**33333https://towardsdatascience.com/top-machine-learning-algorithms-for-classification-2197870ff501**

**Top 6 Machine Learning Algorithms for Classification**

How to Build a Machine Learning Model Pipeline in Python

**Supervised vs. Unsupervised vs. Reinforcement Learning**

The easiest way to distinguish a supervised learning and unsupervised learning is to see whether the data is labelled or not.

**Supervised learning** learns a function to make prediction of a defined label based on the input data.

* Classification: identifies which category an object belongs
* Regression: predicts a continous output

It can be either classifying data into a category (classification problem) or forecasting an outcome ([regression algorithms](https://towardsdatascience.com/top-machine-learning-algorithms-for-regression-c67258a2c0ac" \t "_blank)).

**Unsupervised learning** reveals the underlying pattern in the dataset that are not explicitly presented, which can discover the similarity of data points (clustering algorithms) or uncover hidden relationships of variables (association rule algorithms) …

**Reinforcement learning** is where the agents learn to take actions based on its interaction with the environment, with the aim to maximize rewards. It is most similar to the learning process of human, following a trial-and-error approach.

*For a guide to regression algorithms, please see:*

**[Top 4 Regression Algorithms in Machine Learning](https://towardsdatascience.com/top-machine-learning-algorithms-for-regression-c67258a2c0ac" \t "_blank)**

[A Comprehensive Guide to Implementation and Comparison](https://towardsdatascience.com/top-machine-learning-algorithms-for-regression-c67258a2c0ac" \t "_blank)

[towardsdatascience.com](https://towardsdatascience.com/top-machine-learning-algorithms-for-regression-c67258a2c0ac" \t "_blank)

Sometimes there is an ambiguous line between classification algorithms and regression algorithms. Many algorithms can be used for both classification and regression, and classification is just regression model with a threshold applied. When the number is higher than the threshold it is classified as true while lower classified as false.

In this article, we will discuss top 6 machine learning algorithms for classification problems, including: l*ogistic regression, decision tree, random forest, support vector machine, k nearest neighbour and naive bayes*. Check out the code for model pipeline on my [website](https://www.visual-design.net/code-snippets" \t "_blank).

1. **Logistic Regression**

|  |  |
| --- | --- |
| Logistics regression uses sigmoid function above to return the probability of a label. It is widely used when the classification problem is binary — true or false, win or lose, positive or negative ... |  |

The sigmoid function generates a probability output. By comparing the probability with a pre-defined threshold, the object is assigned to a label accordingly. Check out my posts on [logistic regression](https://towardsdatascience.com/simple-logistic-regression-using-python-scikit-learn-86bf984f61f1" \t "_blank) for a detailed walkthrough.

from sklearn.linear\_model import LogisticRegression  
reg = LogisticRegression()  
reg.fit(X\_train, y\_train)  
y\_pred = reg.predict(X\_test)

[**logistic regression common hyperparameters**](https://towardsdatascience.com/tuning-the-hyperparameters-of-your-machine-learning-model-using-gridsearchcv-7fc2bb76ff27)**:** penalty, max\_iter, C, solver

**2. Decision Tree**

|  |  |
| --- | --- |
| Decision tree builds tree branches in a hierarchy approach and each branch can be considered as an if-else statement. The branches develop by partitioning the dataset into subsets based on most important features. Final classification happens at the leaves of the decision tree. | Text  Description automatically generated |

from sklearn.tree import DecisionTreeClassifier  
dtc = DecisionTreeClassifier()  
dtc.fit(X\_train, y\_train)  
y\_pred = dtc.predict(X\_test)

[**decision tree common hyperparameters**](https://towardsdatascience.com/how-to-tune-a-decision-tree-f03721801680): criterion, max\_depth, min\_samples\_split, min\_samples\_leaf; max\_features

**3. Random Forest**

Diagram

Description automatically generated

As the name suggest, random forest is a collection of decision trees. It is a common type of ensemble methods which aggregate results from multiple predictors. Random forest additionally utilizes bagging technique that allows each tree trained on a random sampling of original dataset and takes the majority vote from trees. Compared to decision tree, it has better generalization but less interpretable, because of more layers added to the model.

from sklearn.ensemble import RandomForestClassifier  
rfc = RandomForestClassifier()  
rfc.fit(X\_train, y\_train)  
y\_pred = rfc.predict(X\_test)

[**random forest common hyperparameters**](https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74): n\_estimators, max\_features, max\_depth, min\_samples\_split, min\_samples\_leaf, boostrap

**4. Support Vector Machine (SVM)**

A picture containing text, antenna

Description automatically generated

Support vector machine finds the best way to classify the data based on the position in relation to a border between positive class and negative class. This border is known as the hyperplane which maximize the distance between data points from different classes. Similar to decision tree and random forest, support vector machine can be used in both classification and regression, SVC (support vector classifier) is for classification problem.

from sklearn.svm import SVC  
svc = SVC()  
svc.fit(X\_train, y\_train)  
y\_pred = svc.predict(X\_test)

[**support vector machine common hyperparameters:**](https://www.vebuso.com/2020/03/svm-hyperparameter-tuning-using-gridsearchcv/)c, kernel, gamma

**5. K-Nearest Neighbour (KNN)**

Chart, scatter chart

Description automatically generated

You can think of k nearest neighbour algorithm as representing each data point in a n dimensional space — which is defined by n features. And it calculates the distance between one point to another, then assign the label of unobserved data based on the labels of nearest observed data points. KNN can also be used for building recommendation system, check out my article on “[Collaborative Filtering for Movie Recommendation](https://towardsdatascience.com/a-beginner-friendly-guide-to-recommender-system-3f5fa2a57c02" \t "_blank)” if you are interested in this topic.

from sklearn.neighbors import KNeighborsClassifier  
knn = KNeighborsClassifier()  
knn.fit(X\_train, y\_train)  
y\_pred = knn.predict(X\_test)

[**KNN common hyperparameters**](https://medium.datadriveninvestor.com/k-nearest-neighbors-in-python-hyperparameters-tuning-716734bc557f)**:**n\_neighbors, weights, leaf\_size, p

**6. Naive Bayes**

Diagram

Description automatically generated

Naive Bayes is based on [Bayes’ Theorem](https://machinelearningmastery.com/bayes-theorem-for-machine-learning/) — an approach to calculate conditional probability based on prior knowledge, and the naive assumption that each feature is independent to each other. The biggest advantage of Naive Bayes is that, while most machine learning algorithms rely on large amount of training data, it performs relatively well even when the training data size is small. Gaussian Naive Bayes is a type of Naive Bayes classifier that follows the normal distribution.

from sklearn.naive\_bayes import GaussianNB  
gnb = KNeighborsClassifier()  
gnb.fit(X\_train, y\_train)  
y\_pred = gnb.predict(X\_test)

[**gaussian naive bayes common hyperparameters**](https://www.analyticsvidhya.com/blog/2021/01/gaussian-naive-bayes-with-hyperpameter-tuning/): priors, var\_smoothing

If you want to know more other machine learning algorithms, check out my list:

Destin Gong

[Destin Gong](https://destingong.medium.com/?source=post_page-----2197870ff501-----------------------------------)

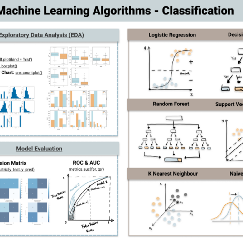
**Practical Guides to Machine Learning**

[View list](https://destingong.medium.com/list/practical-guides-to-machine-learning-a877c2a39884?source=post_page-----2197870ff501-----------------------------------)

7 stories

Graphical user interface, application

Description automatically generated



Graphical user interface, diagram

Description automatically generated

**Build a Classification Model Pipeline**

**1. Loading Dataset and Data Overview**

I chose the popular dataset [Heart Disease UCI](https://www.kaggle.com/ronitf/heart-disease-uci) on Kaggle for predicting the presence of heart disease based on several health related factors.

Table

Description automatically generated

Use df.info()to have a summarized view of dataset, including **data type, missing data and number of records.**

Table

Description automatically generated

**2. Exploratory Data Analysis (EDA)**

**Histogram, grouped bar chart and box plot** are suitable EDA techniques for classification machine learning algorithms. If you’d like a more comprehensive guide to EDA, please see my post “[Semi-Automated Exploratory Data Analysis Process in Python](https://www.visual-design.net/post/semi-automated-exploratory-data-analysis-process-in-python)”

**Univariate Analysis**

Waterfall chart

Description automatically generated

Histogram is used for all features, because all features have been encoded into numeric values in the dataset. This saves us the time for categorical encoding that usually happens during the [feature engineering](https://www.visual-design.net/post/data-transformation-and-feature-engineering-in-python" \t "_blank) stage.

**Categorical Features vs. Target — Grouped Bar Chart**

Chart, bar chart

Description automatically generated

To show how categorical value weigh in determining the target value, grouped bar chart is a straightforward representation.

For example, sex = 1 and sex = 0 have distinctly distribution of target value, which indicates it is likely to contribute more to the prediction of target.

Contrarily, if the target distribution is the same regardless of the categorical features, then very likely they are not correlated.

**Numerical Features vs. Target — Box Plot**

Chart, box and whisker chart

Description automatically generated

Box plot shows how the values of numerical features varies across target groups.

For example, we can tell that “oldpeak” have distinct difference when target is 0 vs. target is 1, suggesting that it is an important predictor. However, ‘trestbps’ and ‘chol’ appear to be less outstanding, as the box plot distribution is similar between target groups.

**3. Split Dataset into Training and Testing Set**

from sklearn.model\_selection import train\_test\_split  
from sklearn import preprocessing

X = df.drop(['target'], axis=1)  
y = df["target"]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state=42)

**4. Machine Learning Model Pipeline**

Text

Description automatically generated

**5. Model Evaluation**

Graphical user interface, text, application, email

Description automatically generated

Below is an abstraction explanation of commonly used evaluation methods for classification models — **accuracy, ROC & AUC and confusion matrix**. Each of the following metrics is worth diving deeper, feel free to visit my article on [logistic regression](https://towardsdatascience.com/simple-logistic-regression-using-python-scikit-learn-86bf984f61f1" \t "_blank) for a more detailed illustration.

**1. Accuracy**

Accuracy is the most straightforward indicator of the model performance. It measure the percentage of accurate predictions: *accuracy = (true positive + true negative) / (true positive + false positive + false negative + false positive)*

**2. ROC & AUC**

|  |  |
| --- | --- |
| ROC is the plot of **true positive rate against false positive rate** at various classification threshold. AUC is the area under the ROC curve, and higher AUC indicates better model performance. |  |

**3. Confusion matrix**

Confusion matrix indicates the actual values vs. predicted values and summarize the **true negative, false positive, false negative and true positive values** in a matrix format.

Table

Description automatically generated

Then we can use seaborn to visualize the confusion matrix in a heatmap.

Table

Description automatically generated with low confidence

Graphical user interface, application

Description automatically generated

Based on three evaluations methods above, random forests and naive bayes have the best performance, whereas KNN is not doing well.

However, this doesn’t mean that random forests and naive bayes are superior algorithms. We can only say that they are more suitable for this dataset where the size is relatively smaller and data is not at the same scale.

Each algorithm has its own preference and require different data processing and feature engineering techniques, for example KNN is sensitive to features at difference scale and multicollinearity affects the result of logistic regression. Understanding the characteristics of each allows us to balance the trade-off and select the appropriate model according to the dataset.

Thanks for reaching so far, if you’d like to read more articles from Medium and also support my work, I really appreciate you signing up Medium Membership using this affiliate [link](https://destingong.medium.com/membership).