Berkeley Engineering | BerkeleyHaas

PROFESSIONAL CERTIFICATE IN MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE

Office Hour #12 with Jessica Cervi

Practical Application 2

Classification Models

A **supervised machine learning** algorithm is one that relies on labeled input data to learn a function that produces an appropriate output when given new unlabeled data.

When we see a pig, we shout "pig!" When it's not a pig, we shout "no, not pig!" After doing this several times with the child, we show them a picture and ask "pig?" and they will correctly (most of the time) say "pig!" or "no, not pig!" depending on what the picture is. That is supervised machine learning

K-Nearest Neighbors

The k-nearest neighbors algorithm is a non-parametric, supervised learning classifier, which uses **proximity** to make classifications or predictions about the grouping of an individual data point. It works off the assumption that **similar points can be found near one another**.

A class label is assigned on the basis of a **majority vote**—KNN performs a voting mechanism to determine the class of an unseen observation. This means that the class with the majority vote will become the class of the data point in question. "Majority voting" technically requires a majority of greater than 50%, which primarily works when there are only two categories. When you have multiple classes—e.g. four categories, you don't necessarily need 50% of the vote to make a conclusion about a class; you could assign a class label with a vote of greater than 25%.

K-Nearest Neighbors

K-nearest neighbor algorithm pseudocode
The following is the pseudocode for KNN:

1.Load the data
2.Choose K value
3.For each data point in the data:
    1. Find the Euclidean distance to all training data samples
    2. Store the distances on an ordered list and sort it
    3. Choose the top K entries from the sorted list
    4. Label the test point based on the majority of classes present in the selected points
4.End

K-Nearest Neighbors

How to choose the optimal value of K

There isn't a specific way to determine the best K value – in other words – the number of neighbors in KNN. This means that you might have to experiment with a few values before deciding which one to go forward with.

When dealing with a two-class problem, it's better to choose an odd value for K. Otherwise, a scenario can arise where the number of neighbors in each class is the same. Also, the value of K must not be a multiple of the number of classes present.

Another way to choose the optimal value of K is by calculating the sqrt(N), where N denotes the number of samples in the training data set.
However, K with lower values, such as K=1 or K=2, can be noisy and subjected to the effects of outliers. The chance for overfitting is also high in such cases.

On the other hand, K with larger values, in most cases, will give rise to smoother decision boundaries, but it shouldn't be too large. Otherwise, groups with a fewer number of data points will always be outvoted by other groups. Plus, a larger K will be computationally expensive.

K-Nearest Neighbors

Here are some of the **advantages** of using the k-nearest neighbors algorithm:
- It's easy to understand and simple to implement
- It can be used for both classification and regression problems
- It's ideal for non-linear data since there's no assumption about underlying data
- It can naturally handle multi-class cases
- It can perform well with enough representative data

Here are some of the **disadvantages** of using the k-nearest neighbors algorithm:
- Associated computation cost is high as it stores all the training data
- Requires high memory storage
- Need to determine the value of K
- Prediction is slow if the value of N is high
- Sensitive to irrelevant features

K-Nearest Neighbors Applications

- **Data preprocessing**: Datasets frequently have missing values, but the KNN algorithm can estimate for those values in a process known as missing data imputation.
- **Recommendation Engines**: the a user is assigned to a particular group, and based on that group's user behavior, they are given a recommendation.
- **Finance**: a can help banks assess risk of a loan to an organization or individual. It is used to determine the credit-worthiness of a loan applicant. Furthermore, it can be uses in stock market forecasting, currency exchange rates, trading futures, and money laundering analyses.
- **Healthcare:** making predictions on the risk of heart attacks and prostate cancer. The algorithm works by calculating the most likely gene expressions

.

Classifier Metrics

Using different metrics for performance evaluation:

- Accuracy
- Confusion matrix
- Precision
- Recall

Confusion Matrix

Confusion Matrix is a performance measurement for the machine learning classification problems where the output can be two or more classes. It is a table with combinations of predicted and actual values.

Predicted Values	Actual Values: Positive (1)	Actual Values: Negative (0)
Positive (1)	TP	FP
Negative (0)	FN	TN

Image Source—https://www.roelpeters.be/glossary/what-is-a-confusion-matrix/

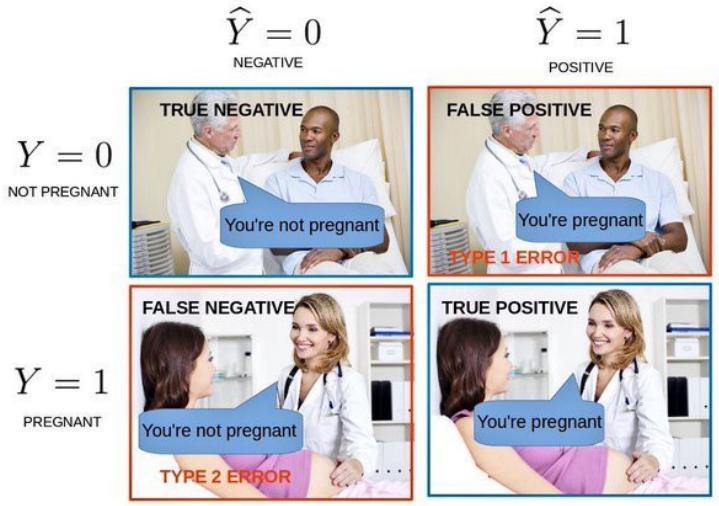

Image Source- https://dzone.com/articles/understanding-the-confusion-matrix

Accuracy

Accuracy simply measures how often the classifier correctly predicts. We can define accuracy as the ratio of the number of correct predictions and the total number of predictions.

Accuracy is useful when the target class is ***well balanced*** but is not a good choice for the unbalanced classes. Imagine the scenario where we had 99 images of the dog and only 1 image of a cat present in our training data. Then our model would always predict the dog, and therefore we got 99% accuracy. In reality, Data is always imbalanced for example Spam email, credit card fraud, and medical diagnosis. Hence, if we want to do a better model evaluation and have a full picture of the model evaluation, other metrics such as recall and precision should also be considered.

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

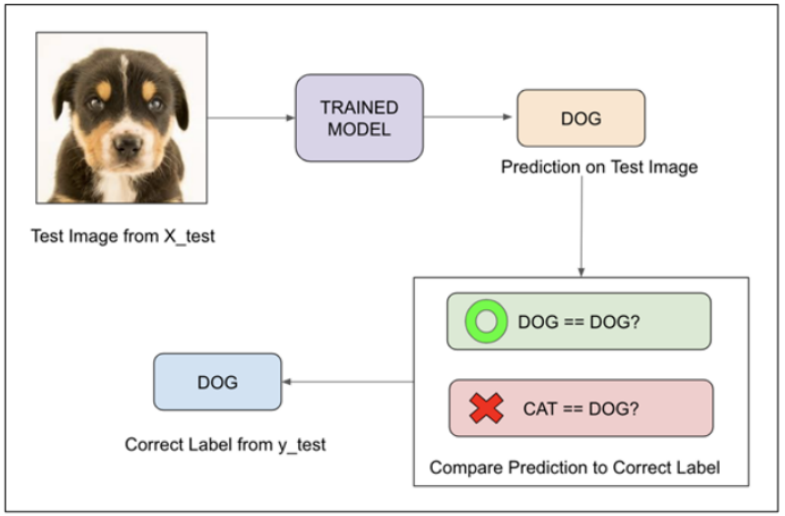

Image Source – https://www.kdnuggets.com/2020/04/performance-evaluation-metrics-classification.html

Precision

Precision explains **how many of the correctly predicted cases actually turned out to be positive**.
Precision is useful in the cases where False Positive is a higher concern than False Negatives.

The importance of *Precision is in music or video recommendation systems, e-commerce websites, etc. where wrong results could lead to customer churn and this could be harmful to the business.*

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

Recall

Recall (Sensitivity)— explains how many of the actual positive cases we were able to predict correctly with our model. It is a useful metric in cases where False Negative is of higher concern than False Positive.

It *is important in medical cases where it doesn't matter whether we raise a false alarm but the actual positive cases should not go undetected!*

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

QUESTIONS?