DQLab

DQLab Project

Customer Churn Prediction using Machine Learning

DQLab Telco

Oktavio Reza Putra
Business Management Final Year Student at
Pembangunan Nasional "Veteran" Yogyakarta University

CUSTOMER CHURN PREDICTION USING MACHINE LEARNING

DQLab

XERATIC

UMN

#DQLABAPL2 GRBVJC

CERTIFICATE

OF COMPLETION

This certificate is proudly presented to

Oktavio Reza Putra

Has Completed in

Customer Churn Prediction using Machine Learning

Feb 11, 2022

WHAT IS THE PROBLEM IN THIS CASE ?

DQLab Telco is a Telco company that already has many branches spread everywhere. Since its establishment in 2019, DQLab Telco has been consistent in paying attention to its customer experience so that customers will not leave it.

Even though it's only a little over 1 year old, DQLab Telco already has a lot of customers who have switched subscriptions to competitors. The management wants to reduce the number of churn customers by using machine learning.

After yesterday I prepared data as well as cleaning in the "Data Science in Telco: Data Cleansing" project, now it's time for me to make the right model to predict customer churn

PROJECT PROGRESSION STAGES

In yesterday's "Data Science in Telco: Data Cleansing" project, I did Cleansing Data. Now, as a data scientist, I am asked to make the right model. In this task, I will do Machine Learning Modeling using last month's data, namely June 2020.

The steps to be taken are:

- Performing Exploratory Data Analysis
- Doing Data Pre-Processing
- Doing Machine Learning Modeling
- Determining the Best Model

LIBRARIES USED

In this analysis, several packages will be used to assist us in conducting data analysis (Page 5 to 7)

Pandas (Python for Data Analysis) is a Python library that focuses on data analysis processes such as data manipulation, data preparation, and data cleaning.

- **read_csv()** is used to read csv files
- **replace()** is used to replace the value
- **value_counts()** is used to count unique from column
- **drop()** is used to remove
- **describe()** is used to view the description of the data
- **value_counts()** is used to count unique from column

LIBRARIES USED

Matplotlib is a Python library that focuses on visualizing data such as plotting graphs. Matplotlib can be used in Python scripts, Python and IPython shells, web application servers, and several other graphical user interface (GUI) toolkits.

- **figure()** is used to create a new figure
- **subplots()** is used to create an image and a set of subplots
- **title()** is used to give a title to the image
- **ylabel()** is used to label the Y-axis of the image
- **xlabel()** is used to label the Y axis of the image
- **pie()** is used to create a pie chart

Seaborn builds plots on top of Matplotlib and introduces additional plot types. It also makes your traditional Matplotlib plots look prettier.

- **countplot()** is used to create a plot with the number of observations in each bin of the categorical variable
- **heatmap()** Plot rectangular data as a color-encoded matrix

LIBRARIES USED

Scikit-learn is a library in Python that provides many Machine Learning algorithms both for Supervised, Unsupervised Learning, or used to prepare data.

- **LabelEncoder()** is used to change the value of a variable to 0 or 1
- **train_test_split()** is used to split data into 2 row sections (Training & Testing)
- **LogisticRegression()** is used to call the Logistic Regression algorithm
- **RandomForestClassifier()** is used to call the Random Forest Classifier algorithm
- **confusion_matrix()** is used to create a confusion matrix
- **classification_report()** is used to create a classification report, which includes the accuracy of the model

Xgboost is a library in Python for the extreme gradient boosting (xgboost) algorithm.

- **XGBClassifier()** is used to call the XG Boost Classifier algorithm

Pickle implements a binary protocol for serializing and de-serializing Python object structures.

- **dump()** is used to store

DΦLab

TOOLS

PYTHON

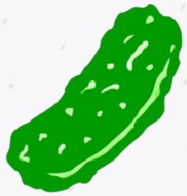
LIBRARY

pandas

matplotlib

seaborn

scikit learn

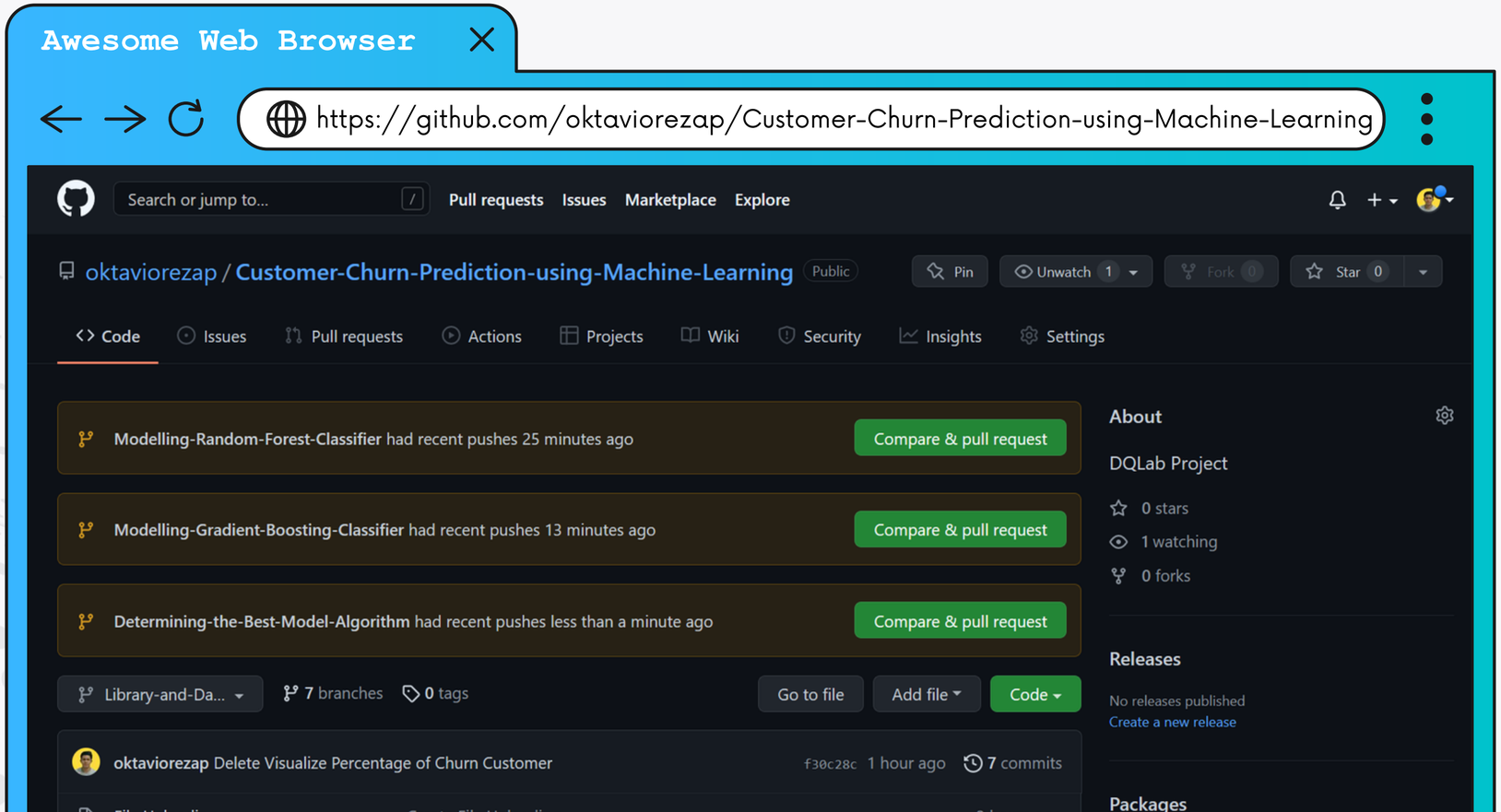


pickleDB

dmlc XGBoost

HOW CAN I ANALYZE THE DATA?

You can open my GitHub at this link below :
https://github.com/oktaviorezap/Customer-Churn-Prediction-using-Machine-Learning



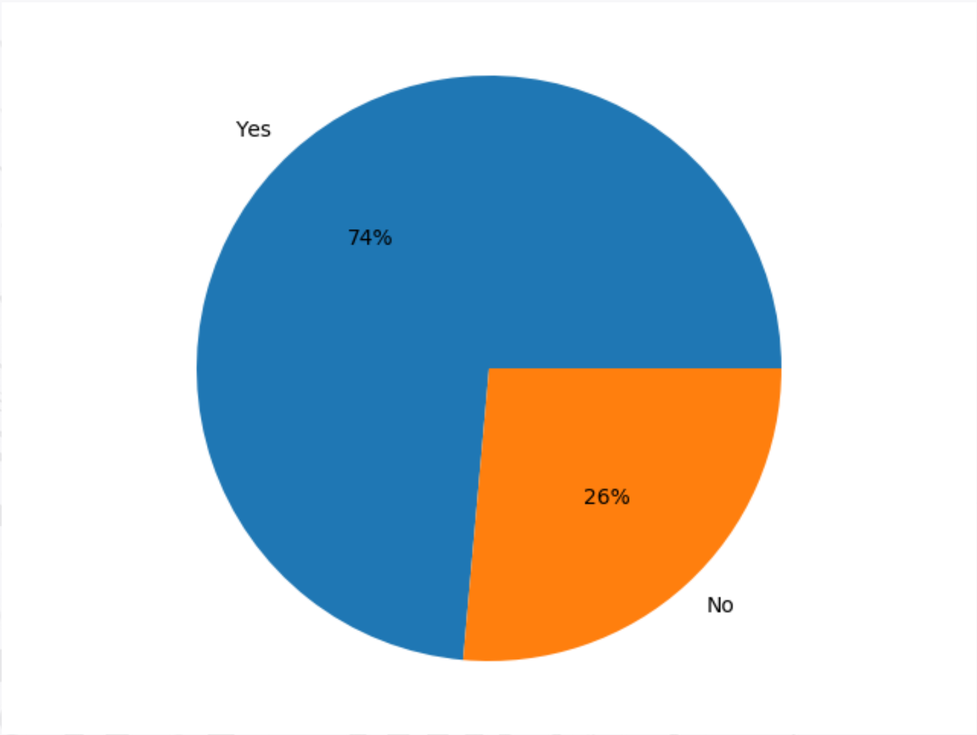
DATA USED

For the dataset used is already provided in csv format, please read through the pandas function in python df_load = pd.read_csv('**https://storage.googleapis.com/dqlab-dataset/dqlab_telco_final.csv**')

The detailed data are as follows:

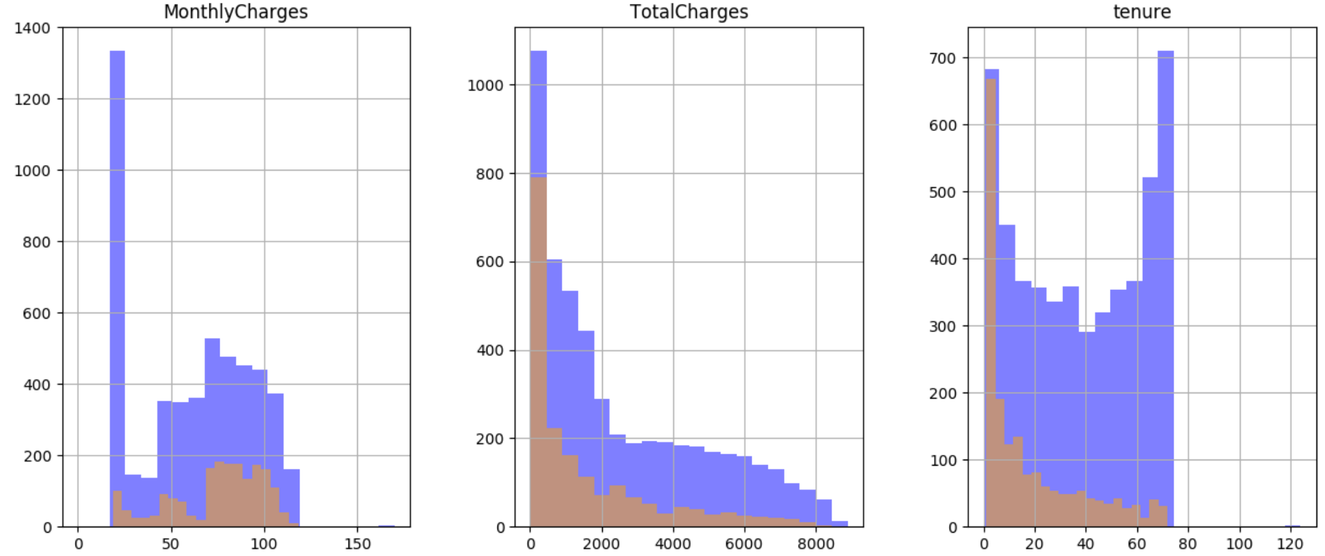
- **UpdatedAt**: Period of Data taken
- **customerID**: Customer ID
- **gender**: Whether the customer is a male or a female (Male, Female)
- **SeniorCitizen**: Whether the customer is a senior citizen or not (Yes, No)
- **Partner**: Whether the customer has a partner or not (Yes, No)
- **tenure**: Number of months the customer has stayed with the company
- **PhoneService**: Whether the customer has a phone service or not (Yes, No)
- **InternetService**: Customer's internet service provider (Yes, No)
- **StreamingTV**: Whether the customer has streaming TV or not (Yes, No)
- **PaperlessBilling**: Whether the customer has paperless billing or not (Yes, No)
- **MonthlyCharges**: The amount charged to the customer monthly
- **TotalCharges**: The total amount charged to the customer
- **Churn**: Whether the customer churned or not (Yes, No)

VISUALIZE PERCENTAGE OF CHURN CUSTOMER (STAGE C.1)



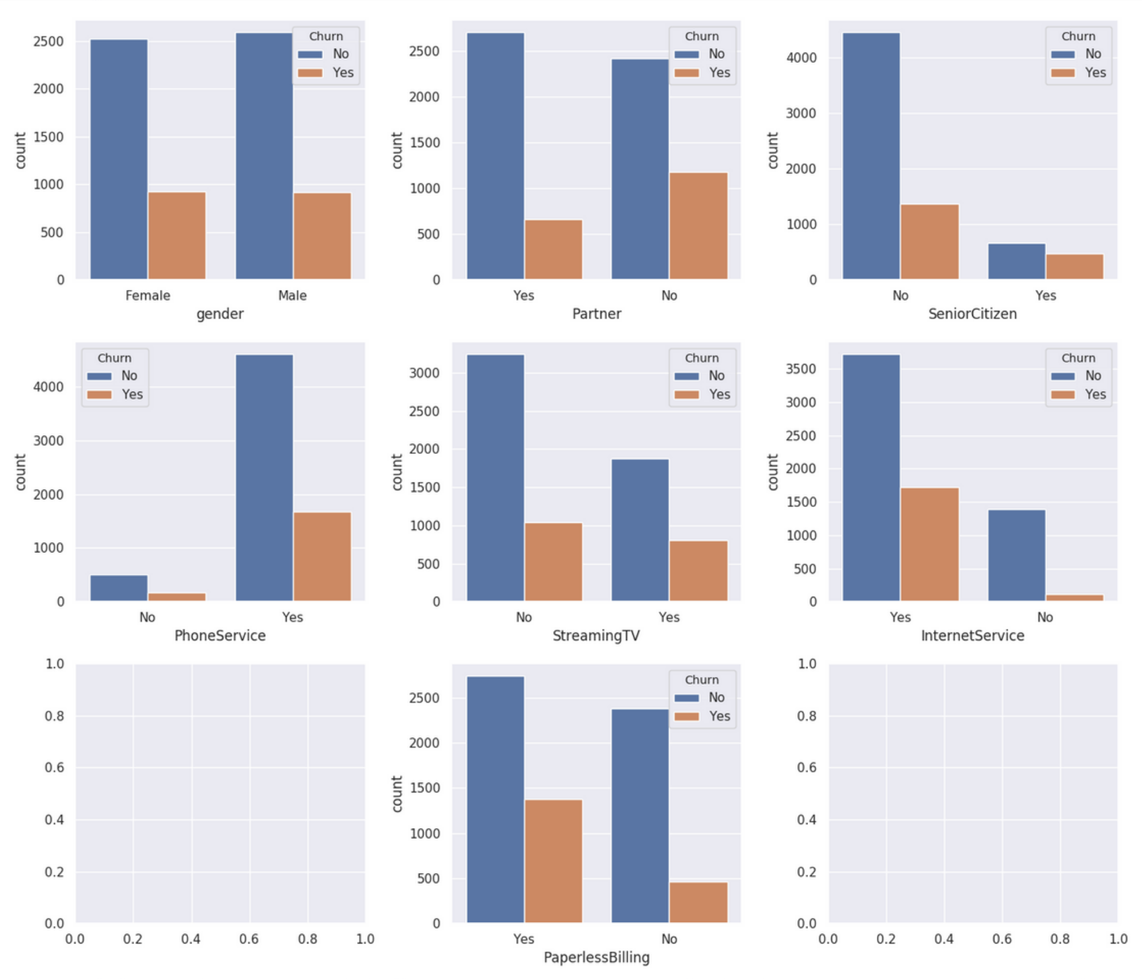
At stage C.1 we can see that the overall distribution of data does not churn customers, with details of **Churn as much as 26%** and **No Churn as much as 74%**.

CREATE EXPLORATORY DATA ANALYSIS (EDA) NUMERIC VARIABLE (STAGE C.2)



At stage C.2, we can see that for **MonthlyCharges** **there is a tendency that the smaller the monthly fee charged, the smaller the tendency to Churn**. For **TotalCharges**, **there seems to be no trend towards Churn customers.** For **tenure**, **there is a tendency that the longer the customer subscribes, the smaller the tendency to Churn**.

CREATE EXPLORATORY DATA ANALYSIS (EDA) CATEGORIC VARIABLE (STAGE C.3)



At stage C.3 we can see that there is no significant difference for people doing churn in terms of **gender (gender)** and **telephone service (PhoneService)**. However, there is a tendency that people who churn are people who do not have partners (**partners: No**), people whose status is senior citizens (**SeniorCitizen: Yes**), people who have TV streaming services (**StreamingTV: Yes**) , people who have Internet service (**internetService: Yes**) and people whose bills are paperless (**PaperlessBilling: Yes**).

DATA PROCESSING

DELETING UNNECESSARY DATA COLUMN

gender	SeniorCitizen	Partner	tenure	PhoneService	StreamingTV	InternetService	PaperlessBilling	MonthlyCharges	TotalCharges	Churn
Female	No	Yes	1	No	No	Yes	Yes	29.85	29.85	No
Male	No	Yes	60	Yes	No	No	Yes	20.50	1198.80	No
Male	No	No	5	Yes	Yes	Yes	No	104.10	541.90	Yes
Female	No	Yes	72	Yes	Yes	Yes	Yes	115.50	8312.75	No
Female	No	Yes	56	Yes	Yes	Yes	No	81.25	4620.40	No

Dataset after deleted unnecessary column. In this deletion task, I excluded "customerID" and "UpdatedAt" columns so that the data can be analyzed easily.

DATA PROCESSING

ENCODING DATA (CHANGE THE DATA TYPE FROM STRING TO NUMERIC)

Persebaran data setelah dilakukan encoding:

	gender	SeniorCitizen	Partner	tenure	PhoneService
count	6950.000000	6950.000000	6950.000000	6950.000000	6950.000000
mean	0.504317	0.162302	0.483309	32.415827	0.903741
std	0.500017	0.368754	0.499757	24.561336	0.294967
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	9.000000	1.000000
50%	1.000000	0.000000	0.000000	29.000000	1.000000
75%	1.000000	0.000000	1.000000	55.000000	1.000000
max	1.000000	1.000000	1.000000	73.000000	1.000000

	StreamingTV	InternetService	PaperlessBilling	MonthlyCharges
count	6950.000000	6950.000000	6950.000000	6950.000000
mean	0.384317	0.783453	0.591942	64.992201
std	0.486468	0.411921	0.491509	30.032040
min	0.000000	0.000000	0.000000	0.000000
25%	0.000000	1.000000	0.000000	36.462500
50%	0.000000	1.000000	1.000000	70.450000
75%	1.000000	1.000000	1.000000	89.850000
max	1.000000	1.000000	1.000000	169.931250

	TotalCharges	Churn
count	6950.000000	6950.000000
mean	2286.058750	0.264173
std	2265.702553	0.440923
min	19.000000	0.000000
25%	406.975000	0.000000
50%	1400.850000	0.000000
75%	3799.837500	1.000000
max	8889.131250	1.000000

DATA PROCESSING

SPLITTING DATASET (CHECK THE PROPORTION)

```
Jumlah baris dan kolom dari x_train adalah: (4865, 10) , sedangkan Jumlah baris dan kolom dari y_train adalah:
(4865,)
Prosentase Churn di data Training adalah:
 0    0.734841
1    0.265159
Name: Churn, dtype: float64

Jumlah baris dan kolom dari x_test adalah: (2085, 10) , sedangkan Jumlah baris dan kolom dari y_test adalah: (2085,)
Prosentase Churn di data Testing adalah:
 0    0.738129
1    0.261871
Name: Churn, dtype: float64
```

DATA PROCESSING

CONCLUSION

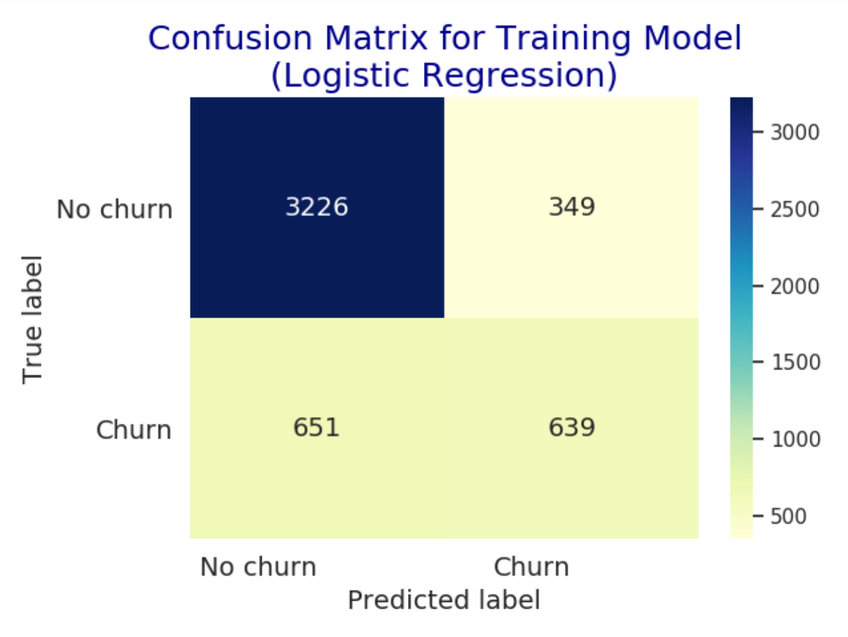
After further analysis, it turns out that there are columns that are not needed in the model, namely the **customer ID number (customerID)** & **the data collection period (UpdatedAt)**, so this needs to be deleted.

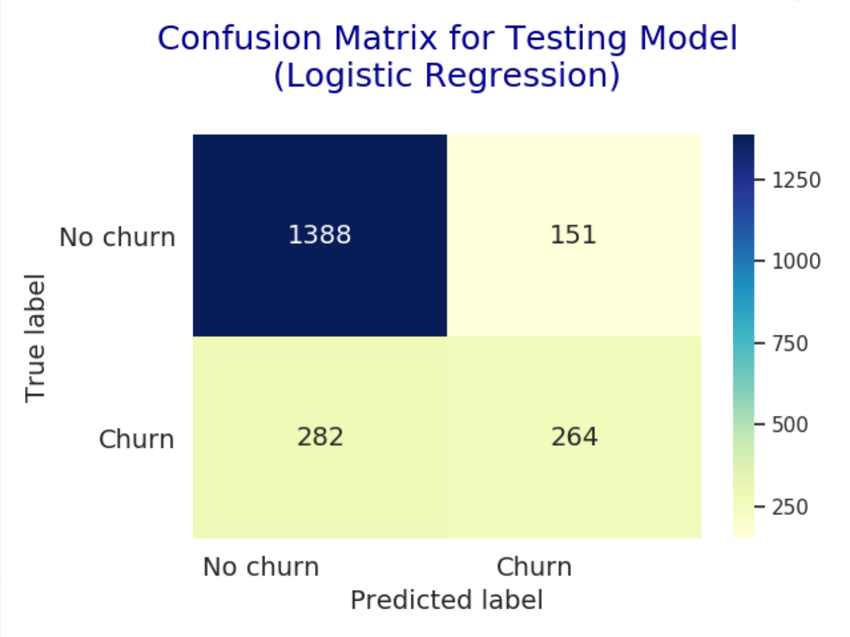
Then we continue to change the value of the data that is still in the form of a string into numeric through encoding, after that, it can be seen that the distribution of the data, especially the min and max columns of each variable, has changed to 0 & 1.

The last step is to divide the data into 2 parts for modeling purposes, After that, it can be seen that the number of rows and columns of each data is appropriate & the percentage of the churn column is also the same as the data at the beginning, this indicates that the data is separated properly and correctly.

MODELLING : LOGISTIC REGRESSION

TRAINING MODEL AND TESTING MODEL PERFORMANCE



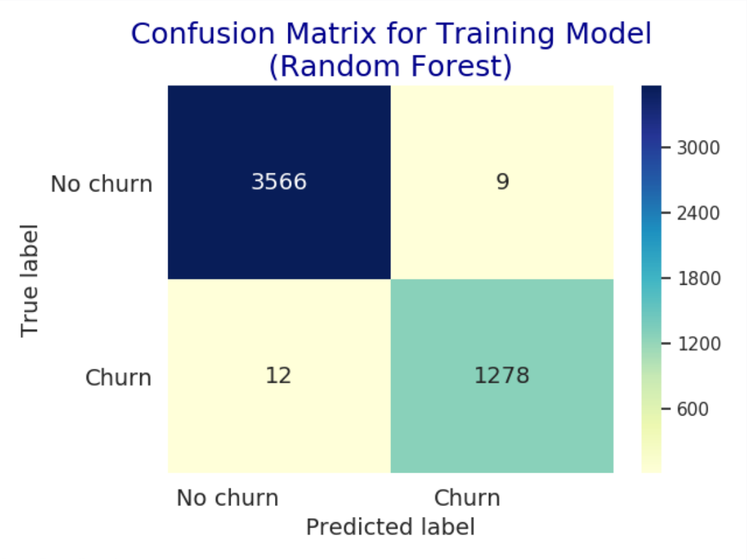


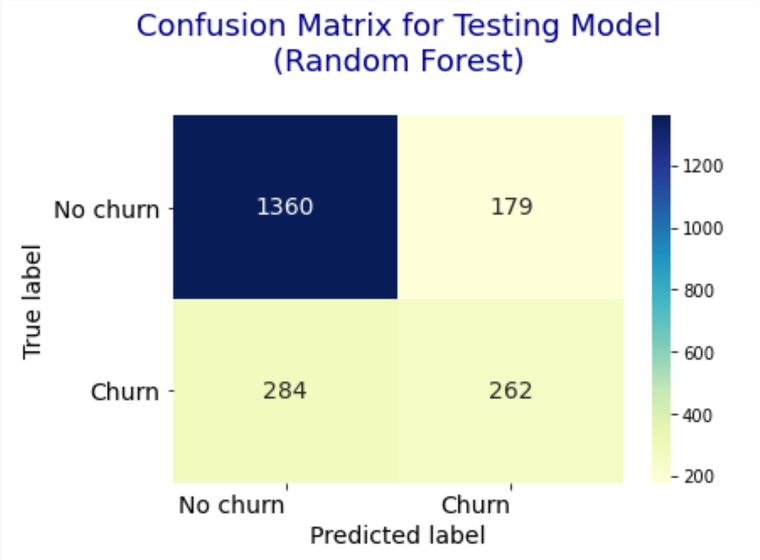
If we use the logistic regression algorithm by calling **LogisticRegression()** from sklearn without adding any parameters, then the resulting model with the default settings from sklearn, for details can be seen in the documentation.

- From the **training data**, it can be seen that the model is able to predict the data with an accuracy of 79%, with details of the correct churn guess, the churn is 639, the non-churn guess that doesn't actually churn is 3226, the correct churn guess is 651 and the correct churn guess. actually not churn is 349.
- From the **data testing**, it can be seen that the model is able to predict the data by producing an accuracy of 79%, with details of the correct churn guess, the churn is 282, the non-churn guess that doesn't actually churn is 1388, the non-churn guess that actually has the churn is 282 and the wrong churn guess. actually not churn is 151.

MODELLING : RANDOM FOREST CLASSIFIER

TRAINING MODEL AND TESTING MODEL PERFORMANCE





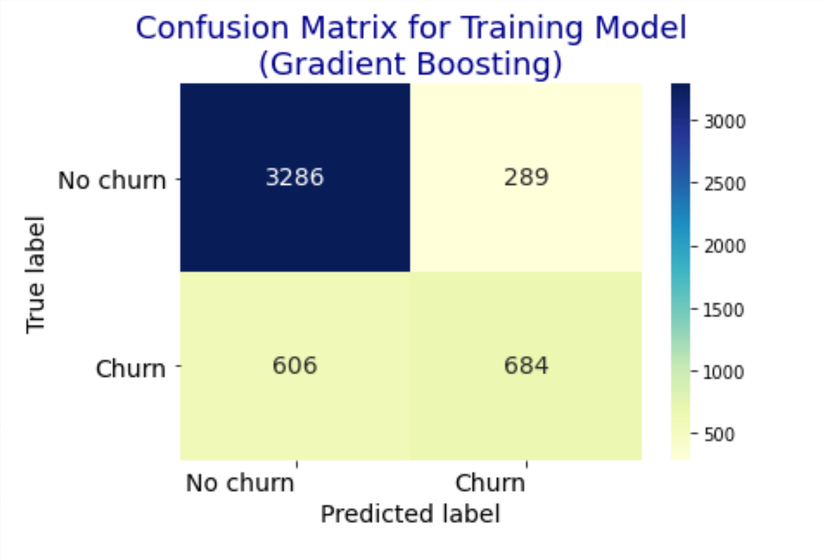
If we use the Random Forest algorithm by calling **RandomForestClassifier()** from sklearn without adding any parameters, then the resulting model with the default settings of sklearn, for details can be seen in the documentation.

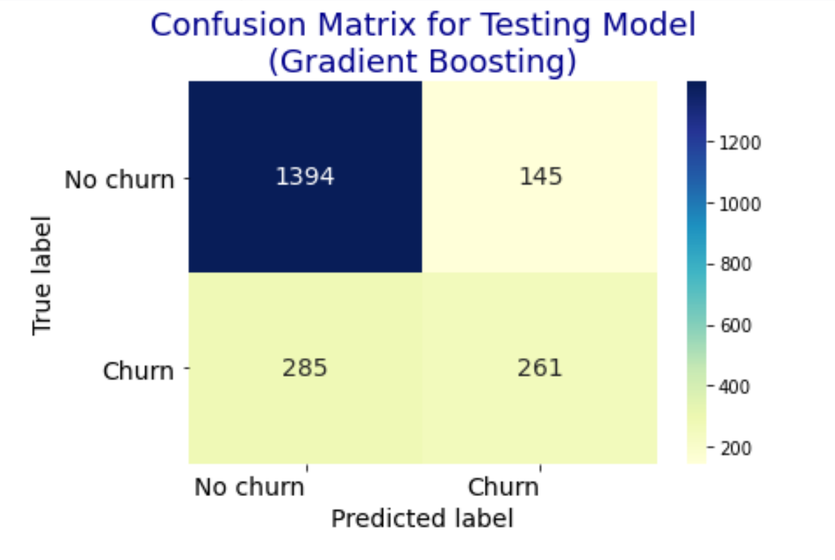
- From the **training data**, it can be seen that the model is able to predict the data with an accuracy of 100%, with details of the correct churn guess, the churn is 1278, the non-churn guess that doesn't actually churn is 3566, the correct churn guess is 12 and the correct churn guess. actually not churn is 9.

- From the **data testing**, it can be seen that the model is able to predict the data by producing an accuracy of 78%, with details of the churn guess that actually churns 262, the guess that doesn't churn that doesn't actually churn is 1360, the guess that doesn't churn that actually churn is 284 and the guess that doesn't churn actually not churn is 179.

MODELLING : GRADIENT BOOSTING CLASSIFIER

TRAINING MODEL AND TESTING MODEL PERFORMANCE





If we use the Gradient Boosting algorithm by calling **GradientBoostingClassifier()** from the sklearn package without adding any parameters, the resulting model with the default settings of sklearn, for details can be seen in the documentation.

- From the **training data**, it can be seen that the model is able to predict the data by producing an accuracy of 82%, with the details of the correct churn guess, the churn is 684, the non-churn guess that doesn't actually churn is 3286, the non-churn guess that actually has the churn is 606 and the wrong churn guess. actually not churn is 289.

- From the **data testing**, it can be seen that the model is able to predict the data by producing an accuracy of 79%, with details of the correct churn guess, the churn is 261, the guess not to churn which actually doesn't churn is 1394, the guess not to churn which is actually true to churn is 285 and the guess to churn which is actually true. actually not churn is 145.

DETERMINING THE BEST MODEL ALGORITHM

Based on the modeling that has been done using Logistic Regression, Random Forest and Extreme Gradiant Boost, it can be concluded that to predict churn from telco customers using this dataset the best model is to use the Logistic Regression algorithm.

This is because the performance of the Logistic Regression model tends to be able to predict equally well in the training and testing phases (79% training accuracy, 79% testing accuracy), on the other hand, other algorithms tend to over-fitting their performance.

However, this does not make us draw the conclusion that if to do any modeling we use Logistic Regression, we still have to do a lot of model experiments to determine which one is the best.

DQLab

Let's Collaborate!

Oktavio Reza Putra

oktaviorezaputra@gmail.com

+62 89630320035