← Go Back to Advanced Machine Learning

Course Content

FAQ - Credit Card Users Churn Prediction

1. How should one approach the Credit Card Users Churn Prediction project?

- Before starting the project, please read the problem statement carefully and go through the criteria and descriptions mentioned in the rubric
- Then you should start with an exploratory analysis of the data.
- This understanding will help you identify the need for pre-processing the data.
- Once the data is ready, you can start with the steps that need to be followed as mentioned in the rubric
  - Build 6 models with original data
  - Build 6 models with oversampled data
  - Build 6 models with undersampled data
  - Choose 3 best models among 18 models built in the previous 3 steps
  - Tune 3 models
- It is important to close the analysis with key findings and recommendations to the business.

2. I am trying to fit a model and getting this error:

```
ValueError: could not convert string to float: 'M'
```

How to resolve?

Please check if the X_train and X_test consists of strings, and then create dummy variables using **pd.get_dummies**

3. I am getting this error while importing SMOTE even after successful installation of imblearn library:

```
ImportError: cannot import name 'delayed' from 'sklearn.utils.fixes' (C:\Users\anac
```

How to resolve?

1. Run **!pip install delayed** in your Jupyter notebook.
2. Restart the kernel and try importing SMOTE again.

4. I am getting this error while trying to tune random forest:

```
NotFittedError: All estimators failed to fit
```

How to resolve?

The Numpy library might not be updated. You can update the Numpy library to the latest version using

!pip install numpy==1.20.3 in your Jupyter notebook

OR

pip install numpy==1.20.3 in Anaconda prompt

5. Do we need to do anything with the income variable, mainly around the signs "K","$", "less"? Should we eliminate these and use a different range?

The category names can be renamed but it is not necessary as it won't affect your model in any way.

6. One column has "abc" values. Can I process by dropping these or replacing them with the most frequent values? Which one is recommended?

Dropping the values is not suggested. You can treat them as missing and replace them using an appropriate method.

7. I Did the capping method during EDA as shown in the supermarket campaign model notebook. But that was then capping the extreme outliers only. what about the remaining outliers. can I treat them after Splitting the data, before model building?

For finding outliers you can use the IQR method. But if you want to remove them based on say IQR method (say 25 and 75 percentiles) then after splitting data your test set might have a different range than your training set for a particular feature for which you want to detect the outliers. In this case, after applying IQR your test set might not represent the training set very well and this can reduce the accuracy of the test set.

It's recommended to remove outliers before splitting the dataset. Treating outliers depends on the business problem we need to analyze whether they are outliers or the values that can be possible.

Outliers need to drop because they can make your model worse. According to the business problem if there are more outliers and cannot be dropped just use any transformations such as log or square root so that we can reduce their effect.

8. Why I am getting a different number of columns after imputation and one-hot encoding?

The extra column you are getting is due to a common mistake while using simple imputer.

Whenever we use simple imputer you should fit only once not multiple times.

It's imputing the value 'married' for the missing value of the education_level column and while one-hot encoding it's creating a column for that.

The attached code is the correct way of doing

```
reqd_col_for_impute = [NAME OF COLUMNS WITH MISSING VALUES]
imputer = SimpleImputer(missing_values=np.nan, strategy="most_frequent")

# Fit and transform the train data
```

```
X_train[reqd_col_for_impute] = imputer.fit_transform(X_train[reqd_col_for_impute])

# Transform the validation data
X_val[reqd_col_for_impute] = imputer.transform(X_val[reqd_col_for_impute])

# Transform the test data
X_test[reqd_col_for_impute] = imputer.transform(X_test[reqd_col_for_impute])
```

9. How to deal with "ValueError: This solver needs samples of at least 2 classes in the data, but the data contains only 1 class"?

The target variable is not encoded properly. The error shows that there is only 1 class. Ensure that you are encoding the target features properly.

10. How to decide which approach should be taken to split the data into different sets and perform cross-validation?

We can take the following two approaches two split the data:

Train/Test:

You can split the data into train and test. Train the model using the training set and report cross-validation on the train set using K-Fold cross-validation. Check the CV score to assess the performance and then use the test set to assess the performance only on the final model.

We should follow this approach when we do not have enough data to create three splits.

For eg, we have 500 data points, splitting the data into 60% train, 20% validation, and 20% test would result in having very limited data points in the train set and our model will not be able to identify the relevant patterns. Hence, train/test split might be an appropriate strategy in such cases.

Train/Validation/Test: We can split the data into train, test, and validation. Train the model using the train set, check the model performance on the validation set and tweak the hyperparameters by checking the performance on the validation set. Use test set to assess the performance only on the final model.

We should follow this approach when we have enough data to create three splits.

For eg, We have 10k data points, splitting the data into 60% train, 20% validation, and 20% test would result in having fair enough data points in the train set and our model might be able to identify the relevant patterns. Hence, we should make a train/test split in such cases.