# Machine Learning

## | Performance Measures

Prof. Dr. Mostafa Elhosseini

https://youtube.com/@ProfElhosseiniTechVerse

# Agenda

- Performance Measures
- Accuracy
- Cross-validation
- Dummy Classifier
- Model Metrics
- How accuracy misleads?
- Imbalanced data Examples (Skewed)

# Accuracy

- Being a machine learner, you should know how to evaluate your classification model.
- You should know what are the different metrics used to show the classifications and prediction score
- **Accuracy** is the widely used metric for evaluating the classification model's predictions.

$$\text{Accuracy} = \frac{\text{Total number of correct predictions}}{\text{Total number of predictions}}$$

# Cross-validation

- A good way to evaluate a model is to use cross-validation
- Use the cross_val_score() function to evaluate your SGDClassifier model using K-fold crossvalidation, with three folds.
- Remember that K-fold cross-validation means splitting the training set into K-folds (in this case, three), then making predictions and evaluating them on each fold using a model trained on the remaining folds

# Cross-validation

```
from sklearn.model_selection import cross_val_score

cross_val_score(sgd_clf, X_train, y_train_5, cv=3, scoring="accuracy")
```

```
array([0.95035, 0.96035, 0.9604 ])
```

- Above 95% accuracy (ratio of correct predictions) on all cross-validation folds

# Dummy Classifier

- A dummy classifier is a simple classifier that serves as a **baseline or reference point** for evaluating the performance of more complex machine learning models. It makes predictions based on **simple rules** or heuristics rather than learning from the training data.
- The purpose of using a dummy classifier is to **establish a benchmark** or a minimum performance level that any real classifier should aim to surpass. If a **sophisticated** machine learning model **fails** to outperform a dummy classifier, it indicates that the model **is not learning anything** meaningful from the data or that there might be issues with the data or the model's implementation.

# Dummy Classifier

- **Scikit-learn**, provides an implementation of the dummy classifier called **DummyClassifier**. It offers several strategies for making predictions:
  - **most_frequent**: The dummy classifier always predicts the most frequent class label in the training data.
  - **stratified**: The dummy classifier generates predictions by respecting the training data's class distribution.
  - **uniform**: The dummy classifier generates predictions uniformly at random.
  - **constant**: The dummy classifier always predicts a constant label that is provided by the user.

# Dummy Classifier

- The default strategy for the DummyClassifier in scikit-learn is "**prior**". When the strategy is set to "prior" (or when no strategy is explicitly specified), the dummy classifier makes predictions based on the class prior probabilities learned from the training data.
- Here's how the "**prior**" strategy works:
  - During the training phase, the dummy classifier computes the class prior probabilities by counting the frequency of each class label in the training data.
  - When making predictions, the dummy classifier selects the class label with the highest prior probability.
  - In other words, the "prior" strategy predicts the most frequent class label from the training data for all the instances in the test set, regardless of their feature values.

# Dummy Classifier

```
from sklearn.dummy import DummyClassifier

dummy_clf = DummyClassifier()
dummy_clf.fit(X_train, y_train_5)
print(any(dummy_clf.predict(X_train)))
```

```
False
```

```
cross_val_score(dummy_clf, X_train, y_train_5, cv=3, scoring="accuracy")
```

```
array([0.90965, 0.90965, 0.90965])
```

# Dummy Classifier

- It has over 90% accuracy! This is simply because only about 10% of the images are 5s, so if you always guess that an image is not a 5, you will be right about 90% of the time.
- This demonstrates why **accuracy is generally not the preferred performance measure** for classifiers, especially when you are dealing with **skewed** datasets (i.e., when some classes are much more frequent than others)

# Model Metrics

- There are many scenarios where **accuracy** metric works well but there are several cases where accuracy metric **might fool you**
- How accuracy misleads?
- This is a case of **imbalanced dataset** (**skewed**) where one class (large no of observations) overpowers another class (small no of observation)
- when some classes are much more frequent than others

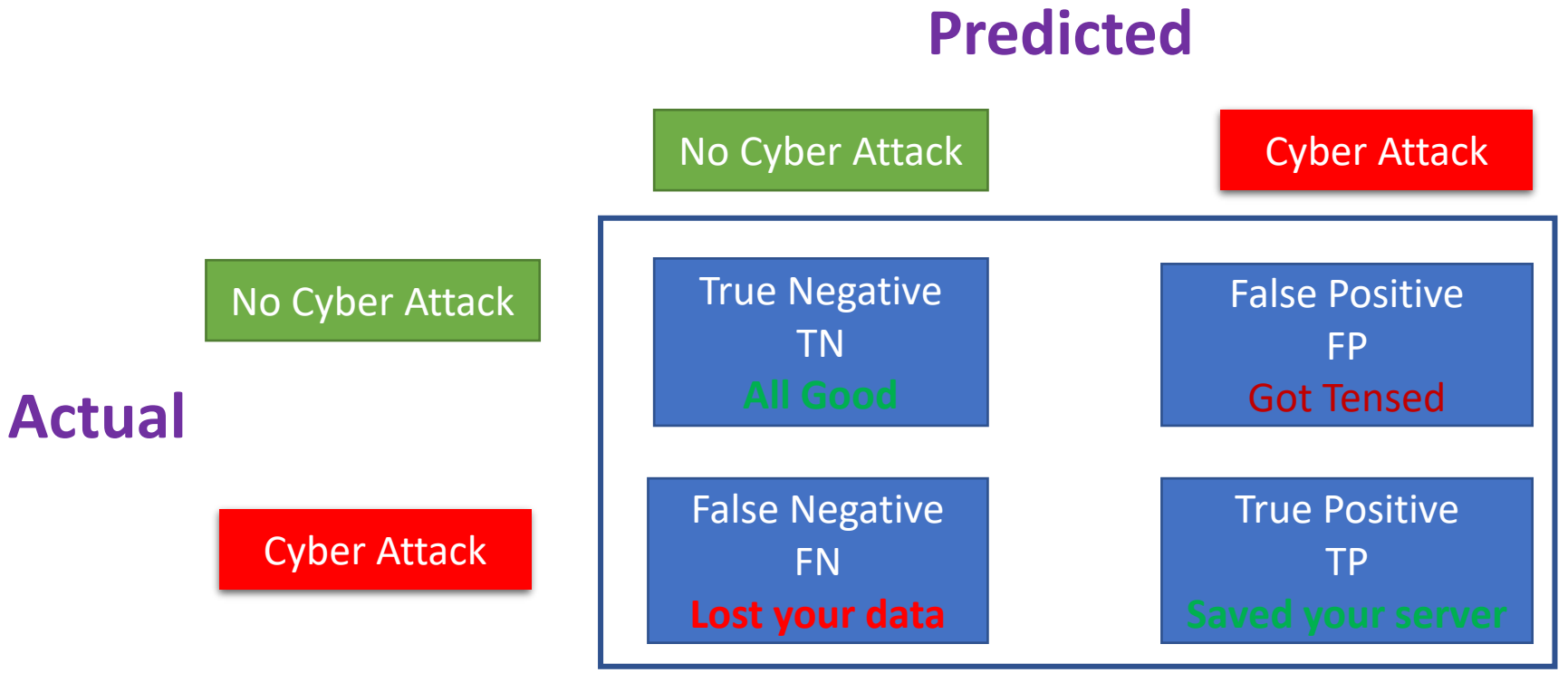
# How accuracy misleads?

- You have built an AI model to detect the intrusion and cyber attacks on your web server with real-time updates. Whenever there's an attack your model will activate an alarm

# How accuracy misleads?

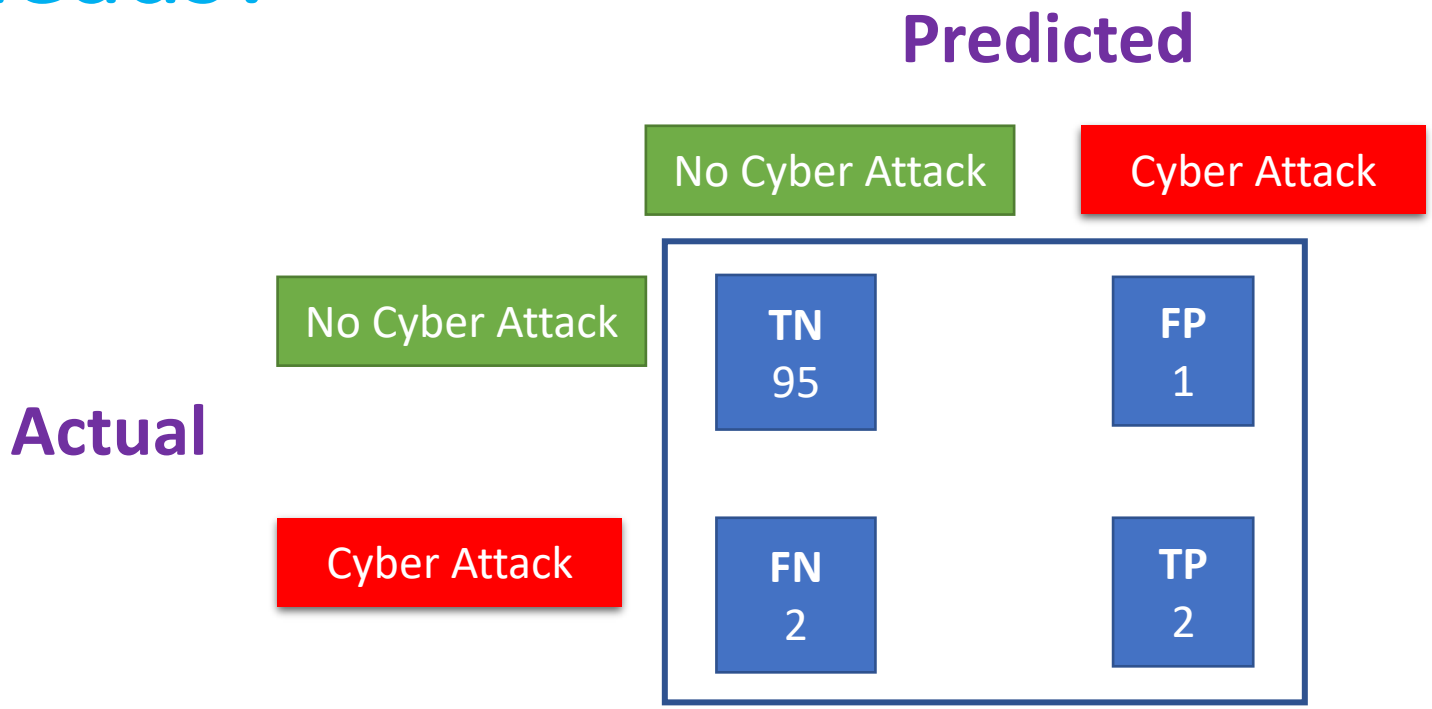
- True Positive (TP)
  - When model predicts churner (the positive class) and the actual class was also churner. It means the model correctly predicted the churner
- True Negative (TN)
  - When the model predicts non churner (the negative class) and the actual class was also non churner. It means the model correctly predicted the non churner class.
- False Positive (FP)
  - Model incorrectly predicted non churner as a churner customer
- False Negative (FN)
  - It means the model incorrectly predicted churner as a non churner customer

# How accuracy misleads?




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

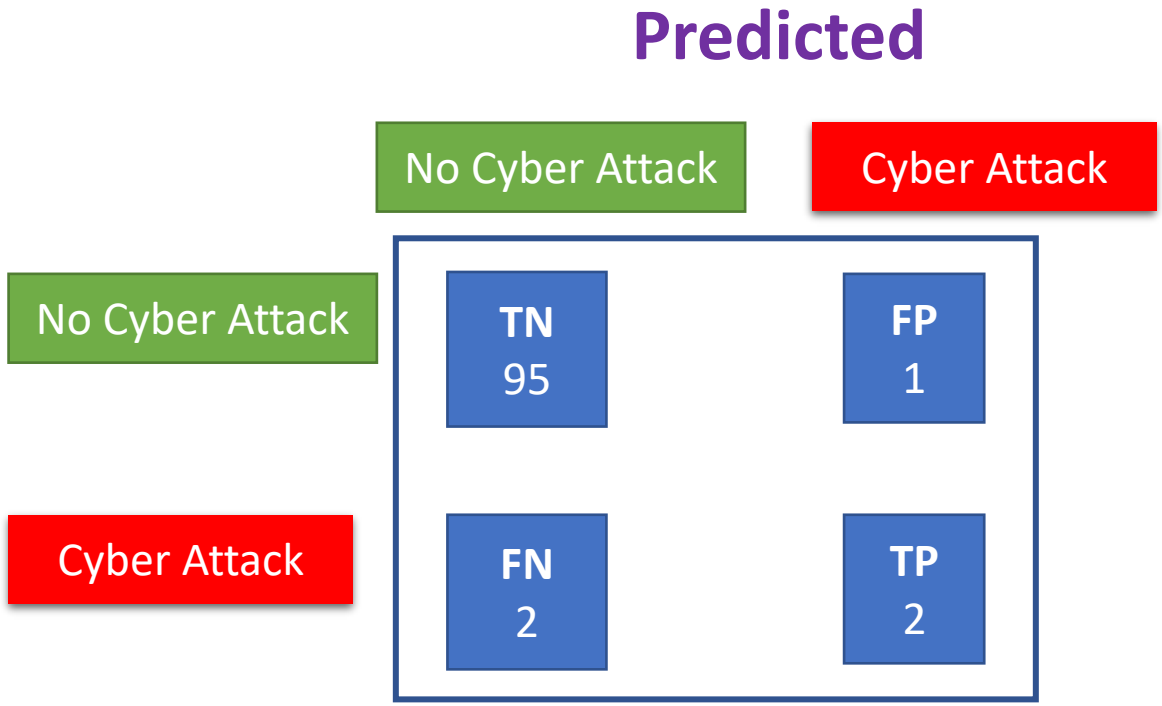
# How accuracy misleads?




- Out of 100 cases, there are 4 cyber attacks (TP and FN) and 96 non-cyber attacks (FP and TN)
- Out of 96 non-cyber attacks, 95 were correctly predicted. But out of 4 cyberattacks, only 2 were identified correctly. Not good

# How accuracy misleads?

- $Accuracy = \frac{2+95}{2+1+2+95} = 97\%$
- Well, done
- But …
- If you have **dumb classifier** that classify every sample as non-cyber attack, then also we will have **96% accuracy** which is misleading



# Imbalanced data Examples

- Disease detection: when rate of disease is very low
- Shoplifter
- Safe movie for kids
- The positive class — disease or Criminals — is greatly outnumbered by the negative class.

- These types of problems are examples of the fairly common case in data science when accuracy is not a good measure for assessing model performance

# Confusion Matrix

- This demonstrates why accuracy is generally not the preferred performance measure for classifiers, especially when you are dealing with skewed datasets (i.e., when some classes are much more frequent than others)
- A much better way to evaluate the performance of a classifier is to look at the confusion matrix.
- To compute the confusion matrix, you first need to have a set of predictions, so they can be compared to the actual targets.
- **Next Lecture**