Classification Project

Introduction/Premise

The main purpose of this project is to not only show that I can apply what I learned from a recent Machine Learning course that I took on Udemy, but also to extend my understanding of those topics further. This project focuses on classification techniques.

The original premise of this project is to find the most accurate way to classify/predict if the customers are going to file claims or not using a plethora of features (as provided in the Inputs chart). This is a solid start, but I understand that there are always techniques to improve on this information.

Inputs

These are the inputs provided as well as what modifications I made to them in order to run the algorithms.

Features	Description	Modifications/Encodings
ID	Numerical identifier used in place of names.	• I deleted this feature.
AGE	The age of the person in the form of the following groupings: • 65+ • 16-25 • 26-39 • 40-64	• I encoded this feature as an ordinal value.
GENDER	The customer's gender: • Male • Female	• I encoded this feature as an nominal value.
RACE	The customer's race which is categorized as either majority or minority. • Majority • Minority	• I encoded this feature as an nominal value.
DRIVING_EXPERIENCE	The customer's driving experience categorized as follows:	• I encoded this feature as an ordinal value.

	• 0-9y • 10-19y • 20-29y • 30y+	
EDUCATION	The customer’s level of education categorized as follows: • High school • None • University	• I encoded this feature as an nominal value.
INCOME	The customer’s level of income categorized as follows: • Upper class • Poverty • Working class • Middle class	• I encoded this feature as an ordinal value.
CREDIT_SCORE	The customer’s credit score as a fraction of one in decimal form.	• I encoded this feature as an nominal value.
VEHICLE_OWNERSHIP	A Boolean value of whether or not the customer owns their vehicle. • Zero (0) means that they do not • One (1) means that they do	• Converted the data type from float to integer
VEHICLE_YEAR	This feature is grouped into two values: • Before 2015 • After 2015	• I encoded this feature as an nominal value.
MARRIED	A boolean value of whether the customer is married or not.	• Converted the data type from float to integer
CHILDREN	A boolean value representing whether the customer has children or not	• Converted the data type from float to integer
POSTAL_CODE	This is the 5-digit postal code of where the customer resides.	• I deleted this feature.

ANNUAL_MILEAGE	The approximate mileage that the customer puts on their vehicle.	• No modifications necessary
VEHICLE_TYPE	The type of vehicle that the customer is insuring. The values are: • Sedan • Sports car	• I encoded this feature as an nominal value.
SPEEDING_VIOLATIONS	The number of speeding tickets that the customer has received.	• No modifications necessary
DUIS	The number of DUIs that the customer has.	• No modifications necessary
PAST_ACCIDENTS	The number of past accidents that the customer has had.	• No modifications necessary
OUTCOME	If the customer made/filed a claim. • Zero (0) means that they did not • One (1) means that they did	• Converted the data type from float to integer

Other Data Preprocessing

From all of the research that I have found, the best technique for imputing missing data is using the MICE technique. I selected the credit score feature for the imputed value/feature for this as there are many more possible values for that feature than for the annual mileage feature. The annual mileage feature is largely in increments of thousands (1000) of miles.

The Naïve Bayes algorithm used the MinMax scaler. The other algorithms utilized the StandardScaler. Scikit-Learn provided both of the scalers. After that, I reshaped the outcome value, so it would work properly with the algorithms.

All ordinal features were encoding using Ordinal Encoding functionality from Scikit-Learn. Additionally, all nominal features were encoding using One Hot Encoding functionality from Scikit-Learn.

Additionally, I split the data set in order for the data to work with the supervised learning algorithms. I set up the split so that two-thirds (2/3) of the data is for the training data and the remainder (one-third) is set aside for testing the models.

<u>Algorithms Utilized</u>

These are the algorithms that I used as well as some useful notes:

Algorithm	**Sub-Algorithm***	**Notes about Decisions Made**	**Cross Validation Score*** **
Naïve Bayes	**-**	• Used the MinMaxScaler instead of the StandardScaler	.9896
Decision Tree	-	-	1.0
Random Forest	-	-	1.0
KNN	n_neighbors=20	-	0.9624
KNN (25 iterations)	-	25 iterations of the KNN algorithm	0.9624 (min: 0.9606 & max: 0.9724)
SVM/SVC	kernel='rbf'	-	.9991
	kernel='linear'	-	1.0
	kernel='poly'	-	1.0
	kernel='sigmoid'	-	0.9803
Logistic Regression	-	-	1.0
Neural Network	-	• I used two layers. • The first (dense) layer included 8 units. I used the kernel initializer function and the relu activation function, while receiving the 23 features as the input dimension • The second (dense) layer included 1 unit. I used the kernel initializer function and the sigmoid activation function (since this is a binary classification). • I ran the algorithm 100 times (epochs) • The model was compiled with the binary_crossentropy loss function and utilized the adam optimizer	1.0

* When I used an algorithm multiple times with differences to the parameters, I refer to them as Sub-Algorithms for the purposes of this chart.

** This value was the highest cross validation score provided (rounded to 4 decimal places). The 25 neighbors value provided in the un-looped KNN was a ‘best guess’ without running through a loop of values. The optimal value that the KNN loop provided for n_neighbors was 45, instead of 25.
*** I rounded the cross-validation score to four decimal points if they were more than 4 digits to the right of the decimal point.

Conclusions

You really cannot go too wrong with any of the algorithms provided the material provided. If it was up to me, I would go with one of the following three algorithms: Random Forest, SVM using the linear kernel, or SVM using the poly kernel. My worry with the Decision tree is the potential for bias long term. While the Neural Network achieves a percent score as well, I would want a little more time to train it for me to feel comfortable with using it as “THE algorithm” to solve this problem.

Areas for improvement in Future

Due to the timing of when I completed this project, I wanted to make sure to post what I can right away, but that is the logical next step. That said, I do realize that there are additional outputs that I can add on to this project. Some of them include the following:

- The next logical step is to add some visualizations of each of the models
- Another addition to this project in the near future is to add predictions to each of the models
- Once I learn some better techniques to eliminate outlier data, I would like to implement it into this project