Table of Contents

Upload Dataset 1
KNN - Past Payment Information 1
KNN - Personal Characteristics 3
Naive Bayes - Past Payment Information 5
Naive Bayes - Personal Characteristics 9
Initial Statistics for Dataset 9
Machine Learning Algorithm Accuracy Table 11

Upload Dataset

```
clc; clear all; close all
credit = readtable('CreditTaiwan.xlsx');
credit(:,1) = []; %remove first column

%Rename Column Headings in Data
credit.Properties.VariableNames{'sex'} = 'Gender';
credit.Properties.VariableNames{'defaultpaymentnextmonth'}
 = 'Defaults';

%Rename Target Variable into Categorical Classifications
credit.Defaults = categorical(credit.Defaults,[0 1],{'Not
 Defaulted' 'Defaulted'});

%Produce Summary of the Dataset
disp('Summary')
summary(credit.Defaults)

%Create three new columns with averages of payment history
credit.pay = mean(credit{:,6:11},2);
credit.billed = mean(credit{:,12:17},2);
credit.previouspayment = mean(credit{:,18:23},2);
```

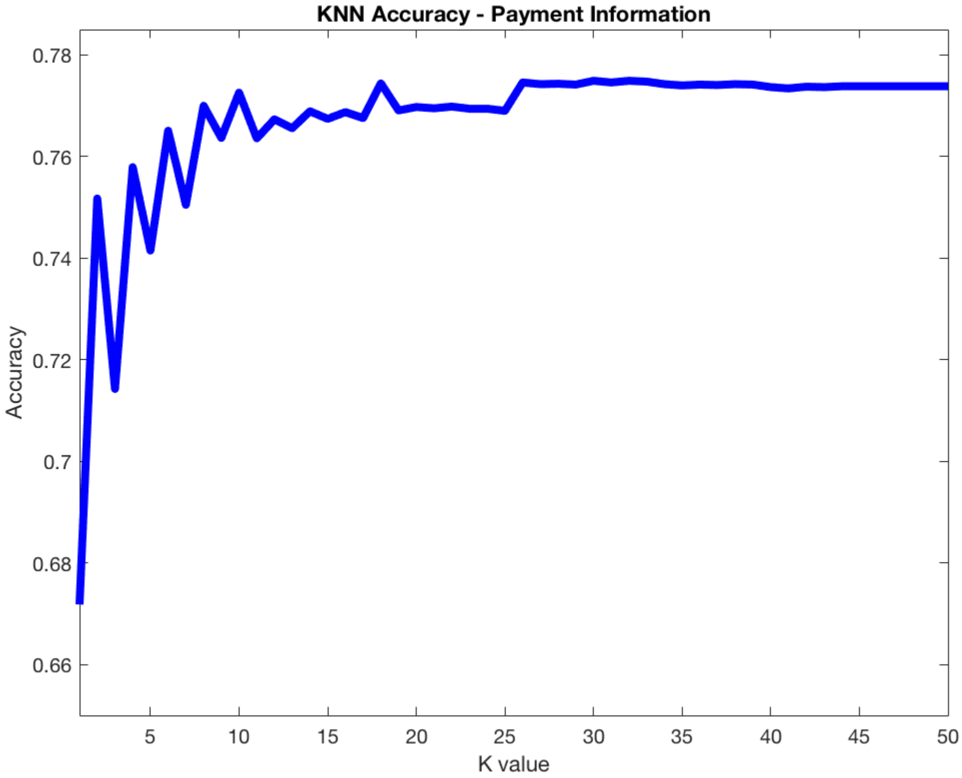
```
Warning: Variable names were modified to make them valid MATLAB
 identifiers. The
original names are saved in the VariableDescriptions property.
Summary
     Not Defaulted      23364
     Defaulted           6636
```

KNN - Past Payment Information

```
%Create Test and Training Sets
cv = cvpartition(30000, 'holdout',0.4); %Define cv

X1 = credit(:,end-2:end-1);
Y1 = credit.Defaults;
Xtrain = X1(training(cv),:);
```

```
Ytrain = Y1(training(cv),:);
Xtest = X1(test(cv),:);
Ytest = Y1(test(cv),:);

%Display Training and Test Set
disp('Training set')
tabulate (Ytrain)
disp('Test set')
tabulate (Ytest)

for k = 1:50
    mdlKNN1 = fitcknn(Xtrain, Ytrain, 'NumNeighbors',
 k, 'Distance', 'euclidean'); %Create Model
    rloss = resubLoss(mdlKNN1); %Percentage error in model
    PredictedResponseKNN1 = predict(mdlKNN1, [Xtest]);   %Predict
 response variable using Test Set
    PModelKNN1 = [PredictedResponseKNN1 Ytest];   %Matrix of predicted
 response and actual response in test set
    PredictionKNN1 = nnz( PModelKNN1(:,1) == PModelKNN1(:,2)); %Number
 of Correct Predictions
    AccuracyKNN1 = PredictionKNN1/length(PModelKNN1); %Accuracy of
 Model
    AccuracyKNN1Matrix(k,:) = [k AccuracyKNN1]; %Accuracy of Model in
 Matrix
end

%Plot KNN Accuracy
KNN1 = plot(AccuracyKNN1Matrix(:,1),
 AccuracyKNN1Matrix(:,2), 'b', 'LineWidth',4)
title('KNN Accuracy - Payment Information')
xlabel('K value')
ylabel('Accuracy')
axis([1,50,0.65,0.785])

%Extract figure
savefig('KNN1Pay.fig');

KNN1accuracy = max(AccuracyKNN1Matrix(:,2))
```

```
Training set
          Value    Count   Percent
  Not Defaulted    14078    78.21%
      Defaulted     3922    21.79%
Test set
          Value    Count   Percent
  Not Defaulted     9286    77.38%
      Defaulted     2714    22.62%

KNN1 = 

  Line with properties:

              Color: [0 0 1]
          LineStyle: '-'
          LineWidth: 4
             Marker: 'none'
         MarkerSize: 6
    MarkerFaceColor: 'none'
              XData: [1×50 double]
              YData: [1×50 double]
              ZData: [1×0 double]

  Use GET to show all properties


KNN1accuracy =

    0.7749
```



KNN - Personal Characteristics

```
%Create Test and Training Sets
cv = cvpartition(30000, 'holdout',0.4); %Define cv

X2 = credit(:,2:5);
Y2 = credit.Defaults;
Xtrain = X2(training(cv),:);
Ytrain = Y2(training(cv),:);
Xtest = X2(test(cv),:);
```

```
Ytest = Y2(test(cv),:);

%Display Training and Test Set
disp('Training set')
tabulate (Ytrain)
disp('Test sets')
tabulate (Ytest)

%KNN
for k = 1:50
    mdlKNN2 = fitcknn(Xtrain, Ytrain, 'NumNeighbors',
 k, 'Distance', 'euclidean'); %Create Model
    rlossKNN2 = resubLoss(mdlKNN2); %Percentage Error in Model
    PredictedKNN2 = predict(mdlKNN2, [Xtest]);  %Predict response
 variable using Test Set
    PModelKNN2 =[PredictedKNN2 Ytest];  %Matrix of predicted response
 and actual response in test set
    PredictionKNN2 = nnz( PModelKNN2(:,1) == PModelKNN2(:,2)); %Number
 of Correct Predictions
    AccuracyKNN2 = PredictionKNN2/length(PModelKNN2); %Accuracy of
 Model
    AccuracyKNN2Matrix(k,:) = [k AccuracyKNN2]; %Accuracy of Model in
 Matrix
end

%Plot KNN Accuracy
KNN2 = plot(AccuracyKNN2Matrix(:,1),
 AccuracyKNN2Matrix(:,2), 'r', 'LineWidth',4)
title('KNN Accuracy - Personal Characteristics')
xlabel('K value')
ylabel('Accuracy')
axis([1,50,0.65,0.785])

%Extract figure
savefig('KNN2Pay.fig')

KNN2accuracy = max(AccuracyKNN1Matrix(:,2))
```

```
Training set
          Value    Count   Percent
  Not Defaulted    14075     78.19%
      Defaulted     3925     21.81%
Test sets
          Value    Count   Percent
  Not Defaulted     9289     77.41%
      Defaulted     2711     22.59%

KNN2 = 

  Line with properties:

              Color: [1 0 0]
          LineStyle: '-'
          LineWidth: 4
```

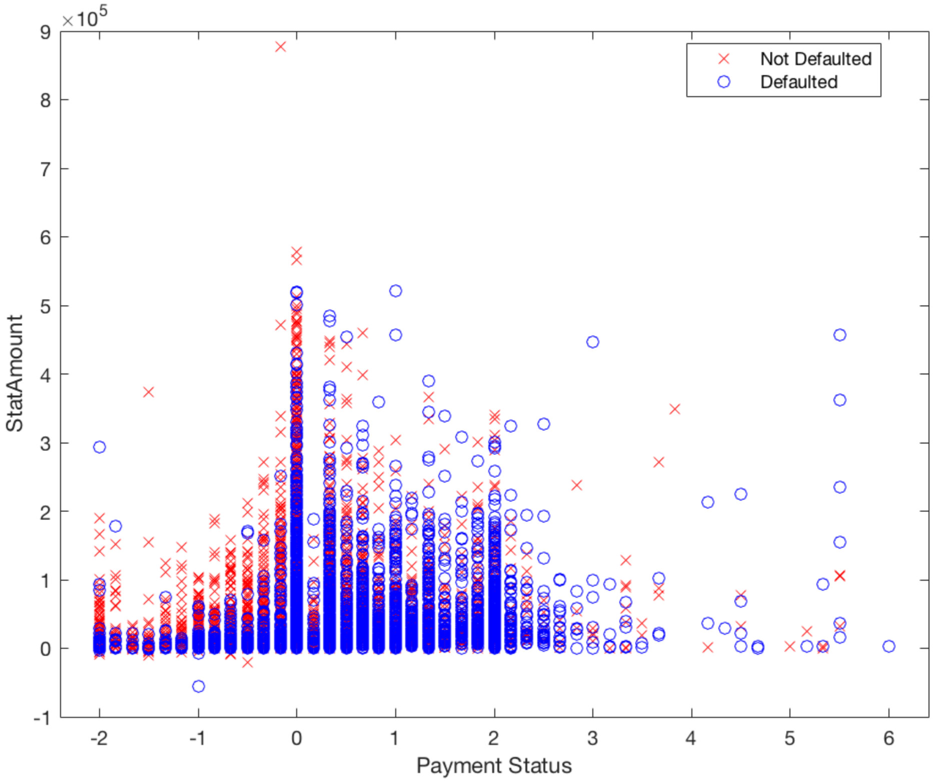
```
             Marker: 'none'
         MarkerSize: 6
    MarkerFaceColor: 'none'
              XData: [1×50 double]
              YData: [1×50 double]
              ZData: [1×0 double]

  Use GET to show all properties


KNN2accuracy =

    0.7749
```

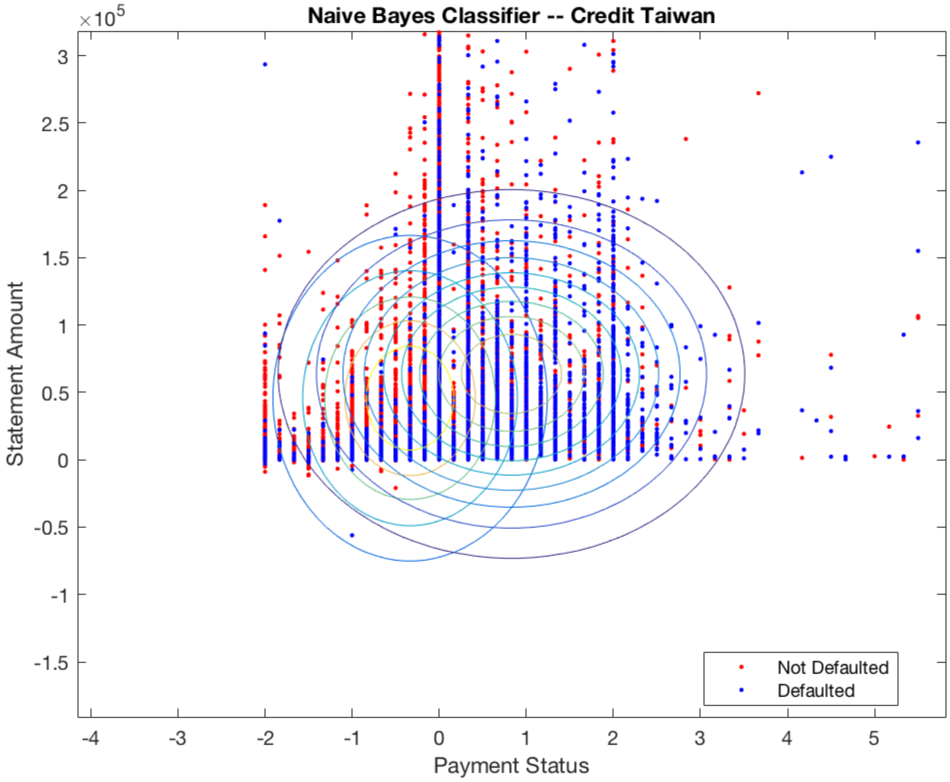
Naive Bayes - Past Payment Information

```
%Create Training and Test sets
X3 = credit(:,end-2:end-1);
Y3 = credit.Defaults;

cv = cvpartition(30000, 'holdout',0.4); %Define cv

Xtrain = X3(training(cv),:);
Ytrain = Y3(training(cv),:);
Xtest = X3(test(cv),:);
```

```
Ytest = Y3(test(cv),:);

%Display Training and Test Set
disp('Training set')
tabulate (Ytrain)
disp('Test sets')
tabulate (Ytest)

%scatter plot
gscatter(Xtrain{:,1}, Xtrain{:,2},
 Ytrain(:,1), 'rb', 'xo') %Scatterplot for training set
xlabel('Payment Status')
ylabel('StatAmount')
N = size(credit,1);

%Extract figure
savefig('NB1.fig')
NB_PPI = openfig('NB1.fig')

%Create NB Model
Mdl = fitcnb(Xtrain,Ytrain, ...
    'ClassNames',{'Not Defaulted', 'Defaulted'});
NotDefaultedIndex = strcmp(Mdl.ClassNames, 'Not Defaulted');
PredictionNB1 =
 Mdl.DistributionParameters(NotDefaultedIndex,1); %Create Model

%Create NB Econtours
gscatter(Xtrain{:,1}, Xtrain{:,2}, Ytrain(:,1), 'rb'); %Plot Econtour
h=gca;
hold on

Params = cell2mat(Mdl.DistributionParameters);
Mu = Params(1:4,1:2); % Extract the means
Sigma = zeros(2,2,2);
for j = 1:2
    Sigma(:,:,j) = diag(Params(2*j,:)).^2; %Create diagonal covariance
 matrix
    xlim = Mu(j,1) + 4*[1 -1]*sqrt(Sigma(1,1,j));
    ylim = Mu(j,2) + 4*[1 -1]*sqrt(Sigma(2,2,j));
    ezcontour(@(x1,x2)mvnpdf([x1,x2],Mu(j,:),Sigma(:,:,j)),[xlim
 ylim])
        %Draw contours for the multivariate normal distributions
end
title('Naive Bayes Classifier -- Credit Taiwan')
xlabel('Payment Status')
ylabel('Statement Amount')

%Extract figure
savefig('NB2.fig')

hold off

%Accuracy of NB Model
NB_Predict = predict(Mdl, [Xtest]);
```

```
NB_Predict1 = [NB_Predict Ytest];
Prediction = nnz(NB_Predict1(:,1) == NB_Predict1(:,2));
AccuracyNB1 = Prediction/length(NB_Predict1)

Training set
          Value    Count   Percent
  Not Defaulted    14021     77.89%
      Defaulted     3979     22.11%
Test sets
          Value    Count   Percent
  Not Defaulted     9343     77.86%
      Defaulted     2657     22.14%

NB_PPI =

  Figure (2) with properties:

      Number: 2
        Name: ''
       Color: [0.9400 0.9400 0.9400]
    Position: [360 271 560 420]
       Units: 'pixels'

  Use GET to show all properties


AccuracyNB1 =

    0.8039
```


×10^5
9
8
7
6
5
4
3
2
1
0
-1
StatAmount
-2
-1
0
1
2
3
4
5
6
Payment Status
× Not Defaulted
○ Defaulted


Naive Bayes Classifier -- Credit Taiwan
×10^5
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
Statement Amount
-4
-3
-2
-1
0
1
2
3
4
5
Payment Status
● Not Defaulted
● Defaulted

Naive Bayes - Personal Characteristics

```
%Creat Training and Test sets
X4 = credit(:,2:5);
Y4 = credit.Defaults;

cv = cvpartition(30000, 'holdout',0.4); %Define cv

Xtrain = X4(training(cv),:);
Ytrain = Y4(training(cv),:);
Xtest = X4(test(cv),:);
Ytest = Y4(test(cv),:);

%Display Training and Test Set
disp('Training set')
tabulate (Ytrain)
disp('Test sets')
tabulate (Ytest)

%Predict Model
Mdl = fitcnb(Xtrain,Ytrain, ...
    'ClassNames',{'Not Defaulted', 'Defaulted'});
NotDefaultedIndex = strcmp(Mdl.ClassNames, 'Not Defaulted');
estimates = Mdl.DistributionParameters(NotDefaultedIndex,1); %Create
 Model

%Accuracy of NB Model
NB_Predict = predict(Mdl, [Xtest]);
NB_Predict1 = [NB_Predict Ytest];
Prediction = nnz(NB_Predict1(:,1) == NB_Predict1(:,2));
AccuracyNB2 = Prediction/length(NB_Predict1)
```

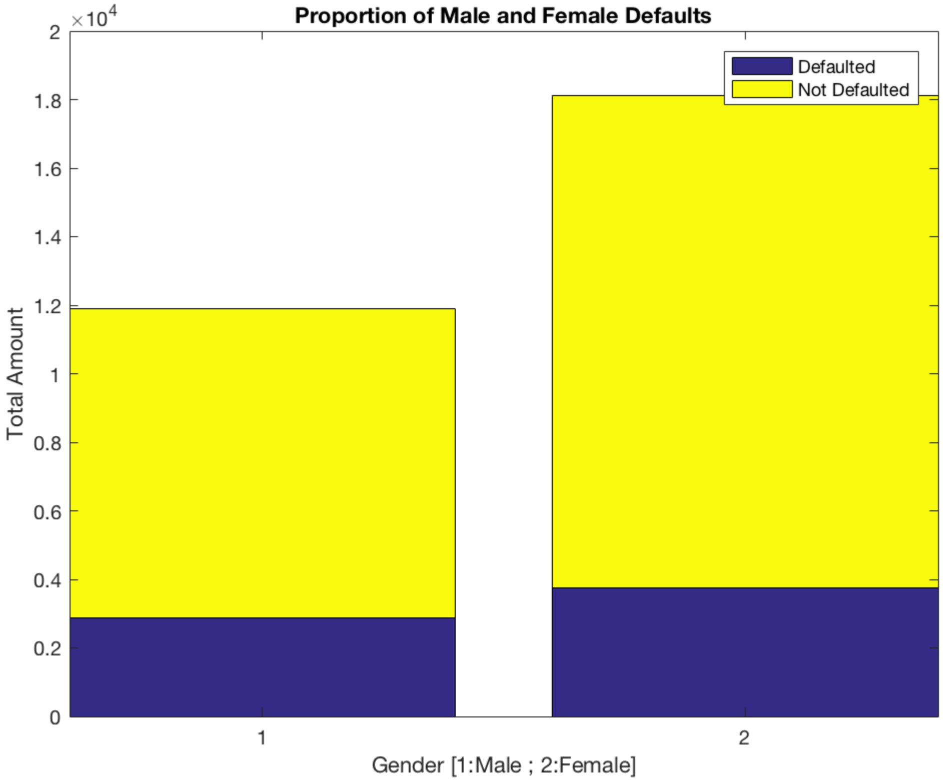
```
Training set
          Value    Count   Percent
  Not Defaulted    13997     77.76%
      Defaulted     4003     22.24%
Test sets
          Value    Count   Percent
  Not Defaulted     9367     78.06%
      Defaulted     2633     21.94%

AccuracyNB2 =

    0.7804
```

Initial Statistics for Dataset

```
%Rename Gender Variable into Categorical Classifications
credit.Gender = categorical(credit.Gender,[1 2],{'male' 'female'});

%Extract number of Males vs Females defaults
```

```
maleDefaulted = nnz(credit.Gender == 'male' & credit.Defaults
 == 'Defaulted');
maleNotDefaulted = nnz(credit.Gender == 'male' & credit.Defaults
 == 'Not Defaulted');
femaleDefaulted = nnz(credit.Gender == 'female' & credit.Defaults
 == 'Defaulted');
femaleNotDefaulted = nnz(credit.Gender == 'female' & credit.Defaults
 == 'Not Defaulted');

%Create Matrix of Gender defaults
GenderDefaultsMatrix =[ maleDefaulted maleNotDefaulted;
 femaleDefaulted femaleNotDefaulted];

%Create table of Matrix
Columns = {'Defaulted', 'NotDefaulted'};
Rows = {'Male', 'Female'}; %Label columns and rows

GenderDefaultsTable =
 array2table( GenderDefaultsMatrix, 'VariableNames',
 Columns, 'RowNames', Rows ) %Generate Table

%Create bar plot of Gender Defaults
bar(GenderDefaultsMatrix, 'stacked')
title('Proportion of Male and Female Defaults')
xlabel('Gender [1:Male ; 2:Female]')
ylabel('Total Amount')
legend('Defaulted','Not Defaulted')

%Extract figure
savefig('GenderMatrix.fig')
```

```
GenderDefaultsTable =

              Defaulted    NotDefaulted
              _________    ____________

    Male      2873          9015
    Female    3763         14349
```



Machine Learning Algorithm Accuracy Table

```
%Extract accuracy
maxaccuracyKNN1 = max(AccuracyKNN1Matrix(:,2));
maxaccuracyKNN2 = max(AccuracyKNN2Matrix(:,2));
maxaccuracyNB1 = max(AccuracyNB1);
maxaccuracyNB2 = max(AccuracyNB2);

%Matrix
MaximumMatrix = [maxaccuracyKNN1 maxaccuracyKNN2; maxaccuracyNB1
 maxaccuracyNB2];

%Create table of Matrix
Columns1 = {'PaymentHistory', 'PersonalCharacteristics'};
Rows1 = {'Naive Bayes', 'KNN'};
MLAccuracyTable = array2table( MaximumMatrix, 'VariableNames',
 Columns1, 'RowNames', Rows1 )
```

```
MLAccuracyTable =

                   PaymentHistory    PersonalCharacteristics
                   ______________    _______________________

    Naive Bayes    0.77492           0.77408
    KNN            0.80392           0.78042
```

Published with MATLAB® R2016b