Predicting Online Shopper Behavior With ML Classification Models

By J. Michael Barbiere

OESON Learning Global Training and Internship Program

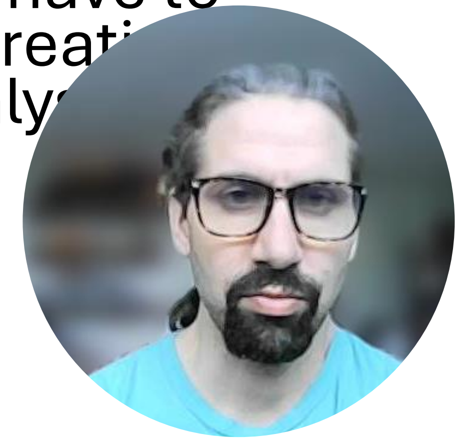
Description of The Problem

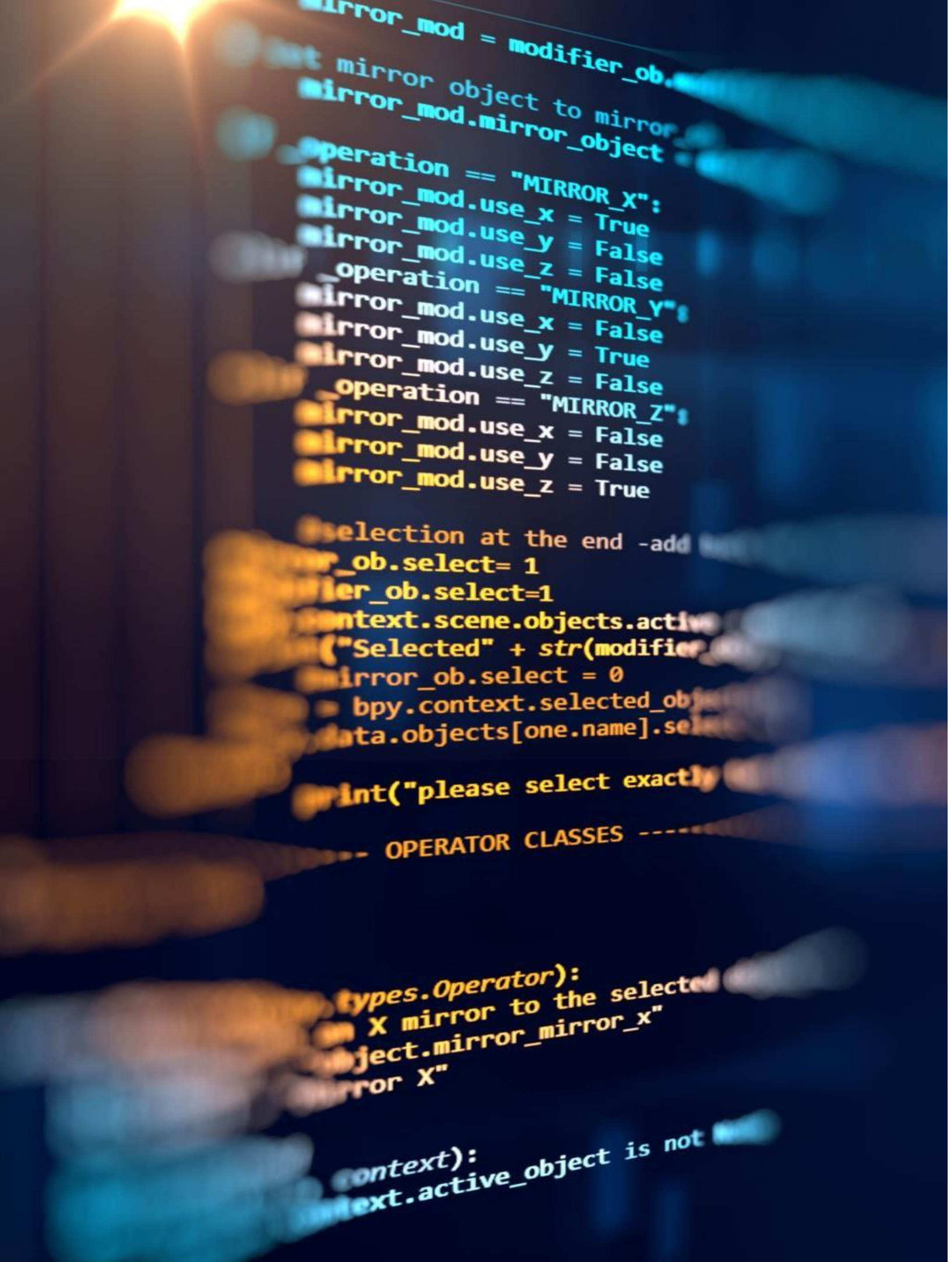
Data Source?

This dataset is a fictitious representation of a reputable online e-commerce platform. Every day, customers visit the website from various search engines, or by directly entering the URL, and browse the website's products.

Will they purchase?

A certain proportion of visiting customers make a purchase, but have to provide some demographic information as part of the account creati process. The products themselves are not important to this analy instead other variables that can help explain customer behavior (regarding the purchase-making decision).





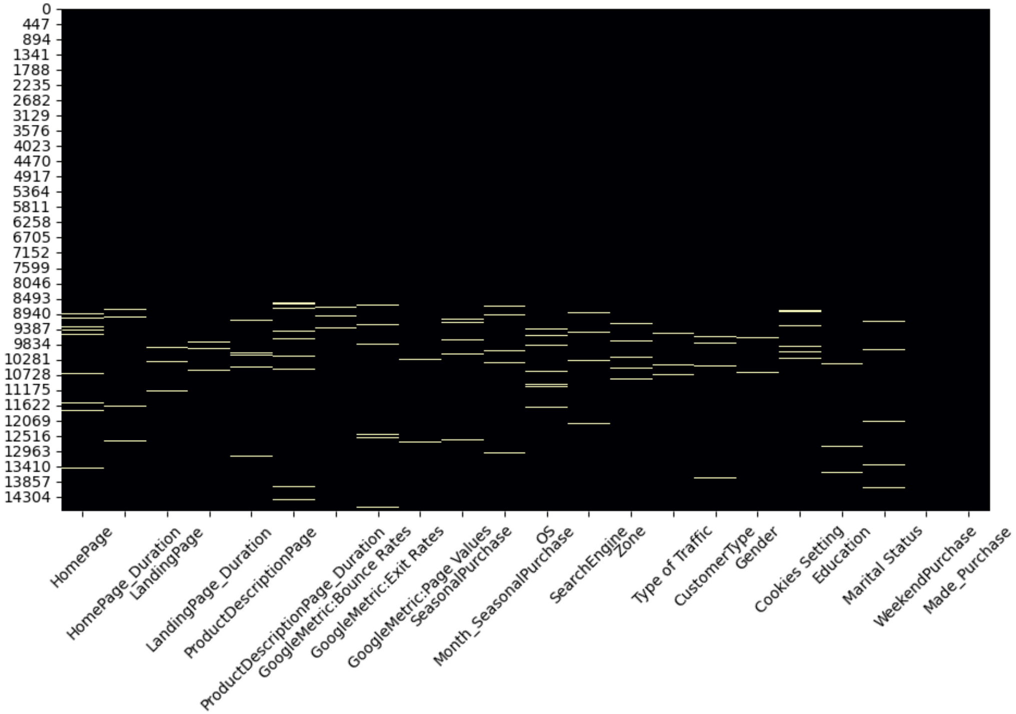
Dataset Description

Target Variable -- "Made Purchase"

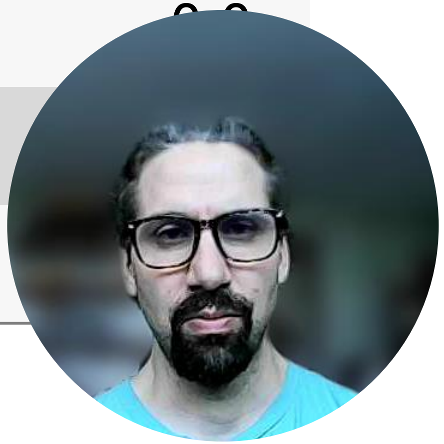
Explanatory Variables --
HomePage,
HomePage_Duration,
LandingPage,
LandingPage_Duration,
ProductDesriptionPage,
ProductDescriptionPage_Duration,
Bounce Rate,
Exit Rate, Page Value,
SeasonalPurchase,
SeasonalPurchase_Month,
OS, Search engine,
Time_Zone,
Visitor,
Gender, Cookies

Exploratory Data Analysis

- Shape of dataset (rows, columns) equals `(14731, 22)`.
- `Are there any missing values? Outliers? Mislabeled variables?`
- `.dropna() method`



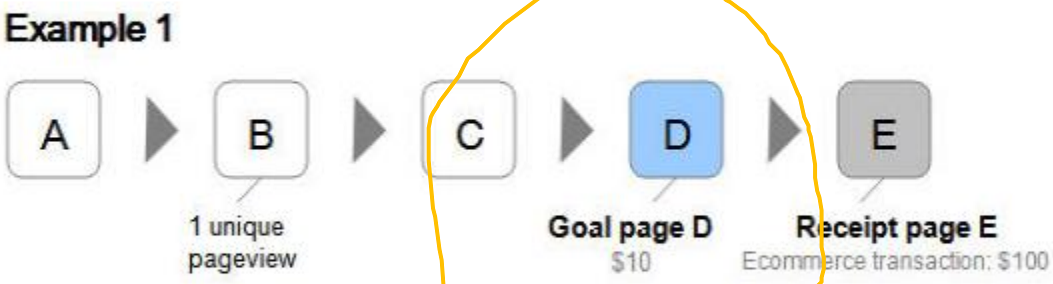
	TOTAL	%
homepage	0	0.0
home_page_duration	0	0.0
weekendpurchase	0	
marital_status	0	
education	0	

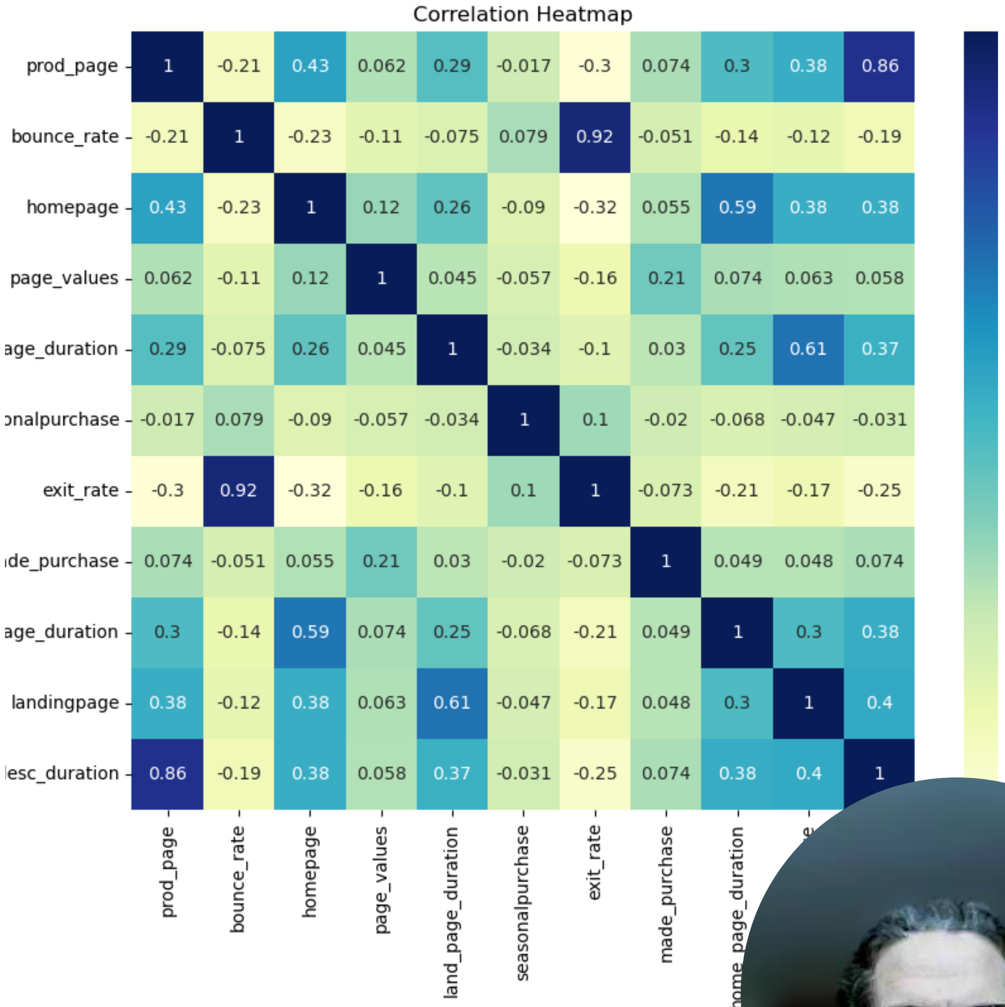


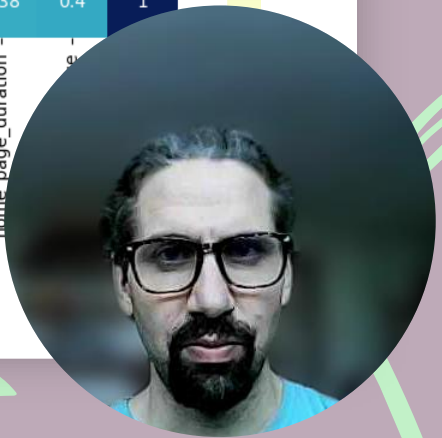
Correlation Matrix

- Used to determine which variables are correlated with each other.
- Page Value – 0.21% correlation

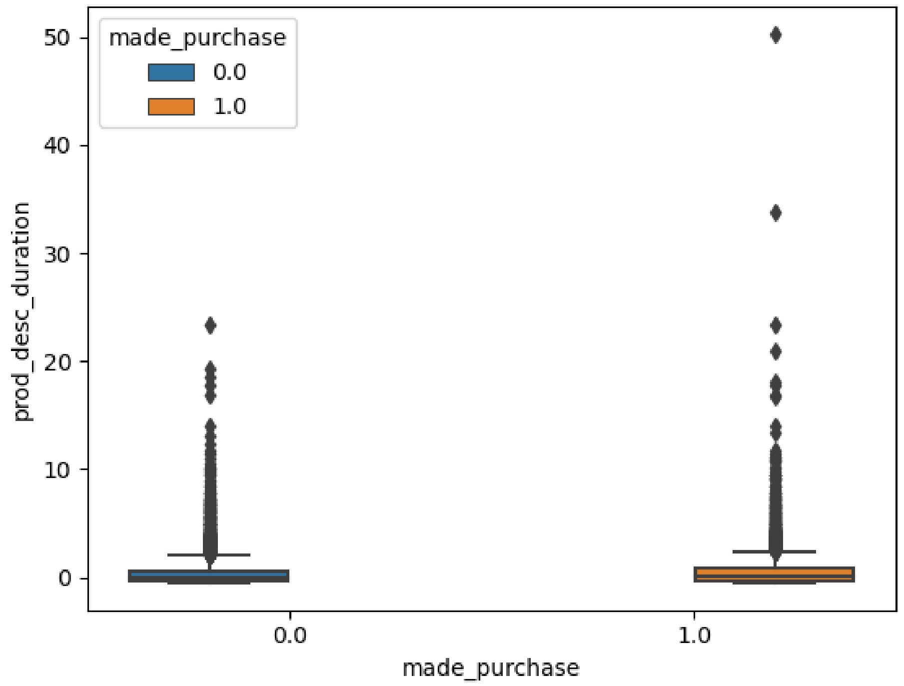
<u>**Ecommerce Revenue + Total Goal Value**</u>
<u>**Number of Unique Pageviews for Given Page**</u>

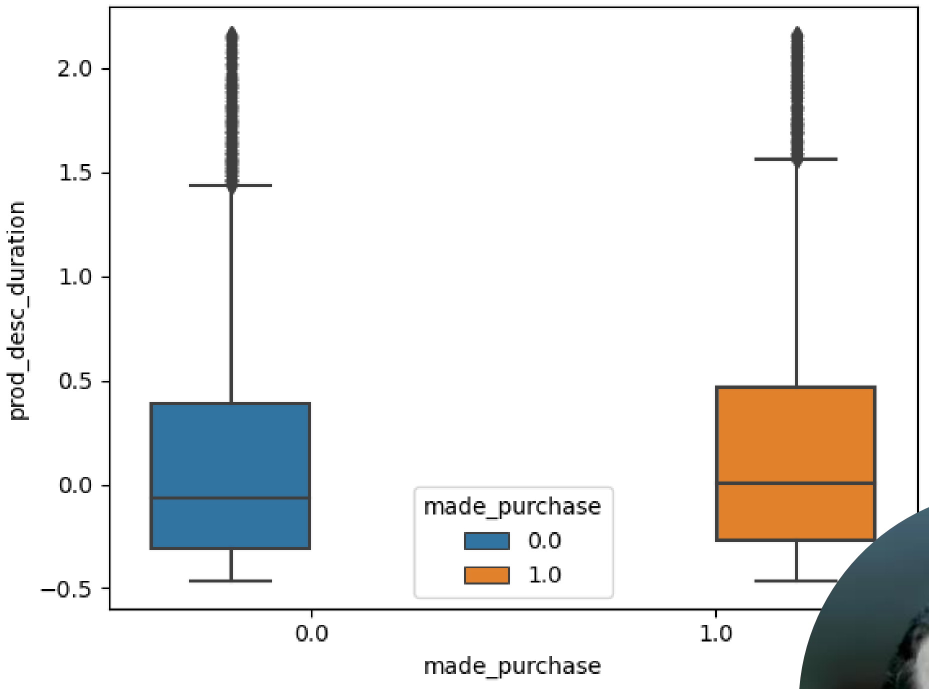






Outlier Detection and Removal (IQR Method)





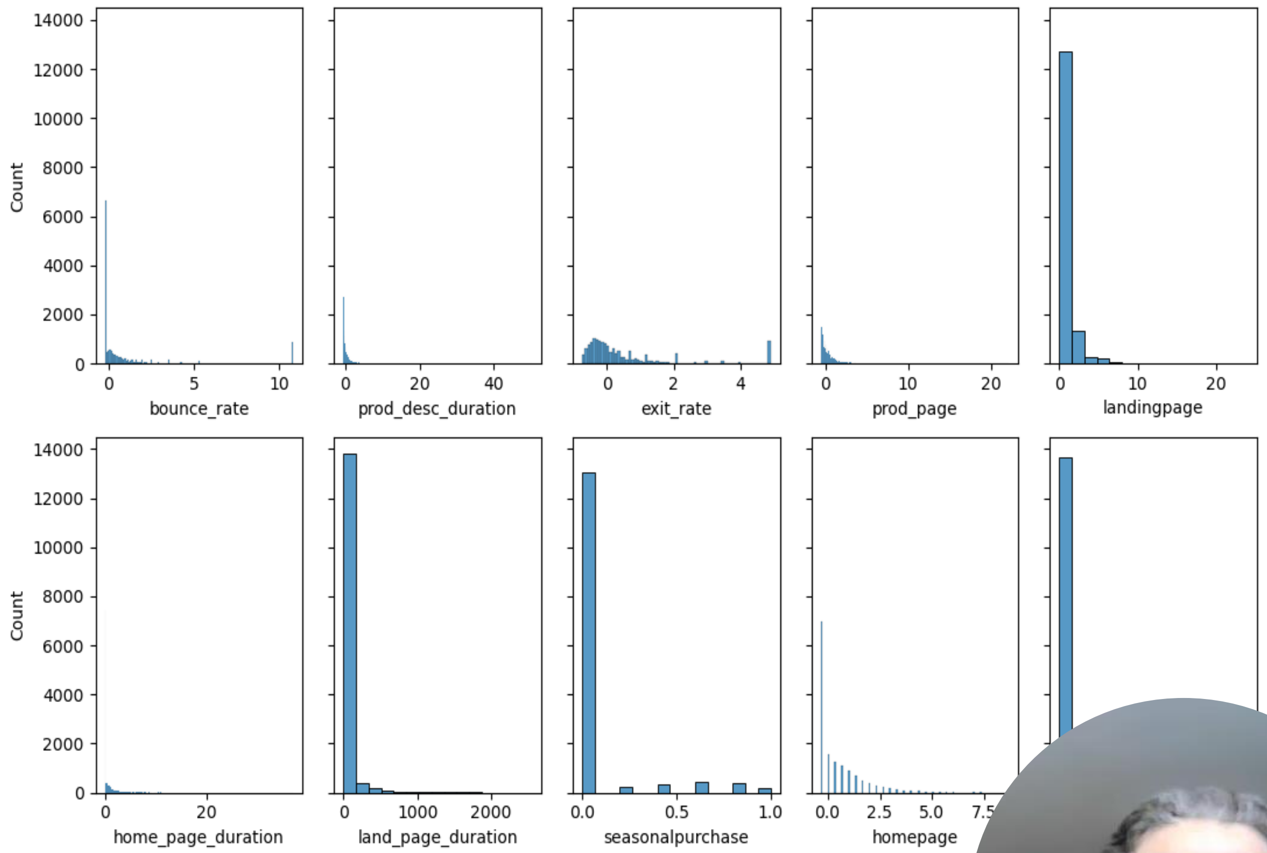

```
(8214, 22) --> New Shape of the dataset
The number of outliers dropped is: 3594
```

Robust Scaler

- Useful for handling datasets that have many outliers:

  value = (value – median) / (p75 – p25)

- The “with_centering” argument controls whether the value is centered to zero (median is subtracted) and defaults to True.
- The “with_scaling” argument controls whether the value is scaled to the IQR (standard deviation set to one) or not and defaults to True.



Create Dummy Variables

```
Index: 8214 entries, 2 to 14730
Data columns (total 11 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   weekendpurchase         8214 non-null   float64
 1   marital_status          8214 non-null   object
 2   education               8214 non-null   object
 3   cookies                 8214 non-null   object
 4   gender                  8214 non-null   object
 5   customertype            8214 non-null   object
 6   traffic_type            8214 non-null   float64
 7   zone                    8214 non-null   float64
 8   searchengine            8214 non-null   float64
 9   os                      8214 non-null   float64
 10  month_seasonalpurchase  8214 non-null   object
```

```
df_dummies = pd.get_dummies(
data=df_new,
 columns=['weekendpurchase', 'marital_status', 'education', 'cookies',
 'gender', 'customertype', 'traffic_type', 'zone',
'searchengine', 'os', 'month_seasonalpurchase'])
```

Model Assembly

- Determination of best classification models to use for the analysis.
    - **Logistic Regression –** binary classifier and prediction model.
    - **Decision Trees** -- supervised learning algorithm that can be used for both classification and regression tasks.
    - **Random Forest** -- an ensemble method that combines multiple decision trees to improve performance.
    - **Support Vector Machine** -- powerful classification algorithm that finds the best hyperplane to separate data points.
    - **Gaussian Naïve Bayes** -- a probabilistic classifier based on Bayes' theorem with the assumption of feature independence.

Results

- Peformance Metrics for Logistic Regression

- precision recall f1-score support
-
- 0.0 1.000 1.000 1.000 1701
- 1.0 1.000 1.000 1.000 1010
-
- accuracy 1.000 2711
- macro avg 1.000 1.000 1.000 2
- weighted avg 1.000 1.000 1.000

Overall Performances

- Accuracy - $\frac{\text{True Positives+True Negatives}}{\text{Total Instances}}$
- Jaccardi Score - $\frac{\text{True Positives}}{\text{True Positives + False Positives + False Negatives}}$
- F1 Score - $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision+Recall}}$

	TREE	FOREST	SVM	NB
Accuracy	1.0	1.0	1.0	1.0
Jaccardi Score	1.0	1.0	1.0	1.0
1 Score	1.0	1.0	1.0	1.0

Overfitting Concerns

Because all of the classification models scored perfectly on each of the performance metrics, there are some obvious concerns about having over-fit the sampling data

STRATEGIES

- **Regularization** – using Ridge and Lasso, add a cost function to reduce overfitting for the models.
  - **Effect on Coefficients**:
    - Lasso regularization can **shrink coefficients to exactly zero**.
    - It performs **feature selection**, effectively excluding less relevant features.
- **Feature Selection** – analyzing which variables are most impor and only including those; based upon one or more methods.

Results of Implementing Coefficient Penalty

Ridge

- print(ridge.score(X_train, y_train))
- print(ridge.score(X_test, y_test))

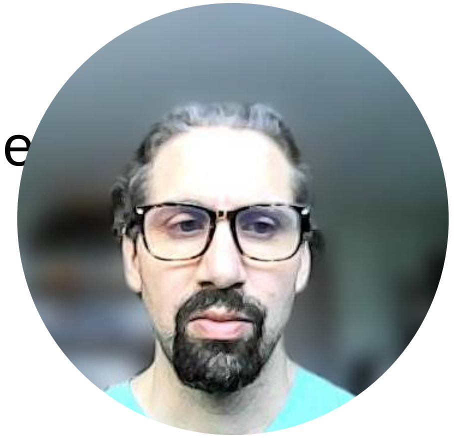
```
0.9999999430066427
0.9999999411590402
```

Lasso

- print(lasso.score(X_train, y_train))
- print(lasso.score(X_te=st, y_test))

```
0.821141370993343
0.8210478995200383
```

Lasso: particularly useful when feature sparsity is desired or when some
are expected to have no impact.

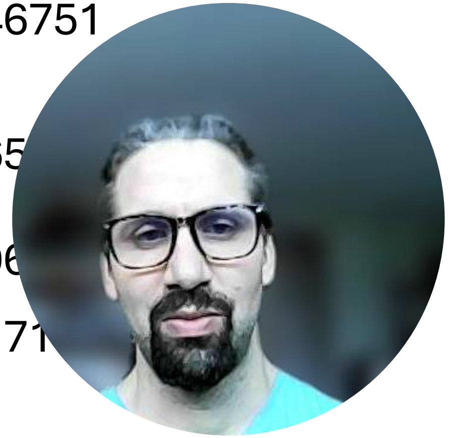


Feature Selection Using SelectKBest

- Determine column importance by using an Univariate Statistical Test
- Import SelectKBest from SciKitLearn and assign value to each column in the dataset.
- Remove all but the highest scoring features, to include in the model assembly.

Inference of Variables

	column	value
6	page_values	512.717203
35	exit_rate	41.439457
19	bounce_rate	15.982253
10	prod_desc_duration	14.774279
25	customertype_Returning_Visitor	13.995952
28	traffic_type_2.0	13.989402
44	customertype_New_Visitor	12.148054
67	marital_status_Single	10.846751
1	month_seasonalpurchase_Nov	9.9365
52	made_purchase	9.4506
84	traffic_type_4.0	7.18171



Results

	precision	recall	f1-score	support
False	0.671	0.952	0.787	1701
True	0.725	0.212	0.328	1010
accuracy			0.677	2711
macro avg	0.698	0.582	0.557	2711
weighted avg	0.691	0.677	0.616	2711

```
array([[1620,   81],
       [ 796,  214]], dtype=int64)
```

Addition Classification Models

	Tree	Forest	SVM	NB
Accuracy	0.676872	0.678716	0.682036	0.672814
Jaccardi Score	0.224092	0.238636	0.247818	0.188472
F1 Score	0.676872	0.678716	0.682036	0.672814

Conclusion

- After properly pre-processing the dataset, and fitting the above five classification models, it's been determined through several performance assessments that the models have substantial predictive ability; however, they were overfitting the dataset.
- Using Cross-Validation techniques, the models didn’t express any overfitting for repeated performances on different subsets of data.
- Tuning the model with Lasso technique reduced the accuracy to 82% though; which suggests that some feature pruning was necessary for this dataset.
- Manually selecting only the best 10 features resulted in a much more realistic accuracy for the models – 67%.
- The original objective has been satisfied - determining which customers are likely to make a purchase on the given e-commerce website; this is thanks to the features selection process using SelectKBest.
- Most important variables: Page Values, Exit Rate, Bounce Rate, Product Description Page, Returning Customer, etc.

Thank you!

Mike Barbiere

Oeson Learning: Data Science Internship

mike@analyzen.pro

03/17/2024