Szczepan “Ray” Sadowski

Problem set 9

Python Code

Exercise 1:

# Impute missing values (categorical variables)

train.Gender.fillna("Female",inplace=True)

train.Married.fillna("No",inplace=True)

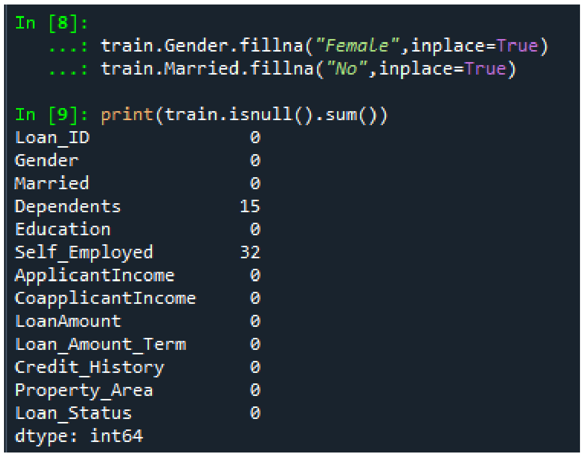
Exercise 2:

# Impute missing values (numerical variables)

train.fillna(train.median(),inplace=True)

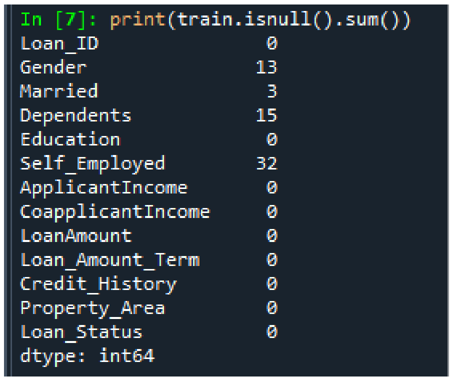
Output

Exercise 1:



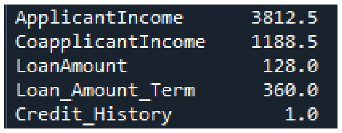
```
In [8]:
   ...: train.Gender.fillna("Female",inplace=True)
   ...: train.Married.fillna("No",inplace=True)

In [9]: print(train.isnull().sum())
Loan_ID              0
Gender               0
Married              0
Dependents          15
Education            0
Self_Employed       32
ApplicantIncome      0
CoapplicantIncome    0
LoanAmount           0
Loan_Amount_Term     0
Credit_History       0
Property_Area        0
Loan_Status          0
dtype: int64
```

Exercise 2:



```
In [7]: print(train.isnull().sum())
Loan_ID              0
Gender              13
Married              3
Dependents          15
Education            0
Self_Employed       32
ApplicantIncome      0
CoapplicantIncome    0
LoanAmount           0
Loan_Amount_Term     0
Credit_History       0
Property_Area        0
Loan_Status          0
dtype: int64
```

train.isnull().sum()



```
ApplicantIncome      3812.5
CoapplicantIncome    1188.5
LoanAmount            128.0
Loan_Amount_Term      360.0
Credit_History          1.0
```

train.median()

Szczepan “Ray” Sadowski

Problem set 9

Python Code

Exercise 3:

```
#Split train data for cross
validation
from sklearn.model_selection import
train_test_split
x_train,x_cv,y_train,y_cv =
train_test_split(X,y,test_size=0.3)
```

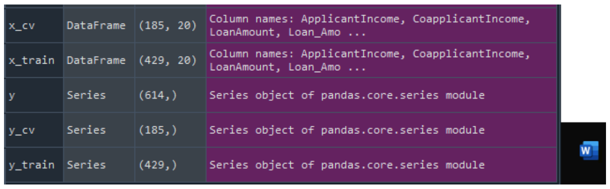
Exercise 4:

```
#Predict values using kNN

pred_test=kNN.predict(test)

#Write test results in csv file

predictions=pd.DataFrame(pred_test,columns=['pre
dictions']).to_csv('knn_prediciton.csv')
```

```
np.unique(pred_test, return_counts=True)
```

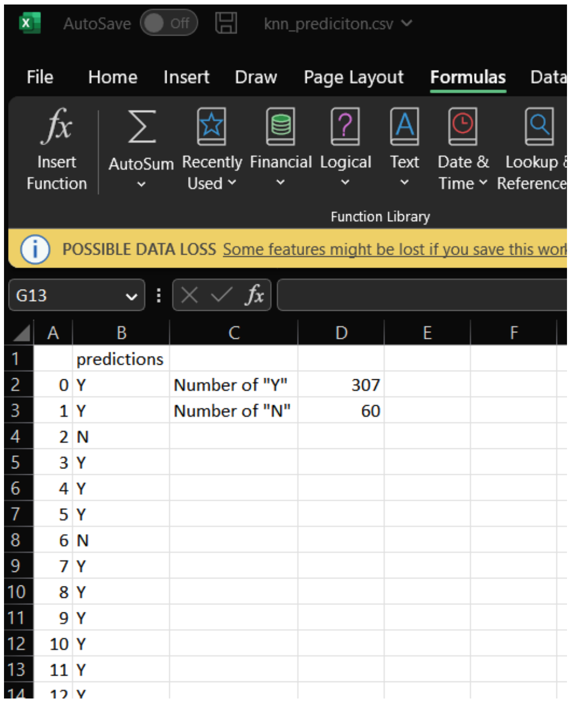
Output

Exercise 3:



x_cv	DataFrame	(185, 20)	Column names: ApplicantIncome, CoapplicantIncome, LoanAmount, Loan_Amo ...
x_train	DataFrame	(429, 20)	Column names: ApplicantIncome, CoapplicantIncome, LoanAmount, Loan_Amo ...
y	Series	(614,)	Series object of pandas.core.series module
y_cv	Series	(185,)	Series object of pandas.core.series module
y_train	Series	(429,)	Series object of pandas.core.series module

Exercise 4:



	A	B	C	D	E	F
1		predictions				
2	0	Y	Number of "Y"	307		
3	1	Y	Number of "N"	60		
4	2	N				
5	3	Y				
6	4	Y				
7	5	Y				
8	6	N				
9	7	Y				
10	8	Y				
11	9	Y				
12	10	Y				
13	11	Y				
14	12	Y				

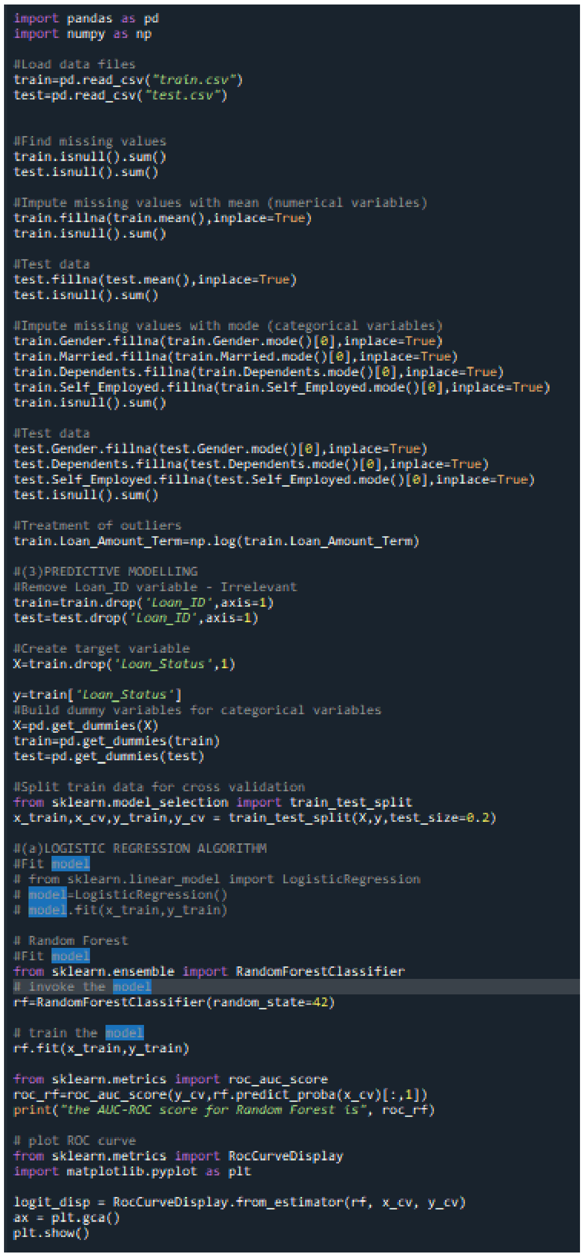
```
(array(['N', 'Y'], dtype=object), array([ 60, 307],
dtype=int64))
```

Szczepan “Ray” Sadowski

Problem set 9

Python Code

Exercise 5:

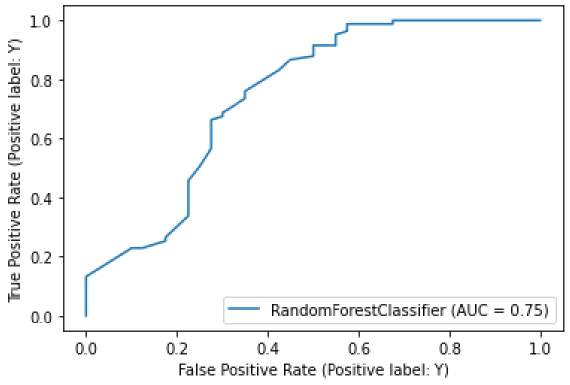


```
import pandas as pd
import numpy as np

#Load data files
train=pd.read_csv("train.csv")
test=pd.read_csv("test.csv")


#Find missing values
train.isnull().sum()
test.isnull().sum()

#Impute missing values with mean (numerical variables)
train.fillna(train.mean(),inplace=True)
train.isnull().sum()

#Test data
test.fillna(test.mean(),inplace=True)
test.isnull().sum()

#Impute missing values with mode (categorical variables)
train.Gender.fillna(train.Gender.mode()[0],inplace=True)
train.Married.fillna(train.Married.mode()[0],inplace=True)
train.Dependents.fillna(train.Dependents.mode()[0],inplace=True)
train.Self_Employed.fillna(train.Self_Employed.mode()[0],inplace=True)
train.isnull().sum()

#Test data
test.Gender.fillna(test.Gender.mode()[0],inplace=True)
test.Dependents.fillna(test.Dependents.mode()[0],inplace=True)
test.Self_Employed.fillna(test.Self_Employed.mode()[0],inplace=True)
test.isnull().sum()

#Treatment of outliers
train.Loan_Amount_Term=np.log(train.Loan_Amount_Term)

#(3)PREDICTIVE MODELLING
#Remove Loan_ID variable - Irrelevant
train=train.drop('Loan_ID',axis=1)
test=test.drop('Loan_ID',axis=1)

#Create target variable
X=train.drop('Loan_Status',1)

y=train['Loan_Status']
#Build dummy variables for categorical variables
X=pd.get_dummies(X)
train=pd.get_dummies(train)
test=pd.get_dummies(test)

#Split train data for cross validation
from sklearn.model_selection import train_test_split
x_train,x_cv,y_train,y_cv = train_test_split(X,y,test_size=0.2)

#(a)LOGISTIC REGRESSION ALGORITHM
#Fit model
# from sklearn.linear_model import LogisticRegression
# model=LogisticRegression()
# model.fit(x_train,y_train)

# Random Forest
#Fit model
from sklearn.ensemble import RandomForestClassifier
# invoke the model
rf=RandomForestClassifier(random_state=42)

# train the model
rf.fit(x_train,y_train)

from sklearn.metrics import roc_auc_score
roc_rf=roc_auc_score(y_cv,rf.predict_proba(x_cv)[:,1])
print("the AUC-ROC score for Random Forest is", roc_rf)

# plot ROC curve
from sklearn.metrics import RocCurveDisplay
import matplotlib.pyplot as plt

logit_disp = RocCurveDisplay.from_estimator(rf, x_cv, y_cv)
ax = plt.gca()
plt.show()
```

Output

Exercise 5:

```
the AUC-ROC score for Random Forest is
0.7466867469879518
```


True Positive Rate (Positive label: Y)
False Positive Rate (Positive label: Y)
RandomForestClassifier (AUC = 0.75)