Preprocessing data for Predicting Online Shoppers Purchasing Intention

Pre-processing data for revenue predictor based on Machine Learning.

*From <*[*https://medium.com/analytics-vidhya/preprocessing-data-for-predicting-online-shoppers-purchasing-intention-ml-ba78186b7e85*](https://medium.com/analytics-vidhya/preprocessing-data-for-predicting-online-shoppers-purchasing-intention-ml-ba78186b7e85)*>*

Once a user logs into an online shopping website, knowing whether the person will make a purchase or not holds a massive economical value. A lot of current research is focused on real-time revenue predictors for these shopping websites. In this article, we will start building a revenue predictor for one such website. We will elaborate on the data pre-processing part here, and you can proceed to the [second article](https://medium.com/@isurudissanayake/ospi-mul-randomforests-156acdb73fd9) of the series for more details on the predictor model.

The data set can be found on kaggle- [Online shoppers intention](https://www.kaggle.com/roshansharma/online-shoppers-intention) — along with a detailed description of the features.

*The dataset consists of feature vectors belonging to 12,330 sessions. The dataset was formed so that each session would belong to a different user in a 1-year period to avoid any tendency to a specific campaign, special day, user profile, or period.*

What is pre-processing and why should we do it?

Every real world dataset contains incomplete and inconsistent data points. They might also lack in certain behaviors or trends, and is likely to contain many errors. Converting these data into a format that the predictor can understand in called pre processing. Every data scientist spends most of his/her time on pre-processing operations.

5% 
19% 
What data scientists spend the most time doing 
Building training sets: 3% 
Cleaning and organizing data: 60% 
Collecting data sets: 19% 
Mining data for patterns: 9% 
Refining algorithms: 4% 
Other: 5% 

credits- google

Importing libraries and data set.

The very first step in pre processing is importing the libraries. We used [pandas](https://pandas.pydata.org/) to import, export and maintain dataframes, and [numpy](https://numpy.org/) for matrix operations on the datset. [Sklearn](https://scikit-learn.org/stable/) was used for data analysis and making machine learning models as explained in the rest of the article. Matplotlib was used to plot and visualize data during various analyses.

import numpy as np 
import pandas as pd 
f rom 
f rom 
f rom 
f rom 
f rom 
f rom 
f rom 
f rom 
f rom 
sklearn . preprocessing import OneHotEncoder 
sklearn . impute import Simplelmputer 
sklearn . model selection import train_test_split 
sklearn. svm import SVC 
sklearn 
import metrics 
sklearn . metrics import classification _ report, confusion_matrix, 
sklearn . preprocessing import StandardSca1er 
sklearn . ensemble import RandomForestC1assifier 
sklearn . ensemble import IsolationForest 
accuracy score 
import matplotlib. pyplot as plt 

The dataset was then imported, and was separated into X(input features) and y(labels).

pd . read_csv( ' online_shoppers_intention . csv ' 
dataSet = 
m : dataSet.shape[e] 
features 
print(features) 
print (dataSet . Administrative . describe() ) 
X : dataset.iloc[•., 
Y dataset.iloc[•., 
[ 'Administrative ' 
17:18] 
' Administrative Duration ' 
' Informational ' , 
' Informational Duration ' 
' prod 
uctRe1ated' , 
, ' Month' , 
' Revenue'] 
'ProductRe1ated Duration ' 
' BounceRates ' 
' ExitRates ' 
' OperatingSystems ' 
' Browser ' 
' Region' , ' TrafficType ' 
' PageVa1ues ' 
' VisitorType ' 
' SpecialDa 
'Weekend' 
count 
mean 
std 
m In 
25% 
75% 
max 
Name : 
12316. eeeøøe 
2.317798 
3.322754 
e. eeeøøe 
e. eeeøøø 
1 . eeeøøø 
4. eeeøøe 
27. eeeøøe 
Administrative, dtype: 
float64 

Handling missing data points

There can be random missing data points in the dataset, which if not handled properly may raise errors later, or may lead to inaccurate inferences. First, we found out if there are any missing values. The value next to each feature name shows the number of missing data points per each column.

dataSet . isnu11() 
Administrative 
Administrative Duration 
Informational 
Informational Duration 
ProductRe1ated 
Product Related Duration 
BounceRates 
ExitRates 
PageVa1ues 
SpecialDay 
Month 
OperatingSystems 
Browser 
Region 
TrafficType 
VisitorType 
Weekend 
Revenue 
dtype: int64 
14 
14 
14 
14 
14 
14 
14 
14 
0 
0 
0 
0 
0 
0 
0 
0 
0 
0 

There are two ways to handle the missing values. Deleting the entire row with the missing data points, or fill the missing values with either the mean, median, mode or the most frequently appearing value in the corresponding column. Since only 12330 data points were available for us, we used the sklearn’s SimpleImputer function to replace the missing values with means- for numerical data and most frequent- for catagorical data.

xo - 
mf = 
Imp 
XO : • 
Imp 
Imp 
Imp 
Imp 
Imp 
mean 
— X.i10cC:, 
— X.i10cC:, 
— X.iloc[:, 
— X.i10cC:, 
— X.i10cC:, 
— X.iloc[:, 
— X.i10cC:, 
10:17] . values 
.values 
4:51 .values 
3:41 .values 
5:10] .values 
#month.... weekend 
#homepage 
outus 
#conta c tus 
#homepage_ time 
#aboutus time 
#contactus time.... speciaL_day 
nan, strategy: ' most_frequent ' ) 
mf. fit _ transform(xo) 
mf. fit _ transform (XI) 
mf. fit _ transform(X2) 
mf. fit _ transform(X3) 
= nan, 
strategy: ' mean ' 
X4[X4 < 0] = 
np.nan 
: imp_mean . fit _ transform(X4) 
X5[X5 < 0] : 
np.nan 
imp_mean . fit _ transform(X5) 
X6[X6 < 0] : 
np.nan 
X6 : imp_mean . fit _ transform(X6) 
X: 
X: 
X: 
X: 
X: 
X: 
np . append (XI , axis—I) 
np . append (X, X2 , axls=l) 
np . append (X, X5, axls=l) 
np . append (X, X3 , axls=l) 
np . append (X, X6 , axls=l) 
np . append (X, Xe, axls=l) 
df: np . append (X, y, axls=l) 

dataSet — 
' Weekend' ' Revenue' ] ) 
— pd. DataFrame(df, 
columns : 
' Administrative ' 
' Informational , 
' ProductRe1ated ' , 
' BounceRates ' 
' Administrative Duration ' 
' Informational Duration ' 
' ProductRe1ated Duration' , 
' ExitRates% 
' PageVa1ues ' 
' SpecialDay' , 'Month' , 
' OperatingSystems ' 
' Browser ' 
' Region' , 'TrafficType ' 
' VisitorType ' 
dataSet . isnu11() 
.sum() 
Administrative 
Administrative Duration 
Informational 
Informational Duration 
ProductRe1ated 
ProductRe1ated Duration 
80unceRates 
ExitRates 
PageVa1ues 
SpecialDay 
Month 
OperatingSystems 
Browser 
Region 
Traffic Type 
Visi torType 
Weekend 
Revenue 
dtype: int64 

Imputed dataset- no missing values

Handling catagorical data

The dataset consists of 10 numerical and 8 categorical attributes.

*In statistics, a categorical variable is a variable that can take on one of a limited, and usually fixed number of possible values, assigning each individual or other unit of observation to a particular group or nominal category on the basis of some qualitative property.*

As all the operations inside a Machine learning based predictor are mathematical, it’s clear that we can’t give inputs such as Months ; ‘January’, ‘February’ etc to the model. The easiest way to handle these type of data is [Label Encoding](https://medium.com/@contactsunny/label-encoder-vs-one-hot-encoder-in-machine-learning-3fc273365621), where each category in a particular attribute is encoded by a unique number; January=0, February=1 etc. (Check out our full code [here](https://github.com/Isuru-Dissanayake/OSPI-kaggle))

While this method yields acceptable results, the predictor model could also be biased towards some of the categories which has been encoded with a numerically higher value. (Eg; December=11 and January=0). To avoid this effect, we used [Onehot encoding](https://medium.com/@contactsunny/label-encoder-vs-one-hot-encoder-in-machine-learning-3fc273365621) for our dataset. After the encoding, initial 18 input features increased to 58.

Part of the dataset after onehot encoding of catagorical data

Selecting the best features

As we had 58 input features, we needed to select the features that had the largest effect on the revenue, and remove those that didn’t have considerable effect on the revenue. This step is highly important to enable faster training and to avoid complicating of the model unnecessarily. There are many tools to investigate the effect of each feature on the revenue. We used sklearn.ensemble’s [selectkbest](https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html) to find out the highest scoring features.

#seLecting the best features 
from sklearn . feature _ selection import SelectK8est 
from sklearn . feature_selection import chi2 
bestfeatures = k:1Ø) 
fit : 
pd . DataF rame(fit . scores 
dfscores = 
pd . DataF rame (X. columns) 
dfcolumns : 
pd . concat( Cdfcolumns, dfscores] , axis—I) 
featureScores : 
featureScores . columns = 
[ ' Specs' ' Score' ] #naming the dataframe columns 
print(featureScores. n1argest(1Ø, 'Score' ) ) #print 10 best features 
175126. 808512 
5 
8 
1 
3 
4 
2 
2€ 
53 
14 
Specs 
ProductRe1ated Duration 
PageVa1ues 
Administrative Duration 
Informational Duration 
ProductRe1ated 
Administrative 
Informational 
Month 11 
vi sitorl 
Months 
Score 
862583. 
469223 
4037. 253088 
34539. 1643€9 
19324. 711554 
1133. 965531 
358. 508157 
223. 548231 
IIS. 339482 
54. 9971€8 

Another function that we used for feature selections is Sklearn’s [ExtratreesClassifier](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html).

*This class implements a meta estimator that fits a number of randomized decision trees (a.k.a. extra-trees) on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.*

from sklearn . ensemble import ExtraTreesC1assifier 
import matplotlib . pyplot as plt 
model = ExtraTreesC1assifier() 
model . fit (X, y) 
#pLot graph of feature importances for better visualization 
pd . Series (model . featu re_importances 
feat_importances : 
index—X. columns 
feat _ importances . n1argest(20) . ' barh ' ) 
plt . show() 
/home/amaya/anaconda3/1ib/python3.7/ site-packages/sklearn/ensemble/forest . py : 245: FutureWarnin 
g: The default value of n _ estimators will change from 10 in version 0.2e to lee in 0.22. 
FutureWarning) 
"10 in version e. 20 to lee in e. 22. " 
Yowserl 
region2 
æekendl 
region4 
væekwnd2 
region3 
trowser2 
regionl 
Month 11 
Informational Duration 
Informational 
Traff,c Type 
aounceRates 
kiministrative Duratlon 
kimmistrative 
ProductRelated 
ExitRates 
productRelated Duration 
Pa3eValues 
coo 
005 
010 
015 
020 
025 

In order to visualize the extent of correlation among the input features, and that between the input features and the revenue, we used pandas’s [corr](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html)function.

import seaborn as sns 
f, ax = 8)) 
corr = dataSet. corr() 
sns . heatmap(corr, mask:np . 
squa re: True, ax—ax) 
dtype=np. bool ) , 
cmap=sns . 
10, 

Mministrative 
kiministrative Duration 
Informational - 
Informational Duration - 
ProductRelated - 
ProductRelated Duration - 
BounceRates 
ExitRates 
PageValues - 
*ecialDay 
CDeratingSystems 
Browser 
Region 
TrafficType 
Vkekend 
Revenue 
roo 
075 
" -0.50 
- 025 
coo 
—0.25 

From the above analysis, we selected 12 best features out of the 58. They were ‘Administrative’, ‘Administrative\_Duration’, ‘ Informational’, ‘Informational\_Duration’, ‘ ProductRelated’, ‘ProductRelated\_Duration’, ‘BounceRates’, ‘ExitRates’, ‘PageValues’, ‘Month11’, ‘Traffic\_Type’ and ‘visitor1’ (Month11 and visitor1 were results of one hot encoding, corresponding to month November and visitor type, returning visitor)

Outliers

*In statistics, an****outlier****is a****data****point that differs significantly from other observations. An****outlier****may be due to variability in the measurement or it may indicate experimental error; the latter are sometimes excluded from the****data****set.*

Before implementing the prediction model, we needed to investigate about any such outliers in our data set. We plotted our data in scatter plots, and we found something interesting.

*Most of the data points (customers) that were at a glace outliers, ie, had ridiculously large ProductRelated\_Duration or Informational\_Duration actually ended up buying something off the website.*

y = dataset[featuresC2]] #informationaL 
x = Ci for i in range(m)] 
plt . 
pit . show() 
25 
20 
15 
10 
5 
O 
2000 
4000 
EOOO 
moo 
10000 
12000 

y = dataset[featuresC8]] #pageVaLues 
x = Ci for i in range(m)] 
pit . 
pit . show() 
250 
200 
150 
2000 
4000 
8000 
10000 
12000 

So instead of cropping or deleting the outlying data points, we calculated an abnormality score for each customer, and introduced the score as a new feature to the predictor model.

m: dataset. shape[e] 
dataset . iloc[:, 
dataset . iloc[:, 
-1] 
clf = 
Out: c If. fit _ predict (X) 
Out 2: c If. score_samples (X) 
print (Out) 
print (Out2) 
Out2= 
Out 2: Out2. transpose() 
Out 2: np.reshape(Out2, (m, 1)) 
X: np. axis—I) 
Out: Out*(-l) 
Out: Out . transpose() 
Out: np.reshape(Out,(m,1)) 
X: np. axis—I) 
X. shape 

[1 1-1 
[ -ο .45341988 
-0.4290393 ] 
(12330, 60) 
-Θ. 39507024 
-ο. 50942328 . 
-ο. 40730611 
-Θ .37028447 

Two new features added based on their abnormality score

Train, validation and Test sets

We separated 1850 data points as test dataset. Then we used sklearn’s train-test-split feature to randomly separate a portion of the dataset as the val set and proceeded to the prediction model. We modified the prediction model until we get a satisfactory accuracy while using the val set to validate the results. Then we used the same model on the test dataset.

The final prediction accuracy was around 94%. You can read all about the model [here](https://medium.com/@isurudissanayake/ospi-mul-randomforests-156acdb73fd9) in the second article of the series, or check out the [code](https://github.com/Isuru-Dissanayake/OSPI-kaggle) in out Github repository here.

Cheers!

[Analytics Vidhya](https://medium.com/analytics-vidhya?source=post_sidebar--------------------------post_sidebar-)

Analytics Vidhya is a community of Analytics and Data Science professionals. We are building the next-gen data science ecosystem <https://www.analyticsvidhya.com>

Follow

174

* + [Machine Learning](https://medium.com/tag/machine-learning)
  + [Data Science](https://medium.com/tag/data-science)
  + [Pre Processing](https://medium.com/tag/pre-processing)
  + [Kaggle](https://medium.com/tag/kaggle)
  + [Sklearn](https://medium.com/tag/sklearn)

174 claps

WRITTEN BY

[Minikirani Amaya Dharmasiri](https://medium.com/@minikiraniamayadharmasiri?source=follow_footer--------------------------follow_footer-)

Follow

209

[Discover Medium](https://medium.com/about?autoplay=1&source=post_page-----ba78186b7e85----------------------)

Welcome to a place where words matter. On Medium, smart voices and original ideas take center stage - with no ads in sight. [Watch](https://medium.com/about?autoplay=1&source=post_page-----ba78186b7e85----------------------)

*From <*[*https://medium.com/analytics-vidhya/preprocessing-data-for-predicting-online-shoppers-purchasing-intention-ml-ba78186b7e85*](https://medium.com/analytics-vidhya/preprocessing-data-for-predicting-online-shoppers-purchasing-intention-ml-ba78186b7e85)*>*