ONLINE SHOPPER IN USING IBM WATSON

1. INTRODUCTION

1.1 Overview

Retail shopping is continuing to shift to E-commerce shopping and as a result the dynamics of shopping is changing around the world. E-commerce has already become a major form of retail market. Online customers often browse pages of e-commerce sites before they place orders or abandon their browsing without purchase. This information can help businesses to better cater to customer preferences and help both the business and customers mutually by recommending products specific to each customer and therefore increasing sales for the businesses. However, most of the time customers visiting these online websites may not make any purchase at all. This could be for various reasons i.e., Price of product or window shopping. It is important to predict customer's purchasing intention so that retention measures like e.g., recommending suitable products can be taken to convert potential customers into purchasers.

Currently, the closest existing solution to this problem has been the recommendation system. Where previous purchase information from a customer is processed to predict the types of products a customer would be interested in. There is evidence to suggest that retention measures such as an apt recommendation system plays an extremely important part in converting sales. Here the prediction of customer purchase intention can help strategize different marketing strategies and could be added to the mechanism of the recommendation system[13] of an ecommerce retailer.

An example could be that if the ML solution predicts a strong customer purchase intention, then maybe the recommend system could recommend a higher quality or a more expensive product as it can be inferred that the user would be willing to consider a better or more expensive product if their intention to purchase a type of product is very strong. If the solution predicts a low purchase intention, then recommendation system could referred products that are on discount or products with special offers i.e. "Buy one get one free". Later this historical data of how customer intention changes with such recommendation can also be studied and be applied to improve the recommendation system itself further. However, this report focuses only to the extent of predicting customer purchase intention.

1.2 Purpose

This project aims to use the information customers may leave in the form of the trace of browsing history data or user information when they visit an online shopping site. With the help of this information, the project aims to predict online shoppers' purchasing intention by using clickstream and session information data. The project aims to create a machine learning model based on this information to predict customer’s purchasing intension. The Objective of the project is to build a Machine Learning that can predict customer purchase intention as accurately as possible.

We will be using classification algorithms such as Logistic Regression, K-Means, Random Forest. We will train and test the data with these algorithms. From this, the best model is selected and saved in pkl format. We will also be deploying our model locally using Flask.

2. LITERATURE SURVEY

2.1 Existing Problem

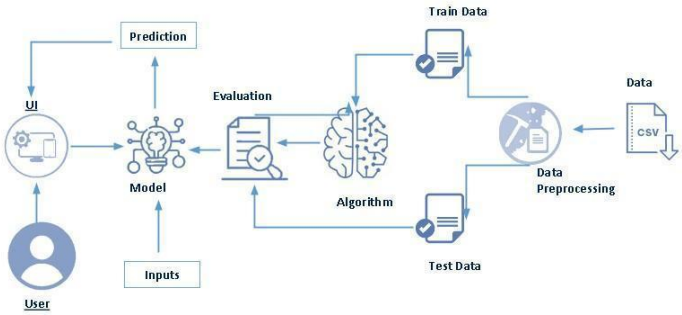
One of the problems in online shopping is predicting the behaviour of customers which will help to improve the quality of service. Unfortunately relatively few research studies in the literature attempted to tackle this problem; the majority of the research studies cannot achieve accurate prediction. Numerous machine learning techniques are suitable for prediction. Nevertheless, there is also shortage in adequate comparative studies that specify the most suitable techniques for the prediction process.

2.2 Proposed Solution

This Project presents a comparative study among few common techniques in the literature for predicting the customer behaviour. Those techniques are Logistic Regression, K-Means, Random Forest... The comparative study is based on realistic data gathered from a number of online shopping sites.

3. THEORITICAL ANALYSIS

3.1 Block Diagram



3.2 Hardware / Software Designing

To complete this project, you must required following software's, concepts and

packages

1. Anaconda navigator and pycharm:

a. Refer the link below to download anaconda navigator

b. Link : https://youtu.be/1ra4zH2G4o0

2. Python packages:

■ Open anaconda prompt as administrator

■ Type "pipinstallnumpy" and click enter.

■ Type “pipinstallpandas” and click enter.

■ Type “pipinstallscikit-learn” and click enter.

■ Type ”pipinstall matplotlib” and click enter.

■ Type ”pipinstallscipy” and click enter.

■ Type ”pipinstallpickle-mixin” and click enter.

■ Type ”pipinstallseaborn” and click enter.

■ Type “pipinstallFlask” and click enter.

4. EXPERIMENTAL INVESTIGATIONS

You must have prior knowledge of following topics to complete this project.

● ML Concepts

○ Supervised learning: https://www.javatpoint.com/supervised-machine-learning

○ Unsupervised learning: https://www.javatpoint.com/unsupervisedmachine-learning

○ Regression and classification

■ Decision tree: https://www.javatpoint.com/machine-learning-decision-treeclassification-algorithm

■ Random forest: https://www.javatpoint.com/machine-learning-random-forest-algorithm

■ KNN: https://www.javatpoint.com/k-nearest-neighbor-algorithm-formachine-learning

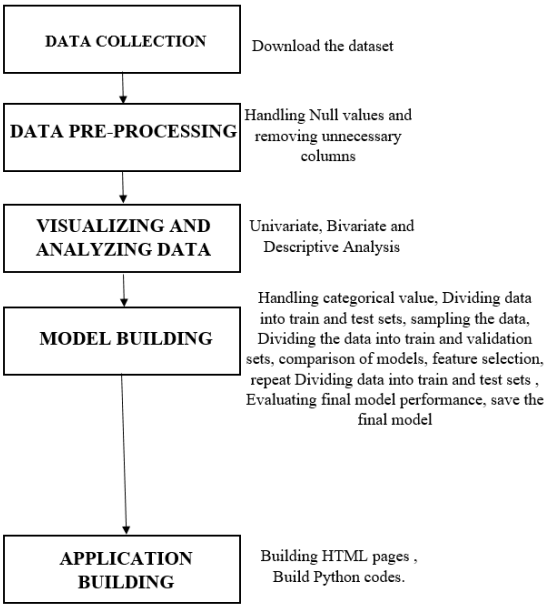
■ Xgboost: https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/

■ Evaluation metrics: https://www.analyticsvidhya.com/blog/2019/08/11-

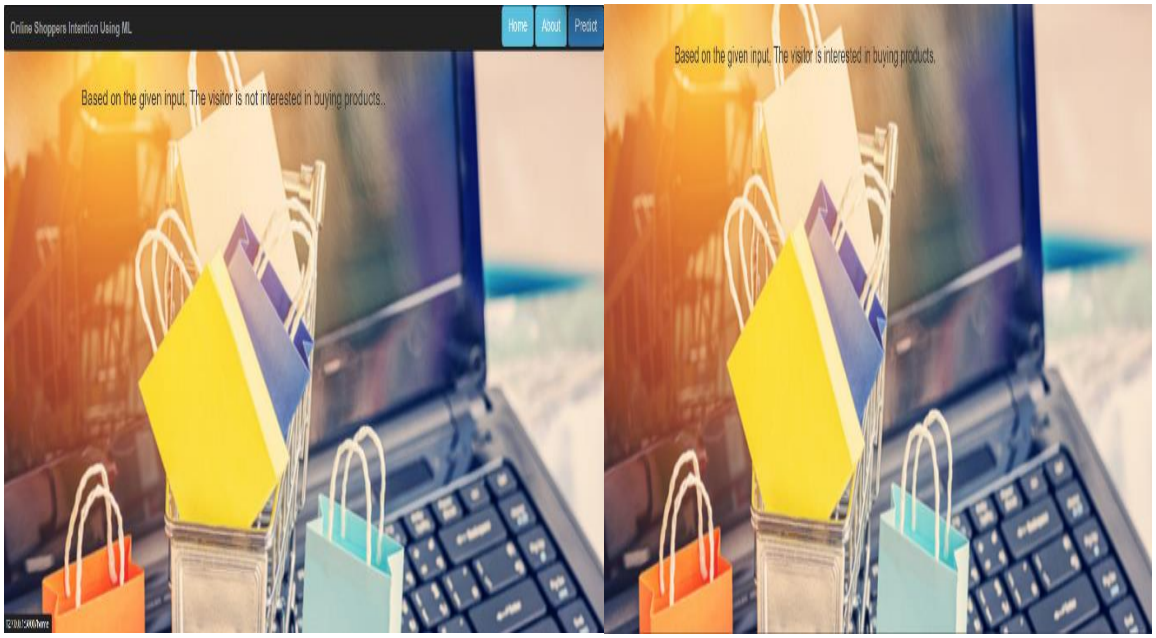
important-model-evaluation-error-metrics/

○ Flask Basics : https://www.youtube.com/watch?v=lj4I_CvBnt0

5. FLOWCHART



6. RESULT



The Project will predict whether the customer will buy the product or not.

7. ADVANTAGES AND DISADVANTAGES

Advantages

- Helps to provide better quality of services to customers.
- Easy to access
- Easy to predict whether a customer will purchase a product or not
- Predicting the behaviours of customers.

Disadvantage

- May not be that much familiar for normal people.
- Initial Implementation cost

8. APPLICATIONS

- Ecommerce
- Sales process
- Price prediction

9. CONCLUSIONS

The project aim was to build a solution that can predict customer purchase intention with as high an accuracy as possible. The highest accuracy It has managed to achieve was 89.9% accuracy . It was recorded as the highest accuracy by comparing and ranking the various model's performances with each other. It can be seen that Random Forest model performed the best among the various algorithms implemented. After testing the algorithm with various states of pre-processed data it can be concluded that different machine learning models would work better with different types of pre-processed data. I have shown the accuracy of the models based on different states of pre-processed at in the Results and Evaluation section. Comparing the results after the dataset had been cleaned for irregularities. The dataset was then transformed to suit the machine learning algorithms

10. FUTURE SCOPE

The customer purchase intention prediction can be combined to the ecommerce website's product recommendation system. Further work can be done to see if recommending products based on customer intension could have an impact on increases sales or not. I.e. Does recommending discounts and special deals to customer who have no intention to purchase choose to buy instead?

This can also be used to recommend more expensive or higher quality products to a customer if they already have a strong intention to purchase a similar product. As it is likely they would be open to consider other more expense alternative product . The Artificial Neural Network model I built has potential to be Hyperparameter optimized further. Perhaps, in the future if there are more hardware capabilities then one can perform a Random Search , Grid Search or Bayesian Optimization with granular parameter intervals to obtain best hyperparameter results. There is also the potential of implementing reinforcement learning to solve this problem and researching how Reinforcement Learning solution compares to Artificial Neural Networks and other algorithms used in this report. It would also be useful to collect and perform the research on a larger dataset by which algorithm performance can be measured better because of better training with a larger dataset.

11.BIBLIOGRAPHY

- **https://www.researchgate.net/publication/346846529_Using_machine_learning_to_predict_the_next_purchase_date_for_an_individual_retail_customer**
- **https://www.w3schools.com/css/css_rwd_templates.asp**
- **https://smartinternz.com/**

APPENDIX

app.py

```
from flask import Flask, render_template, request
import numpy as np
import pickle
model = pickle.load(open('rfos.pkl', 'rb'))
app = Flask(__name__)
@app.route("/")
def about():
    return render_template('home.html')
@app.route("/about")
def home():
    return render_template('about.html')
@app.route("/predict")
def home1():
    return render_template('predict.html')
@app.route("/submit")
def home2():
    return render_template('submit.html')
@app.route("/pred", methods=['POST'])
```

```
def predict():
    Administrative = request.form['Administrative']
    Administrative_Duration = request.form['Administrative_Duration']
    Informational = request.form['Informational']
    Informational_Duration = request.form['Informational_Duration']
    ProductRelated = request.form['ProductRelated']
    ProductRelated_Duration = request.form['ProductRelated_Duration']
    BounceRates = request.form['BounceRates']
    ExitRates = request.form['ExitRates']
    PageValues = request.form['PageValues']
    SpecialDay = request.form['SpecialDay']
    Month = request.form['Month']
    OperatingSystems = request.form['OperatingSystems']
    Browser = request.form['Browser']
    Region = request.form['Region']
    TrafficType = request.form['TrafficType']
    VisitorType = request.form['VisitorType']
    Weekend = request.form['Weekend']

    total = [[int(Administrative), float(Administrative_Duration), int(Informational), float(Informational_Duration),
            int(ProductRelated), float(ProductRelated_Duration), float(BounceRates), float(ExitRates),
            float(PageValues), float(SpecialDay), int(Month), int(OperatingSystems), int(Browser), int(Region),
            int(TrafficType), int(VisitorType), int(Weekend)]]
    print(total)
    prediction = model.predict(total)
    print(prediction)
    if prediction == 0:
        text = 'The visitor is not interested in buying products.'
    else:
        text = 'The visitor is interested in buying products'
    return render_template('submit.html', prediction_text=text)
if __name__ == "__main__":
    app.run(debug=False)
```

OSI.IPYNB

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
plt.style.use('fivethirtyeight')
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from sklearn.preprocessing import MinMaxScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.cluster import KMeans
from sklearn.metrics import classification_report,confusion_matrix
from sklearn.decomposition import PCA
from sklearn.model_selection import cross_val_score
import pickle
```

```
df=pd.read_csv(r'D:\Online shopper
intensions\Dataset\online_shoppers_intention.csv')
df.head()
df.info()
df.shape
plt.figure(figsize=(10,10))
plt.subplot(121)
df['Revenue'].value_counts().plot(kind='pie',autopct='%.1f%%')
plt.subplot(122)
df['VisitorType'].value_counts().plot(kind='pie',autopct='%.1f%%')
plt.figure(figsize=(16,4))
plt.subplot(131)
plt.scatter(df['Administrative'],df['Administrative_Duration'],color='r')
plt.subplot(132)
plt.scatter(df['Informational'],df['Informational_Duration'],color='g')
plt.subplot(133)
plt.scatter(df['ProductRelated'],df['ProductRelated_Duration'],color='y')
pd.crosstab(df['Revenue'],df['SpecialDay'])
pd.crosstab([df['Month'],df['VisitorType']],df['Revenue'])
df.describe(include='all')
df.isnull().sum()
le=LabelEncoder()
df['Month']=le.fit_transform(df['Month'])
df['VisitorType']=le.fit_transform(df['VisitorType'])
df['Weekend']=le.fit_transform(df['Weekend'])
df['Revenue']=le.fit_transform(df['Revenue'])
dfKmeans=df.drop('Revenue',axis=1)
scaler=MinMaxScaler()
scaled_df=scaler.fit_transform(dfKmeans)
dfKmeans=pd.DataFrame(scaled_df,columns=dfKmeans.columns)
dfKmeans.head()
n_cluster=range(1,10,1)
sse=[]
for i in n_cluster:
    k=KMeans(n_clusters=i)
    ypred=k.fit(scaled_df)
    sse.append(k.inertia_)
sse
plt.figure(figsize=(12,6))
plt.plot(n_cluster,sse,marker='*',markersize=20)
km=KMeans(n_clusters=4)
ypred=km.fit_predict(dfKmeans)
pca=PCA(n_components=2)
dfPCA=pca.fit_transform(dfKmeans)
dfPCA
dfPCA=pd.DataFrame(dfPCA,columns=['PCA 1','PCA 2'])
dfPCA.head()
```

```
dfPCA['cluster']=ypred
plt.figure(figsize=(10,10))
sns.scatterplot(dfPCA['PCA 1'],dfPCA['PCA
2'],hue=dfPCA['cluster'],palette=['red','yellow','green','blue'])
plt.scatter(km.cluster_centers_[:,0],km.cluster_centers_[:,1],color='black',s=300
,marker='*',label='centroid')
plt.legend()
x=df.drop('Revenue',axis=1)
y=df['Revenue']
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=10)
def logisticReg(x_train, x_test, y_train, y_test):
    lr=LogisticRegression()
    lr.fit(x_train,y_train)
    ypred=lr.predict(x_test)
    print('***LogisticRegresson***')
    print('Confusion matrix')
    print(confusion_matrix(y_test,ypred))
    print('Classification report')
    print(classification_report(y_test,ypred))
def randomForest(x_train, x_test, y_train, y_test):
    lr=RandomForestClassifier()
    lr.fit(x_train,y_train)
    ypred=lr.predict(x_test)
    print('***RandomForestClassifier***')
    print('Confusion matrix')
    print(confusion_matrix(y_test,ypred))
    print('Classification report')
    print(classification_report(y_test,ypred))
def compareModel(x_train, x_test, y_train, y_test):
    logisticReg(x_train, x_test, y_train, y_test)
    print('-'*100)
    randomForest(x_train, x_test, y_train, y_test)

compareModel(x_train, x_test, y_train, y_test)
rf=RandomForestClassifier()
rf.fit(x_train,y_train)
ypred=rf.predict(x_test)
cv=cross_val_score(rf,x,y,cv=5)
np.mean(cv)
pickle.dump(rf,open('rfos.pkl','wb'))
```