PREDICTIVE ANALYTICS

SEC 01

ALY 6020, FALL 2022

WEEK 2: MODULE 3 MIDWEEK PROJECT 3

SUBMITTED BY: Akash Raj, Ananya Sharma, Vaibhav Arora, Vaibhav Jain

CRN: 70916

SUBMITTED TO: Dr. Mary Donhoffner

DATE: October 10, 2022

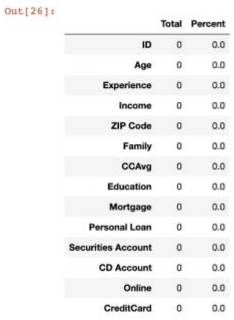
INTRODUCTION

We are developing a logistic regression model for personal loans in this mid-week project. Here, we are attempting to forecast to decide whether to approve or reject personal loans. Logistic regression is a supervised learning algorithm with a qualitative independent variable. This assignment aims to identify the best suitable variables that increase the likelihood of loan acceptance.

DATA CLEANING

For the given dataset we have 5000 rows of data and 14 variables. From Fig 1, we can say that there are no missing values in the dataset. But that does not mean our data is clean. We calculated descriptive statistics to get our data. Please refer to Figure 2. Since Experience has negative values in this dataset, we used the following steps to remove the ambiguity from column experience.

1. Create two separate data frames with records where the experience value is greater than or less than 0.
2. Retrieve a list of Customer IDs from the data frame that contains records with negative experience values.
3. Iterating through the list of Customer IDs.
4. Get the corresponding ID's age and education level from the negative experience data frame.
5. Filter the records in the positive experience data frame based on the age and education value obtained.

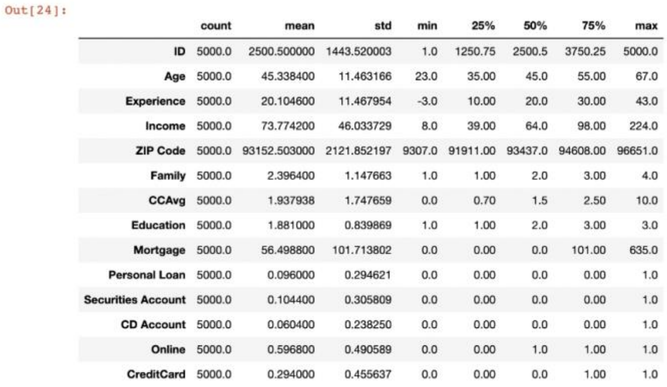


Out[26]:

	Total	Percent
ID	0	0.0
Age	0	0.0
Experience	0	0.0
Income	0	0.0
ZIP Code	0	0.0
Family	0	0.0
CCAvg	0	0.0
Education	0	0.0
Mortgage	0	0.0
Personal Loan	0	0.0
Securities Account	0	0.0
CD Account	0	0.0
Online	0	0.0
CreditCard	0	0.0

Fig 1: Checking Null values

6. Calculate the median experience value from the filtered data frame and save it in the variable "experience."
7. If the filtered data frame is empty, use the negative experience data frame to calculate the median experience value.
8. Substitute the absolute value of the median "experience" values for the negative experience.



Out[24]:

	count	mean	std	min	25%	50%	75%	max
ID	5000.0	2500.500000	1443.520003	1.0	1250.75	2500.5	3750.25	5000.0
Age	5000.0	45.338400	11.463166	23.0	35.00	45.0	55.00	67.0
Experience	5000.0	20.104600	11.467954	-3.0	10.00	20.0	30.00	43.0
Income	5000.0	73.774200	46.033729	8.0	39.00	64.0	98.00	224.0
ZIP Code	5000.0	93152.503000	2121.852197	9307.0	91911.00	93437.0	94608.00	96651.0
Family	5000.0	2.396400	1.147663	1.0	1.00	2.0	3.00	4.0
CCAvg	5000.0	1.937938	1.747659	0.0	0.70	1.5	2.50	10.0
Education	5000.0	1.881000	0.839869	1.0	1.00	2.0	3.00	3.0
Mortgage	5000.0	56.498800	101.713802	0.0	0.00	0.0	101.00	635.0
Personal Loan	5000.0	0.096000	0.294621	0.0	0.00	0.0	0.00	1.0
Securities Account	5000.0	0.104400	0.305809	0.0	0.00	0.0	0.00	1.0
CD Account	5000.0	0.060400	0.238250	0.0	0.00	0.0	0.00	1.0
Online	5000.0	0.596800	0.490589	0.0	0.00	1.0	1.00	1.0
CreditCard	5000.0	0.294000	0.455637	0.0	0.00	0.0	1.00	1.0

Fig 2: Descriptive Statistics

Exploratory Data Analysis

Now we will do exploratory data analysis and explore the data before creating a predictive model.

First, we create a scatter plot and corresponding regression line for experience with age, income, CC average, and mortgage. From the figure given below, we can observe that there is a direct correlation between age and experience whereas there is no linear correlation that can be watched for income, cc average, and mortgage with experience.

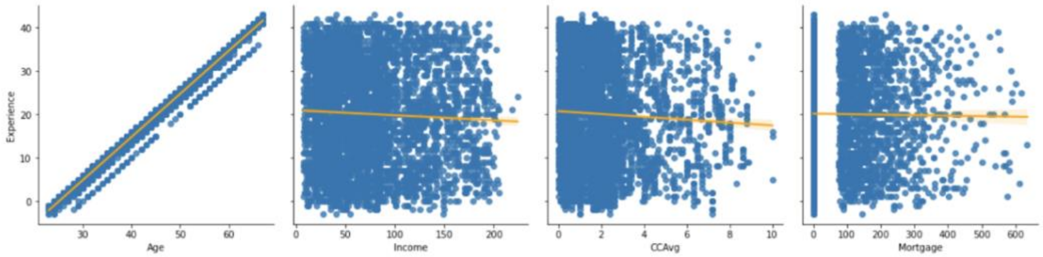


Fig 3: Scatter Plot

Now we plot the box plots to check the range and outliers. We observe that there are a lot of outliers in income, mortgage, and CC average variables. To get a more reliable model we will remove those outliers.

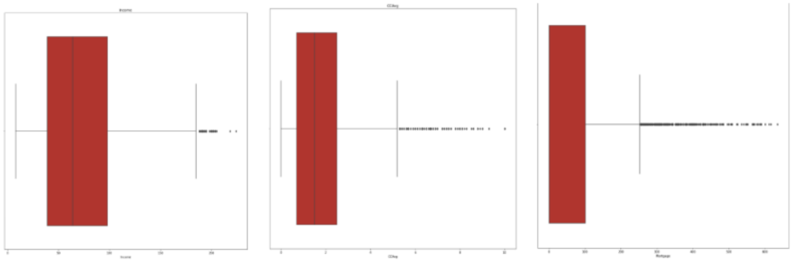


Fig 4 Box Plot

In the distribution plot of the mortgage, we can see that the data is right-skewed.

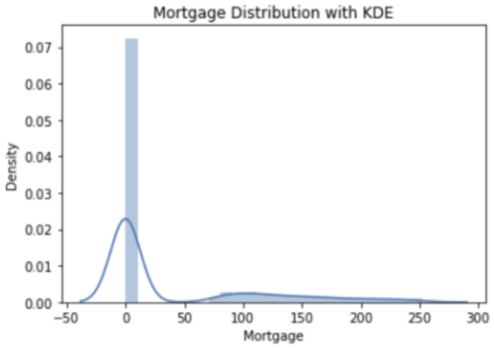


Fig 5: Distribution Plot of Mortgage

To find patterns, we have created count plots/histograms.

We can observe that most account holders do not have securities account with the bank. In terms of credit cards, whether a customer has a credit card or not they still have applied for a personal loan.

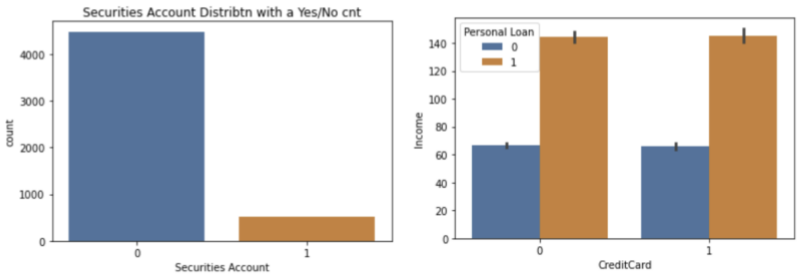


Fig 6: Count Plot

To understand if there is any dependency or correlation in independent features we create a correlation, Matrix. In the heat map given below, we observe that only age and experience have a high correlation as we also witnessed in the scatter plot above apart from that there is a weak correlation in the independent variables. So we do not have to treat data with multicollinearity.

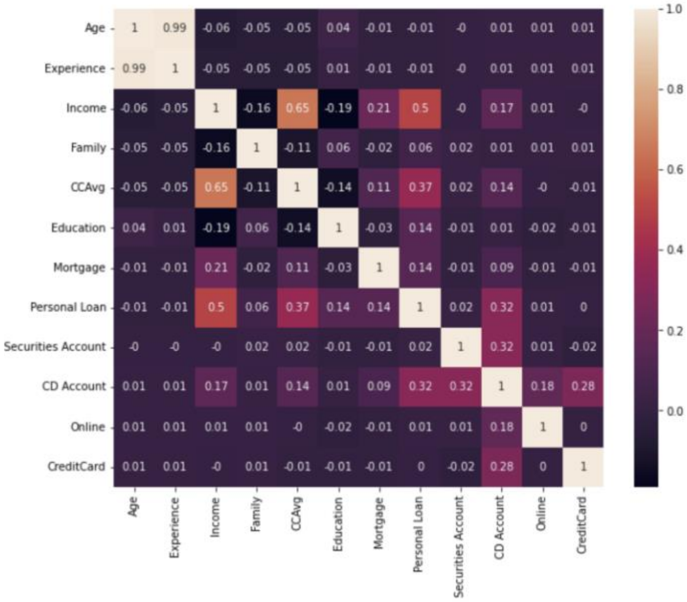


Fig 7: Correlation Matrix

Now, we compare education level and income based on the loan secured we observe that people with an advanced degree have the highest income and therefore correspond to the highest number of personal loans followed by people with a graduate degree and undergraduate degree respectively.

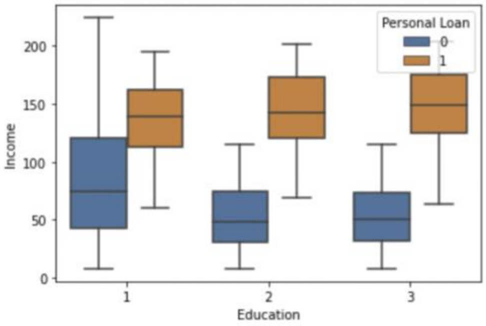


Fig 8 Box plot – Income/Education

Now we understand the users who have a certificate of deposit, CD account with the bank. We can see that most users do not have CD accounts with the bank. From the users who do not have a CD Account, most people have not secured a personal loan. However, most people who have a certificate of deposit account have secured a personal loan.

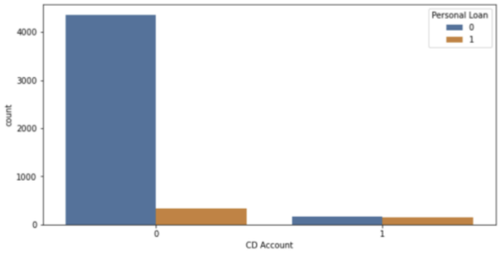


Fig 9(a): Histogram – CD Amount

The plot below shows people with higher credit cards monthly average spend are more likely to secure a personal loan.

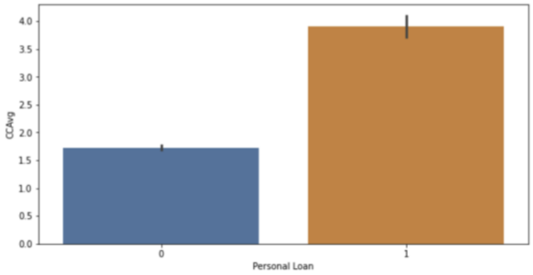


Fig 9(b): Histogram - Personal Loan

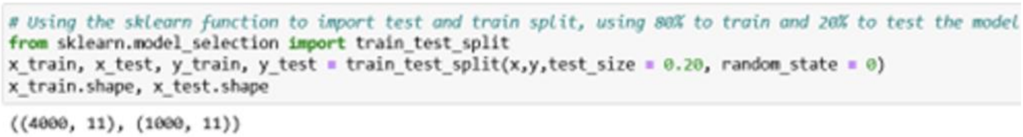
Now that we are done with the EDA and no further data processing is needed, we are ready to build the model.

PREDICTIVE MODELLING

In this section we will create linear regression model and follow the steps given below:

Part 1: Training and Testing Split

To make sure that the model is robust, the model is trained on the training set and validated on the testing set. Validating the model on unseen data ensures that the model would be able to perform equally good on any new set. Generally, as in this case, the split is done in the 80:20 ratio of training and testing set. The function 'train_test_spltit' from sklearn model selection can be used to split the data.



```
# Using the sklearn function to import test and train split, using 80% to train and 20% to test the model
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.20, random_state = 0)
x_train.shape, x_test.shape
```

```
((4000, 11), (1000, 11))
```

Fig 10: Train-Test Split

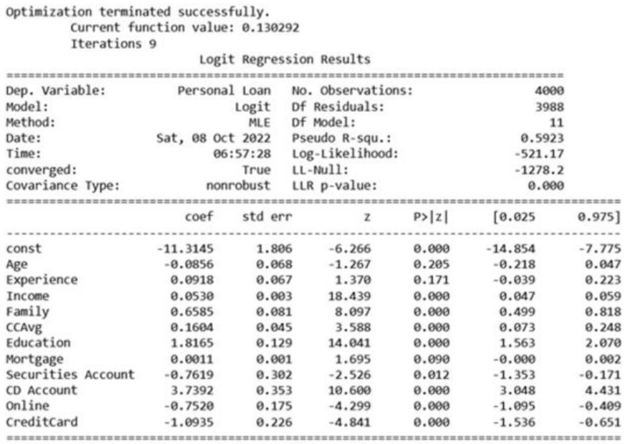
Observation:

The resulting training and testing set has 11 features and 4,000 observations in the training set, and 1,000 observations in the testing set.

Part 2: Fitting a Logistic Regression Model

Logistic regression is a statistical method that is used for building machine learning models where the dependent variable is dichotomous: i.e., binary. Logistic regression is used to describe data and the relationship between one dependent variable and one or more independent variables. The independent variables can be nominal, ordinal, or of interval type.

The name "logistic regression" is derived from the concept of the logistic function that it uses. The logistic function is also known as the sigmoid function. The value of this logistic function lies between zero and one. Logit function from statsmodels API can be used to train a logistic regression model. The advantage of using Statsmodels API is that it gives summary function that can be used to select the features.

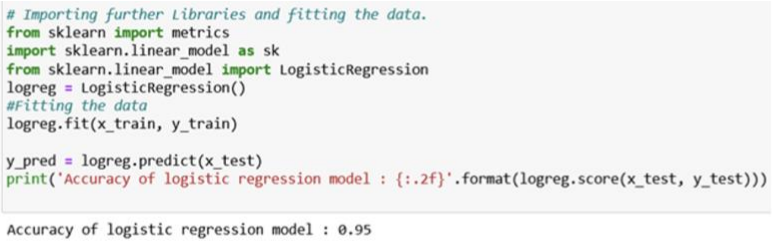


```
Optimization terminated successfully.
         Current function value: 0.130292
         Iterations 9
                           Logit Regression Results
==============================================================================
Dep. Variable:          Personal Loan   No. Observations:                 4000
Model:                          Logit   Df Residuals:                     3988
Method:                           MLE   Df Model:                           11
Date:                Sat, 08 Oct 2022   Pseudo R-squ.:                  0.5923
Time:                        06:57:28   Log-Likelihood:                -521.17
converged:                       True   LL-Null:                       -1278.2
Covariance Type:            nonrobust   LLR p-value:                     0.000
======================================================================================
                         coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------------
const                -11.3145      1.806     -6.266      0.000     -14.854      -7.775
Age                   -0.0856      0.068     -1.267      0.205      -0.218       0.047
Experience             0.0918      0.067      1.370      0.171      -0.039       0.223
Income                 0.0530      0.003     18.439      0.000       0.047       0.059
Family                 0.6585      0.081      8.097      0.000       0.499       0.818
CCAvg                  0.1604      0.045      3.588      0.000       0.073       0.248
Education              1.8165      0.129     14.041      0.000       1.563       2.070
Mortgage               0.0011      0.001      1.695      0.090      -0.000       0.002
Securities Account    -0.7619      0.302     -2.526      0.012      -1.353      -0.171
CD Account             3.7392      0.353     10.600      0.000       3.048       4.431
Online                -0.7520      0.175     -4.299      0.000      -1.095      -0.409
CreditCard            -1.0935      0.226     -4.841      0.000      -1.536      -0.651
======================================================================================
```

Fig 11: Logistic Summary

Part 3: Building a Linear Regression

Once the logistic regression model is finalized, it can be trained in Sklearn package, and make it available for use or deployment.



```
# Importing further Libraries and fitting the data.
from sklearn import metrics
import sklearn.linear_model as sk
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
#Fitting the data
logreg.fit(x_train, y_train)

y_pred = logreg.predict(x_test)
print('Accuracy of logistic regression model : {:.2f}'.format(logreg.score(x_test, y_test)))
```

```
Accuracy of logistic regression model : 0.95
```

Fig 12: Regression Fitting

Observation:

The accuracy of trained logistic regression model is 95%.

PREDICTION AND MODEL EVALUATION

Once the model has been trained, it is important to evaluate the model before putting it in action. There are three metrics that are used to evaluate a classification model like logistic regression – accuracy, precision, and recall, using a confusion matrix.

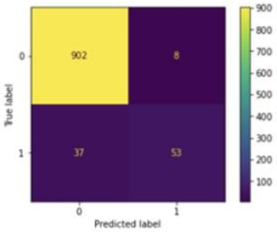
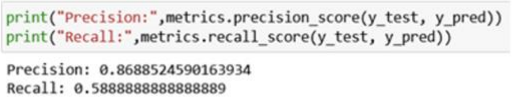


Fig 13: Confusion Matrix

Observation:

Accuracy of the mode is 95%, precision of the model is 86%, and recall of the model is ~59%.



```
print("Precision:",metrics.precision_score(y_test, y_pred))
print("Recall:",metrics.recall_score(y_test, y_pred))

Precision: 0.8688524590163934
Recall: 0.5888888888888889
```

Fig 14: Precision and Recall

Questions

1. **What were the three most significant variables?**

   Ans: The three most significant Variables are Education, Cash Credit Account, and CD account.

2. **Of those three, which had the most negative influence on loan acceptance?**

   Ans: Credit Card Seems to be having a negative effect on the Loan Acceptance.

3. **How accurate was the model overall and what was the precision rate?**

   Ans: The overall accuracy of the model was 95%. The precision is around 86% and the recall is around 58%

Conclusion

After analyzing the bank dataset with logistic regression, I discovered that the target variables are Mortgage, Income, CCAvg, and Personal Loan. The overall model's accuracy is 95%, and the precision rate is 93%. Except for four variables, the p-values for most variables are less than 0.05.

References

[1.] Banoula, M. (2022, September 13). *An Introduction to Logistic Regression in Python*. Simplilearn.com. Retrieved October 10, 2022, from https://www.simplilearn.com/tutorials/machine-learning-tutorial/logistic-regression-in-python

[2.] Mulani, S. (2020, October 19). *Detection and Removal of Outliers in Python - An Easy-to Understand Guide*. AskPython. Retrieved October 10, 2022, from https://www.askpython.com/python/examples/detection-removal-outliers-in-python

[3.] Mahmood, M. S. (2020, December 26). *https://towardsdatascience.com/practical-implementation-of-outlier-detection-in-python*. Retrieved October 2, 2022, from Towards Data Science.

[4.] GeeksforGeeks. (2022a, August 23). Logistic Regression using Statsmodels. Retrieved October 8, 2022, from https://www.geeksforgeeks.org/logistic-regression-using-statsmodels/

APPENDIX

Importing Libraries

```
# Loading Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Reading the file

```
#Converted the Excel to CSV format for easy readibility.
df=pd.read_csv("bank_personal_loan.csv")


df


# Dimensions of the Data set
df.shape


# Information about the data set
df.info()
# The data set has majorly values in the Integer data type
```

Data Cleaning

Checking Null Value Percentage

```
def missing_check(df):
    total = df.isnull().sum().sort_values(ascending=False) # Total number of null values
    percent = (df.isnull().sum()/df.isnull().count()).sort_values(ascending=False) # Percentage of values that are null
    missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent']) # Putting the above two together
    return missing_data

missing_check(df)


#Statistical Modelling
```

```
df.describe().transpose()
# We can see that experience is coming negative.
#We can try to remove the negative value so that it does not cause a left skew


negExp = df.Experience < 0
negExp.value_counts()


df[df['Experience'] < 0]['Experience'].value_counts()


df_positive_experience = df[df['Experience'] > 0]
df_negative_experience = df[df['Experience'] < 0]
negative_experience_id_list = df_negative_experience['ID'].tolist()

for id in negative_experience_id_list:
    age = df.loc[np.where(df['ID']==id)]['Age'].tolist()[0]
    education = df.loc[np.where(df['ID']==id)]['Education'].tolist()[0]
    positive_experience_filtered = df_positive_experience[(df_positive_experience['Age'] == age)
&
                                            (df_positive_experience['Education'] == education)]
    if positive_experience_filtered.empty:
        negative_experience_filtered = df_negative_experience[(df_negative_experience['Age'] ==
age) &
                                            (df_negative_experience['Education'] == education)]
        experience = round(negative_experience_filtered['Experience'].median())
    else:
        experience = round(positive_experience_filtered['Experience'].median())
    df.loc[df.ID == id, 'Experience'] = abs(experience)


df[df['Experience'] < 0]['Experience'].count()


df.Experience.describe()
```

Checking Duplicates

```
#Checking for duplicated data
df[df.duplicated()== True]
# The data does not seem to be having duplicated values


# Dropping the first column of the data set
df.drop(['ID','ZIP Code'],axis=1,inplace=True)
# Removing Zip Code and Id since they are not relevant here
```

Checking Outliers

```
# Plotting Box Plots for various columns to understand the outliers
for feature in df:
    sns.boxplot(df[feature],color='red')
    plt.title(feature)
    plt.figure(figsize=(15,15))

    # The Median Experiance is 20 years
    # The income seems to have some outliers
    #The CC avg also seems to have some outliers


#Removing the outliers for the Income Parameter
for x in ['Income','Mortgage','CCAvg']:
    q75,q25 = np.percentile(df.loc[:,x],[75,25])
    intr_qr = q75-q25

    max = q75+(1.5*intr_qr)
    min = q25-(1.5*intr_qr)

    df.loc[df[x] < min,x] = np.nan
    df.loc[df[x] > max,x] = np.nan




# Checking the values for the outliers
df.isnull().sum()


#Dropping the Null Values
df = df.dropna(axis = 0)


# Checking the Sum to verify that Outliers have reduced
df.isnull().sum()


sns.distplot(df['Mortgage'])
plt.title('Mortgage Distribution with KDE');


#When we check the skew we find that the Bank Data has a slight Left Skew

df['Mortgage'].skew()
```

Scatter Plot

```
#Plotting scatter Plot between Age of the customer and the Income
plt.scatter(df.Age, df.Income)
```

```
#The graph below does not give very conclusive results
```

```
# checking the -ve values
Removing_Negative_Experience = df.Experience < 0
Removing_Negative_Experience.value_counts()
#
```

```
#Checking the distribution of the data for negative values in the Experience Column
df[df['Experience'] < 0]['Experience'].value_counts()
# The -ve experiences may cause some biases in the analysis.
```

Count Plot

```
# Plotting the Securities Account with the Count
sns.countplot(df['Securities Account'])
plt.title('Securities Account Distribtn with a Yes/No cnt');

#The graph shows that a lot of people don't hold the securities account
```

Bar Plot for understanding relation between credit card and income

```
# Reading the Bar plot for credit card and Income
sns.barplot(x='CreditCard',y='Income', data=df,
            hue='Personal Loan')

plt.show()
```

Correlation amongst Variables

```
#Plotting a correlation Map
plt.figure(figsize=[10,8])
matrix = df.corr().round(2)
sns.heatmap(matrix, annot=True)
plt.show()
#There is a very strong correlation between Age and Experience
#CC Avg and income have a moderate correlation to the tune of 0.65
#Personal Loan is our target variable here
```

```
# Creating a box plot with Education and Income as the main parameters
sns.boxplot(x='Education', y='Income', hue='Personal Loan', data=df);
plt.figure(figsize=(10,5))

# Customers with graduate degrees have a  higher propensity for taking the loan.
```

```
#Count Plot between CD Account and Personal Loan
```

```
plt.figure(figsize=(10,5))
sns.countplot(x='CD Account', data=df, hue='Personal Loan');
#Customers who have a CD account have also taken Personal Loan
#Customers who have not taken CD account do not gravitate towards PL, in majority do not opt for PL


#Count Plot with Personal Loan and Cash Credit Account
plt.figure(figsize=(10,5))
sns.barplot(x='Personal Loan', y='CCAvg', data=df);
#People who had taken Short term working capital  were also more susceptible to taking Personal Loans


#Declaring the value of X and Y for training and testing the Model
y = df[['Personal Loan']]  #Predictor Parameter
x=df.drop('Personal Loan',axis=1) # Target Column


# Using the sklearn function to import test and train split, using 80% to train and 20% to test the model
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.20, random_state = 0)
x_train.shape, x_test.shape


#Summary of Logistic Regression Model
import statsmodels.api as sm
Xlog2 = sm.add_constant(x_train)
logr_model = sm.Logit(y_train, Xlog2)
logr_fit = logr_model.fit()
print(logr_fit.summary())
# Logitic regression uses statsmodels.api
# From the table generated below we see that Education, CC Avg,and CD Account are three importamt variables for this study
# Credit card seems to have a negative effect on personal loans .
```

Fitting Logistic Regression Model

```
# Importing further Libraries and fitting the data.
from sklearn import metrics
import sklearn.linear_model as sk
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
#Fitting the data
logreg.fit(x_train, y_train)

y_pred = logreg.predict(x_test)
print('Accuracy of logistic regression model : {:.2f}'.format(logreg.score(x_test, y_test)))
```

Confusion Matrix

```
#Printing the confusion Matrix

from sklearn.metrics import confusion_matrix
confusion_matrix = confusion_matrix(y_test, y_pred)
print(confusion_matrix)


#Importing confusion Matrix
from sklearn.metrics import plot_confusion_matrix
from sklearn.linear_model import LogisticRegression

logistic_regression= LogisticRegression()
model=logistic_regression.fit(x_train,y_train)
plot_confusion_matrix(logistic_regression, x_test, y_test)
plt.show()


print("Precision:",metrics.precision_score(y_test, y_pred))
print("Recall:",metrics.recall_score(y_test, y_pred))


#The Precision of the model comes out to be 96%


#The Recall comes out to be 44%


# GeeksforGeeks. (2022a, August 23). Logistic Regression using Statsmodels. Retrieved October 8, 2022, from https://www.geeksforgeeks.org/logistic-regression-using-statsmodels/


#Conclusion
#The target parameter- Personal Loan has good correlation with Income, CC Avg, Education, Mortgage and CD Account
```