← Go Back to Advanced Machine Learning

Course Content

FAQs - Model Tuning

1. How do False-negative and False positive values change in the confusion matrix? Are they inversely related?

When we build a model and our target variable has a binary category and then we look at the confusion matrix. There are four elements:
True Positive (observed=1,predicted=1)
False Positive (observed=0,predicted=1)
True Negative (observed=0,predicted=0)
False Negative (observed=1,predicted=0)
The default threshold is 0.5. We change the threshold when we want to decrease or increase the number of false positives or false negatives.
Consider a credit card default problem for a bank. So if the model predicts that a person will default and the person doesn't then the bank might lose a potential customer but if the model predicts a customer would not default and he does, in the scenarios company would lose money. So the bank would want this number to be small.
Assuming defaulter to be 1 and non-defaulter to be 0 then the company would want lower False negatives. So we will try to build the model with a lower threshold so more people will be predicted as defaulters.
As you will try to decrease FN, FP will increase because of the values which were previously predicted as TN might change to FP, that is why prof must have said about the inverse relationship. So if we try to decrease FN via changing threshold all values in the confusion matrix change and FN will decrease but FP will increase.

2. How to deal with outliers?

How to treat outliers is a very tricky topic, it mainly depends on the problem statement. Sometimes we can't even remove outliers or treat them because that's just the way data is and should be kept like that for analysis.
Let us consider an example of a bank
In a bank, account balance might have high outliers as few people will have a high amount in their accounts, but should we be treating this value or removing it? So I think here we have to consider that in practical life situations we cannot remove these accounts and if we replace them with any other value that will make the data biased.

So the best option would be thinking w.r.t. the problem statement. If the problem statement is: Who is more likely to do a credit card default then I think we can remove people with a high balance in their account as they are less likely to default. In another case, if the problem statement is: Who is going to take a home loan then also I think we can remove outliers but if our problem statement is about business loans then we have to consider these outliers having high balance as they might take a loan for their business.

All the above judgments should be made by doing the necessary EDA on the data set to figure out the relationship of balance with the target (credit card default, home loan, business loan, etc.). None of the conclusions should come without the data supporting them.

Replacing the outliers is done when outliers are few in number and we think that this would be because of the wrong imputation. So we remove these outliers and treat them as missing data.

3. Why can't we use the ROC_AUC score in case of a skewed dataset?

This is because a small number of correct or incorrect predictions can result in a large change in the ROC Curve or ROC AUC score, that is why precision and recall are preferred more in case of skewed data.

4. What are imbalanced datasets?

Imbalanced datasets are those where there is a severe skew in the class distribution, such as 1:100 or 1:1000 examples in the minority class to the majority class.

This bias in the training dataset can influence many machine learning algorithms, leading some to ignore the minority class entirely. This is a problem as it is typically the minority class on which predictions are most important.

5. What are the techniques to handle imbalanced datasets?

One approach to addressing the problem of class imbalance is to randomly resample the training dataset. The two main approaches to randomly resampling an imbalanced dataset are to delete examples from the majority class, called **undersampling**, and to duplicate examples from the minority class, called **oversampling.**

6. What is Cross-Validation?

Cross-Validation is a very useful technique for assessing the performance of machine learning models. It helps in knowing how the machine learning model would generalize to an independent data set. You want to use this technique to estimate how accurate the predictions your model will give in practice.

7. What are polynomial Features and when to use them? why do we need additional features, which are going to be derived from existing features? Is it not the case of overfitting?
Polynomial features are used to convert features to a higher degree.
If there are 2 features [a, b], then degree-2 polynomial features are [1, a, b, a^2, ab, b^2].
This is a pre-processing technique after converting to a higher degree we can use the new features for any algorithm like a ridge, lasso, logistic, decision tree, etc.
We use polynomial features when the relationship between the target variable and independent variables is nonlinear.

One way to check whether a dataset is nonlinear is negative r2score. R2score will be negative when the relationship between the target variable and independent variables is nonlinear. Then we introduce the polynomial features to check for higher degree terms.
Polynomial features not necessarily increase overfitting, it increases redundant columns but not overfitting. So we remove all the correlated features when we want to check which features are driving the model if the sole aim of the model is a higher score then these redundant columns don’t hurt.
We can use polynomial features to check for interaction between the features as well by interaction_only=True.

8. How K-Fold Cross-validation works?
Here is the thing:

- If k-fold cross-validation is used to optimize the model parameters, the **training set** is split into k parts.

- Training happens k times, each time leaving out a different part of the training set.
- Typically, the error of these k-models is averaged.
- This is done for each of the model parameters to be tested, and the model with the lowest cross-validated and validation error is chosen.
- The test set has not been used so far.
- Only at the very end, the test set is used to test the performance of the (optimized) model.

```
# example: k-fold cross validation for hyperparameter optimization (k=3)

original data split into training and test set:

|---------------- train ---------------------|         |--- test ---|

cross-validation: test set is not used, error is calculated from
validation set (k-times) and averaged:

|---- train ------------------|- validation -|         |--- test ---|
|---- train ---|- validation -|---- train ---|         |--- test ---|
|- validation -|----------- train -----------|         |--- test ---|

final measure of model performance: model is trained on all training data
and the error is calculated from test set:

|---------------- train ---------------------|--- test ---|
```

- In some cases, k-fold cross-validation is used on the entire data set if no parameter optimization is needed (this is rare, but it happens).
- In this case, there would not be a validation set and the k parts are used as a test set one by one.
- The error of each of these k tests is typically averaged.

```
# example: k-fold cross validation

|----- test -----|------------ train --------------|
|----- train ----|----- test -----|----- train ----|
|------------ train --------------|----- test -----|
```

9. Should we apply the transformations AFTER splitting the data into train/test/validation to prevent data leaks?
Yes. Data preparation must be fit on the training dataset only. That is, any coefficients or models prepared for the data preparation process must only use rows of data in the training dataset.

Once fit, the data preparation algorithms or models can then be applied to the training dataset, and the test dataset.

1. Split Data.
2. Fit Data Preparation on Training Dataset.
3. Apply Data Preparation to Train and Test Datasets.
4. Evaluate Models.

10. How to install imblearn library?

To install imblearn you can use

```
!pip install imbalanced-learn==0.8.0 #in jupyter notebook # for Windows OS
```

```
pip install imbalanced-learn==0.8.0 # in anaconda prompt
```

11. I am getting this error while trying to import SMOTE

```
ImportError: cannot import name 'delayed' from 'sklearn.utils.fixes' (C:\Users\anaconda3
```

How to fix it?

1. Try to restart the kernel and then import SMOTE again
2. Try to reinstall using:

```
!pip install imbalanced-learn==0.8.0
```

```
!pip install delayed
```

Restart the kernel and now try to import SMOTE again

12. Why do we need to do one-hot encoding after splitting the data even though one-hot encoding doesn't lead to data leakage?

We might do one hot encoding after splitting data if we have missing values in categorical columns as we can't do one-hot encoding for variables with missing values

And missing value imputation leads to data leakage, so should be done after splitting the data.

13. How can we tackle the mismatch of the size of data due to different categories after creating dummies for train and test sets.

The feature mismatch issue happens when the features/columns in the test set are different from the train set after creating the dummies. We can align the features using the following code -

train, test = train.align(test, join="outer", axis=1, fill_value=0)

This will create a dummy column for the category that is not present in any of the sets.

‹ Previous

Next ›