HarvardX Capstone Project - CYO - Credit Card Approval

Jamuna Bhaskar

January 09 2020

1 Introduction

The project for which we would develop machine learning algorithm is Credit Card application assessment. There are multiple checkpoints involved in deciding on a credit card approval, out of which income of the customer, any previous default credit obligations and employment status play a significant role.

The purpose of this documentation is to demonstrate the machine learning techniques and the ability to clearly communicate the process and insights gained from this analysis. Here we have used Credit Approval Dataset from UCI Machine Learning Repository. This dataset is a combination of applicant's details and approval decisions.

1.1 Credit Approval Dataset

As mentioned in the dataset description file, this dataset contains both instances of people who got their credit card approved as well as denied. All attribute names and values have been encrypted to protect confidentiality of data. And 5% of the available data has one or more missing values.

Let's take a quick look at the dataset.

```
## Loading required namespace: DT
```

Show 10 entries Search:

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16
1	b	30.83	0.000	u	g	w	v	1.25	t	t	1	f	g	00202	0	+
2	a	58.67	4.460	u	g	q	h	3.04	t	t	6	f	g	00043	560	+
3	a	24.50	0.500	u	g	q	h	1.50	t	f	0	f	g	00280	824	+
4	b	27.83	1.540	u	g	w	v	3.75	t	t	5	t	g	00100	3	+
5	b	20.17	5.625	u	g	w	v	1.71	t	f	0	f	s	00120	0	+
6	b	32.08	4.000	u	g	m	v	2.50	t	f	0	t	g	00360	0	+

Showing 1 to 6 of 6 entries Previous 1 Next

Attribute Information:

- A1: b, a.
- A2: continuous.
- A3: continuous.
- A4: u, y, l, t.
- A5: g, p, gg.
- A6: c, d, cc, i, j, k, m, r, q, w, x, e, aa, ff.
- A7: v, h, bb, j, n, z, dd, ff, o.
- A8: continuous.
- A9: t, f.
- A10: t, f.
- A11: continuous.
- A12: t, f.
- A13: g, p, s.
- A14: continuous.
- A15: continuous.
- A16: +,- (class attribute)

With reference to the above attribute information provided in the documentation, lets give meaningful names to the columns.

Show 10 entries Search:

	gender	age	debt	married	customer	education	ethnicity	yearsemployed	priordefault	employmentstatu
1	b	30.83	0.000	u	g	w	v	1.25	t	t

2	a	58.67	4.460	u	g	q	h	3.04	t	t
3	a	24.50	0.500	u	g	q	h	1.50	t	f
4	b	27.83	1.540	u	g	w	v	3.75	t	t
5	b	20.17	5.625	u	g	w	v	1.71	t	f
6	b	32.08	4.000	u	g	m	v	2.50	t	f

Showing 1 to 6 of 6 entries

Previous 1 Next

The data set has 690 observations of 16 variables.

```
## 'data.frame':    690 obs. of  16 variables:
##  $ gender          : Factor w/ 3 levels "?","a","b": 3 2 2 3 3 3 3 2 3 3 ...
##  $ age             : Factor w/ 350 levels "?","13.75","15.17",..: 158 330 91 127 45 170 181 76 312 257 ...
##  $ debt            : num  0 4.46 0.5 1.54 5.62 ...
##  $ married         : Factor w/ 4 levels "?","l","u","y": 3 3 3 3 3 3 3 3 4 4 ...
##  $ customer        : Factor w/ 4 levels "?","g","gg","p": 2 2 2 2 2 2 2 2 4 4 ...
##  $ education       : Factor w/ 15 levels "?","aa","c","cc",..: 14 12 12 14 14 11 13 4 10 14 ...
##  $ ethnicity       : Factor w/ 10 levels "?","bb","dd",..: 9 5 5 9 9 9 5 9 5 9 ...
##  $ yearsemployed   : num  1.25 3.04 1.5 3.75 1.71 ...
##  $ priordefault    : Factor w/ 2 levels "f","t": 2 2 2 2 2 2 2 2 2 2 ...
##  $ employmentstatus: Factor w/ 2 levels "f","t": 2 2 1 2 1 1 1 1 1 1 ...
##  $ creditscore     : int  1 6 0 5 0 0 0 0 0 0 ...
##  $ driverlicence   : Factor w/ 2 levels "f","t": 1 1 1 2 1 2 2 1 1 2 ...
##  $ citizenship     : Factor w/ 3 levels "g","p","s": 1 1 1 1 3 1 1 1 1 1 ...
##  $ zipcode         : Factor w/ 171 levels "?","00000","00017",..: 70 13 98 33 39 117 56 25 64 17 ...
##  $ income          : int  0 560 824 3 0 0 31285 1349 314 1442 ...
##  $ approvalstatus  : Factor w/ 2 levels "-","+": 2 2 2 2 2 2 2 2 2 2 ...
```

2 Data Transformation

2.1 Binary Conversion of Data

As a first step, lets choose the variables with 2 status values and convert them to binary values, assuming they might have an influence on approval status. This would help us in further analysis when working with classification models.

2.2 Missing Data

With reference to dataset description file, we are aware that there are 5% missing values. By taking a look at the summary, we recognize that missing values are labelled as '?'

```
##  gender       age           debt        married customer   education
##  0:480   ?      : 12   Min.   : 0.000   ?:  6   ? :  6   c      :137
##  1:210   22.67  :  9   1st Qu.: 1.000   l:  2   g :519   q      : 78
##          20.42  :  7   Median : 2.750   u:519   gg:  2   w      : 64
##          18.83  :  6   Mean   : 4.759   y:163   p :163   i      : 59
##          19.17  :  6   3rd Qu.: 7.207                    aa     : 54
##          20.67  :  6   Max.   :28.000                    ff     : 53
##          (Other):644                                     (Other):245
##    ethnicity   yearsemployed    priordefault employmentstatus
##  v      :399   Min.   : 0.000   0:329        0:395
##  h      :138   1st Qu.: 0.165   1:361        1:295
##  bb     : 59   Median : 1.000
##  ff     : 57   Mean   : 2.223
##  ?      :  9   3rd Qu.: 2.625
##  j      :  8   Max.   :28.500
##  (Other): 20
##   creditscore   driverlicence citizenship    zipcode        income
##  Min.   : 0.0   f:374         g:625       00000  :132   Min.   :     0.0
##  1st Qu.: 0.0   t:316         p:  8       00120  : 35   1st Qu.:     0.0
##  Median : 0.0                 s: 57       00200  : 35   Median :     5.0
##  Mean   : 2.4                             00160  : 34   Mean   :  1017.4
##  3rd Qu.: 3.0                             00080  : 30   3rd Qu.:   395.5
##  Max.   :67.0                             00100  : 30   Max.   :100000.0
##                                           (Other):394
##  approvalstatus
##  0:383
##  1:307
##
##
##
##
##
```

The attributes are

- Age - 12 records with missing age value
- Married - 6 records with missing value

- Customer - 6 records with missing value
- Ethnicity - 9 records with missing value

Based on the dependency, either the occurrences can be deleted or plugged in with estimated values.

In the above list, Age is the continuous variable. Lets start to fill in the missing values for age variable. The simplest method is to find out the mean and substitute the NAs. However an accurate method would be to find the most closely correlated variables with age and predict the values using **linear regression**.

```
## Warning: NAs introduced by coercion
```

```
##                 age  debt yearsemployed creditscore income
## age           1.000 0.202         0.396       0.186  0.019
## debt          0.202 1.000         0.301       0.272  0.122
## yearsemployed 0.396 0.301         1.000       0.327  0.053
## creditscore   0.186 0.272         0.327       1.000  0.063
## income        0.019 0.122         0.053       0.063  1.000
```

From the above correlation table, we observe that the closely correlated variable with age is yearsemployed (0.396). So we'll use age and yearsemployed to create linear regression model which in turn would be used to predict missing age values.

```
##
## Call:
## lm(formula = age ~ yearsemployed, data = credit_app, na.action = na.exclude)
##
## Coefficients:
##   (Intercept)  yearsemployed
##        28.448          1.412
```

Records with missing age values are shown below.

	gender	age	debt	married	customer	education	ethnicity	yearsemployed	priordefault	employmentstatus	creditscore	driverlicence	citize
84	1	NA	3.500	u	g	d	v	3.000	1	0	0	t	
87	0	NA	0.375	u	g	d	v	0.875	1	0	0	t	
93	0	NA	5.000	y	p	aa	v	8.500	1	0	0	f	
98	0	NA	0.500	u	g	c	bb	0.835	1	0	0	t	
255	0	NA	0.625	u	g	k	v	0.250	0	0	0	f	
287	1	NA	1.500	u	g	ff	ff	0.000	0	1	2	t	
330	0	NA	4.000	y	p	i	v	0.085	0	0	0	t	
446	1	NA	11.250	u	g	ff	ff	0.000	0	0	0	f	
451	0	NA	3.000	y	p	i	bb	7.000	0	0	0	f	
501	0	NA	4.000	u	g	x	v	5.000	1	1	3	t	
516	0	NA	10.500	u	g	x	v	6.500	1	0	0	f	
609	0	NA	0.040	y	p	d	v	4.250	0	0	0	t	

Then using Linear regression we have predicted age value and below table lists the same rows with predicted value.

	gender	age	debt	married	customer	education	ethnicity	yearsemployed	priordefault	employmentstatus	creditscore	driverlicence	c
84	1	32.68500	3.500	u	g	d	v	3.000	1	0	0	t	
87	0	29.68382	0.375	u	g	d	v	0.875	1	0	0	t	
93	0	40.45276	5.000	y	p	aa	v	8.500	1	0	0	f	
98	0	29.62732	0.500	u	g	c	bb	0.835	1	0	0	t	
255	0	28.80112	0.625	u	g	k	v	0.250	0	0	0	f	
287	1	28.44804	1.500	u	g	ff	ff	0.000	0	1	2	t	
330	0	28.56808	4.000	y	p	i	v	0.085	0	0	0	t	
446	1	28.44804	11.250	u	g	ff	ff	0.000	0	0	0	f	
451	0	38.33428	3.000	y	p	i	bb	7.000	0	0	0	f	
501	0	35.50964	4.000	u	g	x	v	5.000	1	1	3	t	
516	0	37.62812	10.500	u	g	x	v	6.500	1	0	0	f	
609	0	34.45040	0.040	y	p	d	v	4.250	0	0	0	t	

Linear Regression -> Y=a+bX *b -> slope

*a -> intercept

*X -> respective yearsemployed value from table

Age[609] = 28.448 + (1.412 * 4.250) = 34.45040

2.3 Normalization

Next we observe that the continuous variables are of different scale. To overcome this we use Normalization, means transforming the values to the range between 0 and 1. This brings the dataset to a common scale (between 0 and 1) while keeping the distributions of variables the same. Here we use **z-score normalization**.

$$\frac{value - \mu}{\sigma}$$

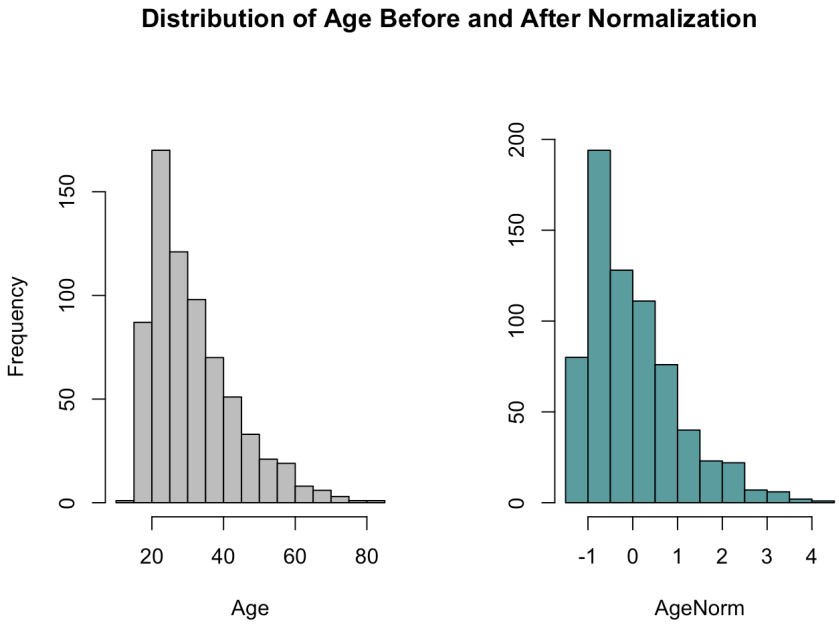
μ - mean value

σ - standard deviation

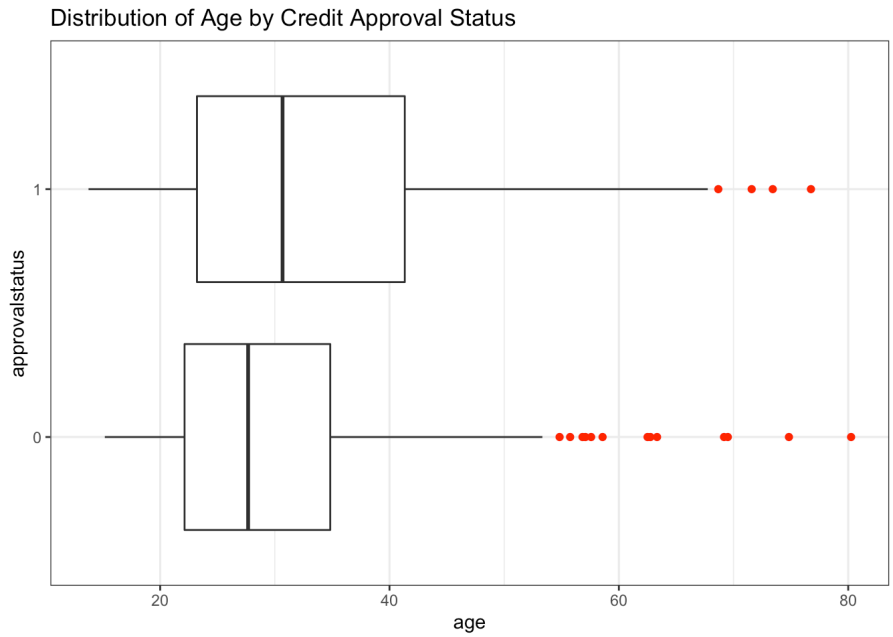
```
sigma = sd(num_data$age, na.rm=TRUE)
sigma
```

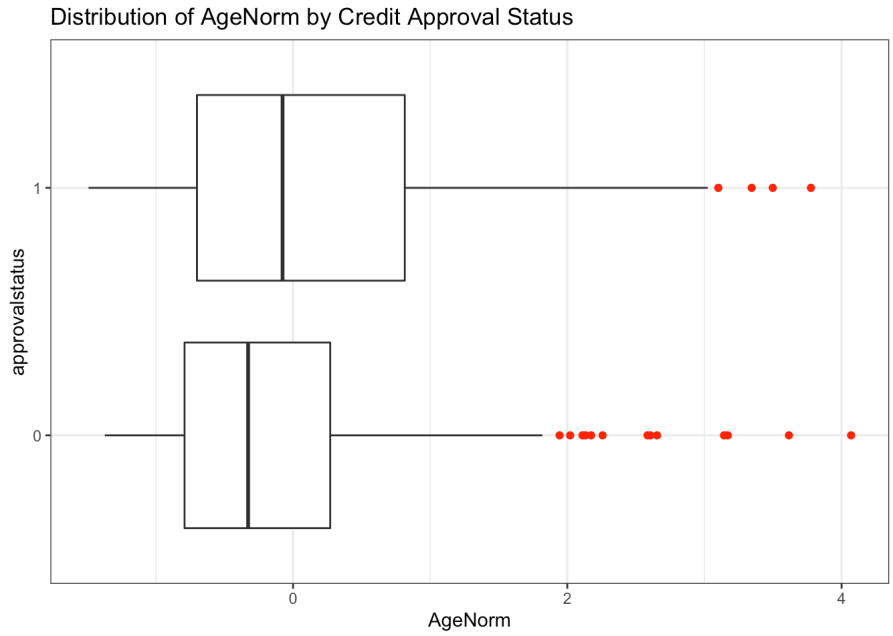
```
## [1] 11.95786
```

Normalize age

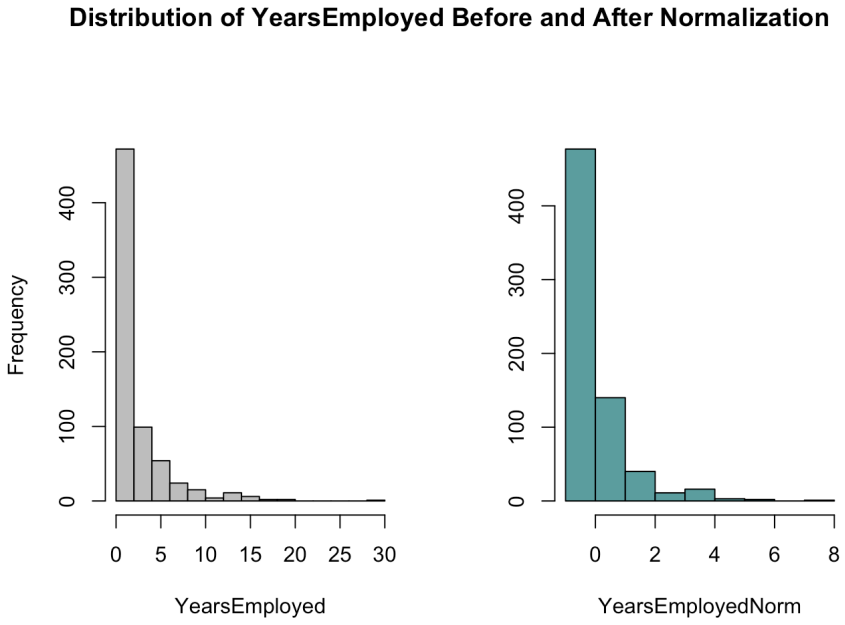


We'll use a boxplot showing the mean value for each group and the quartiles. This can be interpreted as the credit applicants with lower age are less likely to be approved, however there are several outlying applicants with high values that still were not provided approval.





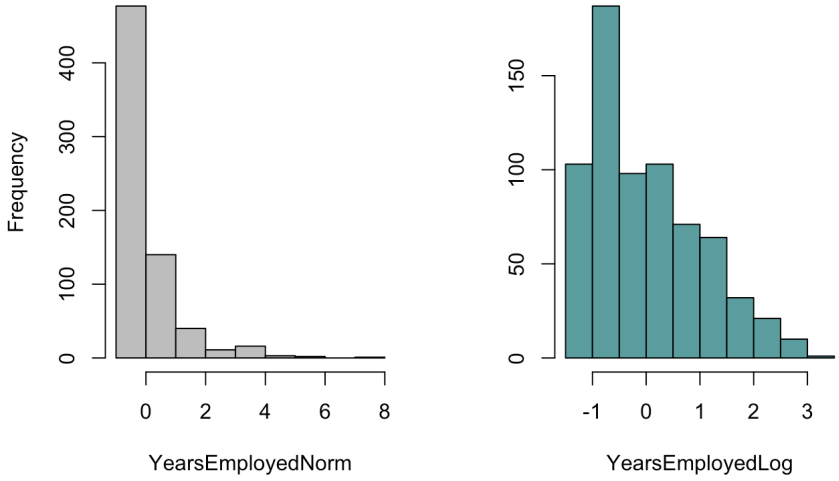
Normalize yearsemployed



2.4 Log Transformation

By looking at these two histograms, we can notice the data is skewed towards right meaning unbalanced distribution. The logarithmic transformation can be used to make highly skewed distributions less skewed.

Distribution of Values Before and After Log Transformation



3 Generate Train & Test

The machine learning approach is to train an algorithm using a dataset for which we do know the actual outcome, and then apply this algorithm in the future to make a prediction when we don't know the actual outcome.

Let's split the dataset into Train and Test sets. Train set would be used to create and train the model and test set would be used to validate the model. We will allocate 75% of the items to Training and 25% items to the Test set.

```
set.seed(1)
split<- sample.split(credit_app$approvalstatus, SplitRatio=0.75)
Train<- subset(credit_app,split==TRUE)
Test <- subset(credit_app, split==FALSE)
```

4 Modeling Approach

4.1 Logistic Regression

The machine learning task is to build an algorithm that returns a prediction for any of the possible values of the features. As mentioned in book, Introduction to Data Science by author Irizarry, data comes in the form of:

- the **outcome** we want to predict and
- the **features** that we will use to predict the outcome

  In our project, the **outcome** we want to predict is approvalstatus which is binary (1 or 0), meaning the application can be either approved or denied. When the outcome is categorical, we refer to the machine learning task as **classification**, and the main output of the model will be a decision rule which prescribes which of the K classes we should predict. One of the classification techniques we're about to use is **Logistic Regression**, as it is intended for binary (two-class) classification problems. It will predict the probability of an instance belonging to the default class, which can be snapped into a 0 or 1 classification.

4.1.1 Model 1

```
LogFit<- glm(approvalstatus~AgeNorm+DebtLog+YearsEmployedLog+CreditScoreLog+IncomeLog,
        data=Train,family=binomial)
summary(LogFit)
```

```
##
## Call:
## glm(formula = approvalstatus ~ AgeNorm + DebtLog + YearsEmployedLog +
##     CreditScoreLog + IncomeLog, family = binomial, data = Train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2673  -0.7915  -0.5417   0.7383   2.0550
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -0.15554    0.10891  -1.428 0.153241
## AgeNorm           0.05415    0.11380   0.476 0.634214
## DebtLog           0.03894    0.11253   0.346 0.729296
## YearsEmployedLog  0.56982    0.12425   4.586 4.52e-06 ***
## CreditScoreLog    0.97231    0.13611   7.144 9.10e-13 ***
## IncomeLog         0.40035    0.11439   3.500 0.000466 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 710.42  on 516  degrees of freedom
## Residual deviance: 530.57  on 511  degrees of freedom
## AIC: 542.57
##
## Number of Fisher Scoring iterations: 4
```

```
LogPred<- predict(LogFit,newdata=Train, type="response")
table(predicted = LogPred>0.5,actual = Train$approvalstatus)
```

```
##          actual
## predicted   0   1
##     FALSE 244  84
##     TRUE   43 146
```

4.1.1.1 Observation

From the distribution of actual and predicted values, 146 are true positive and 244 are true negatives. This means, out of 517 observations, 244 are correctly predicted as denied and 146 are correctly predicted as approved, so here the accuracy of prediction is **75%**.

4.1.2 Model 2

Let's take a look at the p-values from MODEL-1 for each coefficients. A p-value less than 0.05 (typically ≤ 0.05) is statistically significant. In the above summary, AgeNorm and DebtLog are not significant (p-value > 0.05). Now lets refine the model by excluding the non-significant ones, AgeNorm and DebtLog.

```
##
## Call:
## glm(formula = approvalstatus ~ YearsEmployedLog + CreditScoreLog +
##     IncomeLog, family = binomial, data = Train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2640  -0.7907  -0.5409   0.7414   2.0249
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -0.1570     0.1088  -1.443 0.148941
## YearsEmployedLog   0.5942     0.1174   5.060 4.18e-07 ***
## CreditScoreLog     0.9722     0.1355   7.174 7.27e-13 ***
## IncomeLog          0.4054     0.1140   3.558 0.000374 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 710.42  on 516  degrees of freedom
## Residual deviance: 530.94  on 513  degrees of freedom
## AIC: 538.94
##
## Number of Fisher Scoring iterations: 4
```

```
##          actual
## predicted   0   1
##     FALSE 242  84
##     TRUE   45 146
```

4.1.2.1 Observation

Out of 517 observations, 242 are correctly predicted as denied and 146 are correctly predicted as approved and there is no improvement in the accuracy. The accuracy is still **75%** as in MODEL-1.

4.1.3 Inference

As the accuracy remains the same in both models, we will use MODEL-2 (the model excluding non-significant variables) for validation.

4.1.4 Apply the prediction on Test

Let's apply the model on test data.

```
##          actual
## predicted  0  1
##     FALSE 88 20
##     TRUE   8 57
```

4.1.4.1 Observation

145 (88+57) out of 173 observations are correct predictions which is **84%** accuracy.

4.2 K – Nearest Neighbour

KNN algorithm is one of the most used supervised learning algorithm. It's a non-parametric algorithm as it does not make any assumptions on the underlying data distribution. And this works based on feature similarity. Basically it collects the features of applicants comparing with similar features in the dataset. So this is useful to predict the approval status of a new applicant, without going through all the steps.

Let's start by choosing a K value. Usually an odd number is chosen if the number of classes is 2. The optimal value can be found by incrementing it.

K=1

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 252  73
##          1  35 157
##
##                Accuracy : 0.7911
##                  95% CI : (0.7535, 0.8254)
##     No Information Rate : 0.5551
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.57
##
##  Mcnemar's Test P-Value : 0.0003704
##
##             Sensitivity : 0.8780
##             Specificity : 0.6826
##          Pos Pred Value : 0.7754
##          Neg Pred Value : 0.8177
##              Prevalence : 0.5551
##          Detection Rate : 0.4874
##    Detection Prevalence : 0.6286
##       Balanced Accuracy : 0.7803
##
##        'Positive' Class : 0
##
```

K=3

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 249  56
##          1  38 174
##
##                Accuracy : 0.8182
##                  95% CI : (0.7822, 0.8505)
##     No Information Rate : 0.5551
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.629
##
##  Mcnemar's Test P-Value : 0.07953
##
##             Sensitivity : 0.8676
##             Specificity : 0.7565
##          Pos Pred Value : 0.8164
##          Neg Pred Value : 0.8208
##              Prevalence : 0.5551
##          Detection Rate : 0.4816
##    Detection Prevalence : 0.5899
##       Balanced Accuracy : 0.8121
##
##        'Positive' Class : 0
##
```

K=5

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 250  73
##          1  37 157
##
##                Accuracy : 0.7872
##                  95% CI : (0.7494, 0.8218)
##     No Information Rate : 0.5551
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5624
##
##  Mcnemar's Test P-Value : 0.0008465
##
##             Sensitivity : 0.8711
##             Specificity : 0.6826
##          Pos Pred Value : 0.7740
##          Neg Pred Value : 0.8093
##              Prevalence : 0.5551
##          Detection Rate : 0.4836
##    Detection Prevalence : 0.6248
##       Balanced Accuracy : 0.7768
##
##        'Positive' Class : 0
##
```

The Optimal K

From observing the above results, confusion matrix for K=3 is better than K=1. The overall accuracy for K=3 is **82%** whereas for K=1 is **79%**. When we check for K=5, overall accuracy is **80%**. K being the tuning parameter, we need to have a mechanism to find the optimal K. We want to pick the K that maximizes the accuracy and minimizes the error.

```
ks[which.max(accuracy$train)]
```

```
## [1] 3
```

```
max(accuracy$train)
```

```
## [1] 0.8201161
```

Now we know the optimal K is 3, lets apply that on test set.

```
##  Accuracy
## 0.8034682
```

4.3 Chi-Squared test

The Chi Squared is a test for independence between two variables. We'll use this test to check if approval status is independent of ethnicity. The null hypothesis is that there's no relationship between ethnicity and approval.

```
## # A tibble: 10 x 3
##    ethnicity  Freq approvalstatus
##        <int> <int>          <int>
##  1         1     9              4
##  2         2    59             25
##  3         3     6              2
##  4         4    57              8
##  5         5   138             87
##  6         6     8              3
##  7         7     4              2
##  8         8     2              1
##  9         9   399            169
## 10        10     8              6
```

```
## Warning in chisq.test(tbl[2:3]): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  tbl[2:3]
## X-squared = 18.471, df = 9, p-value = 0.03008
```

4.3.1 Observation

The resulting p-value is less than 0.05 so we cannot reject the null hypothesis.

4.3.2 Inference

More investigation is needed to make sure if the results are due to chance and are not significant in terms of supporting the idea being investigated.

5 Conclusion

We are able to conclude that the most significant attributes in determining the outcome of a credit application are Income, Years of Employment and Credit Score. In this project, models are built with focus on applying the classification techniques such as Logistic Regression and KNN. Accuracy predicted by both the methods are very similar.

Future work could include combination of techniques to produce improved accuracy to avoid the risk of approving a credit card to someone that should have been denied.