Intelligent Insurance Subscription Prediction System

Y.J. QUEK, A.N.A. NAZARI, N.H. HAMDAN, M.F. JAFRI

***Abstract*— This paper presents some methods together with its justification and analysis of few experiments conducted regarding user or drivers' preferences on classify whether driver is safe or unsafe and choosing car insurance package for those who drive unsafe. A set of data were used to conduct the experiments which can be found as an open source dataset at Kaggle. There are 2 platforms used to conduct the experiments, namely Jupyter Notebook where python code is used to do the analysis and modelling while the second platform is Rapidminer which is used to do some rapid process of our experiments and data exploration. All categorical values are converted into numerical form before loaded into Rapidminer and execute at Jupyter Notebook. Different results and accuracy for each experiment is obtained due to different algorithm implemented on each experiment. Therefore, data preparation and feature engineering are crucial part that need to be done before proceeding with the next following task.**

I. INTRODUCTION

Mainly, this project is implemented due to the requirement for BITI2223 Machine Learning subject which is aimed to apply the knowledge of data mining and machine learning technique towards a certain case study or scenario. A group of students consists of 4 members are freely to choose any dataset from any domain that is suitable to mine useful and hidden information from the chosen dataset. For this paper, a dataset of car insurance which consist of various feature that affect the driver or customers preferences on choosing which type of insurance that fits their requirements is chosen. The purpose of this project is aimed (1) to provide one model to determine whether the driver is a safe driver or dangerous driver based on selected criteria, (2) to suggest a plan insurance based on driver's criteria, (3) to create a Machine Learning model to solve problem, make a data exploration to do data set, and to find the insight of the data. Few milestones must be followed due to project evaluation: (1) proposal submission, (2) final presentation for this mini project.

II. PREPARATION OF DATA

A. Choose and retrieve the dataset

The dataset is retrieved from *https://www.kaggle.com/* which consist variety of open source dataset where most of them were contributed by online community that works on various data science project.

III. TOOLS

When it comes to data mining process, Rapidminer is used to do some rapid prototyping as well as performing the desired modelling techniques that suits the problem to be solved. Jupyter Notebook is also used to do some additional analysis which is not really available on Rapidminer such as some analysis graphs is much simpler to generate using Jupyter Notebook and easy to understand.

A. Retrieve, Replace and Balancing data

Basically, the data is imported into Rapidminer or Jupyter Notebook to make some early analysis regarding the type of the data whether it is observable or not. In this project, the dataset provided from Kaggle is highly not balanced which will cause bias to occur if we do not do any preprocessing to the data. Therefore, under sampling is used to stratified sample the class which consists of higher number of rows to make it balance between the 2 class. Imputer in Scikit-Learn preprocessing is used to handle the missing value consists in the dataset. All the missing value will be replaced with the median value.

B. Data Exploration using Python

Jupyter Notebook is an open-source web application where python codes can be debugged based on a certain portion or segment of code. Data exploration can be done by using rapid miner and Jupyter Notebook.

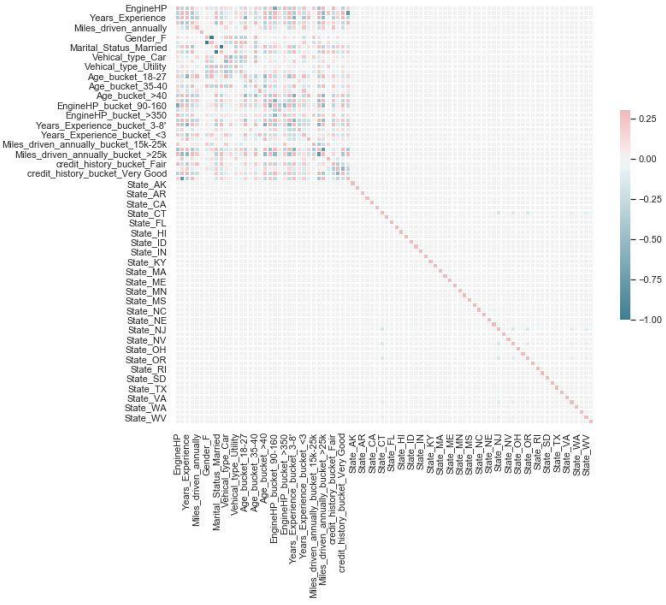

Figure 1: The correlation graph shows that attribute that does not correlated with each other's will be indicate in dark blue in color

C. Attribute Generation

This is where the new attribute is generated from the existing attribute. For example, State attribute. There are a lot of states

assign for each instance. The same state is categorized with regions; North, East, West and South. All this region will become the new attribute which derived from State attribute.

State	North_us	South_us	East_us	West_us
IL	0	0	0	1
NJ	1	0	0	0
CT	0	1	0	0
CT	0	1	0	0
WY	0	0	0	1
DE	0	0	1	0
NJ	1	0	0	0
ME	1	0	0	0
CA	0	0	0	1

Figure 2: The table shows that the new attribute is generated from the State attributes.

D. Data Exploration

An approach similar to initial data analysis, whereby a data analyst uses visual exploration to understand what is in a dataset and the characteristics of the data, rather than through traditional data management systems. These characteristics can include size or amount of data, completeness of the data, correctness of the data, possible relationships amongst data elements or files/tables in the data.

i. Boxplot

Boxplot is a standardized way of displaying the distribution of data based on a five number summary (“minimum”, first quartile (Q1), median, third quartile (Q3), and “maximum”). It can tell you about your outliers and what their values are. It can also tell you if your data is symmetrical, how tightly your data is grouped, and if and how your data is skewed.

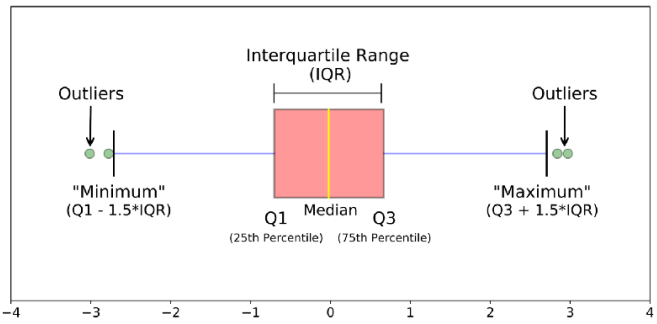

Figure 3: Diagram shows there’s an outlier in the boxplot.

ii. Histogram

A graphical display of data using bars of different heights. In a histogram, each bar groups numbers into ranges. Taller bars show that more data falls in that range. A histogram displays the shape and spread of continuous sample data.

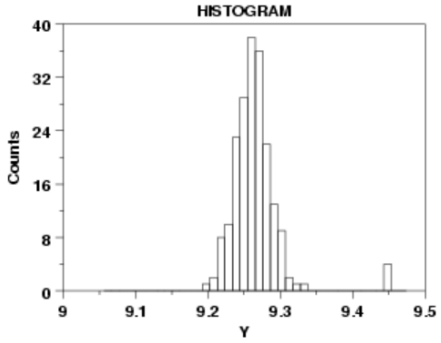

Figure 4: Diagram shows there’s an outlier in the histogram diagram.

E. Feature Reduction

Number of features increases as one-hot-encoding is applied to some features. Some features are meant to be redundancy towards the dataset. Eliminate them by using below techniques:

i. Principle Component Analysis (PCA)

```
# Use PCA from sklearn.decompostion to find principal components
from sklearn.decomposition import PCA

pca = PCA(n_components=10)
X_pca = pd.DataFrame(pca.fit_transform(X))
```

```
print(X_pca.head(5))
```

```
             0           1           2         3         4         5  \
0 -11205.870349  -84.311154    8.705676 -1.209805 -2.543893 -0.134821
1  -2313.591364   92.576508  208.884612  0.987784 -3.481602  0.511398
2  -7384.747473  221.872622 -133.775572 -3.211167  1.468912  1.151773
3  -3172.780968   11.636658   28.146495 -6.966624 -1.521874  0.243743
4  -7720.925258  -96.805444  -56.440236  1.714797  3.450408 -0.529509

          6         7         8         9
0  0.389585  0.253792 -0.905826  0.414131
1 -0.764217 -0.548837  0.132461 -0.310542
2 -1.060503 -0.538993 -0.267172  0.561441
3 -1.123501 -1.228541  0.529195  0.303065
4  0.033012 -0.871933 -0.708578 -0.349515
```

Figure 2: PCA Dimension Reduction to only 10 components

ii. Backward Elimination
iii. Forward Elimination
iv. Brute Force
v. Evolutionary Component (Optimization)

F. Feature Selection

A dataset consists of various features but not all features are important for data mining process. The importance of a certain features can be identified using selection technique as follows:

i. Select by Weight

This operator selects only those attributes of the dataset whose weights satisfy the specified criterion with respect to the input weights. Input weights are provided through the *weights* input port. The criterion for attribute selection by weights is specified by the *weight relation* parameter.

ii. Select by Random

The Select by Random operator selects attributes randomly from the dataset. If the *use fixed number of attributes* parameter is set to true, then the required number of attributes is specified through the *number of attributes* parameter. The randomization can be changed by changing the seed value in the corresponding parameters.

G. Modelling

The model is formed from several numbers of instances alongside with various features. Various algorithms are used to obtained the best model while features that is less important were ignored. The dataset is then split into 2 which are testing, training. Below is the algorithm used while conducting the experiment:

i. Ensemble Model (Random Forest)

```
# Try with Random Forest Classfier

from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(n_estimators=400)
rf.fit(X_train, y_train)
y_pred = rf.predict(X_test)

estimator = rf.estimators_[0]

from sklearn.tree import export_graphviz
# Export as dot file
export_graphviz(estimator, out_file='tree.dot',
                #feature_names = [0,1,2,3,4,5,6,7,8,9],
                #class_names = [0,1],
                rounded = True, proportion = False,
                precision = 2, filled = True)

# Convert to png using system command (requires Graphviz)
#from subprocess import call
#call(['dot', '-Tpng', 'tree.dot', '-o', 'tree.png', '-Gdpi=600'])

accuracy_randomForest = accuracy_score(y_test, y_pred)
recall_randomForest = recall_score(y_test, y_pred)
conf_mat_randomForest = confusion_matrix(y_true=y_test, y_pred=y_pred)
precision_randomForest = precision_score(y_test, y_pred)
f1_score_randomForest = f1_score(y_test, y_pred)

print("Accuracy: %.2f%%" % (accuracy_randomForest * 100.0))
print("Recall Score", recall_randomForest)
print("Precision Score: ", precision_randomForest)
print("Confusion Matric Random Forest: ", conf_mat_randomForest)
print("F1 Score: ", f1_score_randomForest)
```

```
Accuracy: 50.50%
Recall Score 0.49924357034795763
Precision Score:  0.5032405642394205
Confusion Matric Random Forest:  [[1360 1303]
 [1324 1320]]
F1 Score:  0.5012340991076515
```

Figure 2: Python code to run Random Forest Classifier

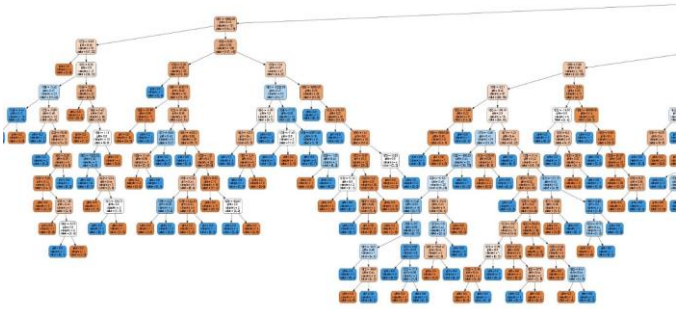

Figure 3: Part of the Random Forest Classifier

ii. Non-linear Support Vector Machine (SVM)

```
# Try with Non Linear SVM Model without PCA dimension reduction

svmnonlinear_noPca = svm.NuSVC(kernel='rbf',gamma='scale', probability=True)
svmnonlinear_noPca.fit(X_train,y_train)
y_pred = svmnonlinear_noPca.predict(X_test)

accuracy_nonlinearsvm = accuracy_score(y_test, y_pred)
recall_nonlinearsvm = recall_score(y_test, y_pred)
conf_mat_nonlinearsvm = confusion_matrix(y_true=y_test, y_pred=y_pred)
precision_nonlinearsvm = precision_score(y_test, y_pred)
f1_score_nonlinearsvm = f1_score(y_test, y_pred)

print("Accuracy Non-linear: %.2f%%" % (accuracy_nonlinearsvm * 100.0))
print("Recall: ", recall_nonlinearsvm)
print("Precision: ", precision_nonlinearsvm)
print("Confusion Matric: ", conf_mat_nonlinearsvm)
print("F1 Score: ", f1_score_nonlinearsvm)
```

```
Accuracy Non-linear: 50.03%
Recall:  0.8744326777609682
Precision:  0.4991364421416235
Confusion Matric:  [[ 343 2320]
 [ 332 2312]]
F1 Score:  0.6355140186915889
```

Figure 4: Python Code to run Support Vector Machine

iii. Logistic Regression

```
# Try with Logistic Regression

lr = LogisticRegression()
lr.fit(X_train,y_train)
y_pred = lr.predict(X_test)

accuracy_logistic = accuracy_score(y_test, y_pred)
recall_logistic = recall_score(y_test, y_pred)
conf_mat_logisticRegression = confusion_matrix(y_true=y_test, y_pred=y_pred)
precision_logisticRegression = precision_score(y_test, y_pred)
f1_score_logisticRegression= f1_score(y_test, y_pred)

print("Accuracy: %.2f%%" % (accuracy_logistic * 100.0))
print("Recall Score", recall_logistic)
print("Precision Score: ", precision_logisticRegression)
print("F1 Score: ", f1_score_logisticRegression)
print("Confusion Matric Logistic: ", conf_mat_logisticRegression)
```

```
C:\Users\Sky\Anaconda3\envs\mini-project\lib\site-packages\sklearn\linear_mod
will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
Accuracy: 49.56%
Recall Score 0.4519667170953101
Precision Score:  0.4931902600082542
F1 Score:  0.47167949477008086
Confusion Matric Logistic:  [[1435 1228]
 [1449 1195]]
```

Figure 5: Python Code to run Logistic Regression

H. Accuracy and Measurement

EXPERIMENT	ALGORITHM	F1-Score
1	Random Forest	50.12%
2	Non-Linear SVM	63.55%
3	Logistic Regression	47.17%

Table 1: Model with F1-Score before feature engineering

EXPERIMENT	ALGORITHM	F1-Score
1	Random Forest	48.39%
2	Non-Linear SVM	51.10%
3	Logistic Regression	54.25%

Table 2: Model with F1-Score after feature engineering

The accuracy is measured by a few techniques namely:

i. Confusion Matrix

A confusion matrix of our safe driver dataset will be produced 2 by 2 table formed by counting the number of 4 outcomes of binary classifier. We can name them as TP, FP, TN, and FN.

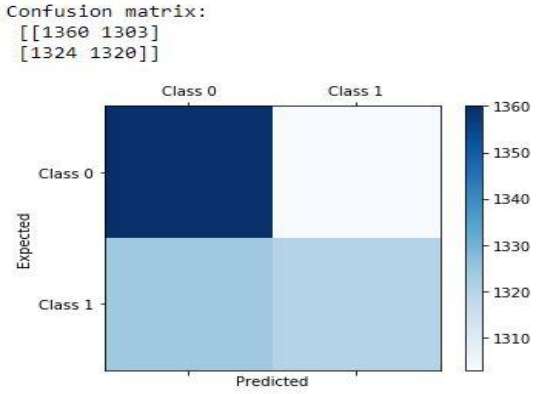

Figure 6: Example of the confusion Matrix which consist of TP = 1320, TN =1360, FP = 1324, FN = 1303.

ii. Accuracy

Accuracy is one metric for evaluating classification models. Accuracy is measure by Number of correct predictions which contains TP and TN divided by Total number of predictions. This evaluating metric is not suitable for certain cases such as imbalance class in a dataset. In imbalance class, certain class has greater amount of dataset than the other class. This makes the prediction inaccurate as the total amount of data is highly imbalance. This is called as Unbalance classification problems. [4] For example, we have an all our data predict on the unsafe driver. We will get the high prediction of accuracy, but we do not have any single data predicted of safe driver. This is due to the imbalanced dataset.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Figure 7: Formula of the accuracy

iii. Recall

Recall is the ability of a model to find all the relevant cases within a dataset. The precise definition of recall is the number of true positive divided by the number of true positive plus the number of true negatives. [3] In our case, our recall will be number of correctly identified unsafe driver divided by the number of correctly identified unsafe driver plus the unsafe driver that is incorrectly labelled as safe driver.

$$recall = \frac{true\ positives}{true\ positives\ + false\ negatives}$$

Figure 8: Formula of the Recall

iv. Precision

Precision attempt to find the proportion of positive identification was correct. In recall measurement, we only measure for the accuracy of the how likely our data able to predict the correct model, but precision taken account of the data how many times the models makes false positive prediction. For example, precision taken account of the data on how many times safe driver is prediction as unsafe driver.

$$\text{Precision} = \frac{TP}{TP + FP}$$

Figure 9: Formula of the Precision

v. F1 Score

F1-score is the harmonic means of the precision and recall by taking both metrics into account. So, in our case we have to maximize the F1 score.

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

Figure 10: Formula of the F1 Score

Table 1 and table 2 above shows the accuracy of the models before and after feature engineering, we can clearly see that accuracy has dropped after feature engineering being applied. After applied min max scalar and PCA dimension reduction, the accuracy dropped might due to lose of important data or feature. Other method of feature engineering needs to be applied in order to increase its accuracy or f1 score.

I. Conclusion

From all the comparisons of the experiments, it can be concluded that non-linear SVM obtained the best model with accuracy of 63.55%. We believe that this dataset is non-linear and models such as XGBBoost, LightBoost, Neural Network or Deep Learning method able to achieve higher accuracy because those are the model or technique that is really good in handling non-linear data.

ACKNOWLEDGEMENT

First of all, we would like to express our gratitude to our lecturer, Associate Professor Dr Choo Yun Huoy for her guidance and patience in giving the knowledge required by the students. A special thanks to the dataset contributor Nelson who open sourced the dataset on Kaggle for research purposes. Last but not least, a special thanks also to our team members who

gave their full commitments by contributing ideas in order to make this project becomes successful.

REFERENCES

[1] Classification: Precision and Recall | Machine Learning Crash Course | Google Developers https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

[2] Basic evaluation measures from the confusion matrix https://classeval.wordpress.com/introduction/basic-evaluation-measures/

[3] Beyond Accuracy: Precision and Recall Will Koehrsen- https://towardsdatascience.com/beyond-accuracy-precision-and-recall-3da06bea9f6c

[4] Measuring just Accuracy is not enough in machine learning, A better technique is required... Aakash Bindal - https://medium.com/datadriveninvestor/measuring-just-accuracy-is-not-enough-in-machine-learning-a-better-technique-is-required-e7199ac36856