JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING AND INFORMATION TECHNOLOGY

"Vehicle Insurance Prediction"

SUBMITTED TO:

DR. MUKTA GOYAL

SUBMITTED BY:

RAHI AGARWAL	**9921103145**
NAMAN JHANWAR	**9921103191**

Course Name: Open Source Software Lab

Course Code: 15B17CI575

Program: B. Tech. CSE

3rd Year 5th Sem

2023 - 2024

Abstract

An insurance policy is an arrangement by which a company undertakes to provide a guarantee of compensation for specified loss, damage, illness, or death in return for the payment of a specified premium. There are multiple factors that play a major role in capturing customers for any insurance policy. Here we have information about demographics such as age, gender, region code, and vehicle damage, vehicle age, annual premium, policy sourcing channel. Based on the previous trend, this data analysis and prediction with machine learning models can help us understand what are the reasons for news popularity on social media and obtain the best classification model.

Problem Statement

Our client is an Insurance company that has provided Health Insurance to its customers. Now they need the help in building a model to predict whether the policyholders (customers) from the past year will also be interested in Vehicle Insurance provided by the company.

An insurance policy is an arrangement by which a company undertakes to provide a guarantee of compensation for specified loss, damage, illness, or death in return for the payment of a specified premium. A premium is a sum of money that the customer needs to pay regularly to an insurance company for this guarantee.

Building a model to predict whether a customer would be interested in Vehicle Insurance is extremely helpful for the company because it can then accordingly plan its communication strategy to reach out to those customers and optimize its business model and revenue.

Data Description

We have a dataset which contains information about demographics (gender, age, region code type), Vehicles (Vehicle Age, Damage), Policy (Premium, sourcing channel) etc. related to a person who is interested in vehicle insurance. We have 381109 data points available.

Feature Name	Type	Description
id	(continous)	Unique identifier for the Customer.
Age	(continous)	Age of the Customer.
Gender	(dichotomous)	Gender of the Custome
Driving_License	(dichotomous)	0 for customer not having DL, 1 for customer having DL.
Region_Code	(nominal)	Unique code for the region of the customer.
Previously_Insured	(dichotomous)	0 for customer not having vehicle insurance, 1 for customer having vehicle insurance.
Vehicle_Age	(nominal)	Age of the vehicle.
Vehicle_Damage	(dichotomous)	Customer got his/her vehicle damaged in the past. 0 : Customer didn't get his/her vehicle damaged in the past.
Annual_Premium	(continous)	The amount customer needs to pay as premium in the year.
Policy_Sales_Channel	(nominal)	Anonymized Code for the channel of outreaching to the customer ie. Different Agents, Over Mail, Over Phone, In Person, etc.
Vintage	(continous)	Number of Days, Customer has been associated with the company.
Response (Dependent Feature)	(dichotomous)	1 for Customer is interested, 0 for Customer is not interested.

MODEL USED :-

Decision Tree

Decision Trees are non-parametric supervised learning methods, capable of finding complex non-linear relationships in the data. Decision trees are a type of algorithm that uses a tree-like system of conditional control statements to create the machine learning model. A decision tree observes features of an object and trains a model in the structure of a tree to predict data in the future to produce output. For classification trees, it is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

Logistic Regression

Logistic regression is named for the function used at the core of the method, the logistic function.

The logistic function, also called the sigmoid function, was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

CODE AND SNIPPETS

```
# %% import pandas as pd

# train=pd.read_csv("./TRAIN-HEALTH INSURANCE CROSS SELL PREDICTION.csv",usecols=[0,1,2,3,5,6,7,8,11])

# %% train.head()

# %%


# %% [markdown]
# # data cleaning

# %% def vehicleAge_toInt(vehicleage):
if vehicleage =='> 2 Years':
        return 3     elif vehicleage =='1-
2 Year':
        return 2     elif vehicleage=='<
1 Year':
        return 1

# %%
print(train.columns)

# %%
train['Vehicle_Age_int']=train['Vehicle_Age'].apply(vehicleAge_toInt)

# %% train.head()

# %% def gender_to_int(gender):
if gender=='Male':
        return 1     elif
gender=='Female':         return 2


# %%
train['gender']=train['Gender'].apply(gender_to_int)


# %% train.head()

# %% def vehicle_damage_to_int(dam):
if dam=='Yes':         return 1     elif
dam=='No':         return 2

# %%
train['vehicle_damage']=train['Vehicle_Damage'].apply(vehicle_damage_to _int)

# %% train
```

```
# %% [markdown]
#

# %%
train.drop(['id','Gender','Vehicle_Age','Vehicle_Damage'],inplace=True, axis=1)

# %% train.head()

# %%


# %%
X=train.drop('Response',axis=1)
Y=train['Response']

# %% train.columns

# %%
from sklearn.model_selection import train_test_split

# %%
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.33,rando m_state=42)

# %%
from sklearn.linear_model import LogisticRegression

# %%
log_model=LogisticRegression()

# %%
log_model.fit(X_train,Y_train)

# %%
prediction=log_model.predict(X_test)

# %%
from sklearn.metrics import classification_report

# %%
print(classification_report(Y_test,prediction))

# %%
from sklearn.metrics import confusion_matrix

# %%
confusion_matrix(Y_test,prediction)

# %%
accuracy=log_model.score(X_test,Y_test)
```

```
# %% print(accuracy)

# %% import joblib
joblib.dump(log_model, 'logistic_regression_model.pkl')

# %%
# this model is around 87% accurate which is quite realisitic  This model has
been trained using LogisticRegression model
# The columns used in for classifiaction are age
Age,Driving_License,Previously_Insured,Annual_Premium  Response,Vehicle
_Age,gender,vehicle_damage

# usage of annual premium has significantly improved the accuracy of the model
```

start.py

```
import joblib
import pandas as pd
model = joblib.load('random_forest_model.pkl')

def default_dataframe(list1):
    df = pd.Series(0, index=list1)

    df = pd.DataFrame(df).T
    return df
def get_user_input(df):
    predictions = model.predict(df)
    return predictions
def features_list(df,val):
    for i in range(len(val)):
        df.iloc[0][i]=val[i]
    return df
def default_dataframe(feature_list):
    return pd.DataFrame(columns=feature_list)
def features_list(dataframe, values):
    dataframe.loc[len(dataframe)] = values
    return dataframe
if __name__ == "__main__":
    print("Machine Learning Model Predictions")

    list1 = ['Age', 'Driving_License', 'Previously_Insured', 'Annual_Premium',
'Vehicle_Age_int', 'gender', 'vehicle_damage']
    df = default_dataframe(list1)

    values = []
    for feature in list1:
        value = input(f"Enter value for {feature}: ")
        values.append(value)

    values = [int(val) if val.isdigit() else float(val) if val.replace('.', '',
1).isdigit() else val for val in values]
```

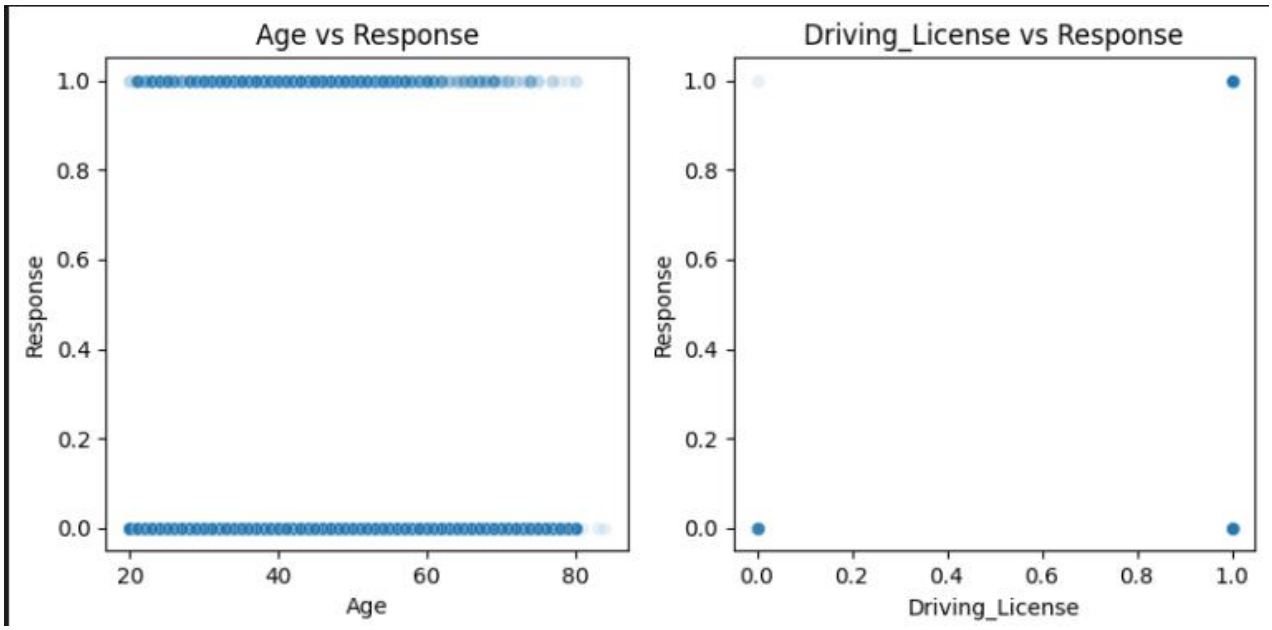
```
df = features_list(df, values)

print(df)
predictions=model.predict(df)
output=""
if(predictions[0]==1):
    output="Yes"
else:
    output="No"
print("prediction: ",output)
```

Result

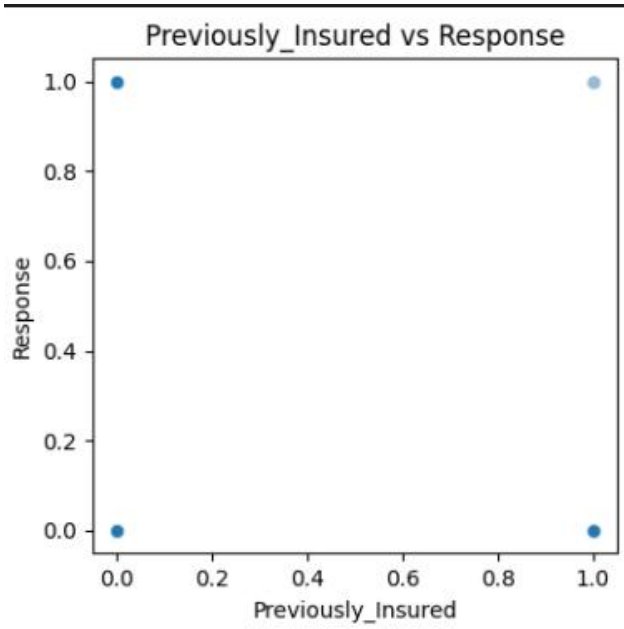
```
Classification Report for logistic_regression:
              precision    recall  f1-score   support

           0       0.88      1.00      0.93      2897
           1       0.00      0.00      0.00       403

    accuracy                           0.88      3300
   macro avg       0.44      0.50      0.47      3300
weighted avg       0.77      0.88      0.82      3300

Model 'logistic_regression' saved to logistic_regression_model.pkl
Classification Report for random_forest:
              precision    recall  f1-score   support

           0       0.89      0.93      0.91      2897
           1       0.28      0.20      0.23       403

    accuracy                           0.84      3300
   macro avg       0.58      0.56      0.57      3300
weighted avg       0.82      0.84      0.83      3300

Model 'random_forest' saved to random_forest_model.pkl
Classification Report for support_vector_machine:
              precision    recall  f1-score   support
...
   macro avg       0.44      0.50      0.47      3300
weighted avg       0.77      0.88      0.82      3300
```

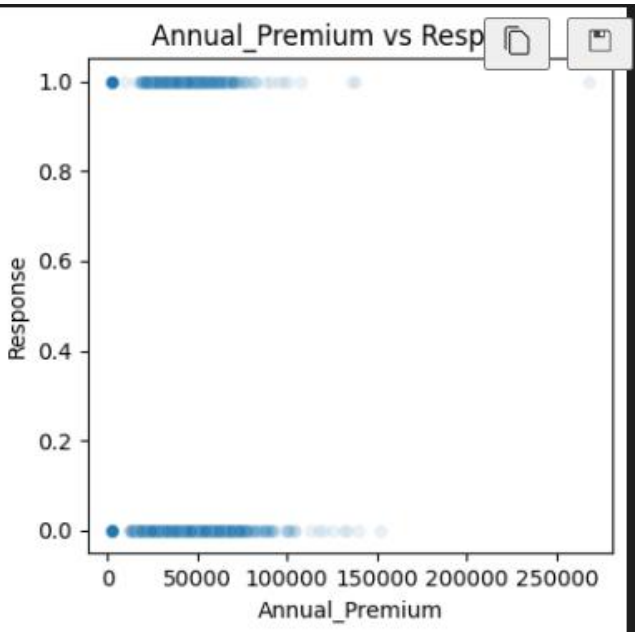
```
Machine Learning Model Predictions
Enter value for Age: 44
Enter value for Driving_License: 1
Enter value for Previously_Insured: 0
Enter value for Annual_Premium: 40000
Enter value for Vehicle_Age_int: 2
Enter value for gender: 1
Enter value for vehicle_damage: 2
   Age  Driving_License  Previously_Insured  Annual_Premium  Vehicle_Age_int  gender  vehicle_damage
0   44                1                   0           40000                2       1               2
prediction:  No
PS C:\Users\user\Downloads\python>
```

Graph


Age vs Response
Response
Age
1.0
0.8
0.6
0.4
0.2
0.0
20
40
60
80

Driving_License vs Response
Response
Driving_License
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


Previously_Insured vs Response
Response
Previously_Insured
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


Annual_Premium vs Resp
Response
Annual_Premium
1.0
0.8
0.6
0.4
0.2
0.0
0
50000
100000
150000
200000
250000

Challenges Faced

- Handling Large dataset.
- Already available methods of hyper-parameter tuning were taking a huge amount of time to process.
- Memory Optimization during hyperparameter tuning.

Conclusion

Starting from loading our dataset, we firstly performed data cleaning and refactoring by outlier detection and normalization of data. Then we covered EDA, feature selection and algorithm selection, and hyperparameter tuning. The Accuracy score obtained for all models was in the range of 68% to 85% before tuning After tuning the models we were able to get an accuracy of approx 87%. But we selected our best model as the model with an accuracy score of 85% considering precision and recall as we have an unequal number of observations in each class in our dataset, so accuracy alone can be misleading.