CHURN CASE

AB testing

Customer Segmentation

Decision Tree

Logistic Regression

Model Evaluation

Customer Lifetime Value

Rui Li

2020/5/15

https://github.com/ruili629/MarketingAnalyticsAssignment

0

Prepocessing

https://github.com/ruili629/MarketingAnalytics
Assignment

Drop some cols or rows

- Reason:
- 1. One single value (Language, Country)
- 2. Abnormal value (Leave only those age<100 or eg. change 1997 into 2020 - 1997 = 23)
- 3. Uncontrollable or usless factor in segmentation (eg. subid)

Age

 # change those > 1900 to 2020 - year

```
subscribers[subscribers.age>100]
```

intended_use	weekly_consumption_hour	num_ideal_streaming_services	retarget_TF	age	male_TF	...	creation_until_cancel_days	cancel_before_trial_end	trial_
access to exclusive content	27.301448	NaN	False	1955.0	False	...	7.0	False	2
access to exclusive content	25.851492	NaN	False	1950.0	False	...	30.0	True	2
expand regional access	27.301448	1.958566	False	1957.0	False	...	NaN	True	2
supplement OTT	31.651317	1.874212	False	1969.0	False	...	4.0	False	2

Fill in the NA

```
values = {'num_weekly_services_utilized': 0, 'weekly_consumption_hour': 0, 'num_ideal_streaming_services': 0,
          'op_sys': 'unknownOp', 'creation_until_cancel_days' : 0, 'revenue_net' : 0, 'join_fee' : 0,
          'payment_type' :'not_paid'}
df = df.fillna(value=values)
```

Change Data Type

```
# change data type
df['preferred_genre'] = df['preferred_genre'].astype(str)
df['package_type'] = df['package_type'].astype(str)
df['intended_use'] = df['intended_use'].astype(str)
df['payment_type'] = df['payment_type'].astype(str)
df['op_sys'] = df['op_sys'].astype(str)
```

1

AB Testing

单击此处添加副标题内容

AB Testing - paid

	paid	all	coversion_rate
base_uae_no_trial_7_day_guarantee	1.0	1	1.000000
high_jpy_14_day_trial	1.0	1	1.000000
low_eur_no_trial	1.0	1	1.000000
low_sar_no_trial	1.0	1	1.000000
low_uae_no_trial	134.0	167	0.802395
base_eur_14_day_trial	11.0	18	0.611111
high_aud_14_day_trial	1.0	2	0.500000
base_uae_14_day_trial	91776.0	227096	0.404129
high_uae_14_day_trial	119.0	325	0.366154
high_sar_14_day_trial	4.0	12	0.333333
low_gbp_14_day_trial	1.0	4	0.250000

Hypothesis setup:

Variant B: high_uae_14_day_trial

Variant A: base_uae_14_day_trial

H0: Variant B and Variant A had the same conversion rates.

HA: Variant B had a lower conversion rate than Variant A

Assumptions:

Variant A represents the population and we can treat the population mean as known and equal to the mean of Variant A.

Result:

$|Z| = 2.2 > 1.64$

AB Testing - cancel_before_trial_end

	cancel	all	churn_rate
base_uae_no_trial_7_day_guarantee	1.0	1	1.000000
high_jpy_14_day_trial	1.0	1	1.000000
low_eur_no_trial	1.0	1	1.000000
low_sar_no_trial	1.0	1	1.000000
low_uae_no_trial	141.0	167	0.844311
base_eur_14_day_trial	11.0	18	0.611111
high_aud_14_day_trial	1.0	2	0.500000
high_sar_14_day_trial	6.0	12	0.500000
base_uae_14_day_trial	103253.0	227096	0.454667
high_uae_14_day_trial	140.0	325	0.430769
low_gbp_14_day_trial	1.0	4	0.250000

Hypothesis setup:

Variant B: high_uae_14_day_trial

Variant A: base_uae_14_day_trial

H0: Variant B and Variant A had the same churn(cancel) rates.

HA: Variant B had a lower churn rate than Variant A

Assumptions:

Variant A represents the population and we can treat the population mean as known and equal to the mean of Variant A.

Result:

$|Z| = 12.22 > 1.64$

AB Testing - renew

	renew	all	coversion_rate
base_eur_14_day_trial	13.0	23	0.565217
high_jpy_14_day_trial	1.0	2	0.500000
high_aud_14_day_trial	2.0	5	0.400000
base_uae_14_day_trial	72971.0	209383	0.348505
high_uae_14_day_trial	129.0	375	0.344000
high_sar_14_day_trial	5.0	15	0.333333
low_uae_no_trial	25.0	88	0.284091
base_uae_no_trial_7_day_guarantee	0.0	2	0.000000
low_gbp_14_day_trial	0.0	1	0.000000

Hypothesis setup:

Variant B: high_uae_14_day_trial

Variant A: base_uae_14_day_trial

H0: Variant B and Variant A had the same renew rates.

HA: Variant B had a lower renew rate than Variant A

Assumptions:

Variant A represents the population and we can treat the population mean as known and equal to the mean of Variant A.

Result:

$|Z| = 9.71 > 1.64$

Optimal Sample Size and Sequential Testing - (paid)

The optimal sample size for each segment is 2616

The challenger wins 0.0% of the time.

Assume P(Xi=1) under H0 = p-varA and P(Xi=1) under H1 = p-treatment.

Set desired type 1 error = 5% and type 2 error = 20%.

Success rate is 80.0%

2

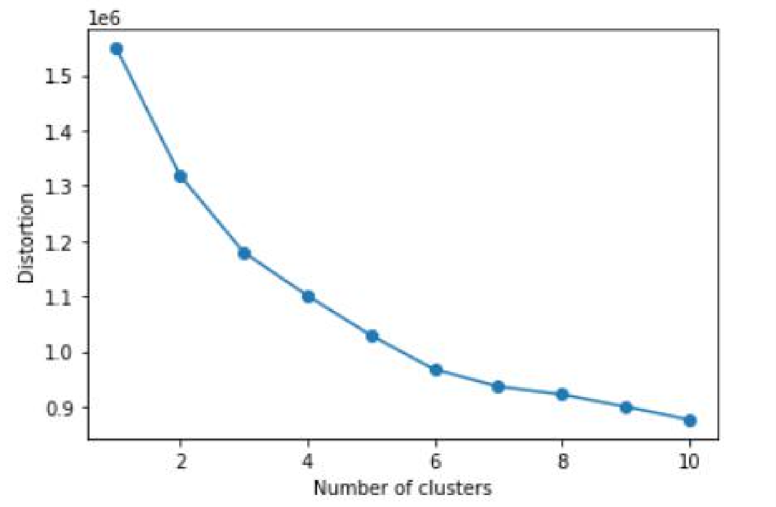
Customer Segmentation

单击此处添加副标题内容

Customer segmentation

Cluster 0 (High value) customer description:

1. They completed less videos but they rated much more often.
2. Customer internet package: base mainly.
3. Preferred genre: comedy and drama mainly.
4. Main Attribution_technical_brand: “sem intent google”, ”email blast” .
5. Plan_type: high_uae_14_day_trial
6. Main Payment_type:Paypal
7. Main Payment_type_Standard:Charter
8. **High Value becasue they pay.**



Cluster	cust_service_mssgs	num_videos_completed	num_videos_rated	retarget_TF	ancel_before_trial_en	initial_credit_card_declined	revenue_net	paid_TF	refund_after_trial_TF
0	0.3726	2.6224	0.0684	0.0438	1.0000	0.0025	4.0263	1.0000	0.2587
1	0.5936	3.0025	0.0005	0.0359	0.0840	0.0827	-0.0026	0.0000	0.0000
2	0.5655	2.9997	0.0008	0.0231	0.0954	0.0952	0.0000	0.0000	0.0000
3	0.5812	3.0539	0.0005	0.0317	0.0726	0.0711	-0.0014	0.0000	0.0000

Cluster	genre_comedy	genre_drama	genre_international	genre_regional	ccess to exclusive co	use_education	expand international ac	se_expand regional acces	package_base
0	0.3424	0.1510	0.0298	0.0549	0.2403	0.0571	0.1391	0.1827	0.3205
1	0.6850	0.2034	0.0291	0.0590	0.4172	0.0000	0.0007	0.0007	0.0000
2	0.0000	0.0007	0.0002	0.0002	0.0000	0.1451	0.3806	0.4208	0.0005
3	0.6585	0.2407	0.0290	0.0479	0.4911	0.0004	0.0009	0.0008	1.0000

Cluster	affiliate	bing	bing_organic	brand sem intent bin	brand sem intent googl	direct_mail	discovery	display	[illegible]
0	0.0387	0.0029	0.0014	0.0174	0.0923	0.0012	0.0171	0.0042	0.0707
1	0.0381	0.0033	0.0012	0.0080	0.0761	0.0005	0.0098	0.0033	0.0247
2	0.0330	0.0036	0.0024	0.0071	0.0733	0.0000	0.0000	0.0011	[illegible]
3	0.0362	0.0037	0.0018	0.0100	0.0772	0.0007	0.0111	0.0037	0.0259

Cluster	facebook	facebook_organic	google_organic	internal	organic	podcast	referral	vod	an_high_uae_14_day_tri
0	0.2464	0.0181	0.0640	0.0153	0.1022	0.0027	0.0452	0.0019	0.0039
1	0.3414	0.0166	0.0505	0.0049	0.0933	0.0053	0.0243	0.0013	0.0024
2	0.3719	0.0149	0.0517	0.0104	0.0899	0.0014	0.0252	0.0000	0.0000
3	0.3592	0.0145	0.0520	0.0047	0.0928	0.0055	0.0245	0.0013	0.0024

3

Decision Tree

Preprocess our input DF
Feature Selection
Process
Visualization

Which features to select?

- After we finish the cleaning data and use get dummies to get our input dataframe, we get a problem:

  Too many variables!

- **Recursive Feature Ranking with recursive feature elimination and cross-validation**
- We choose to use the RFECV, which performs RFE in a cross-validation loop to find the optimal number or the best number of features. Hereafter a recursive feature elimination applied on decision tree classifier or logistic regression with automatic tuning of the number of features selected with cross-validation.
- Classifier: decision tree; Dependent variable: renew;

```
: len(df_input.columns)

: 101

: df_input.columns

: Index(['num_weekly_services_utilized', 'weekly_consumption_hour',
         'num_ideal_streaming_services', 'retarget_TF', 'age',
         'months_per_bill_period', 'monthly_price', 'discount_price',
         'creation_until_cancel_days', 'cancel_before_trial_end',
         ...
         'plan_type_high_sar_14_day_trial', 'plan_type_high_uae_14_day_trial',
         'plan_type_low_gbp_14_day_trial', 'payment_type_Apple Pay',
         'payment_type_CBD', 'payment_type_Najim', 'payment_type_Paypal',
         'payment_type_RAKBANK', 'payment_type_Standard Charter',
         'payment_type_not_paid'],
        dtype='object', length=101)
```

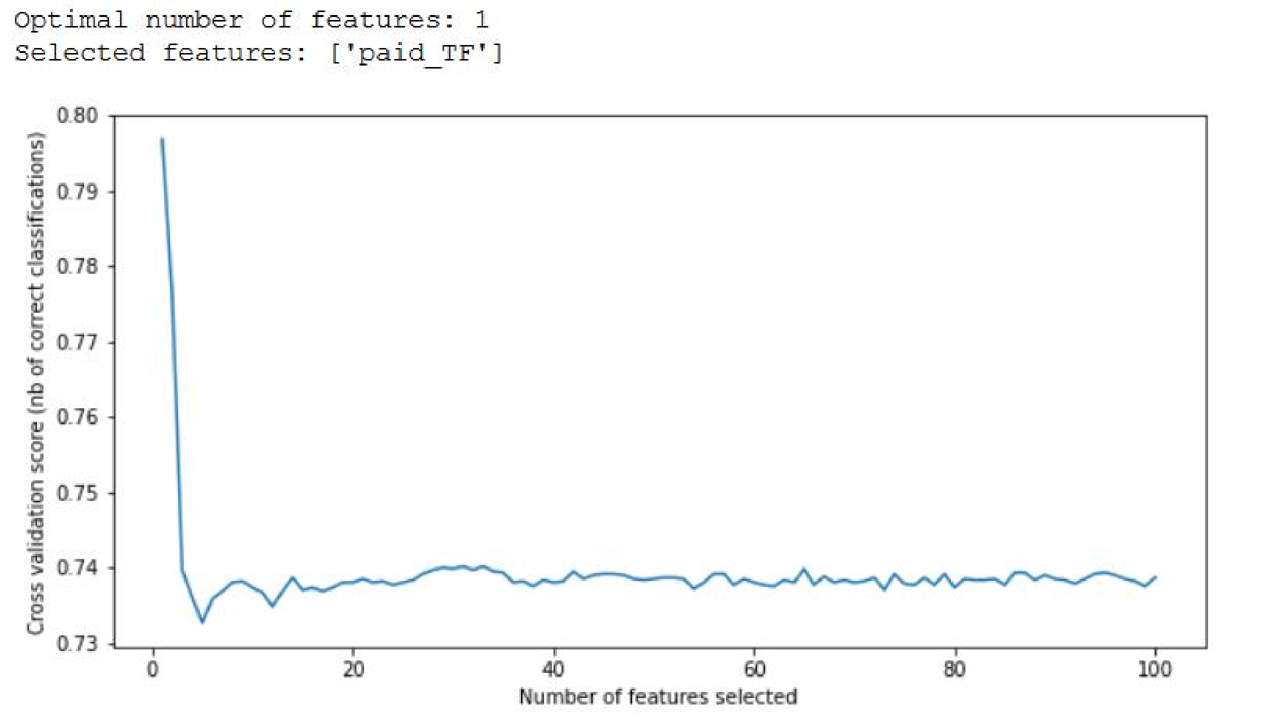
Apply it right away?

- **If we apply the RFECV without further preprocessing, we would get:**

  Optimal number of features: 1

  Selected features: ['paid_TF']

- This is not satisfying since paid_TF is only a a result and it is highly correlated with “renew”



Drop those “result” variable

- Dropping the variables that are highly correlated with “renew”:

  ['paid_TF','payment_type_not_paid','revenue_net','cancel_before_trial_end','renew']

```
a = df_input.corr()['renew']
```

```
a[a>0.5]
```

```
cancel_before_trial_end    0.612260
revenue_net                0.577446
paid_TF                    0.652731
renew                      1.000000
Name: renew, dtype: float64
```

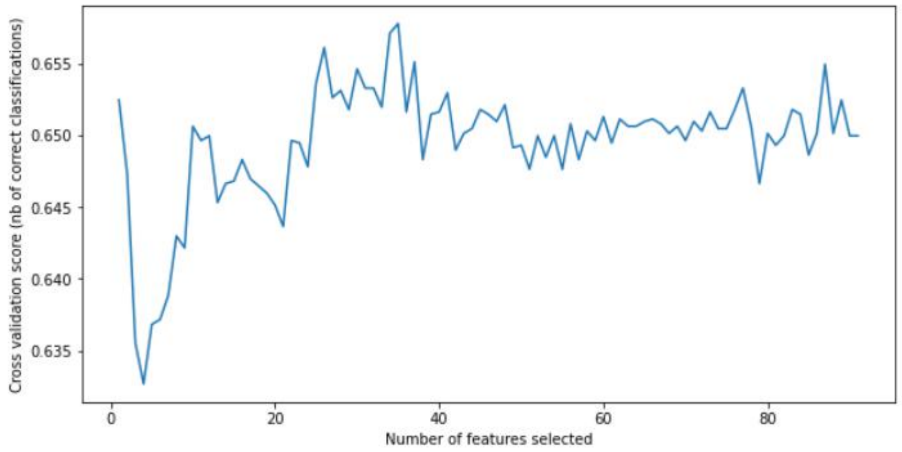
```
a[a<-0.5]
```

```
payment_type_not_paid   -0.652731
Name: renew, dtype: float64
```

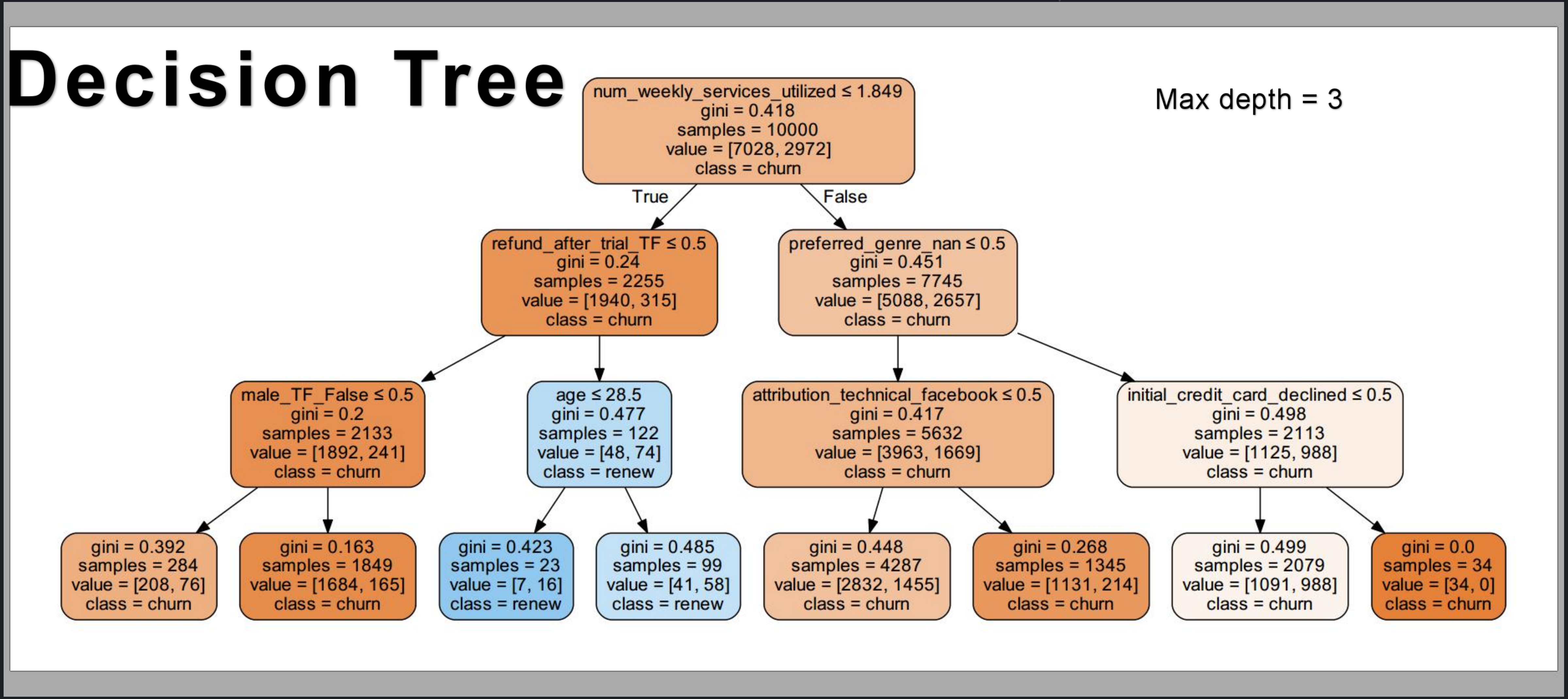
```
feature_selection_class(tree.DecisionTreeClassifier(),
                        X.drop(columns = ['paid_TF', 'payment_type_not_paid', 'revenue_net'
                                         , 'cancel_before_trial_end','creation_until_cancel_days',
                                         'payment_type_Paypal', 'payment_type_RAKBANK',
                                          'payment_type_Standard Charter','num_weekly_services_utilized']),
                        y)
```

After dropping the highly correlated 'result-type'variables like cancel, revenue_net, we ran the RFECV again to get the best number and types of features.

```
Optimal number of features: 35
Selected features: ['weekly_consumption_hour', 'num_ideal_streaming_services', 'retarget_TF', 'age', 'initial_credit_card_declined', 'join_fee', 'refund_after_trial_TF', 'package_type_base', 'package_type_economy', 'package_type_enhanced', 'preferred_genre_comedy', 'preferred_genre_nan', 'intended_use_access to exclusive content', 'intended_use_expand international access', 'intended_use_other', 'intended_use_replace OTT', 'intended_use_supplement OTT', 'male_TF_True', 'attribution_technical_brand sem intent google', 'attribution_technical_email', 'attribution_technical_email_blast', 'attribution_technical_facebook', 'attribution_technical_google_organic', 'attribution_technical_organic', 'attribution_technical_pinterest', 'attribution_technical_referral', 'attribution_technical_search', 'attribution_survey_facebook', 'attribution_survey_referral', 'attribution_survey_search', 'attribution_survey_tv', 'op_sys_Android', 'op_sys_iOS', 'payment_type_CBD', 'payment_type_Najim']
```



To better visualize the tree, I used the max depth =3, yet the best depth tested for now is 10



4

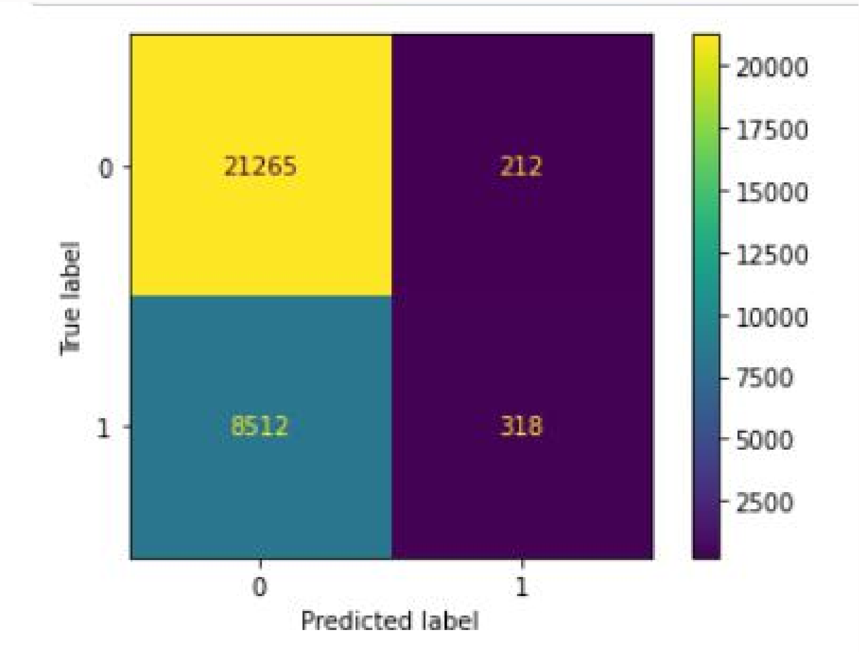
Model Evaluation

1.Under the same process of building decision tree, the logistic regression was also built
2.Comparing the result of Logistic Regression

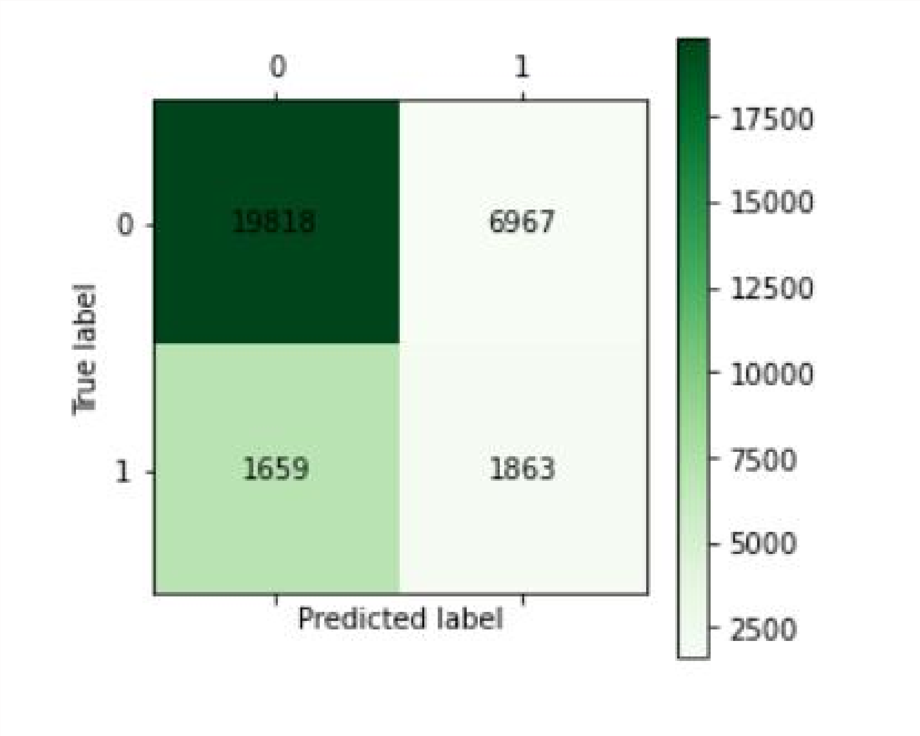
Confusion Matrix

- Comparing Confusion Matrix of two models, Logistic Regression model has a higher true positive rate but als higher false negative rate.

Decision Tree

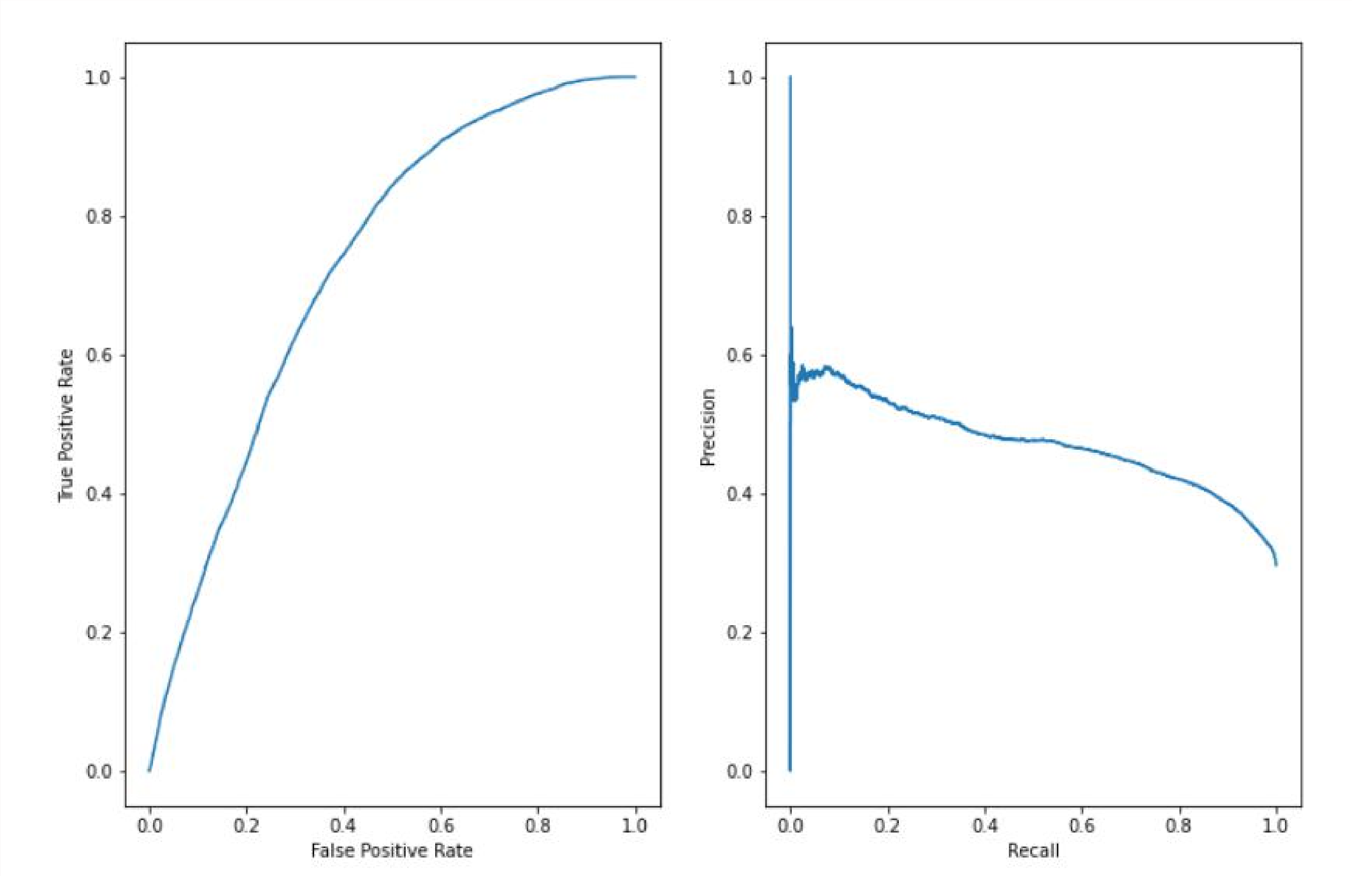


Logistic Regression



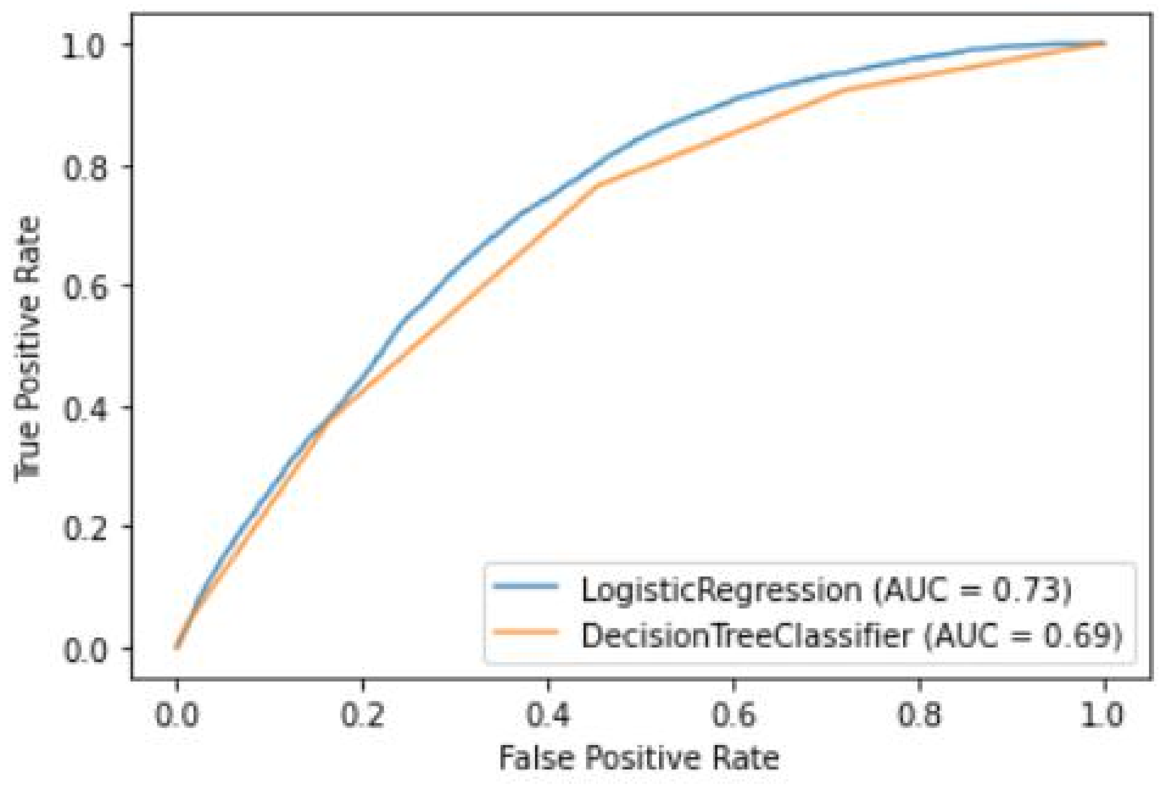
Confusion Matrix

- Since it seems that the logistic regression has a bit better performance than decision tree, and the result of its recall and precision is just satisfying.



ROC Curve

- From the ROC curve, it appears that the logistic regression has a bit better performance



5

Customer Lifetime Value

Using logistic regression to predict

CLV caculation

- As it appears that the logistic regression has a bit better performance, we shall use the logistic model to predict how likely this customer is sill going to renew in the nex period, so as to simulate and get the lifetime value.

```
prob_list = [renew for renew, churn in log.predict_proba(X.drop(columns = drop_cols))]
```

```
prob_list[1:10]
```

```
[0.7261671245082209,
 0.7261671245082209,
 0.8123141328142311,
 0.8123141328142311,
 0.9270975978280358,
 0.8862471829100999,
 0.6174895556199471,
 0.7911038113703474,
 0.9223666593475623]
```

CLV Caculation Method

```
import random

# using r_year = 0.1 as default
r_year = 0.1
# effective discount rate of each month
r = pow(1+r_year,1/12)-1

revenue_list = list(df['revenue_net'])
# 0 if churned
all_churn_list = list(df['revenue_net'])


t=1
while sum(all_churn_list) != 0:

    for i in range(len(prob_list)):
        if all_churn_list[i] != 0:
            if random.random() < prob_list[i]:
                revenue_list[i] = revenue_list[i]+revenue_list[i]/pow(1+r,t)
            else:
                all_churn_list[i] = 0

    t += 1
```

- **Discounted**

$$CLV = \frac{\sum_{i=1}^{n} \sum_{t=1}^{T} (gross\ margin_{it})^{-(1+r)^t}}{n}$$

r = discount rate, *t* indexes time periods