FINAL PRESENTATION

Internship Batch: LISUM09

Data Glacier Virtual Internship 2022

Submitted by: Yihsuan Sun

OUTLINE

- Problem Statement
- Datasets Information
- EDA (Exploratory data analysis)
- Model Selection and Model Building

PROBLEM DESCRIPTION & BUSINESS UNDERSTANDING

XYZ Credit Union, located in Latin America, does well in selling banking products such as: credit cards, deposit accounts, retirement accounts, safe deposit boxes, etc. However, after statistics, they found that their existing customers basically only buy one product, which means that the bank does not perform well in cross-selling. So XYZ Credit Union wants analysts to build models such as marketing models through machine learning to solve their problems.

DATASETS INFORMATION

Train.csv details

Total number of observations	13,647,309
Total number of files	1
Total number of features	48
Base format of the file	csv
Size of the data	2.13 GB

Test.csv details

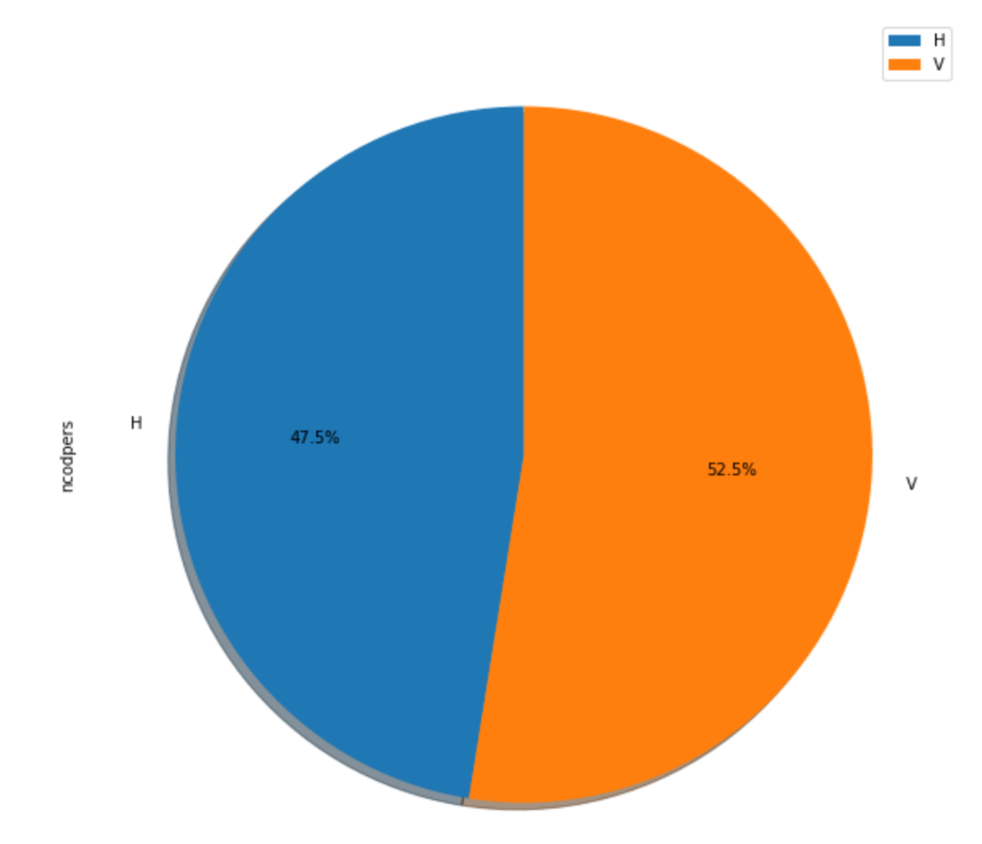
Total number of observations	929,615
Total number of files	1
Total number of features	24
Base format of the file	csv
Size of the data	105 MB

EDA (EXPLORATORY DATA ANALYSIS)

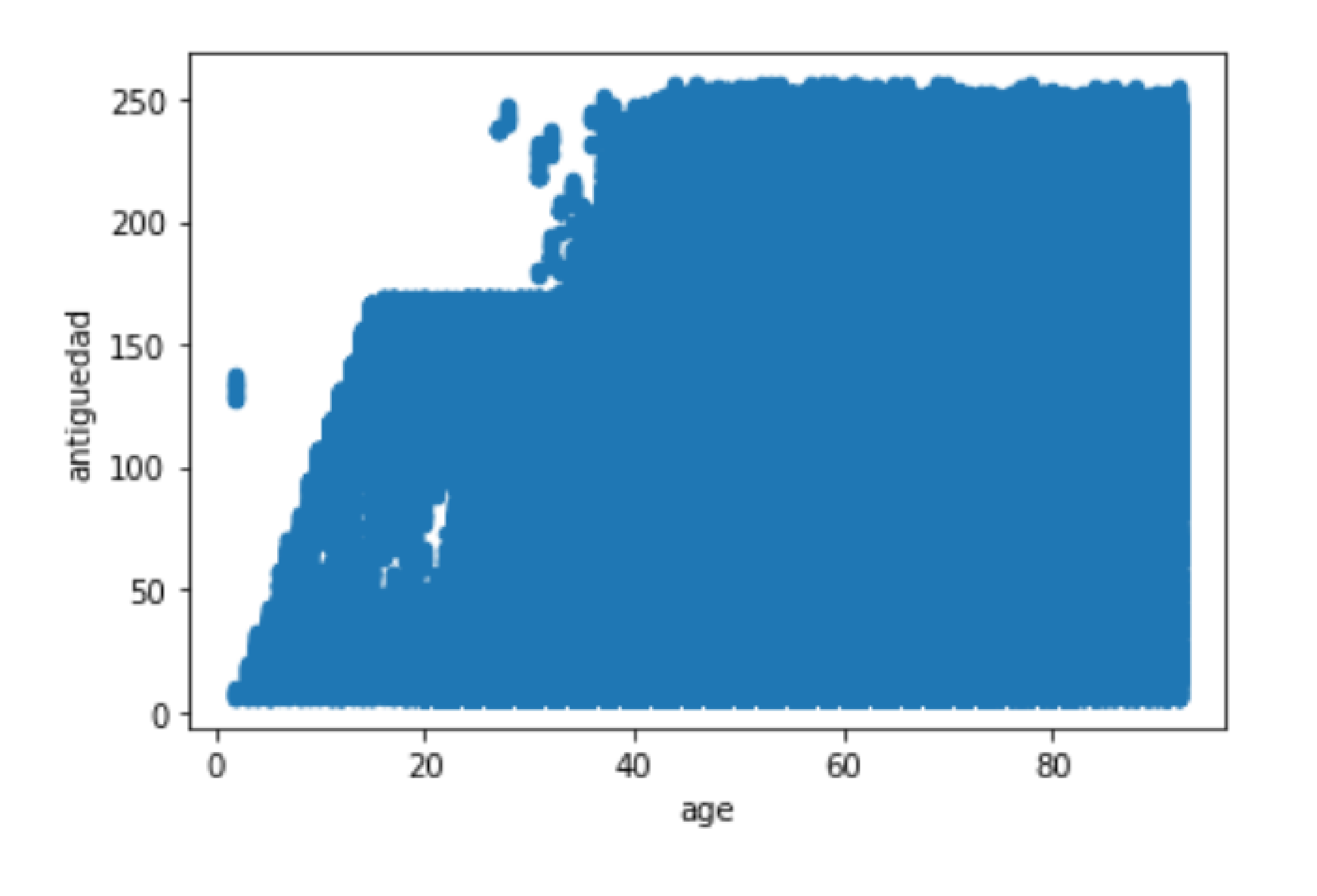
EDA

CUSTOMER'S SEX - COLUMN NAME : SEXO

sexo	ncodpers
H	3421063
V	3784510



CUSTOMER'S AGE VS. CUSTOMER SENIORITY (IN MONTHS)

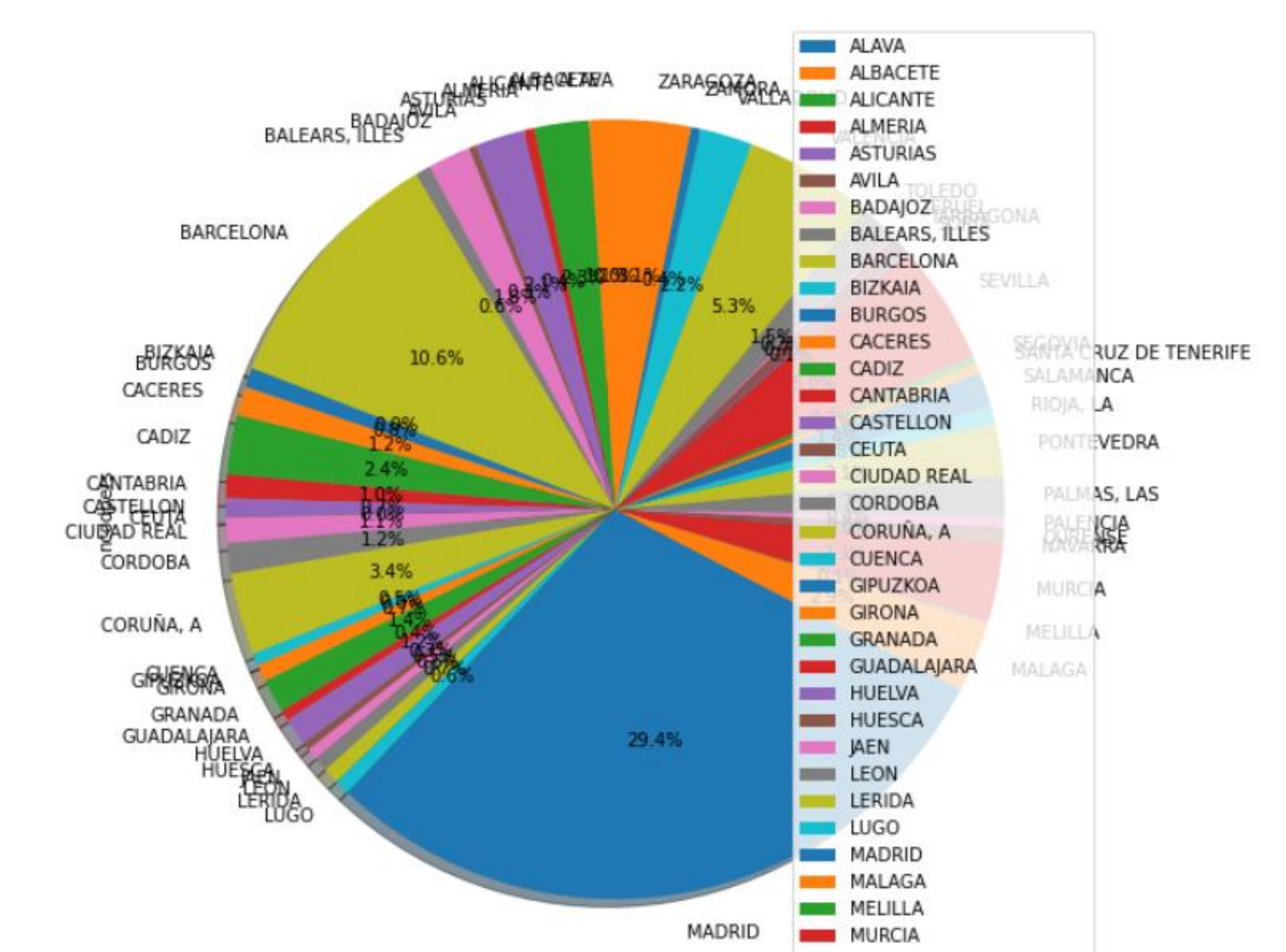


CUSTOMER'S PROVINCE NAME (COUNT)

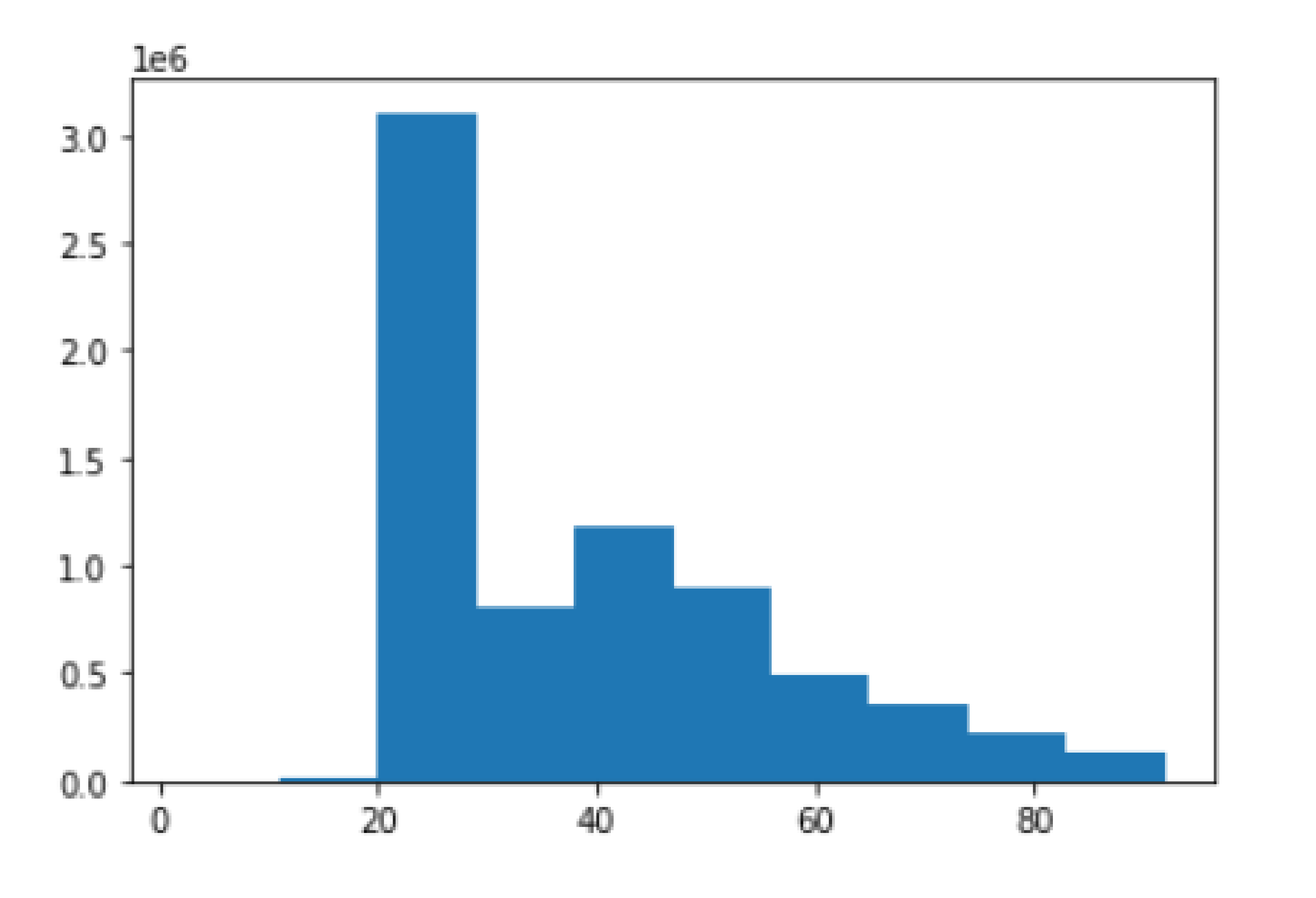
ALAVA	5
ALBACETE	77945
ALICANTE	162647
ALMERIA	30581
ASTURIAS	148461
AVILA	22670
BADAJOZ	128064
BALEARS, ILLES	45551
BARCELONA	760799
BIZKAIA	50
BURGOS	58629
CACERES	85989
CADIZ	175633
CANTABRIA	72042
CASTELLON	53935
CEUTA	2927
CIUDAD REAL	76075
CORDOBA	89996
CORUÑA, A	244916
CUENCA	36099
GIPUZKOA	22
GIRONA	50819
GRANADA	100885
GUADALAJARA	32090
HUELVA	83041
HUESCA	23183

GRANADA	100885
GUADALAJARA	32090
HUELVA	83041
HUESCA	23183
JAEN	34301
LEON	42448
LERIDA	49431
LUGO	45581
MADRID	2119228
MALAGA	210317
MELILLA	4824
MURCIA	236278
NAVARRA	41
OURENSE	46517
PALENCIA	30866
PALMAS, LAS	121606
PONTEVEDRA	153945
RIOJA, LA	54366
SALAMANCA	101863
SANTA CRUZ DE TENERIFE	29166
SEGOVIA	23062
SEVILLA	363568
SORIA	9418
TARRAGONA	49443
TERUEL	13735
TOLEDO	108268
VALENCIA	384133
VALLADOLID	155422
ZAMORA	29474
ZARAGOZA	225218

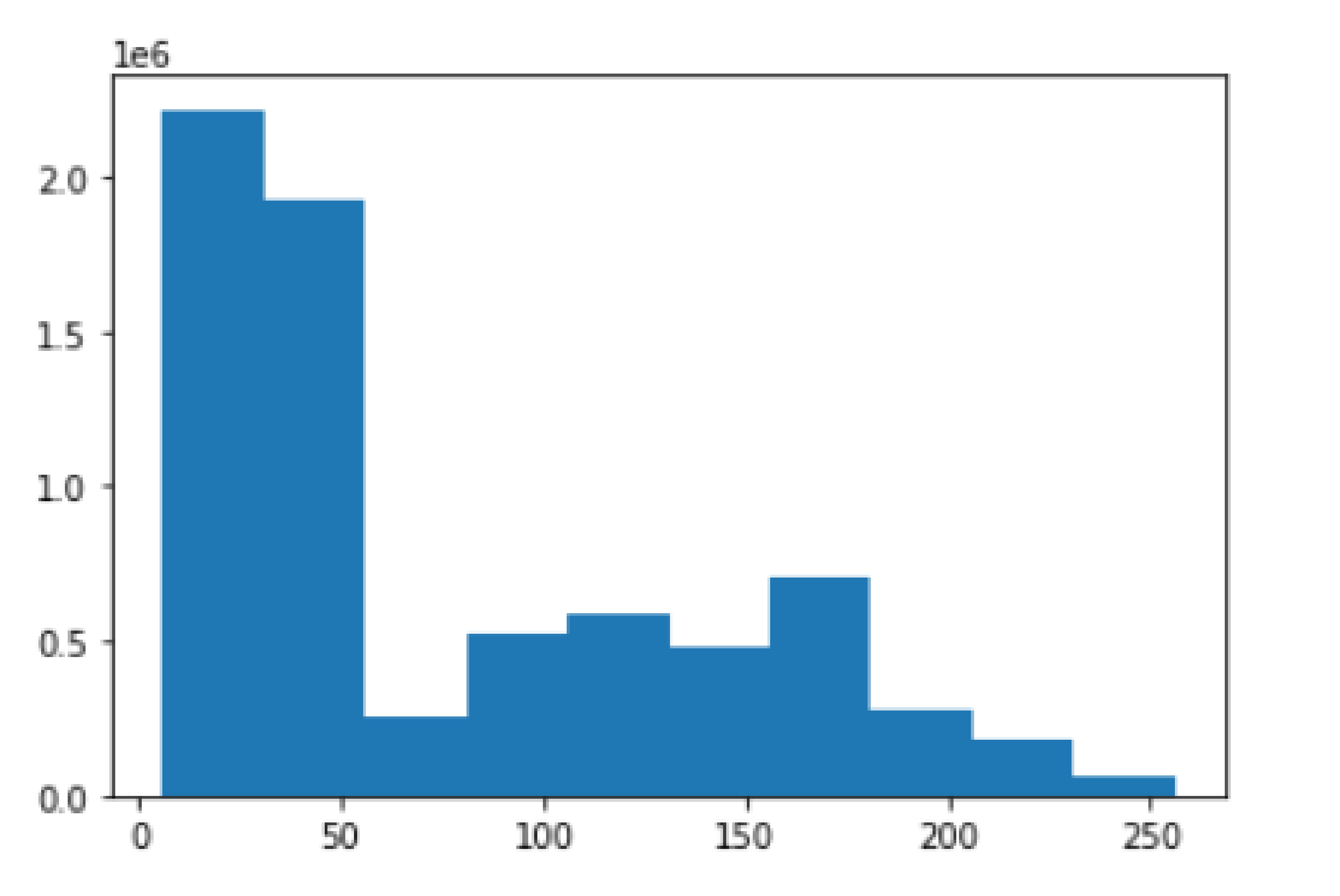
CUSTOMER'S PROVINCE NAME (PIE CHART)



CUSTOMER’S AGE

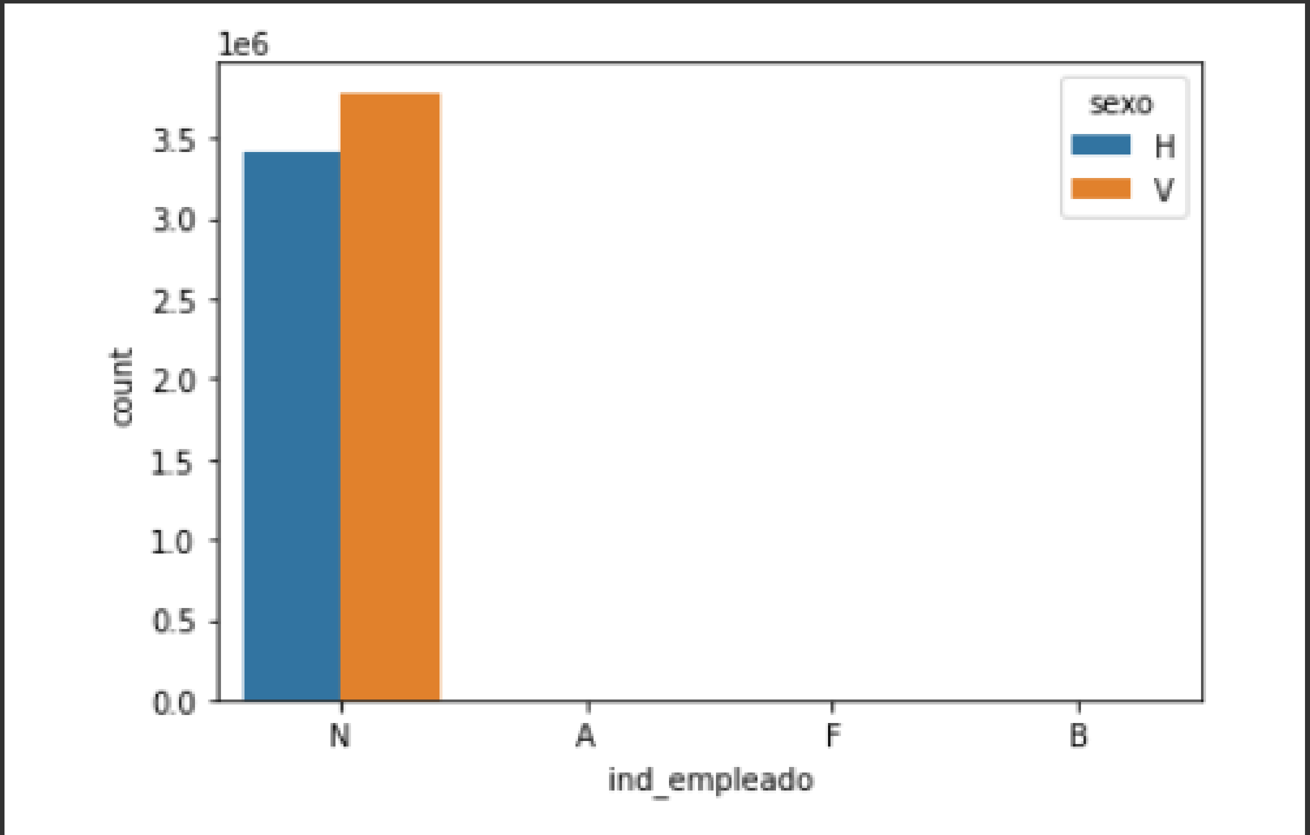


CUSTOMER SENIORITY (IN MONTHS)



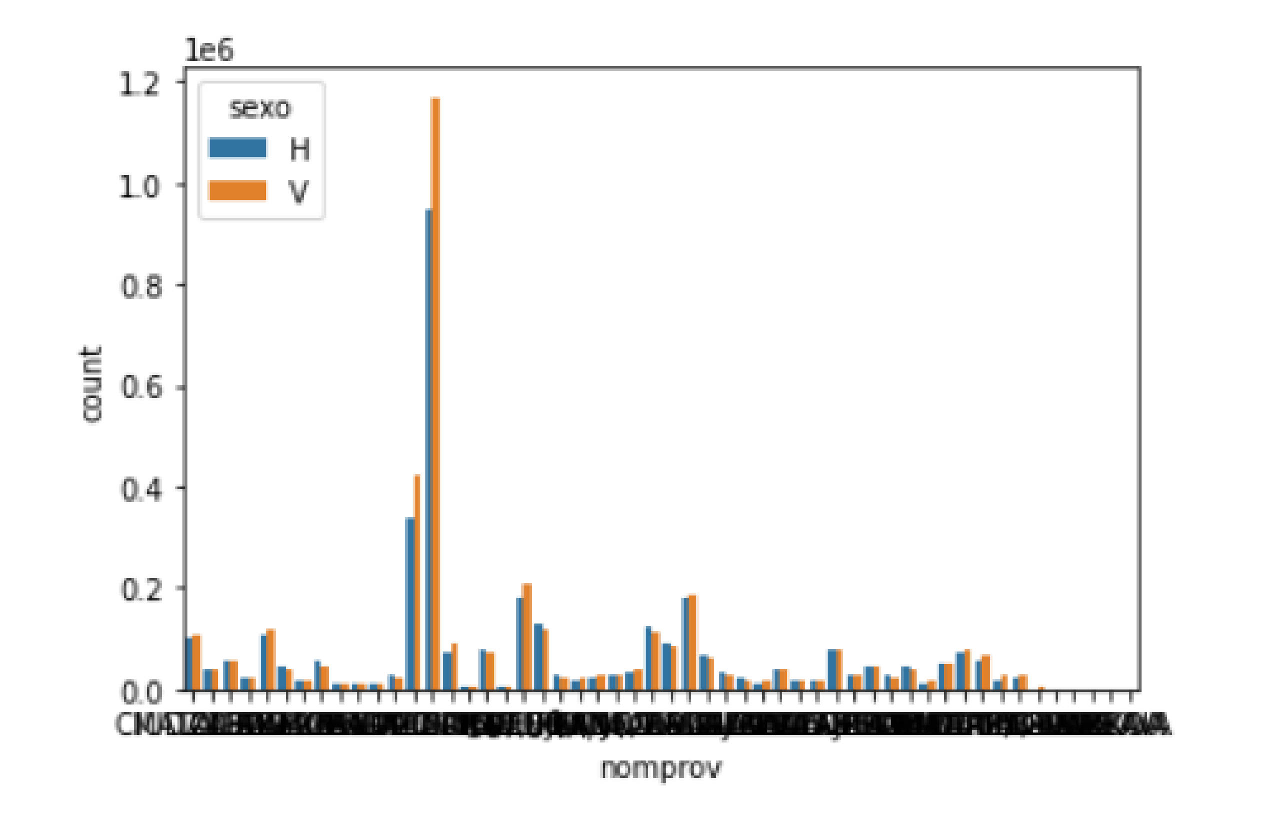
EMPLOYEE INDEX VS. CUSTOMER'S SEX

- Employee index: A active, B ex employed, F filial, N not employee, P pasive



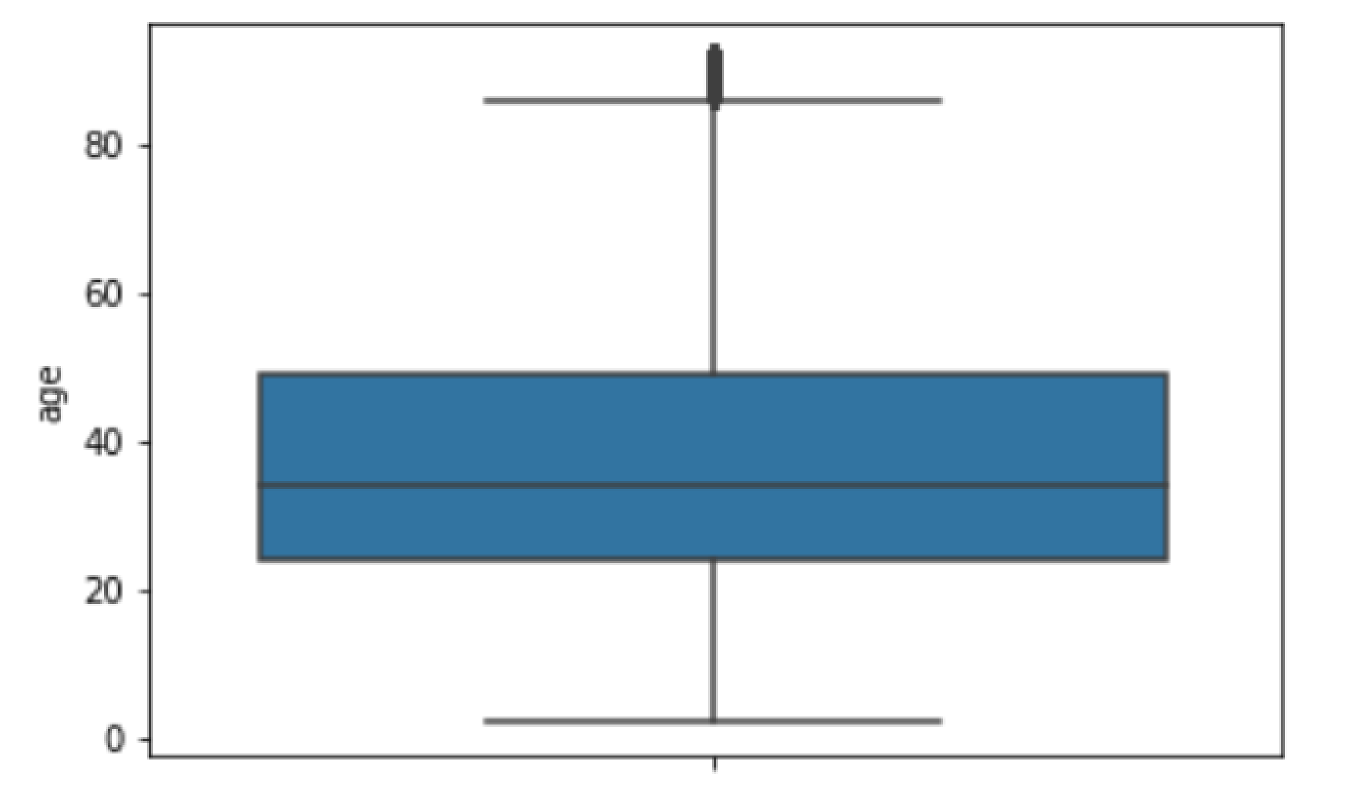
CUSTOMER'S PROVINCE NAME VS. CUSTOMER'S SEX

Max count: MADRID



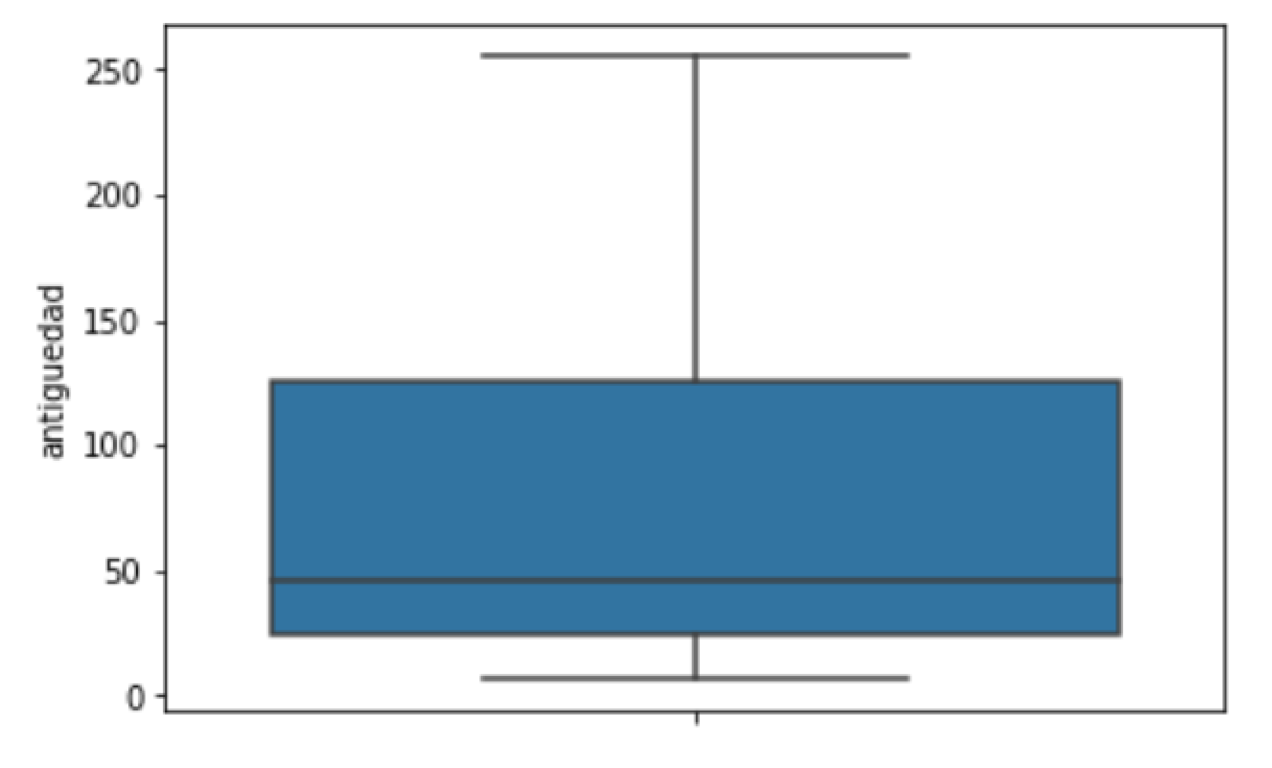
BOXPLOT - AGE

Mean: About 35

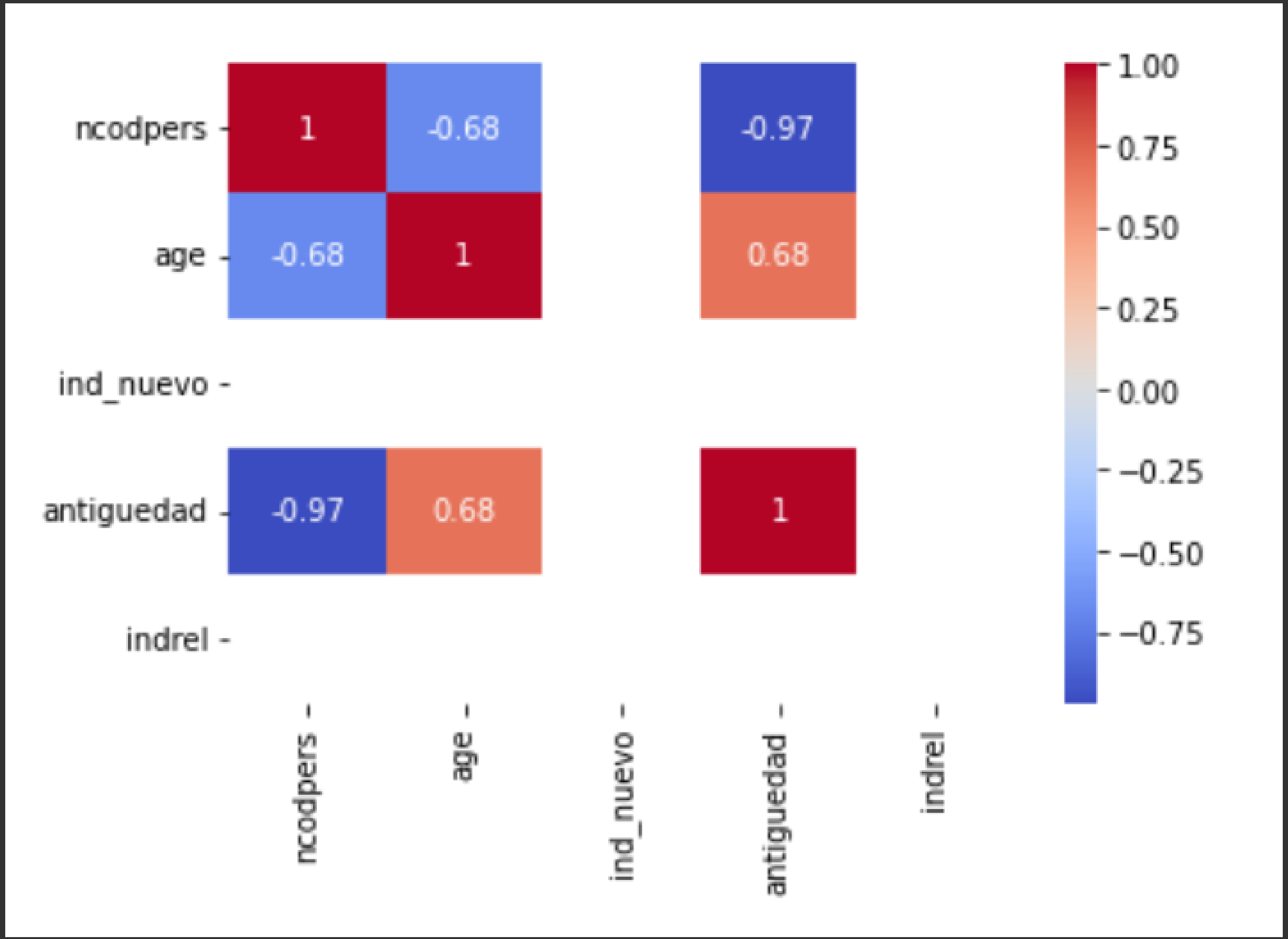


BOXPLOT - CUSTOMER SENIORITY (IN MONTHS)

Mean: About 50



HEATMAP


ncodpers
age
ind_nuevo
antiguedad
indrel
1
-0.68
-0.97
-0.68
1
0.68
-0.97
0.68
1
ncodpers
age
ind_nuevo
antiguedad
indrel
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

MODEL SELECTION AND MODEL BUILDING

Modeling

CHANGE DATA TYPES

Change data types of "age" and "antiguedad" to numeric

```
df["age"] = pd.to_numeric(df["age"])
df["antiguedad"] = pd.to_numeric(df["antiguedad"])
df_test["age"] = pd.to_numeric(df_test["age"])
df_test["antiguedad"] = pd.to_numeric(df_test["antiguedad"])
df_test["renta"] = pd.to_numeric(df_test["renta"], errors='coerce')
df_test = df_test[['sexo', 'ind_nuevo', 'ind_actividad_cliente', 'renta']]
df_test = df_test.dropna()
```

CHOOSE FEATURES TO USE IN MODELING

features= ['ind_nuevo', 'ind_actividad_cliente', 'renta', 'sexo_H']

	ind_nuevo	ind_actividad_cliente	renta	sum	sexo_H	sexo_V
0	0.0	1.0	87218.10	0.0	1	0
1	0.0	0.0	35548.74	0.0	0	1
2	0.0	0.0	122179.11	0.0	0	1
3	0.0	0.0	119775.54	0.0	1	0
5	0.0	0.0	22220.04	0.0	1	0
...	...	...	...	...	...	...
13647302	0.0	0.0	73134.81	0.0	0	1
13647303	0.0	0.0	50945.25	0.0	0	1
13647304	0.0	0.0	43912.17	0.0	0	1
13647305	0.0	0.0	23334.99	0.0	0	1
13647307	0.0	0.0	199592.82	0.0	1	0

10795392 rows × 6 columns

SPLIT DATA TO 80% TRAIN AND 20% TEST

```
#split data to 80% train and 20% test
X_train, X_test, Y_train, Y_test = train_test_split(df2[features],df2['sum'], test_size = 0.2)
X_train.shape,X_test.shape
```

```
((8636313, 4), (2159079, 4))
```

CALCULATE MODEL PERFORMANCE

```
In [34]:
#calculate model performance
def performance_met(model,X_train,Y_train,X_test,Y_test):
    acc_train=accuracy_score(Y_train, model.predict(X_train))
    f1_train=f1_score(Y_train, model.predict(X_train))
    acc_test=accuracy_score(Y_test, model.predict(X_test))
    f1_test=f1_score(Y_test, model.predict(X_test))
    print("train score: accuracy:{} f1:{}".format(acc_train,f1_train))
    print("test score: accuracy:{} f1:{}".format(acc_test,f1_test))
```

LINEAR MODEL

```
#linear model
model_linear = LogisticRegression()
model_linear.fit(X_train,Y_train)
performance_met(model_linear,X_train,Y_train,X_test,Y_test)
```

```
train score: accuracy:0.7277077614023484 f1:0.0
test score: accuracy:0.727239253403882 f1:0.0
```

ENSEMBLE MODEL

```
#ensemble model
model_ensemble= RandomForestClassifier(n_estimators = 20,max_depth=20,n_jobs=-1)
model_ensemble.fit(X_train,Y_train)
performance_met(model_ensemble,X_train,Y_train,X_test,Y_test)
```

```
train score: accuracy:0.8065376972789199 f1:0.6592528161897644
test score: accuracy:0.8054309267979541 f1:0.6577298991986077
```

BOOSTING MODEL

```
#boosting model
model_boosting = AdaBoostClassifier()
model_boosting.fit(X_train,Y_train)
performance_met(model_boosting,X_train,Y_train,X_test,Y_test)
```

```
train score: accuracy:0.7807272617377347 f1:0.6282568367905806
test score: accuracy:0.7807176115371415 f1:0.6286158045841765
```

PREDICTION

```
#use linear model to predict data from test.csv
df3["predict_linear"] = model_linear.predict(df3[features])
```

```
#use ensemble model to predict data from test.csv
df3["predict_ensemble"] = model_ensemble.predict(df3[features])
```

```
#use boosting model to predict data from test.csv
df3["predict_boosting"] = model_boosting.predict(df3[features])
```

```
df3
```

	ind_nuevo	ind_actividad_cliente	renta	sexo_H	sexo_V	predict_linear	predict_ensemble	predict_boosting
0	0	1	326124.90	0	1	0.0	1.0	1.0
3	0	0	148402.98	1	0	0.0	0.0	0.0
4	0	0	106885.80	1	0	0.0	0.0	0.0
6	0	1	96395.88	1	0	0.0	0.0	0.0
9	0	1	68322.72	1	0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...
929608	0	0	70852.20	0	1	0.0	0.0	0.0
929609	0	0	100647.45	1	0	0.0	0.0	0.0
929610	0	1	128643.57	0	1	0.0	1.0	1.0
929612	0	1	72765.27	0	1	0.0	1.0	1.0
929613	0	0	147488.88	0	1	0.0	0.0	0.0

RESULT

Random Forest Classifiers provide the best result.

Accuracy ~ 80%

THANK YOU

Thank you for your listening