JOB-A-THON

Smart Lead Scoring Engine

Problem Statement

A D2C startup develops products using cutting edge technologies like Web 3.0. Over the past few months, the company has started multiple marketing campaigns both offline and digital. As a result, the users have started showing interest in the product on the website. These users with intent to buy product(s) are generally known as leads (Potential Customers).

Leads are captured in 2 ways - Directly and Indirectly. Direct leads are captured via forms embedded in the website while indirect leads are captured based on certain activity of a user on the platform such as time spent on the website, number of user sessions, etc.

Now, the marketing & sales team wants to identify the leads who are more likely to buy the product so that the sales team can manage their bandwidth efficiently by targeting these potential leads and increase the sales in a shorter span of time.

The task is to predict the propensity to buy a product based on the user's past activities and user level information.

Data Understanding

'train.csv' contains the leads information of last 1 year from Jan 2021 to Dec 2021. And also, the target variable indicating if the user will buy the product in the next 3 months or not.

'test.csv' contains the leads information of the current year from Jan 2022 to March 2022. You need to predict if the lead will buy the product in the next 3 months or not.

Variable	**Description**
id	Unique identifier of a lead
created_at	Date of lead dropped
signup_date	Sign up date of the user on the website
campaign_var (1 and 2)	campaign information of the lead
products_purchased	No. of past products purchased at the time of dropping the lead
user_activity_var (1 to 12)	Derived activities of the user on the website
buy	0 or 1 indicating if the user will buy the product in next 3 months or not

Exploratory Data Analysis and Preprocessing

The training dataset contains **19 columns** and **39160 records**. Apart from 'signup_date' and 'created_at' date features, other features are numerical.

First, we checked the following basic information about the data

1. get statistical information of the numerical features
2. check for duplicates
3. check for missing values
4. analyze the target variable

There are no duplicates in the dataset. But we found null values for 'signup_date' and 'product_purchased' columns. This might have happened because of failure in recording the values due to human error or the system couldn't have captured the values. I thought to replace the missing values rather than removing the records as we may lose important information from remaining non-null values.

Imputation of 'null' values:

1. 'product_purchased' : There are **20911 null values** for this feature. If we check the non-null values, the number of past products purchased **ranges from 1 to 4**. So, I replace the null values with '0' as we can assume that we don't know the number of past products for these records.
2. 'signup_date': The date ranges from **February 2015 to March 2022**. I feel this is also an important feature. We found **15113 null values**. Mean replacement is done to impute the signup date.

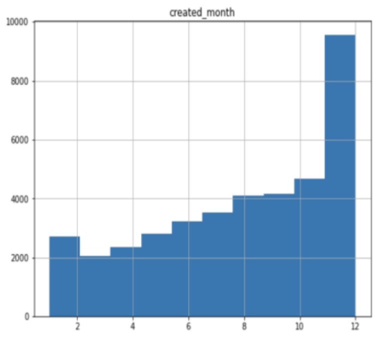
Analysis of target variable 'buy':

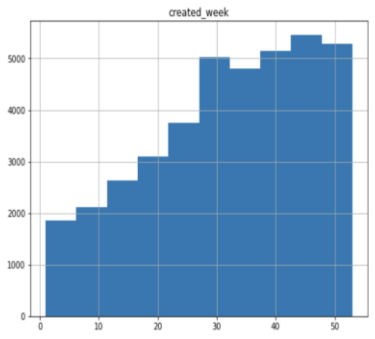
1. There are **37163 records for class 0**(number of users who didn't buy the product) and **1998 records for class 1(**number of users who bought the product). There is a high imbalance in the dataset with respect to the target variable.
2. We are using F1 score of Class 1 as an evaluation metric. So, there is **no need to balance the data** as if the classifier predicts the minority class but the prediction is erroneous and false-positive increases, the precision metric will be low and so as F1 score. Also, if the classifier identifies the minority class poorly, i.e., more of this class wrongfully predicted as the majority class then false negatives will increase, so recall and F1 score will be low. F1 score only increases if both the number and quality of prediction improves. F1 score keeps the balance between precision and recall and improves the score only if the classifier identifies more of a certain class correctly.

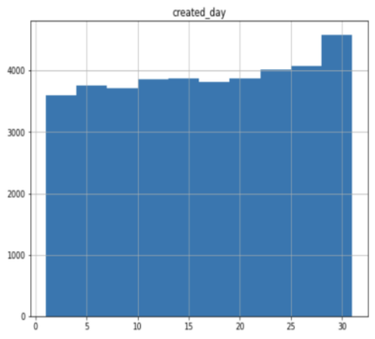
Feature Engineering

Deriving New features:

1. ‘time_delta’: It becomes very crucial to know the number of days between signu date(signup_date) and lead creation date(created_at).
2. ‘created_month’, ‘created_week’, ‘created_day’ : The ‘created_at’ **ranges from January 2021 to December 2021**. I analyzed lead creation month, week, day from the year 2021. It is increasing in a linear fashion each month. So, lead creation month and week become important as we are going to predict if a user will buy the product in the next 3 months or not.

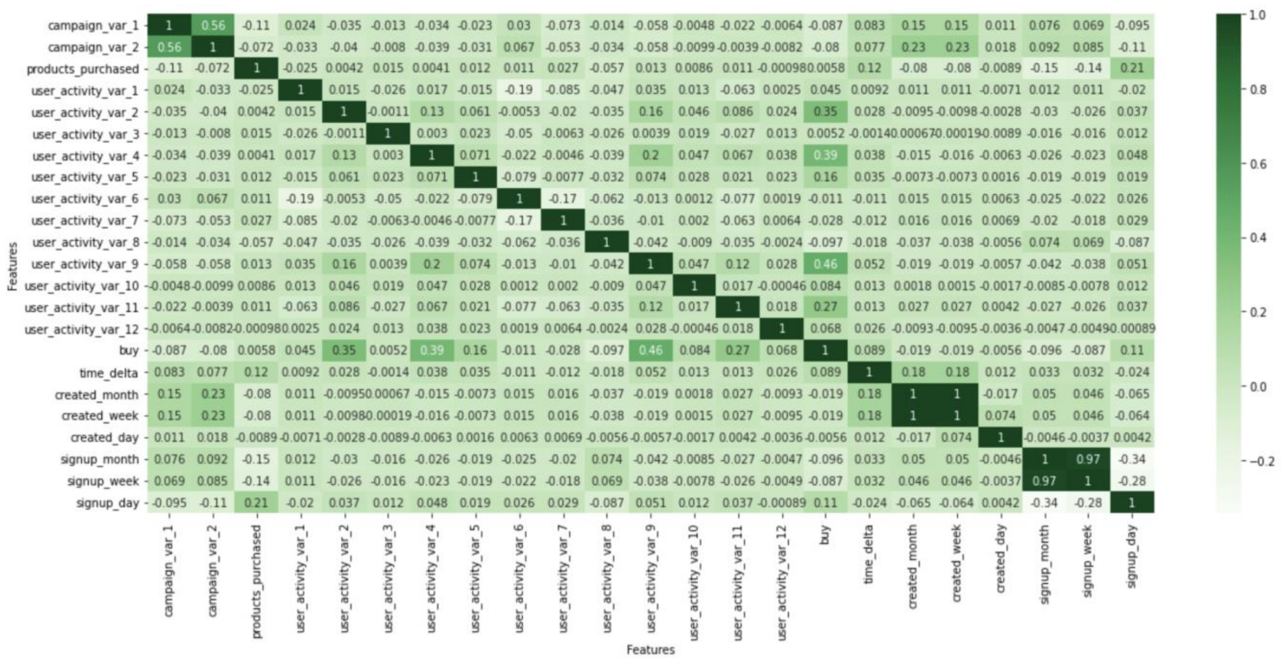






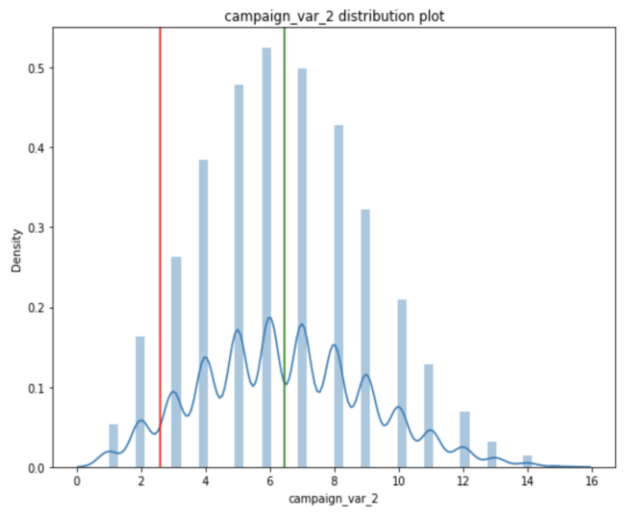
Important Conclusions:

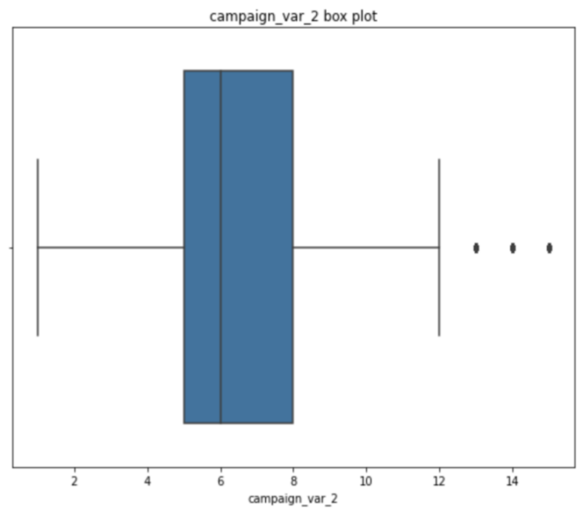
1. campaign_var_1 and campaign_var_2 are highly correlated with each other (corr = 0.56)
2. User activities 9,4,2 and 11 are highly correlated with target variable ‘buy’. (Positive Correlation)
3. created_month and created_week have correlation 1
5. Most of the features follow gaussian distribution and some of the features are left skewed. eg. campaign_var_1 and campaign_var_2
6. Also, time_delta (derived feature) has left skewed distribution. Boxplot of 'time_delta' clearly show some outliers in the feature.



Distribution plots are visualized to understand which features follow Gaussian Distribution.

Box plots are plotted to know the range of the features. It also helped to detect the outliers in the data.





I didn't think to remove the outliers as there are very few outliers present in the dataset. Further, I am using a tree based Random Forest Classifier for predictive modelling. The model handles the outliers implicitly.

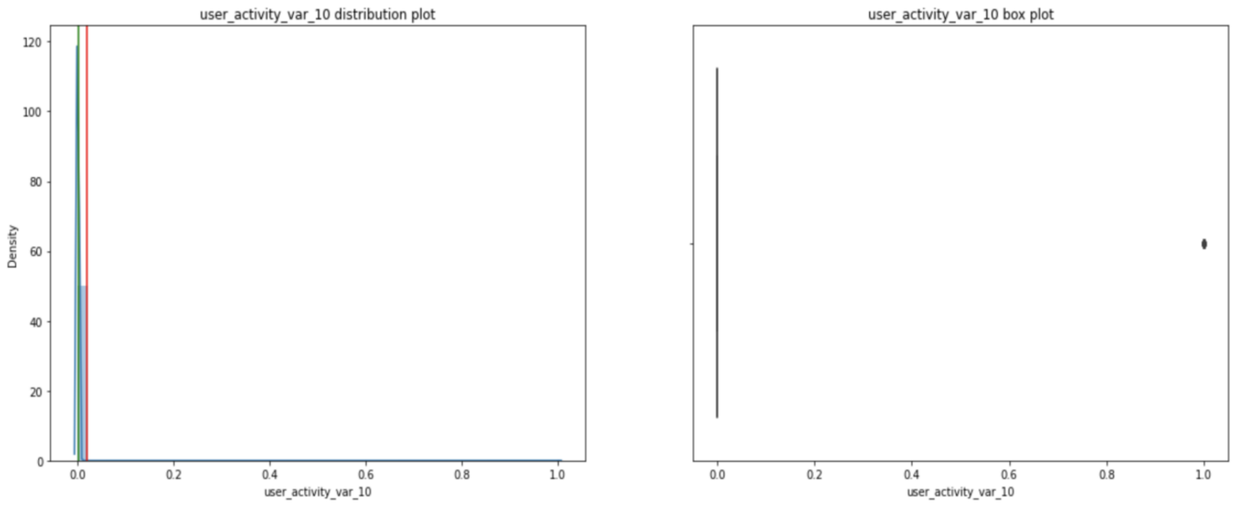
Feature Scaling:

1. Logistic Regression uses Gradient Descent as an optimization technique. Support Vector Machine is a distance-based algorithm. These need data to be scaled. I chose to standardize the data as most features follow Gaussian Distribution (Bell-Shaped Curve).
2. However, Random Forest Algorithm doesn't require data to be scaled as it is a tree-based classifier. It is fairly insensitive to the scale of the feature.

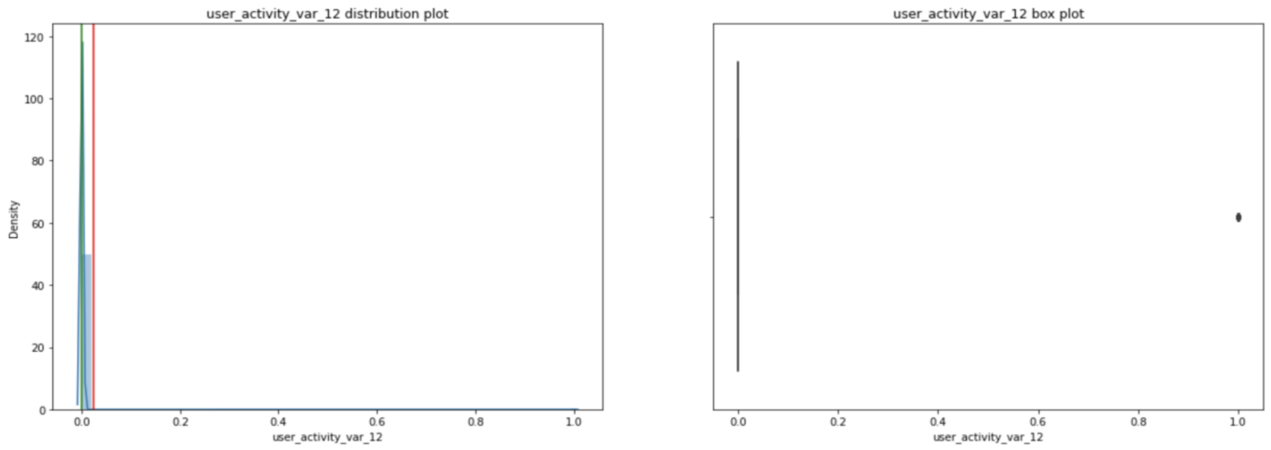
<u>Feature Reduction:</u>

If we visualize the following user activity features, the features don't add much value in the predictive modelling.

1. user_activity_var_10



2. user_activity_var_12



If we remove these from the feature matrix, accuracy doesn't drop. Also, I noticed a slight increase in F1-score of Class 1 (by ~1 percent).

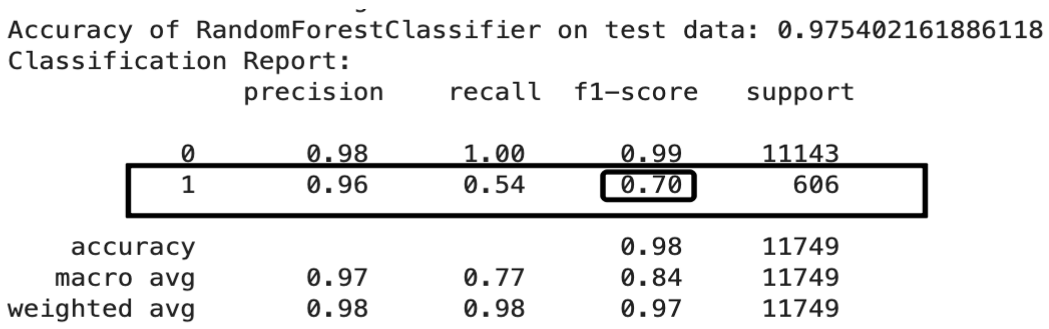
Predictive Modelling

This is a binary classification problem. I trained classification models using the following algorithms:

1. Random Forest Classifier (RF)
2. Support Vector Machine (SVM)
3. Logistic Regression (LR)

We split the data into 70:30 train-test proportion. Then apply 5-fold cross validation to ensure the model does not overfit. After, the classification report and confusion matrix are plotted for each model. It seems that Random Forest and Support vector Machine are performing equally.

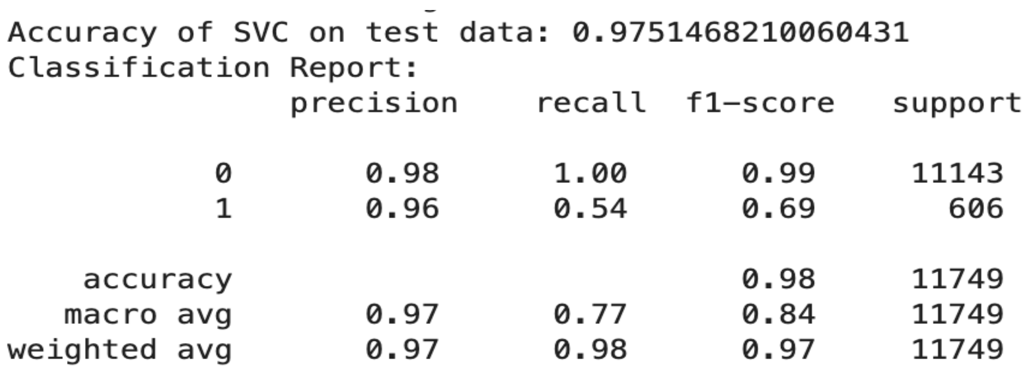
Below are the results for both the models on test data:



```
Accuracy of RandomForestClassifier on test data: 0.975402161886118
Classification Report:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99     11143
           1       0.96      0.54      0.70       606

    accuracy                           0.98     11749
   macro avg       0.97      0.77      0.84     11749
weighted avg       0.98      0.98      0.97     11749
```



```
Accuracy of SVC on test data: 0.9751468210060431
Classification Report:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99     11143
           1       0.96      0.54      0.69       606

    accuracy                           0.98     11749
   macro avg       0.97      0.77      0.84     11749
weighted avg       0.97      0.98      0.97     11749
```

We can see that F1-score for Class 1 is nearly the same i.e. ~70%.

Hyperparameter Tuning

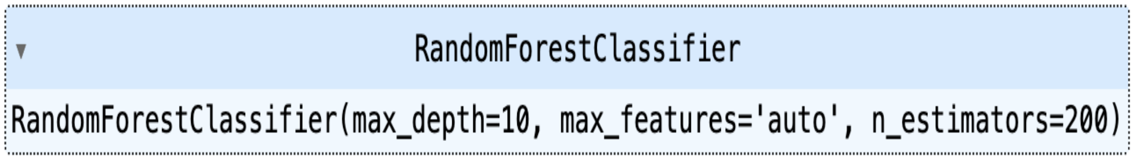
Keeping a few points in mind I chose Random Forest Classifier for further fine tuning.

1. Our data size is fairly high for two classes. The algorithm is a bagging ensemble technique that generates random samples from the dataset with row sampling.
2. Our dataset contains a few outliers and we don't want to remove these. Random forests are robust to outliers since they get averaged out by the aggregation of multiple tree outputs.
3. We don't require data to be scaled for the RF classifier.

I tuned following hyperparameters:

1. n_estimators (200, 300, 400, 500)
2. max_features ('auto', 'sqrt', 'log2')
3. max_depth (4,5,6,7,8,9,10)
4. criterion ('entropy', 'gini')

Following are the best parameters after Grid Search Cross Validation (n_folds = 5):



```
▾ RandomForestClassifier
RandomForestClassifier(max_depth=10, max_features='auto', n_estimators=200)
```

Conclusion

Random Forest Model performed fairly well on the test data.

Test Accuracy: 97.5%

F1-Score for Class 0: 99%

F1-Score for Class 1: 70%

Prediction on Test Data

The same preprocessing mechanisms are applied on test data that we applied for training data.

1. Impute null values
2. Feature Scaling

Finally, the results are stored in submission.csv file.