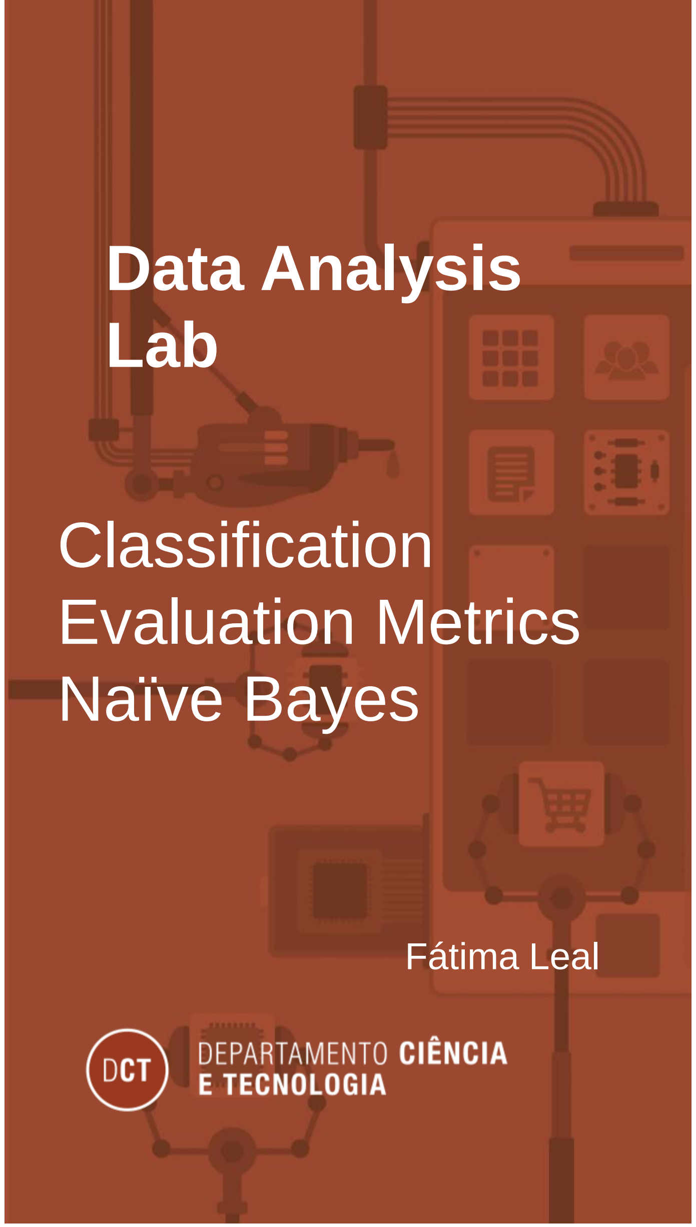


Data Analysis Lab

Classification Evaluation Metrics Naïve Bayes

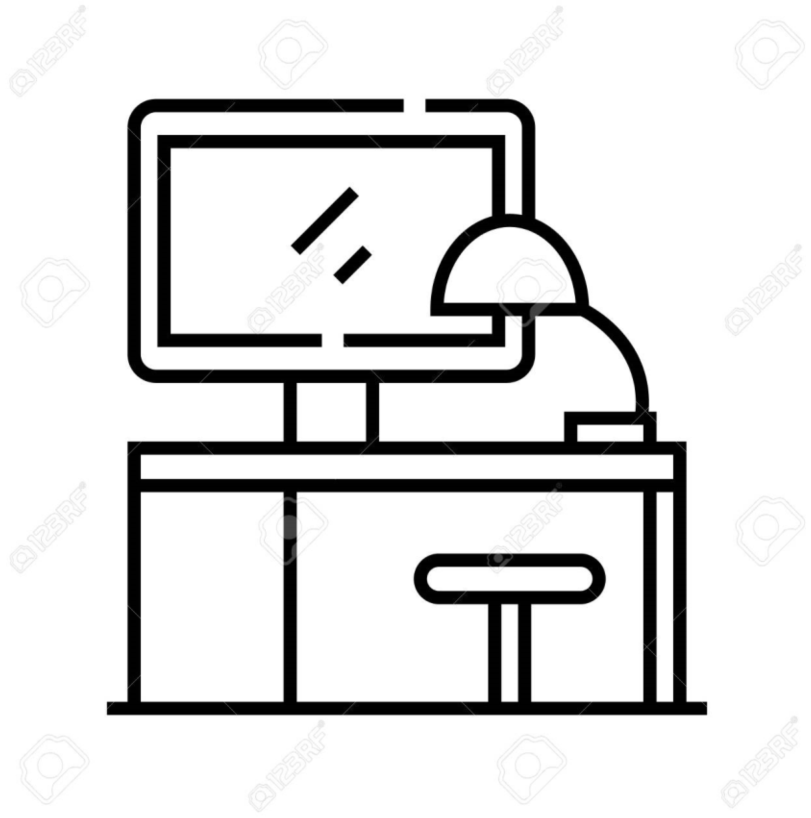
Fátima Leal

DEPARTAMENTO CIÊNCIA E TECNOLOGIA

UNIVERSIDADE PORTUCALENSE

Previous Lesson

- Ridge Regression
- Lasso regression



Outline

- Introduction to Classification
- Evaluation
- Naïve Bayes
- Practical Examples
- Exercises

Introduction to Classification

- **Classification** is a task that requires the **use of machine learning** algorithms that learn how **to assign a class label** to examples from the problem domain.

- Classification is the **process of predicting the class** of given data points.

- **Classes** are sometimes called as **targets/labels or categories**.

- Classification predictive modeling is the task of approximating a mapping function (f) from **input variables** (X) **to discrete output** variables (y).

Evaluation metrics for classification

- Holdout method evaluates the accuracy of classifiers. It splits the data into two sets: tranning and test set.
- The predicted labels are compared to actual labels and accuracy is achieved analysing how many predictions are corrected
- The metrics for classification are: Precision, Recall, and F-measure
- The metrics rely on the number of False Positives (FP), False Negatives (FN), True Positives (TP), and False Positives (FP)
- True Positives, when the model correctly classifies the data point to the class it belongs to
- False Positives, when the model falsely classifies the data point.

Evaluation metrics for classification

- **Precision** is used to calculate the model ability to classify values correctly. It is given by dividing the number of correctly classified data points by the total number of classified data points for that class label.

$$Precision = \frac{TP}{TP + FP}$$

- **Recall** is used to calculate the model ability to predict positive values. But, “How often often does the model predict correct positive values?”. This is calculated by the ration of true positives and the total number of actual positive values.

$$Recall = \frac{TP}{TP + FN}$$

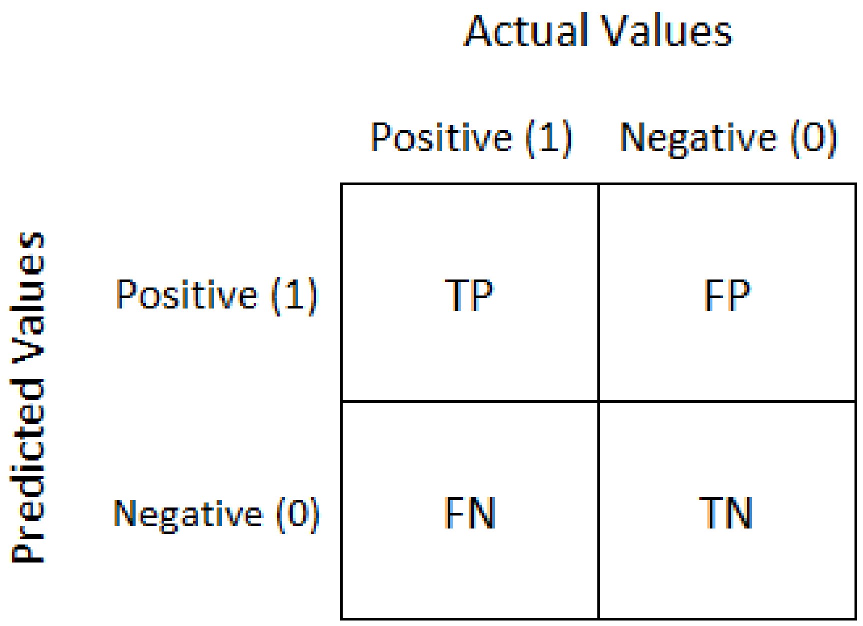
Evaluation metrics for classification

- The **F-measure or F1-score** is a measure of a model's accuracy on a dataset. It is used to evaluate classification systems. The F-measure combines the precision and recall metrics.

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

Confusion matrix

- Supervised learning can be evaluated using an error matrix called confusion matrix
- It allows to visualize the performance of the model



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

Naive Bayes

- Naive Bayes are a family of powerful and easy-to-train classifiers that determine the probability of an outcome given a set of conditions using Bayes' theorem
- Naive Bayes are multipurpose classifiers and it's easy to find their application in many different contexts
- Their performance is particularly good in all those situations where the probability of a class is determined by the probabilities of some causal factors

Naive Bayes

- Let's consider two probabilistic events A and B. We can correlate the marginal probabilities $P(A)$ and $P(B)$ with the conditional probabilities $P(A|B)$ and $P(B|A)$ using the product rule: $P(A \cap B) = P(A|B)P(B)$ and $P(B \cap A) = P(B|A)P(A)$
- Considering that the intersection is commutative, the first members are equal; so we can derive Bayes' theorem: $P(A \cap B) = \frac{P(B|A)P(A)}{P(B)}$

Example

```
from matplotlib import pyplot as plt
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import plot_confusion_matrix, classification_report

X, y = load_iris(return_X_y=True)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5,
random_state=0)
gnb = GaussianNB()
y_pred = gnb.fit(X_train, y_train).predict(X_test)
print("Number of mislabeled points out of a total %d points : %d"
% (X_test.shape[0], (y_test != y_pred).sum()))

print(classification_report(y_test, y_pred))
class_names = load_iris().target_names
plot_confusion_matrix(gnb, X_test, y_test, cmap=plt.cm.Blues,
display_labels=class_names)
plt.show()
```

Exercises

- Using the datasets weatherAUS.csv, Admission_Preditct.csv, and heart.csv
- Do an exploratory analysis of the data set
- Divide the dataset in training and test
- Apply the naive bayes classification model
- Obtain and plot the confusion matrix
- Start to employ the models which you have learnt to your project dataset

References

- Bonaccorso, G. (2020). Mastering Machine Learning Algorithms: Expert techniques for implementing popular machine learning algorithms, fine-tuning your models, and understanding how they work. Packt Publishing Ltd.
- Bonaccorso, G. (2018). Machine Learning Algorithms: Popular algorithms for data science and machine learning. Packt Publishing Ltd.
- Lee, W. M. (2019). Python machine learning. John Wiley & Sons.
- Hauck, T. (2014). scikit-learn Cookbook. Packt Publishing Ltd.
- Montgomery, D. C., Peck, E. A., & Vining, G. G. (2021). Introduction to linear regression analysis. John Wiley & Sons.
- Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression (Vol. 398). John Wiley & Sons.

UNIVERSIDADE
PORTUCALENSE

Do conhecimento à prática.