CS5228 – Tutorial 5

Classification & Regression I (Evaluation)

1. **Basic classification metrics.** Assume that you have trained a binary classifier that aims to predict if a bank customer will default on his/her credit. Your test data contained 4,000 samples and your final model yields the following confusion matrix:

		actual label	
		1 (default)	**0** (no default)
prediction	**1** (default)	**260**	**610**
	0 (no default)	**10**	**3120**

(a) Calculate the *Accuracy*, *Specificity*, *Sensitivity*, *Recall*, *Precision*, and *F1 score*!

Solution:

- $Accuracy = \frac{260+3120}{260+3120+10+610} = 0.84$
- $Specificity = \frac{3120}{3120+610} = 0.84$
- $Sensitivity/Recall = \frac{260}{260+10} = 0.96$
- $Precision = \frac{260}{260+610} = 0.3$
- $F1 = 2 \cdot \frac{0.3 \cdot 0.96}{0.3+0.96} = 0.46$

(b) For each metric, provide a verbal interpretation of the resulting value in the context of predicting a customer's likelihood to default on his or her credit! Discuss if we can be happy with the result, or what result might cause problems in practice!

Solution:

- **Accuracy:** With respect to all predictions for all customers, the classifier will be correct about 84% of the time.

- **Specificity:** If a customer will *not* default, the classifier will very likely predict it correctly (with 84% probability).
- **Sensitivity/Recall:** If a customer will default, the classifier will very likely predict it correctly (with 96% probability).
- **Precision:** If the classifier predicts that a customer will default, it will be only correct 30% of the time
- **F1:** A balanced consideration of both Recall and Precision.

The rather high values for Accuracy and Specificity can be a bit misleading since the dataset is a bit imbalanced given that most customers do not default on the credit. However, the main problem is the low Precision value. It essentially means that the classifier will predict default "too often", i.e., in many cases where the customer would not default. If the bank decides to approve or deny credit based on this result, many customers will unjustifiably not qualify for a credit.

2. **Imbalanced datasets.** One reason why we have different metrics to evaluate the quality of a classifier is because of imbalanced datasets where a majority class contains most of the samples whereas a minority class contains only a fraction of samples (assuming a binary classification task).

   (a) List 5 example applications where you would expect a very imbalanced dataset.

   Solution: In general, every application where we need to predict a rare class qualifies here

   - Credit card fraud
   - Intrusion detection
   - Earthquake / tsunami / etc. warning system
   - Automated missile defense system
   - Suspect / criminal profiling

   (b) While not covered in the lecture, what do you think can be done to address the issue of imbalanced datasets (beyond picking the right metric)?

   Solution:

   - Collect more data, if possible and/or practical
   - Generation of synthetic data, if possible and/or practical
   - Undersampling of majority class

- Oversampling of minority class
- Data augmentation / ”smart” oversampling

3. **Assessing classification errors.** In case of a binary classification, we can make 2 types of errors:

   - False Positives (FP), also called Type I Error
   - False Negatives (FN), also called Type II Error

   In the lecture, we mentioned that in many cases these two types of errors are not equally problematic.

   (a) List 2 example applications where False Positives are more problematic than False Negatives, and vice versa. Provide a brief explanation!

Solution: False Negatives are more problematic than False Positives ⇒ aiming for a high Recall

- Fraud detection: When trying to detect fraudulent transactions in the banking etc., missing a true instance of fraud (false negatives) can result in financial losses for both the customers and the institution. While incorrectly flagging legitimate transactions as fraudulent (false positives) could inconvenience customers and lead to frustration, it’s often more critical to have a high recall to ensure that as many actual cases of fraud are caught as possible.
- Medical Diagnosis: Missing to classify a high-risk patient as such can be lethal to this patient. In contrast, incorrectly classifying a healthy patient as a high-risk one is arguably less problematic (although it is likely to cause these healthy patients to worry). Of course, false positives still are problematic as they can involve additional expensive tests and can worry the patients.

False Positives are more problematic than False Negatives ⇒ aiming for a high Precision

- Spam detection: It’s usually OK to occasionally let a spam email through the filter. However, filtering a non-spam email that might have been very important can have severe consequences. In other words, if we label an email as spam, we want to be very certain about it.
- Recommender systems: We can train a classifier to predict whether a user will like a, say, movie. Since there are countless numbers to choose from, there’s generally no harm to miss out on movies that we should recommend. However, recommending too many movies the users won’t like will negatively affect users’ perception of the quality of the recommender system.

(b) Regarding any difference between errors of Type I and Type II, how would you assess their relative importance for the following application use cases:

- Earthquake warning systems
- Automatic missile defense system

Solution: Intuitively, you don't want to miss any potential earthquake or any missile attack. So ideally the number False Negatives should be 0. However, since these are arguably rare events, aiming for $FN = 0$ is difficult to justify. The problem is that False Positives can also have severe consequences. In case of earthquakes, a false warning may lead to panic and over-reactions. So while False Negatives are arguably worse, the rarity of earthquakes and missile attacks (+ the consequences of False Positives) have to be taken into consideration.

4. **How good is "good enough"?**

(a) Say you trained a binary sentiment classifier that classifies social media posts (e.g., tweets) into "negative" and "positive". Your datasets for training and testing were balanced and sufficiently large. Let's assume that the F1 score of your classifier is 0.85. How would you assess if this is a good result?

Solution:

- A "random guesser" would get an F1 score of around 0.5, so the classifier is certainly much better than that
- Comparing a classifier typically involves a comprehensive comparison with other solutions so see how it performs against state-of-the-art models.
- While an F1 score 1.0 is the theoretical upper bound, in practice the goal is often below that. This is particularly true for such subjective use cases like sentiment analysis where different people might give different sentiments to the same tweet. For example, if people agreed on tweets' sentiments in 90% of the cases, then the classifier is rather close to this number.

(b) Say you trained a classifier that identifies whether an image contains a Car, Boat, or Plane, and the F1-score is very high, say, 0.99. Your datasets for training and testing were balanced and sufficiently large. What might be a reason why the classifier would suddenly perform poorly in practice?

Solution:

- Apart from the size of the dataset and the balance of class labels, another important requirement is that the dataset is *representative.*

- A "bad" dataset would contain only images with planes in the sky, boats on the open sea, and cars on roads. In this case, the classifier might identify planes because of the blue/grey background (sky).
- In short, it's not obvious that the classifiers learned the "correct" pattern.