Team 9 Problem Pipeline

Overview

We have been assigned a problem of Porto Seguro, from the insurance domain. https://www.kaggle.com/c/porto-seguro-safe-driver-prediction.

We have tried to solve the given/ assigned problem by performing following stages.

1. Problem Understanding
2. Data Preparation / Analysis
3. Model Selection
4. Training and Testing the Model
5. Evaluating Model

Solution Approach

The stages are detailed below, following stages describes what we have done, how we have approached to the problem to solve it.

1. **Problem Understanding**

   Our problem required us to predict if a driver will file an insurance claim next year. We have been given two data set – train and test data. We had a challenge in interpreting and understanding the provided data, as to how it relates to features provided in the data set. In order to understand the problem and data in detail, we had to explore further on:

   - Domain – insurance claim, auto insurance premium etc.
   - Found the various factors that affects an auto insurance claim - HDI (Human Development Index) – lifestyle, locality, driving skills and history; Safety ratings of vehicle, etc.

2. **Data Preparation / Analysis**

   a. Examining the data

      The data was provide in CSV format, there was one training data file and one test data file. The data columns names were prefixed as ps_ind_ or ps_reg or ps_car and ps_calc. The exploration for domain understanding helped us to

co-relate the given data sets. We could map some of column names with the factors that affects the insurance claim. This helped us to understand the features and target value.

b. Data analysis

We analyzed the data further to prepare for machine learning model. We found that values given in the data set was in the state that can be used for the model directly and since, all values were numerals; we did not need to apply the data transformation step. There were 57 features in training data file and one target or output. Every record in the data file is associated with an insurance claim made by a driver in the past. The target has a binary value where zero indicates driver has not claimed and one indicates that driver has claimed. The data has also missing feature values.

Sr.#	**Data items**	**Observations**
1.	Features	57
2.	Target / Output	1
3.	Target type	Binary
4.	Total Records	~595 K
5.	Records with target value zero	~573 K
6.	Records with target value one	~22 K

c. Feature engineering & selection

During analysis phase, we realized that our data set has lot of features and we needed to separate relevant and non-relevant features. We explored ways to find correlated features that can be grouped, which would lower the feature count. However, the data set provided no details of what the feature is and in without this information we could not find correlated features and decided to go ahead with the entire list of features.

d. Data Cleaning / Finding Outlier,

We had to address the following issues in the data available to us:

- There were lot of records which had one or more missing feature values
- The number of records where the insurance was not claimed was significantly higher than the record where insurance was claimed

Therefore, we manually cleaned the data set, and removed the rows, which had missing feature values. We came up with a data set with approximately equal numbers of record where the insurance was claimed and not claimed i.e. with the target value '1' & '0' respectively, to avoid overfitting/under fitting.

Resulting data set detail is as follows:

Sr.#	Data items	Observations
1.	Features	57
2.	Target / Output	1
3.	Target type	Binary
4.	Total Records	~12K
5.	Records with target value zero	~6K
6.	Records with target value one	~6K

3. Model Selection

From the data preparation/analysis, stage we concluded that the problem given to us falls in **classification category**.

With the given data setup, we chose to apply decision tree, random forest and neural network classifier. We excluded logistic regression and Naive Bayes considering the feature count and data volume.

4. Training and Testing the model

We have tried to read our data set, which is in csv format using ***pandas*** library.

We explored ways to split the data set in training & test set for model evaluation. We had to choose between K-Fold Cross validation & train test split function and we decided to utilize '***train_test_split'*** function from ***sklearn.model_selection***, since it was easier to implement. It split our data set into train and test with 75% record in train & 25% record in test. Please see the table below:

Sr.#	Data items	Count
1.	Features	57
2.	Target / Output	1
3.	Train_Feature Set	~9K
4.	Train_Target Set	~9K
5.	Test_Feature Set	~3K
6.	Test_Target Set	~3K

The split data set was used in following classifier for training:

1) Random Forest
2) Decision Tree

5. Evaluate the model Accuracy

We have used **model accuracy and confusion matrix from sklearn metrics** library to calculate model accuracy and confusion matrix.

Model Accuracy details are as follows:

1) Random Forest

Random Forest Classifier	
Train Accuracy	***0.996041345942***
Test Accuracy	***0.973175021988***

Confusion Matrix:

	Predicted Positive	Predicted Negative
Labeled as Positive	1113	5
Labeled as Negative	56	1100

2) Decision Tree

Decision Forest Classifier	
Train Accuracy	***1.0***
Test Accuracy	***0.945460942998***

Confusion Matrix:

	Predicted Positive	Predicted Negative
Labeled as Positive	1302	94
Labeled as Negative	61	1385