```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, recall_score, precision_score, f1_score
from sklearn import metrics
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import PolynomialFeatures

import numpy as np
```

```
try:
    df = pd.read_csv("/content/drive/MyDrive/data/takehome/takehome_users (2).csv", encoding='utf-8',index_col=False)
except UnicodeDecodeError:
    # If 'utf-8' fails, try 'ISO-8859-1' encoding
    df = pd.read_csv("/content/drive/MyDrive/data/takehome/takehome_users (2).csv", encoding='ISO-8859-1',index_col=False)
```

```
df.shape
```

```
(12000, 10)
```

```
user_eng=pd.read_csv("/content/drive/MyDrive/data/takehome/takehome_user_engagement.csv",index_col=False)
```

```
user_eng.shape
```

```
(134897, 3)
```

```
user_eng.head()
```

	time_stamp	user_id	visited
0	2014-04-22 03:53:30	1.0	1.0
1	2013-11-15 03:45:04	2.0	1.0
2	2013-11-29 03:45:04	2.0	1.0
3	2013-12-09 03:45:04	2.0	1.0
4	2013-12-25 03:45:04	2.0	1.0

```
# Taking unique user_id

t=user_eng['user_id'].unique()


Unique_user= pd.DataFrame(t, columns=['user_id'])


Unique_user.shape

##WILL USE THIS TO MAKE JOIN AND LABEL OF OUTPUT
```

```
(5610, 1)
```

```
Unique_user.head()
```

	user_id
0	1.0
1	2.0
2	3.0
3	4.0
4	5.0

```
user_eng['Visited_Date'] = user_eng['time_stamp'].apply(lambda x: x.split(' ')[0])


engg_detail =user_eng.drop(columns=['time_stamp'])


engg_detail.head()
```

	user_id	visited	Visited_Date
0	1.0	1.0	2014-04-22
1	2.0	1.0	2013-11-15
2	2.0	1.0	2013-11-29
3	2.0	1.0	2013-12-09
4	2.0	1.0	2013-12-25

```
#converting date type since latest 7 days engg required

engg_detail['Visited_Date'] = pd.to_datetime(engg_detail['Visited_Date'])


# Sort DataFrame by date column in descending order
engg_detail.sort_values(by='Visited_Date', ascending = False, inplace = True)

print(engg_detail)
```

```
        user_id  visited Visited_Date
70763    4051.0      1.0   2014-06-06
131604   7544.0      1.0   2014-06-04
71997    4143.0      1.0   2014-06-04
87466    4812.0      1.0   2014-06-04
98415    5378.0      1.0   2014-06-04
...         ...      ...          ...
60374    3514.0      1.0   2012-06-02
109716   6102.0      1.0   2012-06-01
32373    1995.0      1.0   2012-06-01
26821    1693.0      1.0   2012-05-31
59486    3428.0      1.0   2012-05-31

[134897 rows x 3 columns]
```

```
engg_detail.head(20)
```

	user_id	visited	Visited_Date
70763	4051.0	1.0	2014-06-06
131604	7544.0	1.0	2014-06-04
71997	4143.0	1.0	2014-06-04
87466	4812.0	1.0	2014-06-04
98415	5378.0	1.0	2014-06-04
128864	7375.0	1.0	2014-06-04
58507	3387.0	1.0	2014-06-04
39156	2339.0	1.0	2014-06-04
6173	363.0	1.0	2014-06-04

```
engg_detail.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 134897 entries, 70763 to 59486
Data columns (total 3 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   user_id       134896 non-null  float64
 1   visited       134896 non-null  float64
 2   Visited_Date  134897 non-null  datetime64[ns]
dtypes: datetime64[ns](1), float64(2)
memory usage: 4.1 MB
```

```
# Latest 7 days user eng

engg_2 = engg_detail[(engg_detail['Visited_Date'] < "2014-06-06") & (engg_detail['Visited_Date'] > "2014-05-31")]
```

28718 1781.0 1.0 2014-06-04

```
engg_2.head()
```

	user_id	visited	Visited_Date
131604	7544.0	1.0	2014-06-04
71997	4143.0	1.0	2014-06-04
87466	4812.0	1.0	2014-06-04
98415	5378.0	1.0	2014-06-04
128864	7375.0	1.0	2014-06-04

```
df_eng=engg_2
```

```
#Group by 'user_id' and count the number of unique days each user has visited

df_eng['user_visits_count'] = df_eng.groupby('user_id')['Visited_Date'].transform('nunique')
```

```
<ipython-input-21-d229b3e76b69>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_eng['user_visits_count'] = df_eng.groupby('user_id')['Visited_Date'].transform('nunique')
```

```
df_eng.shape
```

```
(1050, 4)
```

```
df_eng.head(30)
```

	user_id	visited	Visited_Date	user_visits_count
131604	7544.0	1.0	2014-06-04	3
71997	4143.0	1.0	2014-06-04	4
87466	4812.0	1.0	2014-06-04	3
98415	5378.0	1.0	2014-06-04	3
128864	7375.0	1.0	2014-06-04	4
58507	3387.0	1.0	2014-06-04	1
39156	2339.0	1.0	2014-06-04	2
6173	363.0	1.0	2014-06-04	3
93773	5157.0	1.0	2014-06-04	4
58380	3370.0	1.0	2014-06-04	3
32131	1941.0	1.0	2014-06-04	4
1110	63.0	1.0	2014-06-04	4
6110	351.0	1.0	2014-06-04	3
111784	6204.0	1.0	2014-06-04	4
79193	4403.0	1.0	2014-06-04	4

```
len(df_eng['user_id'].unique())
```

```
448
```

[illegible]

```
# Filter the DataFrame to include only users with user_visits_count >= 3
filtered_df = df_eng.groupby('user_id').filter(lambda x: x['user_visits_count'].iloc[0] >= 3)

# Drop duplicate rows based on 'user_id'
unique_users_df = filtered_df.drop_duplicates(subset='user_id')
```

[illegible]

```
selected_columns_df = unique_users_df[['user_id', 'user_visits_count']]
```

23791 1421.0 1.0 2014-06-04 4

```
selected_columns_df.head()
```

	user_id	user_visits_count
131604	7544.0	3
71997	4143.0	4
87466	4812.0	3

```
#if adopted then 1 else 0

selected_columns_df['Adopted_User'] = 1
```

```
<ipython-input-28-b8db41f4a799>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  selected_columns_df['Adopted_User'] = 1
```

```
selected_columns_df.head()
```

	user_id	user_visits_count	Adopted_User
131604	7544.0	3	1
71997	4143.0	4	1
87466	4812.0	3	1
98415	5378.0	3	1
128864	7375.0	4	1

```
selected_columns_df.shape
```

```
(192, 3)
```

```
Adopted_UID=selected_columns_df[['user_id', 'Adopted_User']]
```

```
# Rename 'user_id' column to 'object_id'
Adopted_UID.rename(columns={'user_id': 'object_id'}, inplace=True)
```

```
<ipython-input-32-f86f43243cff>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
    See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
      Adopted_UID.rename(columns={'user_id': 'object_id'}, inplace=True)
```

```
Adopted_UID.head()
```

	object_id	Adopted_User
131604	7544.0	1
71997	4143.0	1
87466	4812.0	1
98415	5378.0	1
128864	7375.0	1

this 192 user or selected_columns_df is list of user those are adopted user they visited 3 diffrent days in latest last 7days

```
user_eng.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 134897 entries, 0 to 134896
Data columns (total 4 columns):
 #   Column        Non-Null Count   Dtype  
---  ------        --------------   -----  
 0   time_stamp    134897 non-null  object 
 1   user_id       134896 non-null  float64
 2   visited       134896 non-null  float64
 3   Visited_Date  134897 non-null  object 
dtypes: float64(2), object(2)
memory usage: 4.1+ MB
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12000 entries, 0 to 11999
Data columns (total 10 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   object_id                   12000 non-null  int64  
 1   creation_time               12000 non-null  object 
 2   name                        12000 non-null  object 
 3   email                       12000 non-null  object 
 4   creation_source             12000 non-null  object
 5   last_session_creation_time  8823 non-null   float64
 6   opted_in_to_mailing_list    12000 non-null  int64
 7   enabled_for_marketing_drip  12000 non-null  int64
 8   org_id                      12000 non-null  int64
 9   invited_by_user_id          6417 non-null   float64
dtypes: float64(2), int64(4), object(4)
memory usage: 937.6+ KB
```

```
df.head()
```

	object_id	creation_time	name	email	creation_source	last_session_creation_time	opted_in_to_mailing_list	enabled_for_market
0	1	2014-04-22 03:53:30	Clausen August	AugustCClausen@yahoo.com	GUEST_INVITE	1.398139e+09	1	
1	2	2013-11-15 03:45:04	Poole Matthew	MatthewPoole@gustr.com	ORG_INVITE	1.396238e+09	0	
2	3	2013-03-19 23:14:52	Bottrill Mitchell	MitchellBottrill@gustr.com	ORG_INVITE	1.363735e+09	0	
3	4	2013-05-21 08:09:28	Clausen Nicklas	NicklasSClausen@yahoo.com	GUEST_INVITE	1.369210e+09	0	
4	5	2013-01-17 10:14:20	Raw Grace	GraceRaw@yahoo.com	GUEST_INVITE	1.358850e+09	0	

```
df.isnull().sum()
```

```
object_id                        0
creation_time                    0
name                             0
email                            0
creation_source                  0
last_session_creation_time    3177
opted_in_to_mailing_list         0
enabled_for_marketing_drip       0
org_id                           0
invited_by_user_id            5583
dtype: int64
```

```
data=df
```

```
#Working on date time
```

```
data['Creation_Date'] = data['creation_time'].apply(lambda x: x.split(' ')[0])
data['Creation_Time'] = data['creation_time'].apply(lambda x: x.split(' ')[1] )
```

```
data.head()
```

	object_id	creation_time	name	email	creation_source	last_session_creation_time	opted_in_to_mailing_list	enabled_for_market
0	1	2014-04-22 03:53:30	Clausen August	AugustCClausen@yahoo.com	GUEST_INVITE	1.398139e+09	1	
1	2	2013-11-15 03:45:04	Poole Matthew	MatthewPoole@gustr.com	ORG_INVITE	1.396238e+09	0	
2	3	2013-03-19 23:14:52	Bottrill Mitchell	MitchellBottrill@gustr.com	ORG_INVITE	1.363735e+09	0	
3	4	2013-05-21 08:09:28	Clausen Nicklas	NicklasSClausen@yahoo.com	GUEST_INVITE	1.369210e+09	0	
4	5	2013-01-17 10:14:20	Raw Grace	GraceRaw@yahoo.com	GUEST_INVITE	1.358850e+09	0	

```
X=data
```

```
X['Creation_day'] = X['Creation_Date'].apply(lambda x: x.split('-')[0])
X['Creation_month'] = X['Creation_Date'].apply(lambda x: x.split('-')[1])
X['Creation_year'] = X['Creation_Date'].apply(lambda x: x.split('-')[2])

X['Creation_hour'] = X['Creation_Time'].apply(lambda x: x.split(':')[0])
X['Creation_minutes'] = X['Creation_Time'].apply(lambda x: x.split(':')[1])
```

```
X.head()
```

	object_id	creation_time	name	email	creation_source	last_session_creation_time	opted_in_to_mailing_list	enabled_for_market
0	1	2014-04-22 03:53:30	Clausen August	AugustCClausen@yahoo.com	GUEST_INVITE	1.398139e+09	1	
		2013-11-15	Poole					
		10:14:20	Grace					

```
X.columns
```

```
Index(['object_id', 'creation_time', 'name', 'email', 'creation_source',
       'last_session_creation_time', 'opted_in_to_mailing_list',
       'enabled_for_marketing_drip', 'org_id', 'invited_by_user_id',
       'Creation_Date', 'Creation_Time', 'Creation_day', 'Creation_month',
       'Creation_year', 'Creation_hour', 'Creation_minutes'],
      dtype='object')
```

```
#Selecting required column

df_new = X[['object_id', 'creation_source','opted_in_to_mailing_list','enabled_for_marketing_drip','org_id','invited_by_user_id', 'Creation_day',
       'Creation_month', 'Creation_year', 'Creation_hour', 'Creation_minutes']]
```

```
df_new.head()
```

	object_id	creation_source	opted_in_to_mailing_list	enabled_for_marketing_drip	org_id	invited_by_user_id	Creation_day	Creation_month	Creation_
0	1	GUEST_INVITE	1	0	11	10803.0	2014	04	
1	2	ORG_INVITE	0	0	1	316.0	2013	11	
2	3	ORG_INVITE	0	0	94	1525.0	2013	03	
3	4	GUEST_INVITE	0	0	1	5151.0	2013	05	
4	5	GUEST_INVITE	0	0	193	5240.0	2013	01	

```
df_new.shape
```

```
(12000, 11)
```

```
df_new['object_id'].unique()
```

```
array([    1,     2,     3, ..., 11998, 11999, 12000])
```

```
# out of 12000 only 190 User comes under Adopted User
```

Joining User details & User engagement

```
data_join=df_new.merge(Adopted_UID, on='object_id', how='left')
```

```
data_join.head()
```

	object_id	creation_source	opted_in_to_mailing_list	enabled_for_marketing_drip	org_id	invited_by_user_id	Creat
0	1	GUEST_INVITE	1	0	11	10803.0	
1	2	ORG_INVITE	0	0	1	316.0	
2	3	ORG_INVITE	0	0	94	1525.0	
3	4	GUEST_INVITE	0	0	1	5151.0	
4	5	GUEST_INVITE	0	0	193	5240.0	

```
data_join.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 12000 entries, 0 to 11999
Data columns (total 12 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   object_id                   12000 non-null  int64  
 1   creation_source             12000 non-null  object 
 2   opted_in_to_mailing_list    12000 non-null  int64  
 3   enabled_for_marketing_drip  12000 non-null  int64  
 4   org_id                      12000 non-null  int64  
 5   invited_by_user_id          6417 non-null   float64
 6   Creation_day                12000 non-null  object 
 7   Creation_month              12000 non-null  object 
 8   Creation_year               12000 non-null  object 
 9   Creation_hour               12000 non-null  object 
 10  Creation_minutes            12000 non-null  object 
 11  Adopted_User                192 non-null    float64
dtypes: float64(2), int64(4), object(6)
memory usage: 1.2+ MB
```

```
data_join['Adopted_User'] = data_join['Adopted_User'].fillna(0).astype(int)
```

```
data_join['Adopted_User'].unique()
```

```
array([0, 1])
```

```
data_join.head()
```

	object_id	creation_source	opted_in_to_mailing_list	enabled_for_marketing_drip	org_id	invited_by_user_id	Creation_day	Creation_month	Creation_
0	1	GUEST_INVITE	1	0	11	10803.0	2014	04	
1	2	ORG_INVITE	0	0	1	316.0	2013	11	
2	3	ORG_INVITE	0	0	94	1525.0	2013	03	
3	4	GUEST_INVITE	0	0	1	5151.0	2013	05	
4	5	GUEST_INVITE	0	0	193	5240.0	2013	01	

```
#NumberofUserAccountCreated Vs Source
```

```
Source_Vs_User_Account = data_join[['creation_source','object_id','invited_by_user_id']].groupby(['creation_source']).count().sort_values(['creation_source
```

```
Source_Vs_User_Account
```

	object_id	invited_by_user_id
creation_source		
SIGNUP_GOOGLE_AUTH	1385	0
SIGNUP	2087	0
PERSONAL_PROJECTS	2111	0
ORG_INVITE	4254	4254
GUEST_INVITE	2163	2163

```
#Source_Vs_AdoptedUser_Vs_NumberofUserAccount
```

```
Source_Vs_AdoptedUser_Vs_UserAccount = data_join[['creation_source','object_id','Adopted_User']].groupby(['creation_source','Adopted_User']).count().sort_v
```

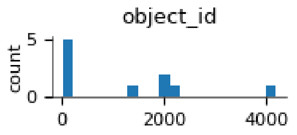
```
Source_Vs_AdoptedUser_Vs_UserAccount
```

		object_id
creation_source	Adopted_User	
PERSONAL_PROJECTS	1	22
SIGNUP_GOOGLE_AUTH	1	25
GUEST_INVITE	1	34
SIGNUP	1	42
ORG_INVITE	1	69
SIGNUP_GOOGLE_AUTH	0	1360
SIGNUP	0	2045
PERSONAL_PROJECTS	0	2089
GUEST_INVITE	0	2129
ORG_INVITE	0	4185

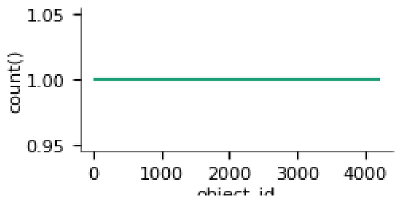
Values

object_id
2500
0
(PERSONAL_PROJECTS, 1)
(SIGNUP_GOOGLE_AUTH, 0)
creation_source,Adopted_User

Distributions



Time series



```
# From Above 2 output we can say that GUEST_INVITE & ORG_INVITE or more effective for more user engagement and also effective for number of adopted user
# SignUp method is 2nd most effective source for adopted user
```

```
#Numberof_Account_Vs_marketing_drip_Vs_mailing_list_Vs_Adopted_User
```

```
Numberof_Account_Vs_marketing_drip_Vs_mailing_list_Vs_Adopted_User = data_join[['object_id','enabled_for_marketing_drip','opted_in_to_mailing_list','Adopte
```

```
Numberof_Account_Vs_marketing_drip_Vs_mailing_list_Vs_Adopted_User
```

			object_id
enabled_for_marketing_drip	**opted_in_to_mailing_list**	**Adopted_User**	
1	**0**	**1**	3
	1	**1**	22
0	**1**	**1**	34
	0	**1**	133
1	**0**	**0**	447
	1	**0**	1320
0	**1**	**0**	1618
	0	**0**	8423

```
# from Above Table

#Highest Adopted user number, 133 User neither opted marketing drip nor mailing list
#Out of total Adopted user, 22 opted for mailing list and marketing drip
#34 only opted for mailing list
```

Making of Machine Learning Model

```
data_join.head()
```

	object_id	creation_source	opted_in_to_mailing_list	enabled_for_marketing_drip	org_id	invited_by_user_id	Creation_day	Creation_month	Creation_
0	1	GUEST_INVITE	1	0	11	10803.0	2014	04	
1	2	ORG_INVITE	0	0	1	316.0	2013	11	

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

```
data_join['creation_source']=le.fit_transform(data_join['creation_source'])
```

```
data_join.head()
```

	object_id	creation_source	opted_in_to_mailing_list	enabled_for_marketing_drip	org_id	invited_by_user_id	Creation_day	Creation_month	Creation_
0	1	0	1	0	11	10803.0	2014	04	
1	2	1	0	0	1	316.0	2013	11	
2	3	1	0	0	94	1525.0	2013	03	
3	4	0	0	0	1	5151.0	2013	05	
4	5	0	0	0	193	5240.0	2013	01	

```
#final_Data=data_join.drop(columns=['org_id','invited_by_user_id'])

final_Data=data_join.drop(columns=['org_id','invited_by_user_id','Creation_day','Creation_month','Creation_year','Creation_minutes','Creation_hour'])
```

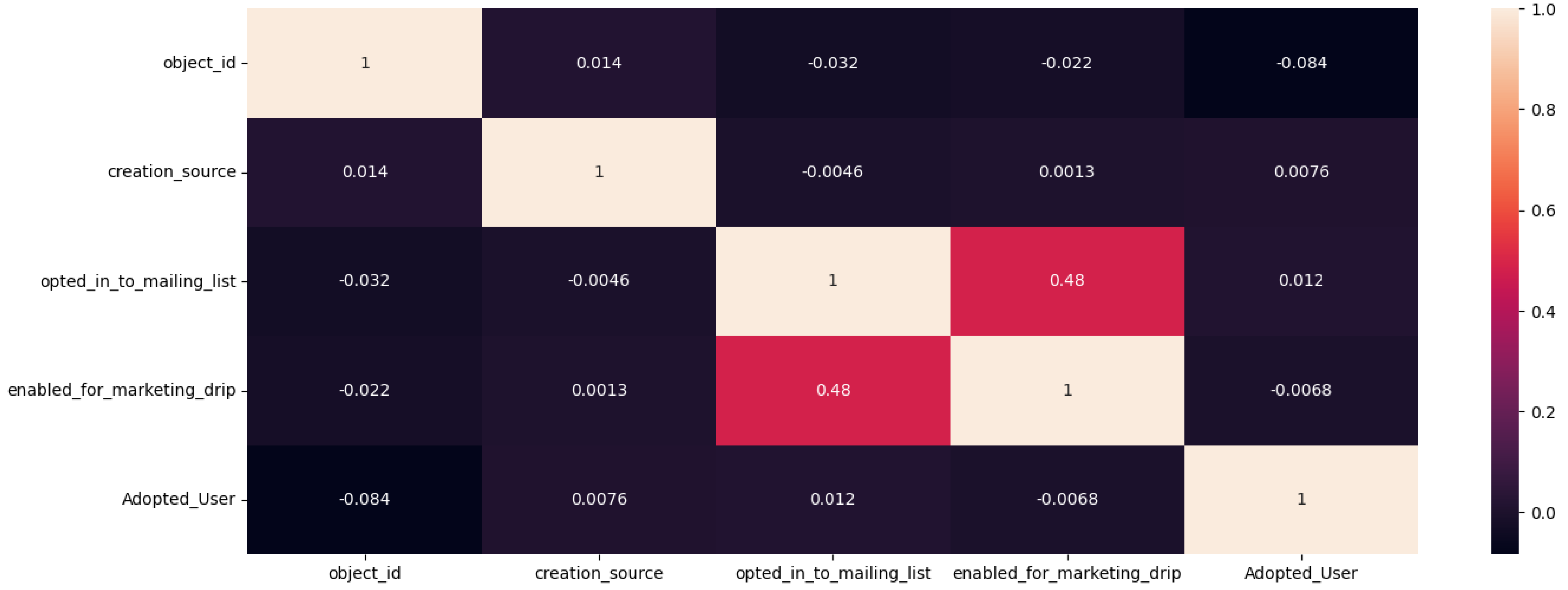
```
final_Data.head()
```

	object_id	creation_source	opted_in_to_mailing_list	enabled_for_marketing_drip	Adopted_User
0	1	0	1	0	0
1	2	1	0	0	0
2	3	1	0	0	0
3	4	0	0	0	0
4	5	0	0	0	0

```
#Heat Map

plt.figure(figsize=(16, 6))# plotting correlation heatmap
sns.heatmap(final_Data.corr(), annot = True)
```

```
    <Axes: >
```



```
# Saperating features and result vectors
X = final_Data.drop('Adopted_User', axis=1).values
y = final_Data['Adopted_User'].values
```

```
X.shape,y.shape
```

```
    ((12000, 4), (12000,))
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=42)
```

```
poly_features_1 = PolynomialFeatures(degree=1)
poly_features_2 = PolynomialFeatures(degree=2)
poly_features_3 = PolynomialFeatures(degree=3)


X_train_poly1 = poly_features_1.fit_transform(X_train)
X_train_poly2 = poly_features_2.fit_transform(X_train)
X_train_poly3 = poly_features_3.fit_transform(X_train)
```

Random Forest Model

```
rfc = RandomForestClassifier(n_estimators=100)
rfc.fit(x_train, y_train)
```

```
    ▾ RandomForestClassifier
    RandomForestClassifier()
```

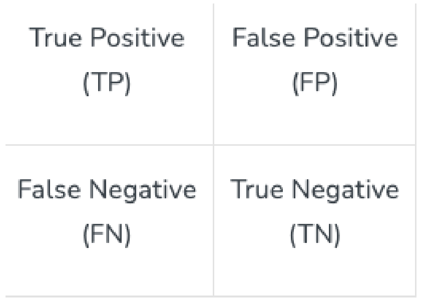
```
predict = rfc.predict(x_test)
acc = accuracy_score(predict, y_test)
pre = precision_score(predict, y_test)
rec = recall_score(predict, y_test)
f1 = f1_score(predict, y_test)


Random_Forest_Table = pd.DataFrame({
    'Metric': ['Accuracy', 'Precision', 'Recall', 'F1 Score'],
    'Score': [acc, pre, rec, f1]
    })
Random_Forest_Table
```

	Metric	Score
0	Accuracy	0.973889
1	Precision	0.019231
2	Recall	0.022727
3	F1 Score	0.020833

```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,predict)
```

```
    array([[3505,   43],
           [  51,    1]])
```



True Positive (TP)	False Positive (FP)
False Negative (FN)	True Negative (TN)

Feature Importance

```
X = final_Data.drop('Adopted_User', axis=1).values
y = final_Data['Adopted_User'].values
pp=final_Data.drop('Adopted_User', axis=1)
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state=42)
rfc = RandomForestClassifier(n_estimators=100)
rfc.fit(x_train, y_train)

feature_importances = pd.DataFrame(rfc.feature_importances_,index = pp.columns,columns=['importance']).sort_values('importance',ascending=False)
```

```
feature_importances
```

	importance
object_id	0.976227
creation_source	0.016476
enabled_for_marketing_drip	0.004312
opted_in_to_mailing_list	0.002986