1-score   support

         0.0       0.76      0.98      0.86      4363
         1.0       0.71      0.14      0.24      1558
micro avg          0.76      0.76      0.76      5921
macro avg          0.74      0.56      0.55      5921
weighted avg       0.75      0.76      0.69      5921
----------------------------------------------------------------
```

Comment: The accuracy is 76% as per Gini

Entropy:

```
----------------------------------------------------------------
Predicted values:
[0. 0. 0. ... 1. 0. 0.]

Confusion Matrix:
[[3972  391]
[1143  415]]
----------------------------------------------------------------
Accuracy :
74.0922141530147
----------------------------------------------------------------
Report :
              precision    recall  f1-score   support

         0.0       0.78      0.91      0.84      4363
         1.0       0.51      0.27      0.35      1558
micro avg          0.74      0.74      0.74      5921
macro avg          0.65      0.59      0.59      5921
weighted avg       0.71      0.74      0.71      5921
----------------------------------------------------------------
```

Comment – This gives the slightly lower accuracy compared to Gini.

DECISION TREE: BOOSTING

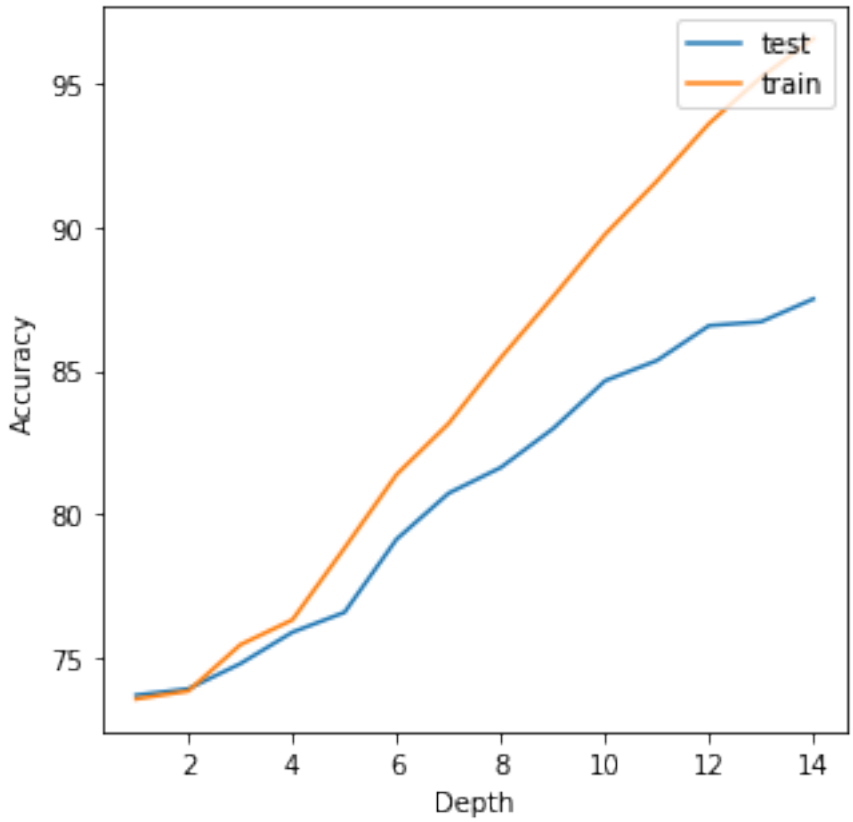
```
------------------------------------------------------------------
Confusion Matrix:
[[4106  257]
[ 377 1181]]
------------------------------------------------------------------

------------------------------------------------------------------
Report :
              precision    recall  f1-score   support
0.0            0.92      0.94      0.93      4363
1.0            0.82      0.76      0.79      1558
micro avg      0.89      0.89      0.89      5921
macro avg      0.87      0.85      0.86      5921
weighted avg   0.89      0.89      0.89      5921
------------------------------------------------------------------

Accuracy is   89.29234926532679
------------------------------------------------------------------
```

Comment: Boosting gives the best output as we can see that the false positives and false negatives are very less.

Depth to Accuracy (Train vs Test):



Comment – I find the depth of 6 to be the best because after that the test and train lines start to divulge and may result in overfitting by a huge margin.

TASK B:

Name of the dataset – Bankloan

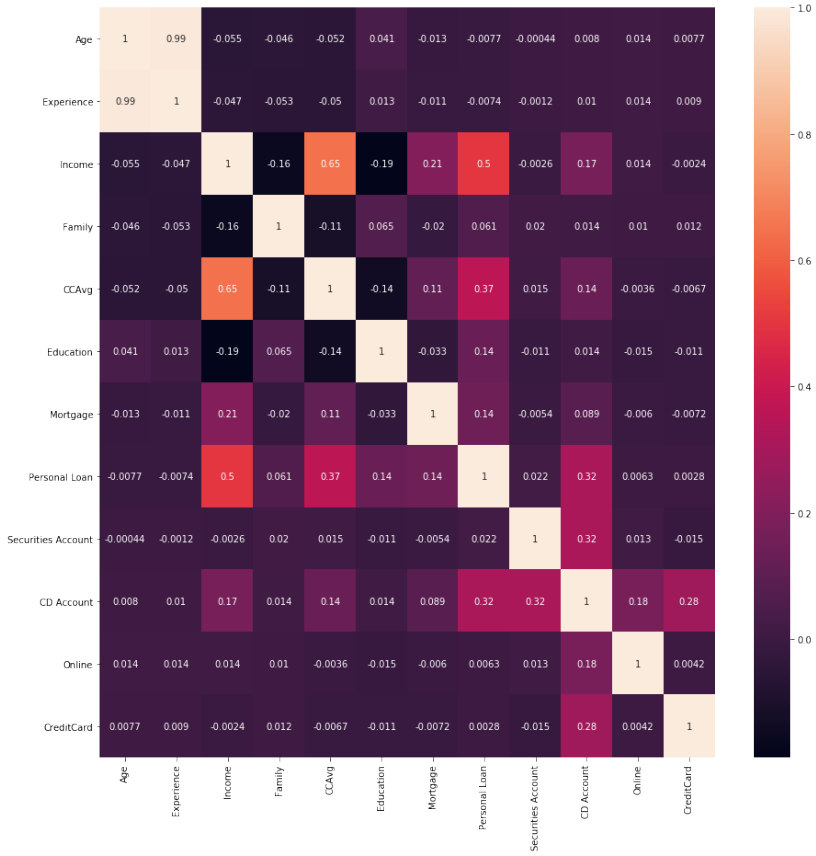
Description of the dataset – The dataset has categorical variables that lead to a binary classification problem to predict default basis the types of loan a customer has (CC, mortgage, personal loan etc.)

Reason – I was interested in looking at banking data because that's the industry I want to start my career in and this dataset gives a proper binary classification problem of whether the customer will default or not. Also the size of the dataset is easier to handle on my old machine.

Approach: Did some exploratory analysis and removed some variables, which were of no use. Looked at correlation matrix and created dummies as well.

Looking at correlation matrix, I can say that none of the variables are highly correlated except age and experience.

Correlation matrix:



SVM:

On Running the usual SVM fit we get the following output:

```
Accuracy on train is  = 0.9825
Accuracy on test is = 0.965
```

SVM best fit:

Tried the following values of the parameters

'C': [0.001, 0.01, 0.1, 1, 10],
'gamma': [0.001, 0.01, 0.1, 1],
'kernel':['linear','rbf']}

Final SVM Comment: In addition to Linear Kernel, as sigmoid kernel did not classify the negative classes, I decided to use Radial Kernel this time.

After executing both liner and radial kernel, for different values of gamma ranging from 0.001 to 1, this time I also changed the value of C which is the margin of error. And I experimented this values by changing them from a scale of 0.001 to 10.

Out of all the fitted models, Best output is achieved by the following parameter values:

C = 10
Gamma = 0.1
Kernel = Radial

The results for this model were as follows:

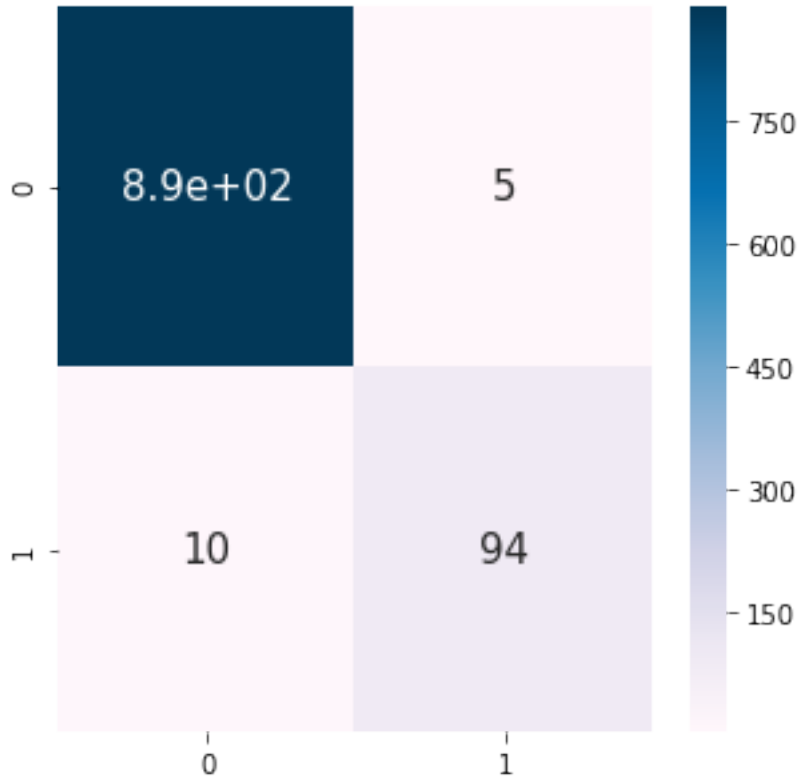
Accuracy on train is: 0.9925
Accuracy on test is: 0.972

Decision Tree: Unrestrained

```
Accuracy on train is: 1.0 Accuracy on test is: 0.985
```

	precision	recall	f1-score	support
0	0.99	0.99	0.99	896
1	0.95	0.90	0.93	104
micro avg	0.98	0.98	0.98	1000
macro avg	0.97	0.95	0.96	1000
weighted avg	0.98	0.98	0.98	1000

Classification Report:



Comment: Looking at the above values, it clearly looks like a case of overfitting. Only 109 observations have been misclassified.

Looking at this data, I tried to prune the decision tree and tried multiple depths but found the max_depth = 3 to be the best fit as per Gini Index

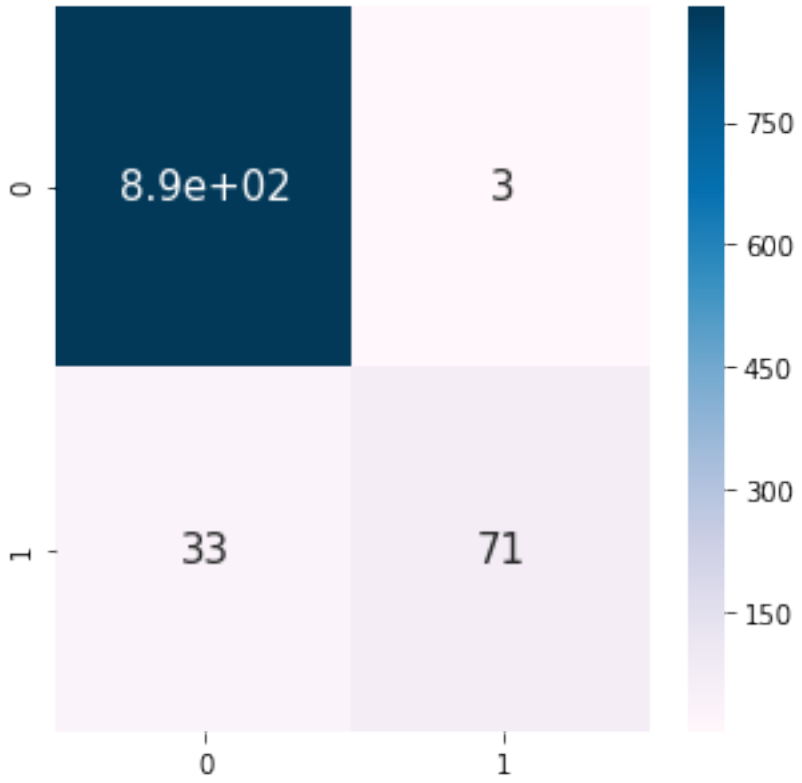
Decision Tree: Max_Depth = 3

Reached a optimum depth of 3 by running different versions

```
Accuracy on train is: 0.97175
Accuracy on test is: 0.964
```

	precision	recall	f1-score	support
0	0.96	1.00	0.98	896
1	0.96	0.68	0.80	104
micro avg	0.96	0.96	0.96	1000
macro avg	0.96	0.84	0.89	1000
weighted avg	0.96	0.96	0.96	1000

Classification plot:



Comment: Looking at the above values, the pruned tree is lightly better however there could still be chances to improve so will try boosting next.

Decision Tree : Boosting

Used the Adaboost classifier with the following parameters:

Depth = 2
Estimators = 100, 150, 200
Learning Rate = 0.1, 0.5, 0.9

This gives the train accuracy of 1 and test accuracy of 0.982 and uses a learning rate of 0.1. This could be overfitting so tried to prune the tree and arrived at the following output:

Decision Tree: Boosting and Pruning

This takes slightly higher time to run but the results are much better.

```
{'learning_rate': 0.1, 'n_estimators': 150}
Accuracy on train set = 0.99525
Accuracy on the test set = 0.988
```