CUSTOMER PURCHASE PREDICTION

Using Machine Learning models to predict user behavior

<u>ME781 DATA MINING/GROUP 3</u>

203100068 MUDIT SAND

170040106 BAVISH K

193100008 JAYKUMAR GUPTA

180100029 BURRA SRIHITH BHARADWAJ

150040010 TANYA GUPTA

Table of Content

NOTE : All the required documents are also submitted on Moodle for better visualization.

Problem Definition

COVID-19 has accelerated the transition from visiting physical stores to online shopping. Predicting customer behavior in the context of e-commerce is gaining importance. It can increase customer satisfaction and sales, resulting in higher conversion rates and competitive advantage, by facilitating a more personalized shopping process.

At PREDICT.AI (http://www.predictai.design/), we aim to HELP GROWING STARTUPS AND BUSINESSES utilize their customer data and build models for PREDICTING CUSTOMER BEHAVIOUR. Comparing models will give further insight into the performance differences in static customer data. Conducting descriptive data analysis visualization will help our clients extract more value from data and make decisions to boost their customer satisfaction.

CUSTOMER REQUIREMENTS	BUSINESS CASE	BARRIER TO ENTRY & EXISTING PRODUCTS/ SERVICES	UNIQUE SELLING PROPOSITION & PROTECTION OF USP
Accurate models	Target startups and small businesses	Companies not wanting to share data	Easy to use Interface
User satisfaction	Publish conclusions from publicly available data	Companies building their own AI Teams	High accuracy models
Increasing revenue	Subscription model like Bloomberg for companies	Google Analytics	Data protection and privacy
24/7 Help and support	Testimonial from clients	NTENT	Branding of USP

Technology landscape assessment

Patents

Jivox Kairos™ is the market's first purchase prediction engine for eCommerce marketing, with a patent granted to Jivox for SCORING USERS BASED ON INTENT FOR ONLINE ADVERTISING. As a global brand marketer, you can confidently use this cutting-edge technology to drive sales, by personalizing messaging in real-time to individual consumers based on their interests and in-the-moment purchase intent.

Built on the Jivox NeuronTM AI and machine learning technology, Kairos "learns" how a product is relevant to a specific consumer based on their purchase intent. Kairos algorithms use these purchase intent signals to score individual users' likelihood and immediacy to make a purchase, and rank products relative to the consumer's interest. The pairing of user scoring and product ranking creates for you the opportune moment to serve the right message with the right product offer at the right time.

Source:
(https://info.jivox.com/kairos-purchase-prediction-ecommerce)

Amex Advance is a data-driven business that partners with companies across the advertising, travel, and service industries to deliver curated personalization services optimized for their customers. Leveraging best-in-class predictive machine learning, deep consumer insights, connectivity capabilities, and an integrated platform, Amex Advance transforms its deterministic data insights into customized solutions to solve partners' key business challenges.

Source : (www.acxiom.com/news/acxiom-amex-advance-launch-new-data-driven-offering-predict-consumer-purchase-intent/)

Libraries Used

NumPy	Pandas	SciKit-Learn	Matplotlib
Plotly	XGBoost	Unittest	Seaborn

Published Literature

Kumar, A., Kabra, G., Mussada, E.K. et al. "Combined artificial bee colony algorithm and machine learning techniques for prediction of online consumer repurchase intention." Neural Comput & Applic 31, 877–890 (2019)

This Literature Focuses on predicting online consumer repurchase intention within the context of shopping malls and consumer characteristics using intelligent techniques. This study integrates characteristics of both consumers as well as shopping malls to predict consumer repurchase intention in the online platform. The experimental results have been analyzed through five machine learning classification models, i.e., decision trees,AdaBoost, random forest (RF), support vector machine (SVM) and neural network (NN) in different settings. In the partitions of data into 70–30 training–testing, among all models, the performance of AdaBoost has the highest sensitivity (0.95%) and accuracy (97.58%).

Chen, Zhen-Yu, and Zhi-Ping Fan. "Distributed customer behavior prediction using multiplex data: a collaborative MK-SVM approach." Knowledge-Based Systems 35 (2012): 111-119.

This paper presents the understanding of customer behavior is a critical success factor. The big databases in an organization usually involve multiplex data such as static, time series, symbolic sequential and textual data which are separately stored in different databases of different sections. It poses a challenge to traditional centralized customer behavior prediction. In this study, a novel approach called collaborative multiple kernel support vector machine (C-MK-SVM) is developed for distributed customer behavior prediction using multiplex data. The alternating direction method of multipliers (ADMM) is used for the global optimization of the distributed sub-models in C-MK-SVM. Computational experiments on a practical retail dataset are reported. Computational results show that C-MK-SVM exhibits better customer behavior prediction performance and higher computational speed than support vector machine and multiple kernel support vector machine.

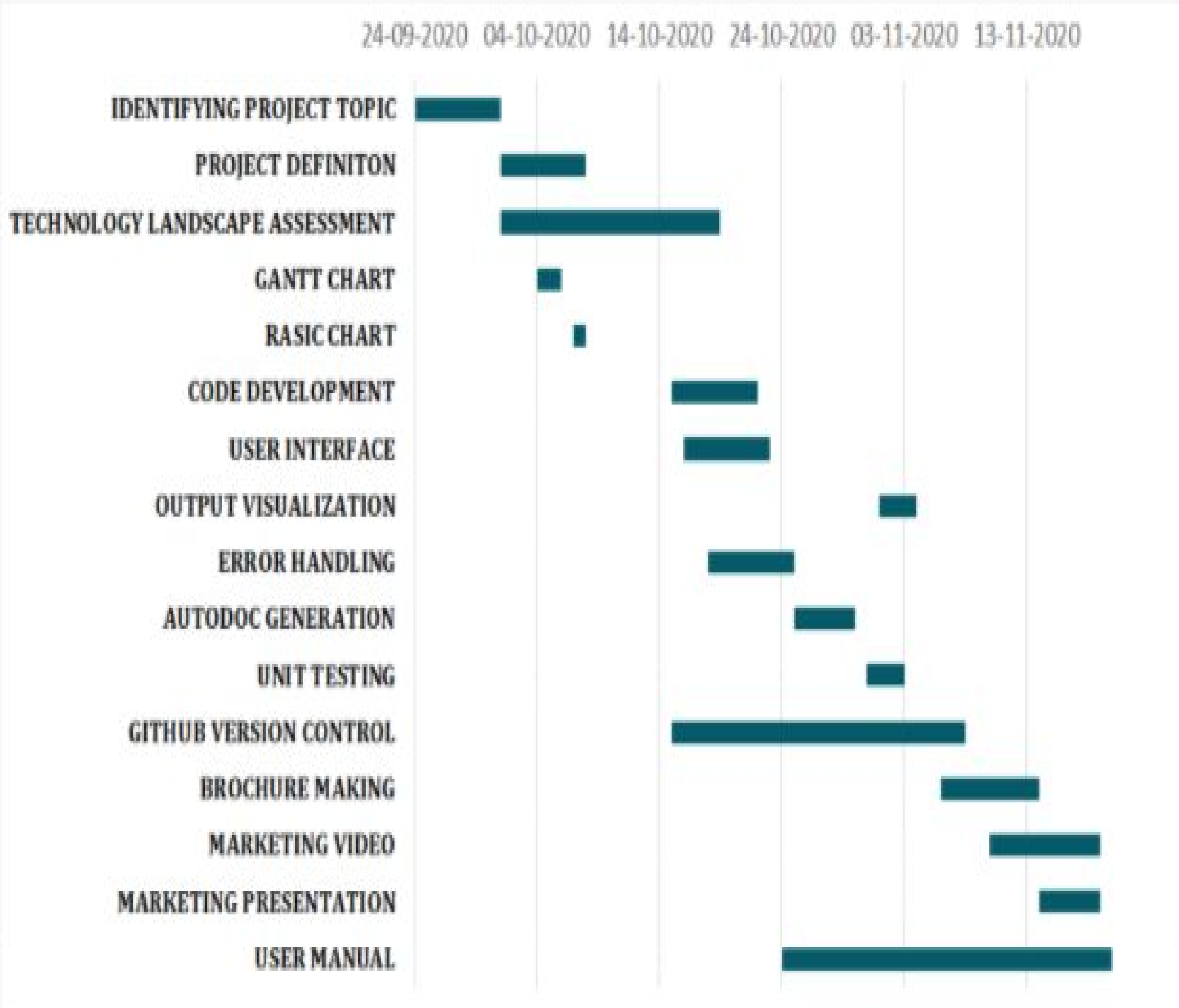
Project planning report

predict.ai

RASIC CHART

Tasks \ People	MUDIT	BAVISH	JAY	SRIHITH	TANYA
OBJECTIVE AND DEFINITION	A	A	S	S	R
TECHNOLOGY LANDSCAPE ASSESSMENT	R	S			S
PLANNING - TIMELINE, GANTT, RASIC CHARTS	A	A	A	A	R
CONCEPTUAL DESIGN - MODEL/DATASET SELECT	A	A	R	S	A
CODE DEVELOPMENT PHASE 1		R	S	S	
CODE DEVELOPMENT PHASE 2	S	R		S	S
MARKETING BROCHURE, PRESENTATION, VIDEO	S		S		R
USER MANUAL & PROJECT REPORT	S	S	S	R	S

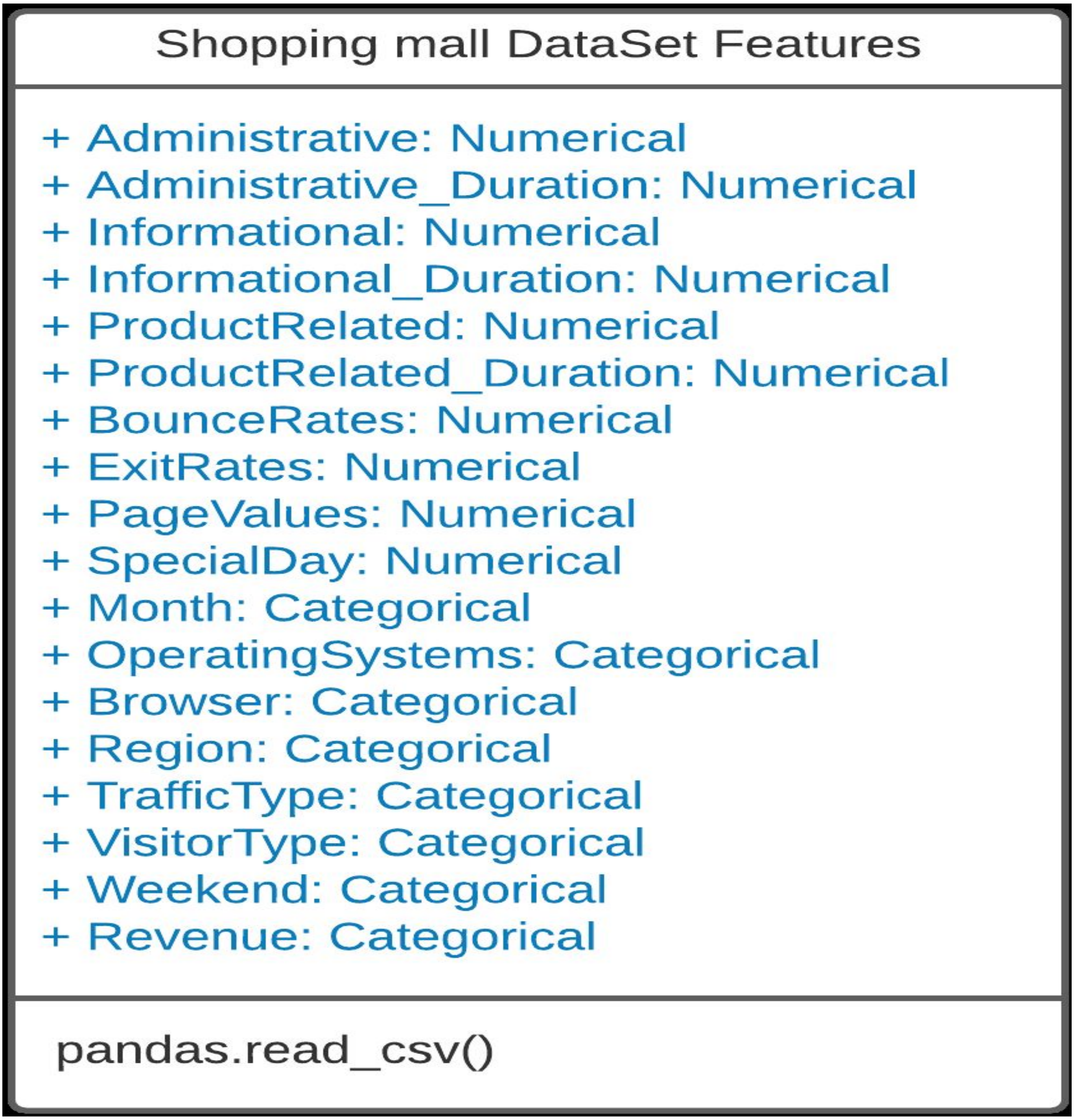
GANTT CHART



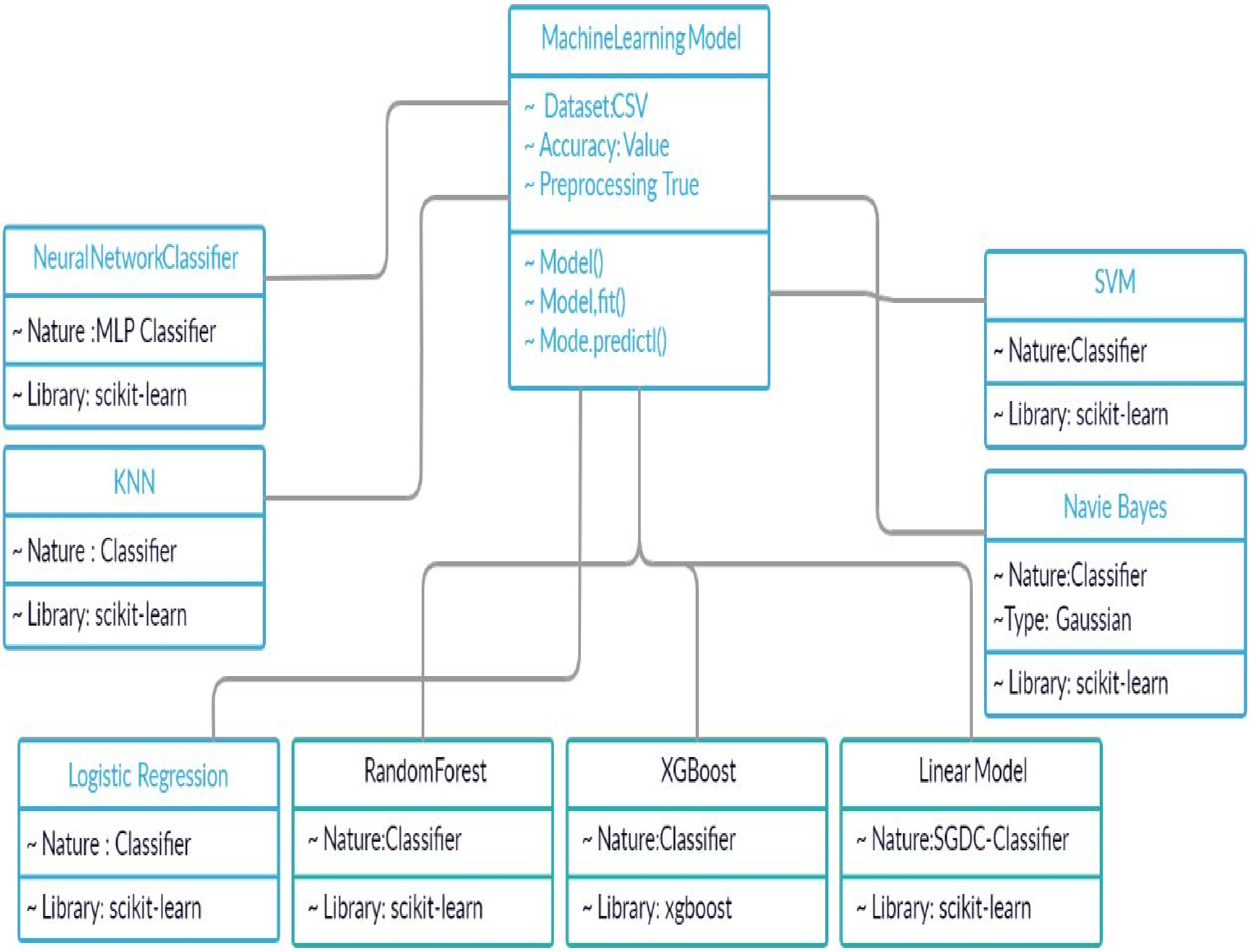
24-09-2020
04-10-2020
14-10-2020
24-10-2020
03-11-2020
13-11-2020
IDENTIFYING PROJECT TOPIC
PROJECT DEFINITON
TECHNOLOGY LANDSCAPE ASSESSMENT
GANTT CHART
BASIC CHART
CODE DEVELOPMENT
USER INTERFACE
OUTPUT VISUALIZATION
ERROR HANDLING
AUTODOC GENERATION
UNIT TESTING
GITHUB VERSION CONTROL
BROCHURE MAKING
MARKETING VIDEO
MARKETING PRESENTATION
USER MANUAL

Conceptual design document

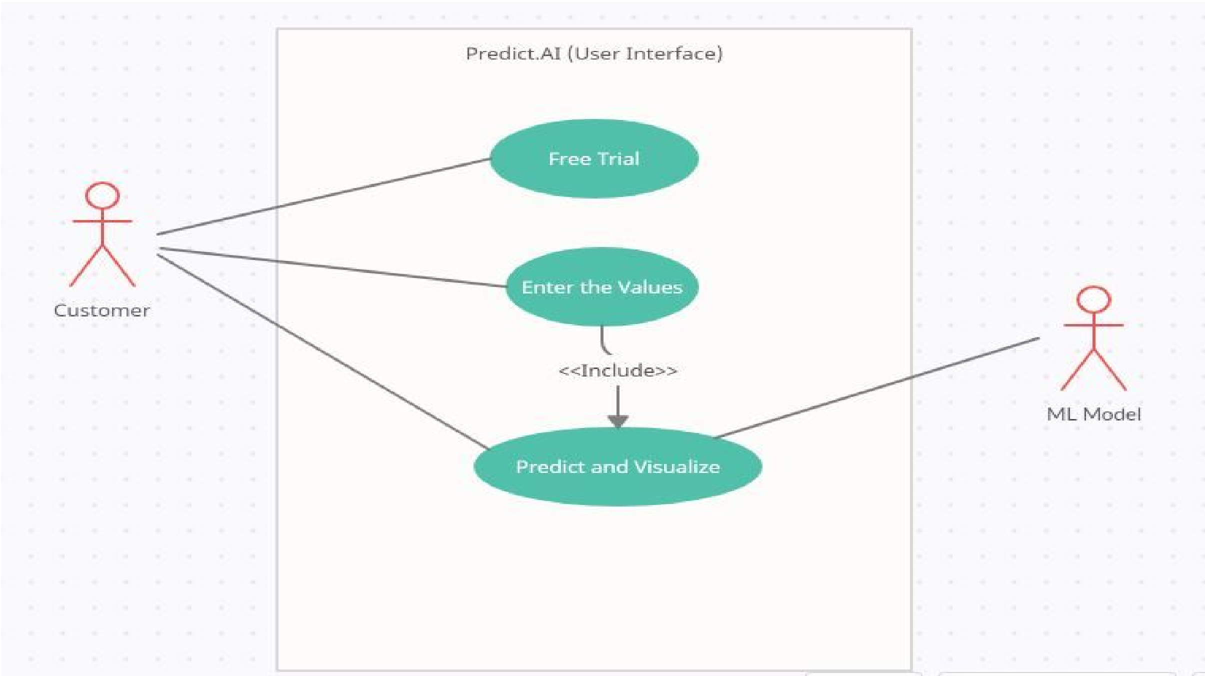
Class Containing The Information about DataSet



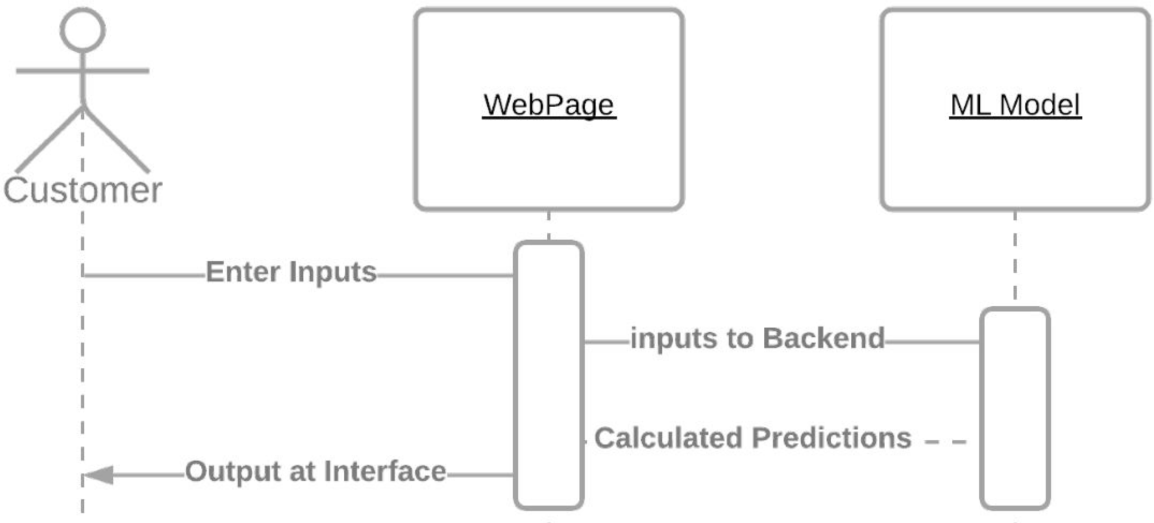
Machine learning Model Class



Use Case



UML Sequence Diagram:



Auto generated code document

final_models.py

Co-authored by Tanya, Bhavik, Mudit, Srihit and Jaykumar as a result of our ME781 Data Mining final project.

Based on Shopping Data set from UCI's Machine Learning Repository

1. **Predict.ai Website** http://www.predictai.design/
2. **GitHub Repo** https://github.com/Tannybuoy/predictai
3. **Demo Video** https://www.youtube.com/watch?v=xFt4cl4daKM/

Importing necessary libraries for running Machine Lerning models

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import RobustScaler, StandardScaler, MinMaxSca
from sklearn.model_selection import cross_val_score, train_test_split, cr
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import roc_auc_score
import xgboost as xgb
from xgboost import XGBClassifier
```

Google Drive Mount in Google Colab

```
from google.colab import drive
drive.mount('/content/drive')
```

Go to directory containing the dataset

```
cd "/content/drive/My Drive/Colab Notebooks"
```

Reading shopping data

```
X_train = pd.read_csv('ShoppingData.csv')
df = X_train.copy()
df.head()
```

Producing dummy variables for categorical data and cleaning data

A dummy dataframe is created to clean the dataset and preprocess

1. **Visitor** - Columns based on New, Returning or Other
2. **Month** - New columns for each month
3. **Class** - Column after changing data type to int

```
dummiesdf = pd.get_dummies(df['VisitorType'])
df.drop('VisitorType', inplace = True, axis = 1)
df['New_Visitor'] = dummiesdf['New_Visitor']
df['Other'] = dummiesdf['Other']
df['Returning_Visitor'] = dummiesdf['Returning_Visitor']

dfmonth = pd.get_dummies(df['Month'])
df.drop('Month', inplace = True, axis = 1)
dfwithdummies = pd.concat([df, dfmonth], axis = 1, sort = False)


dfwithdummies['Class'] = df['Revenue'].astype(int)
dfwithdummies.drop('Revenue', axis = 1, inplace = True)
dfwithdummies['Weekend'] = df['Weekend'].astype(int)
dfwithdummies.drop('Returning_Visitor', axis = 1, inplace = True)
dfcleaned = dfwithdummies.copy()

X = dfcleaned.drop('Class', axis = 1)
Y = dfcleaned['Class'].copy()
```

Checking for Collinearity Between Features and Creating Reducing Feature Size

The cor and heatmap help in visualising correlation between various features. Accordingly we do remove the columns/ pre-process.

```
cor = X.corr()
sns.heatmap(cor, xticklabels=cor.columns,yticklabels=cor.columns)
```

```
def AvgMinutes(Count, Duration):
    if Duration == 0:
```

AvgMinutes function is used to calculate the average time spent by a customer on the given page. It is obtained by dividing the "Count" by "Duration"

```
        output = 0
    elif Duration != 0:
        output = float(Duration)/float(Count)
    return output
```

Three new column features hence get added and six columns can now be dropped

```
Columns = [['Administrative', 'Administrative_Duration'], ['Informational

X['AvgAdministrative'] = X.apply(lambda x: AvgMinutes(Count = x['Administ
X['AvgInformational'] = X.apply(lambda x: AvgMinutes(Count = x['Informati
X['AvgProductRelated'] = X.apply(lambda x: AvgMinutes(Count = x['ProductR
X.drop(['Administrative', 'Administrative_Duration','Informational', 'Inf
```

Correlation matrix is plotted again using sns heatmap to check if the correlation between the above dropped six features has been dealt with

```
cor = X.corr()
sns.heatmap(cor, xticklabels=cor.columns,yticklabels=cor.columns)
```

Quick overview of features

Histogram of all features

```
for idx,column in enumerate(X.columns):
    plt.figure(idx)
    X.hist(column=column,grid=False)
```

Checking for NA values

```
for i in X.columns:
    print('Feature:',i)
    print('# of N/A:',X[i].isna().sum())
```

Visualising no of unique values and the unique values in each column of the training dataset

```
for i in X_train.columns:
    print('####################')
    print('COLUMN TITLE:',i)
    print('# UNIQUE VALUES:',len(X_train[i].unique()))
    print('UNIQUE VALUES:',X_train[i].unique())
    print('####################')
    print()
```

Scaling to normalize data

```
X_copy = X.copy()
rc = RobustScaler()
X_rc=rc.fit_transform(X_copy)
X_rc=pd.DataFrame(X_rc,columns=X.columns)
```

Plotting the histogram obtained post above processing functions

```
for idx,column in enumerate(X_rc.columns):
    plt.figure(idx)
    X_rc.hist(column=column,grid=False)
```

Linear Model with All Features

```
from sklearn import linear_model
from sklearn import metrics
X_train, X_test, y_train, y_test = train_test_split(X_rc,Y,test_size=.2)
```

Linear model

```
model = linear_model.SGDClassifier()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

Accuracy score imported to calculate accuracy

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred)
```

roc_auc_score imported to calculate accuracy

It illustrates in a binary classifier system the discrimination threshold created by plotting the true positive rate vs false positive rate

```
from sklearn.metrics import roc_auc_score
roc_auc_score(y_test, y_pred)
```

Random Forest with all Features

```
from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(max_depth=17, random_state=0)
clf.fit(X_train, y_train)
y_pred1 = clf.predict(X_test)

accuracy_score(y_test, y_pred1)

roc_auc_score(y_test, y_pred1)
```

Finding Important Features then Removing from Dataframe

SelectKBest to obtain a list of importance of each feature column

```
from sklearn import svm
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
list_one =[]


feature_ranking = SelectKBest(chi2, k=5)
fit = feature_ranking.fit(X, Y)

fmt = '%-8s%-20s%s'

for i, (score, feature) in enumerate(zip(feature_ranking.scores_, X.colum
    list_one.append((score, feature))

dfObj = pd.DataFrame(list_one)
dfObj.sort_values(by=[0], ascending = False)
```

On seeing the list, we drop the ones which have a very low weightage and less importance

```
X_rc.drop(['Aug','TrafficType','OperatingSystems','Other','Jul'],axis=1,i

X_train1, X_test1, y_train1, y_test1 = train_test_split(X_rc,Y,test_size=
```

Random Forest Classifier with Feature Selection Dataframe

Now once again we run Random Forest Classifier, but after retaining only the important features as determined by SelectKBest

```
clf1 = RandomForestClassifier(n_estimators= 200, max_depth = 30 )
clf1.fit(X_train1, y_train1)
y_pred2 = clf1.predict(X_test1)

accuracy_score(y_test1, y_pred2)

roc_auc_score(y_test1, y_pred2)
```

XGBoost Classifier with Feature Selection Dataframe

```
model = XGBClassifier(learning_rate = 0.1, n_estimators=150, min_child_we
model.fit(X_train1, y_train1)
y_pred3 = model.predict(X_test1)

accuracy_score(y_test1, y_pred3)

roc_auc_score(y_test1, y_pred3)
```

LogisticRegression with Feature Selection Dataframe

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
log_reg = LogisticRegression(solver='lbfgs',multi_class='multinomial',max
log_reg.fit(X_train1,y_train1)
y_pred4 = log_reg.predict(X_test1)
print(accuracy_score(y_pred4,y_test1))
print(roc_auc_score(y_test1, y_pred4))
```

Gaussian Naive Bayes with Feature Selection Dataframe

```
from sklearn.naive_bayes import GaussianNB
GNB = GaussianNB()
GNB.fit(X_train1,y_train1)
y_pred5 = GNB.predict(X_test1)
print(accuracy_score(y_pred5,y_test1))
print(roc_auc_score(y_test1, y_pred5))
```

KNN classifier with Feature Selection Dataframe

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 6)
knn.fit(X_train1,y_train1)
y_pred6 = knn.predict(X_test1)
print(accuracy_score(y_pred6,y_test1))
print(roc_auc_score(y_test1, y_pred6))
```

SVM Classification with PCA feature reduction technique

```
from sklearn.decomposition import PCA
pca = PCA(n_components=15)
d=pca.fit_transform(X_train1)
e=pca.fit_transform(X_test1)
print(pca.explained_variance_ratio_.sum())


from sklearn.svm import SVC
svm = SVC()
svm.fit(d,y_train1)
y_pred7 = svm.predict(e)
print(accuracy_score(y_pred7,y_test1))
print(roc_auc_score(y_test1, y_pred7))


from sklearn.svm import SVC
svm = SVC()
```

SVM Classification with Feature Selection Dataframe

```
svm.fit(X_train1,y_train1)
y_pred8 = svm.predict(X_test1)
print(accuracy_score(y_pred8,y_test1))
print(roc_auc_score(y_test1, y_pred8))
```

Neural Network Classifier With Feature Selection Dataframe

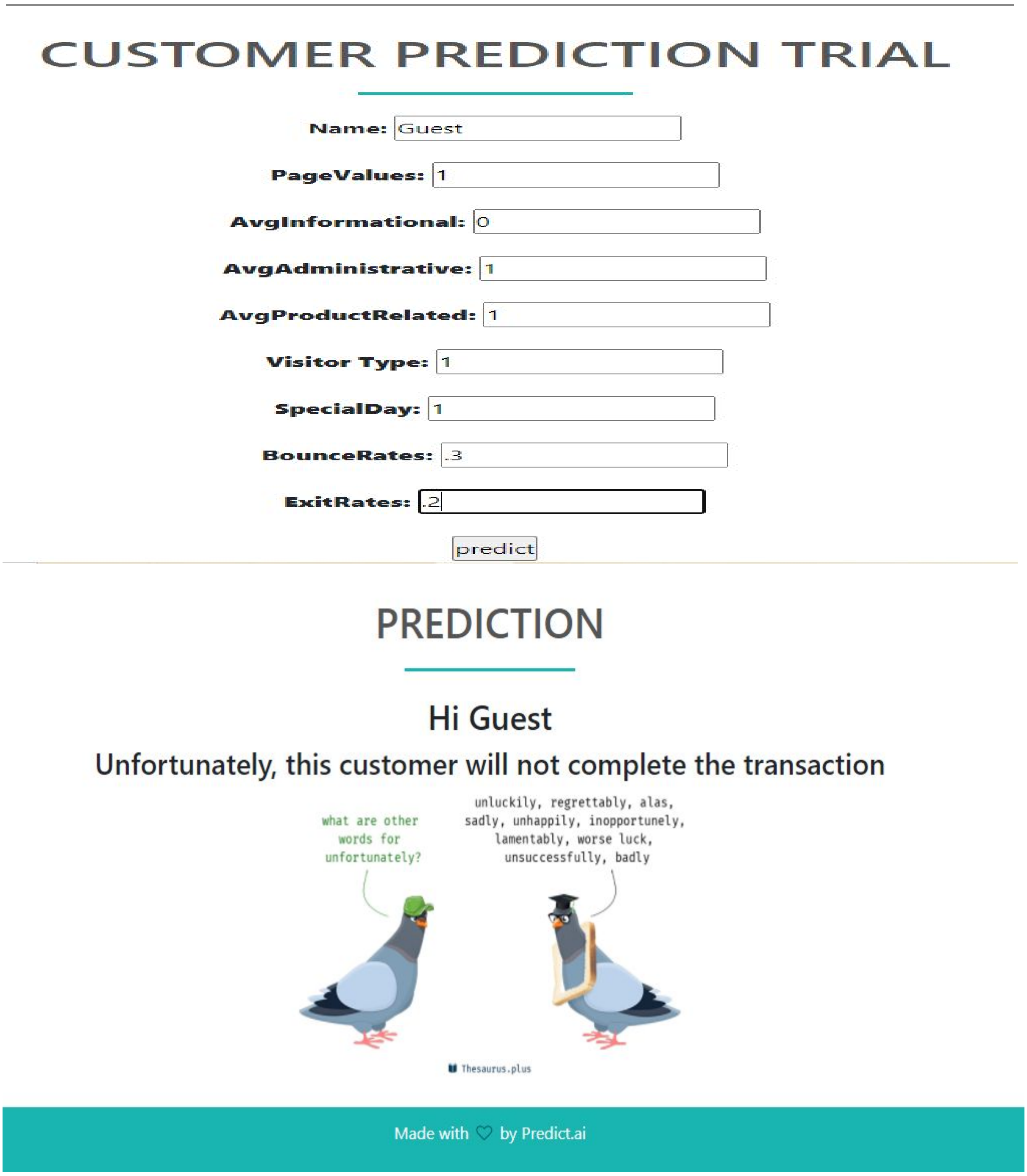
```
from sklearn.neural_network import MLPClassifier

mlp = MLPClassifier(hidden_layer_sizes=(19,19,19), activation='relu', sol
mlp.fit(X_train1,y_train1)
y_pred9= mlp.predict(X_test1)
print(accuracy_score(y_pred9,y_test1))
print(roc_auc_score(y_test1, y_pred9))
```

Screenshots of user interface and output visualization

USER INTERFACE and OUTPUT:

link(http://www.predictai.design/trial)



CUSTOMER PREDICTION TRIAL

Name: Guest

PageValues: 22

AvgInformational: 0

AvgAdministrative: .99

AvgProductRelated: .025

Visitor Type: 0

SpecialDay: .8

BounceRates: -.18

ExitRates: -.35

predict

PREDICTION

Hi Guest

This customer will complete the transaction

Made with ♡ by Predict.ai

Model training and testing report

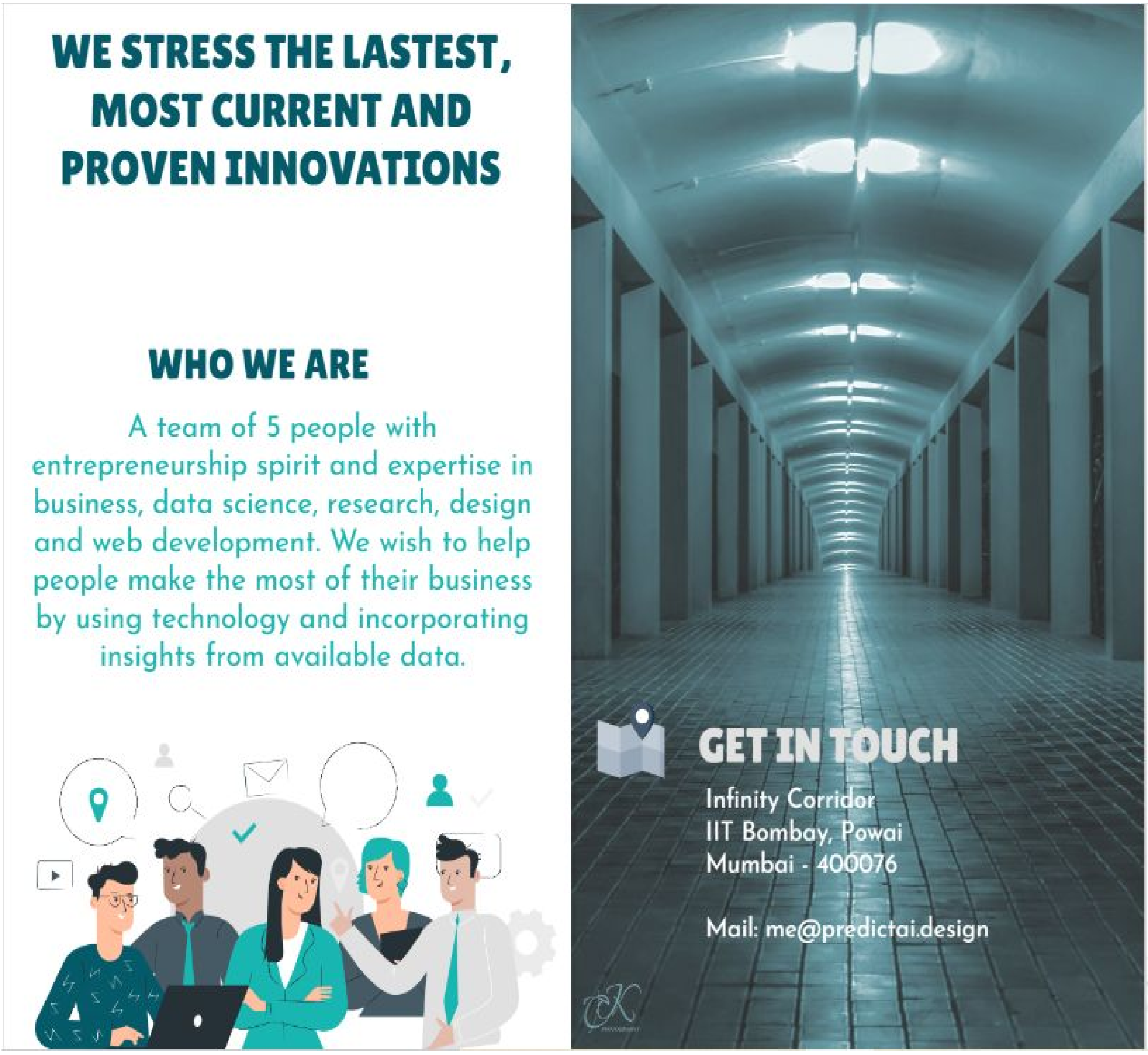
Classifier	Accuracy	R2 accuracy
Logistic regression	0.8892	0.6859
KNN Classifier	0.8819	0.6933
Gaussian Naïve Bayes Classifier	0.6597	0.7475
Neural Network classifier	0.8921	0.729
SVM classifier with Pca Reduction	*0.9026*	*0.7618*
Non linear SVM classifier	*0.9038*	*0.7636*
Random Forest classifier	*0.9042*	*0.7617*
SGDC-classifier	0.8892	0.7465
XGBoost Classifier	*0.9022*	*0.7711*

Product brochure and video presentation

Link of the Video presentation:

https://www.youtube.com/watch?v=xFt4cl4daKM

Broucher Pages:



OUR VALUES

Innovation

We value the trust and respect of our community and coworkers. We commit to becoming a place where we do what's right because we love what's right.

Integrity

We share our clients' aspirations, work vigorously to understand their reality, and align our incentives with their objectives – so they know we're in this for the long haul.

Expertise

We celebrate the power of technology to transform lives. We commit to helping people use technology creatively and ethically.

We deliver transformational results that address your needs.

OUR PRODUCTS

Data Analytics

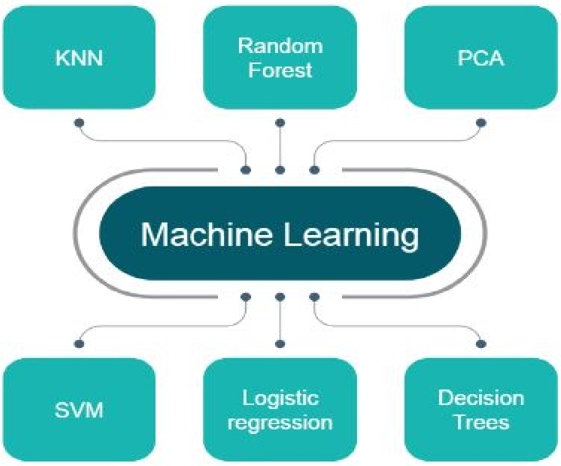
Dashboard to get customer insights from data. View activity across products, contact us for support

Customer Purchase Prediction

Use exit rates, bounce rates, time spent on a page, month and more to predict whether customer will make a purchase. Plan your marketing strategies accordingly.

OUR ALGORITHMS

Our state of the art technology and analytics ensures the highest accuracy and best extraction of information from data. After all, data is the new oil. Make decisions based on data analytics and customer behaviour prediction.



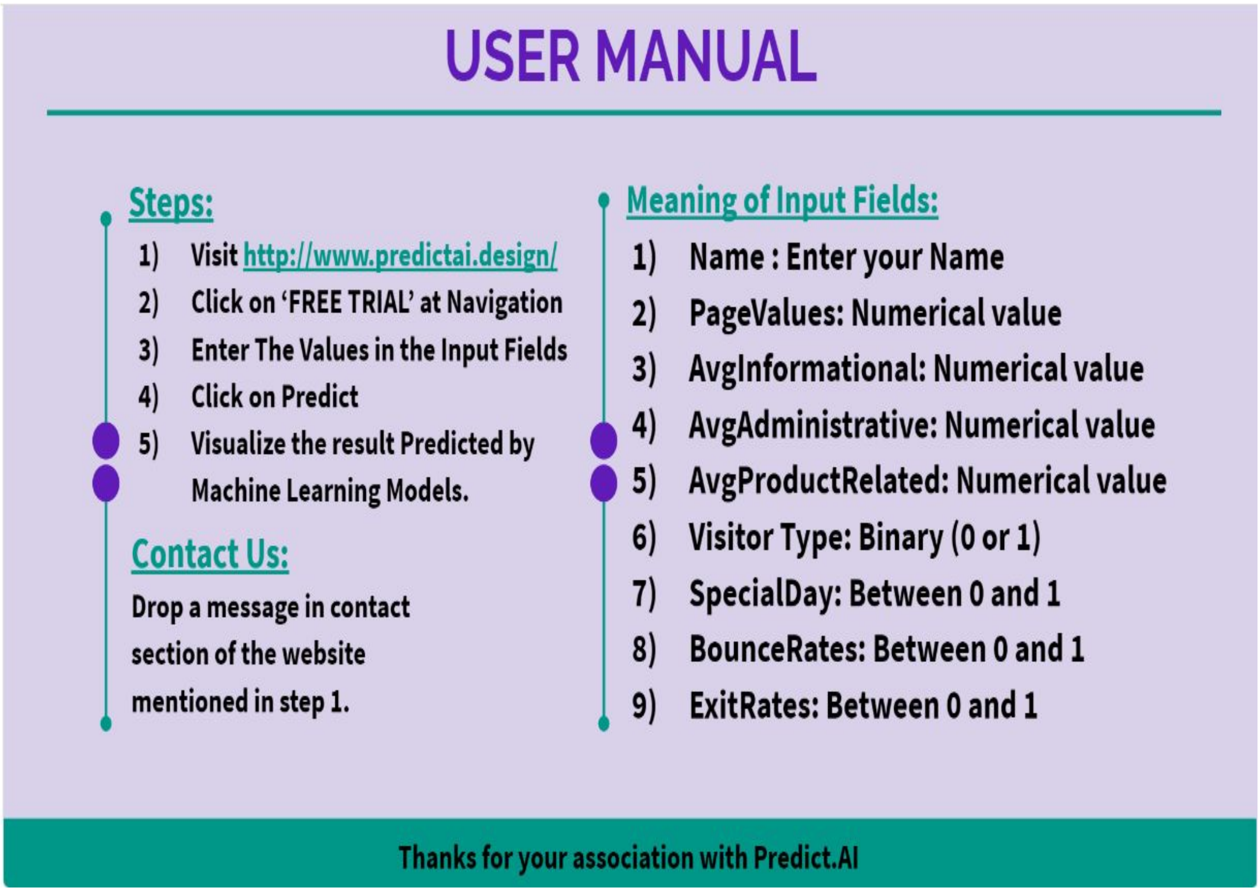
PRICING

Free for hobby project enthusiasts and individuals
Can be customised based on company size and requirements

predict.ai

Driving business development and innovation through technology

User manual



USER MANUAL
Steps:
1) Visit http://www.predictai.design/
2) Click on 'FREE TRIAL' at Navigation
3) Enter The Values in the Input Fields
4) Click on Predict
5) Visualize the result Predicted by Machine Learning Models.
Contact Us:
Drop a message in contact section of the website mentioned in step 1.
Meaning of Input Fields:
1) Name : Enter your Name
2) PageValues: Numerical value
3) AvgInformational: Numerical value
4) AvgAdministrative: Numerical value
5) AvgProductRelated: Numerical value
6) Visitor Type: Binary (0 or 1)
7) SpecialDay: Between 0 and 1
8) BounceRates: Between 0 and 1
9) ExitRates: Between 0 and 1
Thanks for your association with Predict.AI