Smart Internz

Online Shoppers Intension

Team Members

Riyaz Mohammad Arbaz - 20BDS0274
Mir Mohammed Zain – 20BDS0211
Shaik Abdul Gafoor Ul Wadood - 20BCE2570
Tadikonda Sathvik Reddy – 20BBS0205

ABSTRACT

The project aims to explore online shoppers' intention using machine learning techniques. Online shopping has gained tremendous popularity in recent years, and understanding customers' intentions can provide valuable insights for businesses to improve their strategies. By analyzing various factors, such as user behavior, demographics, and browsing patterns, we can predict shoppers' intentions and tailor personalized experiences.

INTRODUCTION

In this project, various machine learning models have been utilized to develop an accurate prediction system for online shoppers' intention. The overarching goal is to gain insights into the factors that significantly influence their decision-making process. By leveraging this knowledge, businesses can make informed decisions to optimize their marketing strategies, enhance customer satisfaction, and ultimately increase their conversion rates.

The models employed in this project include Random Forest, Decision Tree, Logistic Regression, Naive Bayes, Support Vector Machines (SVM), and k-Nearest Neighbors (KNN). Each of these models has distinct characteristics, strengths, and weaknesses that make them suitable for tackling different aspects of the problem.

Throughout the project, each of these models has been evaluated and compared based on various performance metrics such as accuracy, precision, recall, and F1 score. This evaluation process helps identify the strengths and weaknesses of each model and provides insights into their applicability in predicting online shoppers' intention.

By successfully developing a reliable prediction model, businesses can gain a deeper understanding of their customers' behaviour and preferences. This knowledge can be leveraged to optimize marketing efforts, tailor product recommendations, personalize user experiences, and ultimately improve overall customer satisfaction. Additionally, the findings of this project may have broader implications for the field of e-commerce, with potential applications in areas such as targeted advertising, customer segmentation, and market trend analysis.

LITERATURE REVIEW

Existing Problem

Existing approaches to solving the problem of predicting online shoppers' intention typically involve traditional statistical analysis or rule-based techniques. These methods often have limitations in terms of accuracy and scalability. They rely on predefined rules or assumptions and may not capture the complexity of user behaviour and preferences accurately.

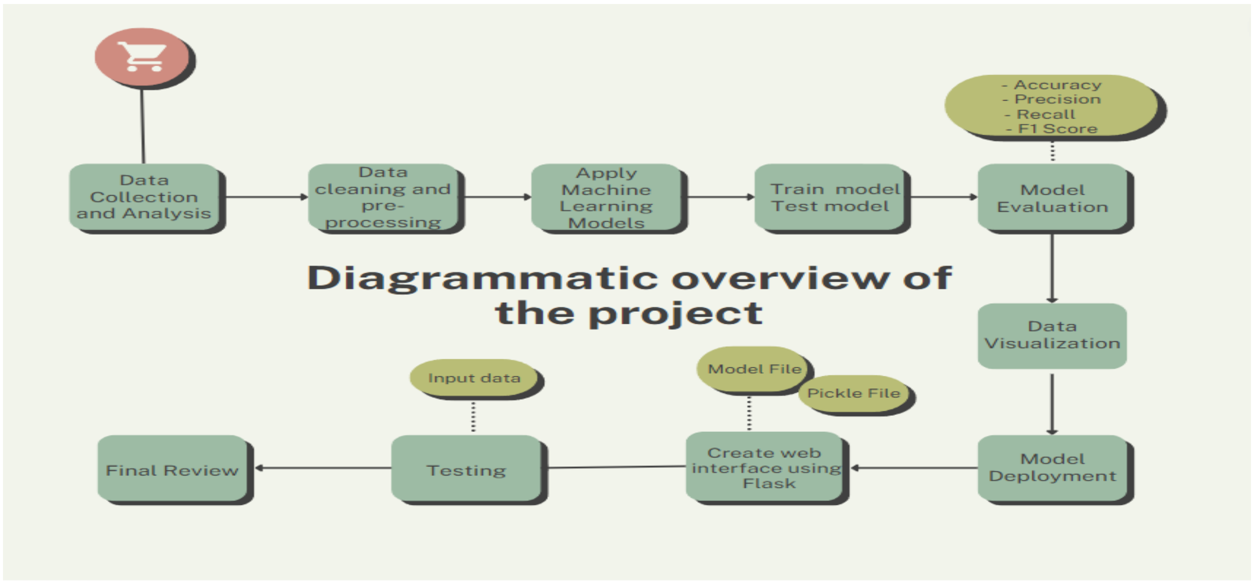
Proposed Solution

In this project, we propose the utilization of machine learning algorithms to predict online shoppers' intention. By leveraging the power of artificial intelligence and data analysis, we can extract meaningful patterns and insights from large datasets. The proposed solution involves training a model on historical data, incorporating relevant features, and utilizing advanced ML algorithms for prediction.

THEORETICAL ANALYSIS –

Block Diagram

The block diagram provides an overview of the project's components and their interactions. It illustrates the flow of data and information within the system, including data collection, preprocessing, model training, and prediction stages.



Hardware/Software Designing

The hardware requirements for this project are minimal as the focus is primarily on software implementation. The software requirements include:

Programming language: Python

Machine learning libraries: Pandas, NumPy, Seaborn, Matplotlib, Flask

Machine learning Models: Random Forest, Decision Tree, Logistic Regression, Naive Bayes, Support Vector Machines (SVM), and k-Nearest Neighbors (KNN).

Development environment: Jupyter Notebook

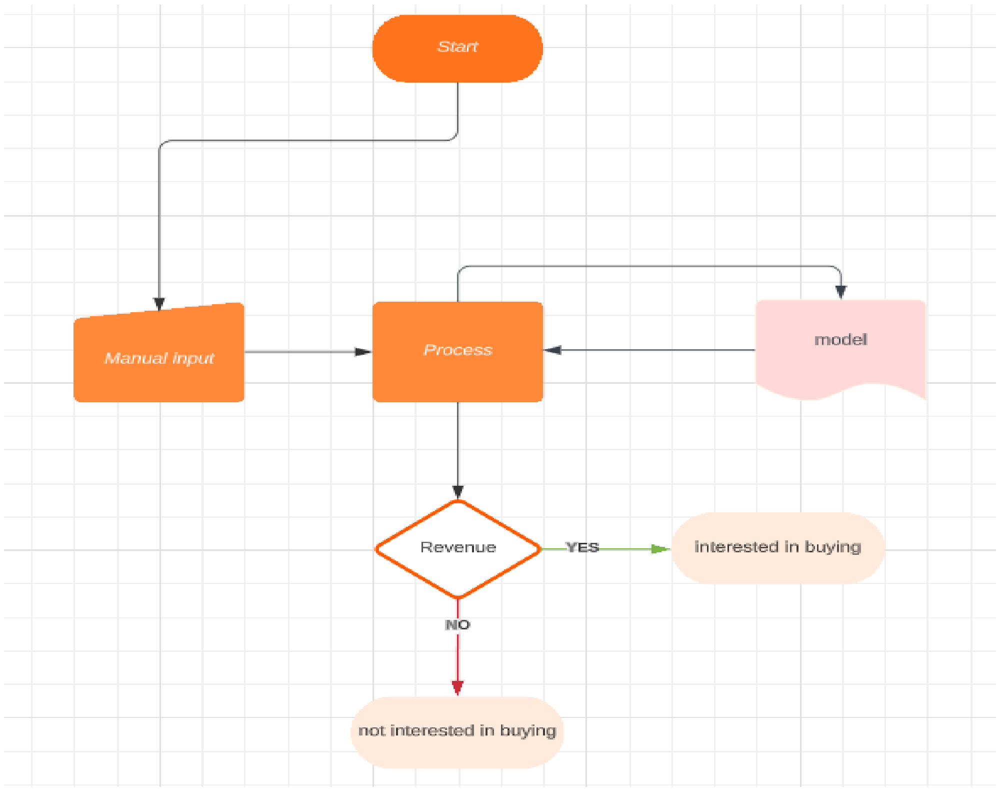
EXPERIMENTAL INVESTIGATIONS

Analysis and Investigation

During the project's development, several experimental investigations were conducted. This involved collecting a dataset of online shopping data, preprocessing the data to remove noise and outliers, and performing exploratory data analysis to gain insights into the dataset. Feature engineering techniques were applied to extract relevant features, and various machine learning algorithms were tested and evaluated for prediction accuracy. The performance of the models was measured using appropriate evaluation metrics such as accuracy, precision, recall, and F1-score.

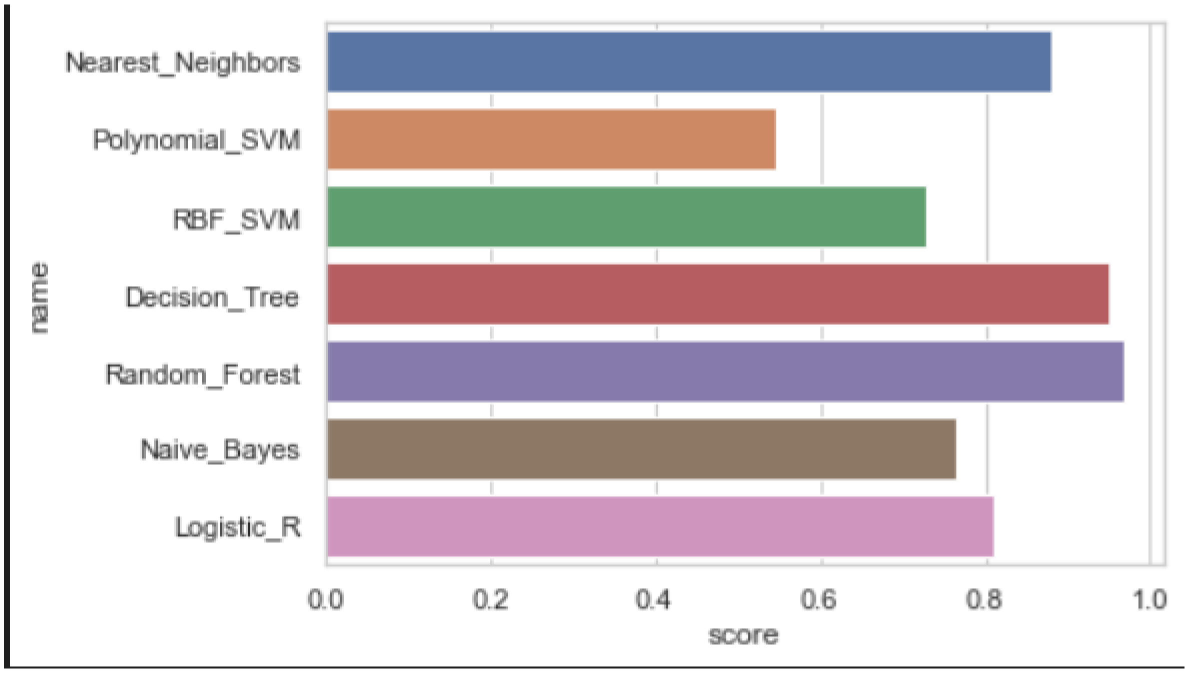
FLOWCHART

The flowchart visually represents the control flow of the solution. It illustrates the steps involved in the data preprocessing, model training, and prediction processes. The flowchart provides a clear overview of the project's workflow and helps understand the sequential steps required for successful implementation.



RESULT

The final findings of the project include the predictive accuracy of the machine learning model in determining online shoppers' intentions. The output of the model provides a probability or classification indicating whether a shopper is likely to make a purchase or abandon the shopping process. Screenshots of the model's performance, including visualizations of key metrics and predicted outcomes, can be included to provide a comprehensive understanding of the results achieved.



ADVANTAGES & DISADVANTAGES

Advantages of the proposed solution:

- Improved prediction accuracy compared to traditional methods
- Personalized customer experiences based on individual preferences
- Optimization of marketing strategies and resource allocation
- Increased conversion rates and customer satisfaction

Disadvantages of the proposed solution:

- Dependency on the availability and quality of training data
- The need for continuous updating and refinement of the model
- Potential privacy concerns related to data collection and analysis

APPLICATIONS

The proposed solution has several potential applications in the e-commerce industry, including:

- Targeted marketing campaigns based on predicted user intentions
- Personalized product recommendations for individual shoppers
- Dynamic pricing strategies based on demand and customer preferences
- Fraud detection and prevention in online transactions
- Optimization of website design and user interface for improved user experience

CONCLUSION

In conclusion, this project demonstrates the effectiveness of machine learning techniques in predicting online shoppers' intention. By analyzing user behavior and relevant features, businesses can gain valuable insights into customers' preferences and optimize their strategies accordingly. The project's findings highlight the advantages and disadvantages of the proposed solution, along with its potential applications in the e-commerce domain.

FUTURE SCOPE

There are several avenues for enhancing and expanding this project in the future, including:

- Incorporating real-time data streams for more accurate predictions
- Implementing advanced deep learning models for improved performance
- Integrating natural language processing techniques for sentiment analysis
- Exploring additional features and data sources to enhance prediction accuracy
- Developing a user-friendly interface or API for easy integration into existing e-commerce platforms

By pursuing these future enhancements, the project can further contribute to the advancement of online shopping experiences and provide valuable insights for businesses in understanding and catering to their customers' needs.

BIBILOGRAPHY

1. Faisal Bin Ashraf (2019) Analysis of Different Predicting Model for Online Shoppersâ€™ Purchase Intention from Empirical Data, 22nd International Conference on Computer and Information Technology: Faisal Bin Ashraf.
2. S.Meher Taj, A.Kumaravel (February 2020) Intentions Of Online Shoppers Prediction By Fuzzy Petri Nets Construction, Volume 9, edn., International Journal Of Scientific & Technology Research: S.Meher Taj.
3. Erevelles, S., Fukawa, N., & Swayne, L (2016) Big Data consumer analytics and the transformation of marketing, edn., Journal of Business Research, 69(2), 897-904.:
4. (2017) Statistics and facts about global e-commerce, edn., Retrieved from https://www.statista.com/topics/871/online-shopping.
5. (2017) Statistics and facts about e-commerce in India, edn., https://www.sta

tista.com/topics/2454/e-commerce-in-india,

6. (2018) Number of active buyers on Alibaba in Bangladesh from 2015 to 2017, edn., https://www.statista.com/statistics/898967/bangladesh-e-commerceexport-distribution

7. Rygielski, Chris, Jyun-Cheng Wang, and David C. Yen. ”Data mining techniques for customer relationship management.” Technology in Society 24.4 (2002): 483-502.

8. Yomi Kastro Inveon (n.d.) Information Technologies Consultancy and Trade, edn., 34335 Istanbul, Turkey:

9. Mobasher, Bamshad, et al. (n.d.) Discovery and evaluation of aggregate usage proï¬les for web personalization, edn., Data mining and knowledge discovery 6.1 (2002): 61-82

10. Shekasta, Michael, et al (2019) New Item Consumption Prediction Using Deep Learning, edn., arXiv preprint arXiv:1905.01686.

11. Llora, Xavier, and Josep M. Garrell (2001) Evolution of decision trees, edn., Forth Catalan Conference on Artiï¬cial Intelligence (CCIA2001).

12. C. Okan Sakar (n.d.) Department of Computer Engineering, Faculty of Engineering and Natural Sciences, Bahcesehir University, edn., 34349 Besiktas, Istanbul, Turkey.

13. Yomi Kastro Inveon (n.d.) Information Technologies Consultancy and Trade, edn., 34335 Istanbul, Turkey.

APPENDIX

A. SOURCE CODE:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('online1.csv')
df.head()
df.describe(include=["object"])
df.info()
df.describe()

df["Month"].value_counts()

df["VisitorType"].value_counts()
df.columns

df.isna().sum()
df = df.drop(df.columns[[0, 1, 2, 3, 11, 12]], axis=1)
df.head();
Uni variante

df[['Region']].boxplot()

sns.heatmap(df.corr())

Plot_Revenue = df['Revenue'].value_counts()
plt.pie(Plot_Revenue,autopct='%.2f',labels = ['yes','no'])
plt.hist(df['Region'])
```

Bivariant

```
plt.plot(df['Region'],df['Revenue'])
plt.xlabel('Region')
plt.ylabel('Revenue')
plt.title('bivariate viz "Region vs Revenue (Which region makes
revenue)"')

plt.figure(figsize=(10,6))

plt.subplot(1,2,1)
plt.scatter(df['TrafficType'],df['Revenue'])
plt.xlabel('TrafficType')
plt.ylabel('Revenue')
plt.title('bivariate viz "TrafficType vs Revenue"')

plt.subplot(1,2,2)
plt.bar(df['VisitorType'],df['Revenue'])
plt.xlabel('VisistorType')
plt.ylabel('Revenue')
plt.title('bivariate viz "VisitorType vs Revenue"')
```

multi variant analysis

```
plt.bar(df['Revenue'],df['Weekend'])
plt.bar(df['Revenue'],df['Month'])

from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

df.columns

df['Revenue'].value_counts()
```

LableEncoding

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

df['Weekend'] = le.fit_transform(df.Weekend)
df['VisitorType'] = le.fit_transform(df.VisitorType)
df['Month'] = le.fit_transform(df.Month)
df['Revenue'] = le.fit_transform(df.Revenue)
```

Break data

```
y = df['Revenue']
y1 =df.drop(columns=['Revenue'],axis=1)
```

```
y1
```

We use oversampling here as the there is dispropotion in the revenue column

```
print(y.value_counts())
print(y1)
from sklearn.tree import DecisionTreeClassifier

Dtree=DecisionTreeClassifier()

Dtree.fit(y1,y)

y.value_counts()
```

Oversampling

```
from imblearn.over_sampling import RandomOverSampler

os=RandomOverSampler(random_state=0)
x_res2,y_res2=os.fit_resample(y1,y)
y_res2.value_counts()
```

We now predict with Decision tree on the attribute columns

```
# Test the Decision Tree model

p=Dtree.predict(x_res2)
p

from sklearn.metrics import classification_report
```

We run a classification report on the revenue for Decision Tree model before the spilting of data

```
print(classification_report(p,y_res2))
```

Splitting the data

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x_res2,y_res2,test_size
=0.2,random_state=0)
```

Create Model

```
from sklearn.ensemble import RandomForestClassifier

rf=RandomForestClassifier(n_estimators=10,criterion='entropy',random_s
tate=0)
```

```
#training the model
rf.fit(x_train,y_train)

# Test the RandomForestClassifier model
pred=rf.predict(x_test)
pred

# Evaluate the model
from sklearn.metrics import
accuracy_score,confusion_matrix,classification_report

accuracy_Randf=accuracy_score(y_test,pred)
conmat=confusion_matrix(y_test,pred)

print(accuracy_Randf)
list_Randf = [accuracy_Randf]

print(conmat)
```

We run a classification report on the revenue for random forest model

```
print(classification_report(y_test,pred))
```

Decision tree model

```
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree
Classifier
from sklearn import metrics

clf = DecisionTreeClassifier()

# Train Decision Tree Classifer
clf = clf.fit(x_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(x_test)
accuracy_DT=metrics.accuracy_score(y_test, y_pred)
list_DT = [accuracy_DT]
accuracy_DT

print(classification_report(y_test,y_pred))
```

SVM

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
# Create and train the SVM model
model_rbf = SVC(kernel='rbf')
model_rbf.fit(x_train, y_train)

# Create and train the SVM-POLY model

model_poly = SVC(kernel="poly")
model_poly.fit(x_train, y_train)

# Make predictions on the test set
y_predrbf = model_rbf.predict(x_test)
y_predpoly = model_poly.predict(x_test)

# Evaluate the model
accuracy_rbf = accuracy_score(y_test, y_predrbf)
accuracy_poly = accuracy_score(y_test, y_predpoly)

#accuracy_SVM = accuracy_score(y_test, y_predl)
list_rbf = [accuracy_rbf]
list_poly = [accuracy_poly]
print("Accuracy SVM:", accuracy_rbf)
print("Accuracy SVM:", accuracy_poly)

#Classification report for SVM
print(classification_report(y_test,y_predrbf))

print(classification_report(y_test,y_predpoly))
```

Logistic Regression

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
model_lr = LogisticRegression()
model_lr.fit(x_train, y_train)

# Make predictions on the test set
y_predlr = model_lr.predict(x_test)

# Evaluate the Logistic Regression model
```

```
accuracy_LR = accuracy_score(y_test, y_predlr)
list_LR = [accuracy_LR]
print("Accuracy:", accuracy_LR)

#Classification report for Logistic Regression
print(classification_report(y_test,y_predlr))
```

KNN

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

# Create and train the KNN model
model_knn = KNeighborsClassifier(n_neighbors=5)  # You can adjust the
number of neighbors (K) as needed
model_knn.fit(x_train, y_train)

# Make predictions on the test set
y_predknn = model_knn.predict(x_test)

# Evaluate the model
accuracy_KNN = accuracy_score(y_test, y_predknn)
list_KNN = [accuracy_KNN]
print("Accuracy:", accuracy_KNN)

#Classification report on KNN
print(classification_report(y_test,y_predknn))
```

Gausian NaveBayes

```
from sklearn.naive_bayes import GaussianNB
nb = GaussianNB()
nb.fit(x_train, y_train)
accuracy = nb.score(x_test, y_test)
accuracy

accuracy_NB = accuracy_score(y_test,y_prednb)
list_NB = [accuracy_NB]
accuracy_NB
```

Comparing classifiers

```
names = ["Nearest_Neighbors", "Polynomial_SVM", "RBF_SVM",
"Decision_Tree", "Random_Forest","Naive_Bayes", "Logistic_R"]
classifiers = [
    KNeighborsClassifier(5),
    SVC(kernel="poly"),
    SVC(kernel="rbf"),
    DecisionTreeClassifier(),
    RandomForestClassifier(),
    GaussianNB(),
    LogisticRegression()
   ]
print(classifiers)
print(names)

scores = []
scores.append(list_KNN)
scores.append(list_poly)
scores.append(list_rbf)
scores.append(list_DT)
scores.append(list_Randf)
scores.append(list_NB )
scores.append(list_LR)

scores

model_perf = pd.DataFrame()
model_perf['name'] = names
model_perf['score'] = scores
model_perf

cm = sns.light_palette("green", as_cmap=True)
s = model_perf.style.background_gradient(cmap=cm)
s
sns.set(style="whitegrid")
ax = sns.barplot(y="name", x="score", data=model_perf)
```