AUTO OPTIMIZER

RELEASE 0.8.9 BETA

Optimization means to find the best value of some function or model. That can be the maximum or the minimum according to some metric. In machine learning, we can construct a function that is a model with a predefined set of training and test data. The input to this function are the model's hyperparameters and the output is the evaluation score. So we are looking for what hyperparameters can produce the best score. Or, the function can be the model itself, and we are looking for what weight parameters to produce the lowest error rate for that given dataset.

"Optimization algorithms lie at the heart of machine learning and artificial intelligence."

GenesisCube

"Science for a comfortable life"

GenesisCube Development Team

Updated March 8, 2023

Contents

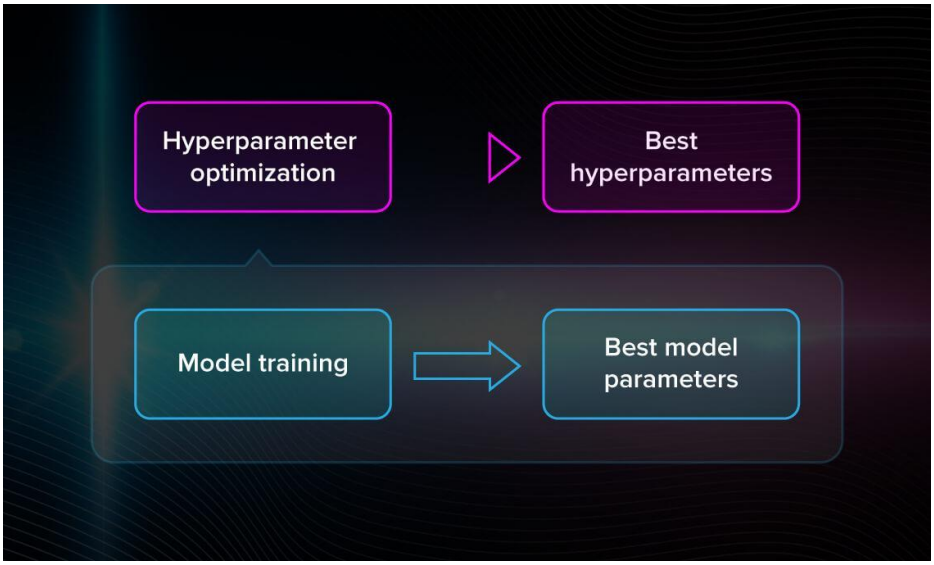
What is Autooptimizer? 3
User's Guide 4
Installation Guide 6
What is new in Autooptimizer 0.8.9 7
Package Tutorials 8
Optimizing Machine Learning Models 8
Clustering: 8
Supervised Learning: 9
Ensemble Learning: 10
Regression Metrics 11
Outlier remover 11
Interquartile Range 11
Z-Score Method 12
Standard Deviation Method 12
Local Outlier Factor 13
Contact and Contributing 14
Autooptimizer License 15

What is Autooptimizer?

The ability to apply function optimization allows us to go to a new level in various stages of machine learning. We may find better models by hyperparameter tuning. Machine learning is doing a lot of optimization behind the scenes. When new algorithm invented or new technique proposed, it is inevitable to explain them in terms of optimization. Hence we also need to understand them from optimization perspectives.

After developing an appropriate model for a machine learning problem, the next step is to choose an optimization technique. we will pair models with efficient optimization algorithms, from stochastic gradient descent to cone programming.

We will cover a breadth of tools, from linear to classification and stochastic/deterministic gradient descent, in the context of practical problems drawn from emerging applications in learning, computer vision, time series analysis, and imaging. With Coding and mathematical exercises we reinforce this package to standard software for optimization.



Optimization algorithms allow us to use machine learning to its potential.

AutoOptimizer package provides tools to automatically optimize machine learning model for a dataset. It allow non-experts and engineers to discover a good predictive model for their machine learning algorithm task quickly, with very little intervention other than providing a dataset. AutoOptimizer uses Exhaustive Search Mechanism with Hyperparameter Tuning for optimizing machine learning models. It also provides evaluation metrics for regression models and ability to delete outliers with several methods.

Exhaustive Search is the process of looking for the most optimal hyperparameters by checking whether each candidate is a good match. In machine learning, we do the same thing but the number of options is quite large and try out all the possible options.

User's Guide

Autooptimizer is the fundamental package for machine learning in Python. It is a Python library that serve optimizer tools for algorithms, regression metrics, and an extra tools for fast operations on outliers.
At the core of the Autooptimizer package, is Exhaustive Search Mechanism (ESM). ESM optimizes the machine learning models with many operations being performed in compiled code for performance.

Cases of use:

- Optimize machine learning models for these algorithms.
    - Clustering
        - DBSCAN
        - K-Means
        - MeanShift
        - Mini Batch K-Means
    - Supervised Learning
        - K-Nearest Neighbor
        - Decision Tree
        - Linear Models
        - Support Vector Machine
    - Ensemble Learning
        - Boosting
        - Bagging
        - Decision Tree
        - Gradient

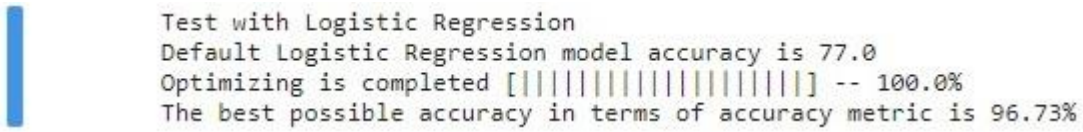
- Evaluation metrics for regression models.
- Remove outliers from dataset with several methods.
    - Interquartile Range
    - Z-Score Method
    - Standard Deviation Method
    - Local Outlier Factor

Prerequisites:

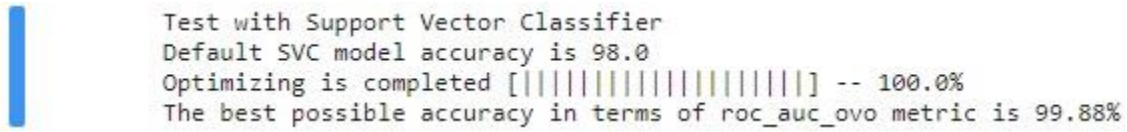
- Python
    - Requires >=3
- Sklearn
- Numpy
- Pandas
- Operating System
    - OS Independent

Experiments and Results: In lab we ran as many experiments with different datasets to make sure that Package performs as expected and optimized. Here are some results to compare.
Note, we ran essential preprocessing steps on datasets.

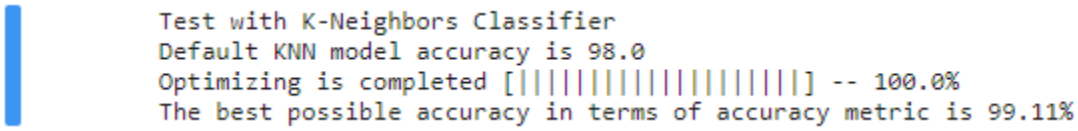
Test “Logistic Regression” algorithm.
Increased accuracy from 77% to 96.73%



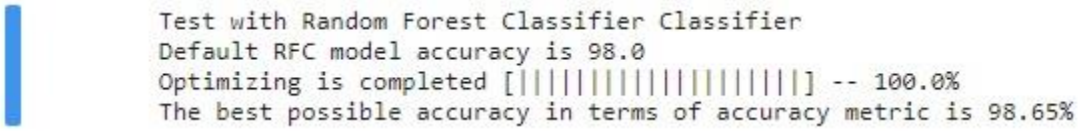
Test “Support Vector Classifier” algorithm.
Increased accuracy from 98% to 99.88%



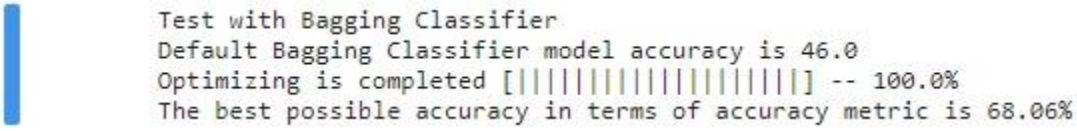
Test “K-Neighbors Classifier” algorithm.
Increased accuracy from 98% to 99.11%



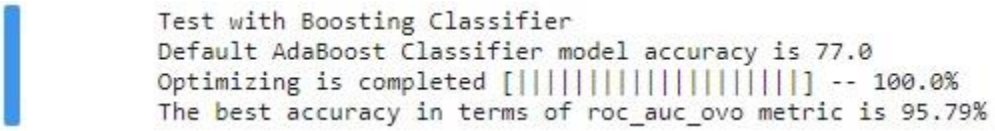
Test “Random Forest Classifier” algorithm.
Increased accuracy from 98% to 98.65%



Test “Bagging Classifier” algorithm.
Increased accuracy from 46% to 68.06%



Test “Ada Boost Classifier” algorithm.
Increased accuracy from 77% to 95.79%



Installation Guide

Autooptimizer is available as package for Microsoft Windows, Mac OS and all Linux distributions.

```
Installing official release:
```

```
pip install autooptimizer
```

```
It is recommended to upgrade pip to latest version.
```

```
python -m pip install --upgrade pip
```

```
Installing in jupyter notebook:
```

```
open anaconda prompt (It is recommended to open as administrator)
pip install autooptimizer
```

```
Upgrade package:
```

```
pip install --upgrade autooptimizer
```

```
Install specific version:
```

```
pip install autooptimizer==0.8.9
```

Pip vs Pip3 What's the Difference?

PIP	PIP3
Pip is a soft link for a particular installer.	Pip3 is an updated version of pip which is used basically for python >3.
The system will use one of your python versions depending on what exactly is first in the system path variable.	When you run pip3, you can be sure that the module will be installed in python 3.
if i use pip, the package will be installed for the python version that i installed last on my computer. That would be 3.9 for my case. So using pip and using pip3.9 will produce the same result for me.	If i use pip3, the package will only be installed for python3. It won't be available for 2.7 or 2.9.

What is new in Autooptimizer 0.8.9

List of all of the issues and requests since the last release. From now on, these upgrades will be considered:

Upgrades and Improvements:

- Improve clustering algorithms optimizer.
    a. MeanShift
    b. K-Means
    c. DBSCAN

New Functions:

- Add another clustering algorithm optimizer.
    a. Mini Batch K-Means
- Add loading progress bar for clustering algorithms.

Previous versions did not consider progress timing, It is now possible to see the timing via progress bar, and you are able to optimize clustering models much better.

Optimizing Machine Learning Models

Tuning machine learning hyperparameters is a tedious yet crucial task, as the performance of an algorithm can be highly dependent on the choice of hyperparameters. Grid and random search are hands-off, but require long run times because they waste time evaluating unpromising areas of the search space. Increasingly, hyperparameter auto tuning is done by automated methods that aim to find optimal hyperparameters in less time using an informed search with no manual effort necessary beyond the initial set-up.

Note, to prevent being mistaken autooptimizer with sklearn classes and functions, in autooptimizer we used lowercase format. In this section, you learn how to apply optimizer to your program.

Clustering:

Clustering machine learning algorithms are grouping similar elements in such a way that the distance between each element of the cluster are closer to each other than to any other cluster.

- DBSCAN
- K-Means
- MeanShift
- Mini Batch K-Means

- from autooptimizer.cluster import dbscan
  - dbscan(x)
- from autooptimizer.cluster import kmeans
  - kmeans(x)
- from autooptimizer.cluster import meanshift
  - meanshift(x)
- from autooptimizer.cluster import minibatchkmeans
  - minibatchkmeans(x)

'x' should be your independent variable or feature's values. The output of the program is **prepared optimized model** to use in your program for prediction, with best possible accuracy.

Example:

```
from autooptimizer.cluster import dbscan
optimized_model = dbscan(x)
optimized_model.fit(x)
```

The default metric is “silhouette” for clustering algorithms. And can be changed with the scoring parameter. (x, scoring).

Supervised Learning:

Supervised learning is the types of machine learning in which machines are trained using well labelled training data, and on basis of that data, machines predict the output.

- K-Nearest Neighbor
  - KNeighborsClassifier
  - KNeighborsRegressor
- Decision Tree
  - DecisionTreeClassifier
  - DecisionTreeRegressor
- Linear Models
  - LogisticRegression
  - LinearRegression
- Support Vector Machine
  - SupportVectorClassifier
  - SupportVectorRegressor

- from autooptimizer.neighbors import kneighborsclassifier
  - kneighborsclassifier(x, y)
- from autooptimizer.neighbors import kneighborsregressor
  - kneighborsregressor(x, y)
- from autooptimizer.tree import decisiontreeclassifier
  - decisiontreeclassifier(x, y)
- from autooptimizer.tree import decisiontreeregressor
  - decisiontreeregressor(x, y)
- from autooptimizer.linear_model import logisticregression
  - logisticregression(x, y)
- from autooptimizer.linear_model import linearregression
  - linearregression(x, y)
- from autooptimizer.svm import svc
  - svc(x, y)
- from autooptimizer.svm import svr
  - svr(x, y)

'x' should be your independent variable or feature's values and 'y' is target variable or dependent variable. The output of the program is **prepared optimized model** to use in your program for prediction, with best possible accuracy.

Example:

```
from autooptimizer.neighbors import kneighborsclassifier

optimized_model = kneighborsclassifier(x, y)

optimized_model.fit(x, y)
```

The default metric is “roc_auc_ovo” for classification algorithms and “neg_mean_squared_error” for regression algorithms. The default metric can be changed with the scoring parameter. (x, y, scoring).

Ensemble Learning:

In machine learning, ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone.

- Boosting
  - AdaBoostClassifier
  - AdaBoostRegressor
- Bagging
  - BaggingClassifier
  - BaggingRegressor
- Decision Tree
  - ExtraTreesClassifier
  - RandomForestClassifier
  - RandomForestRegressor
- Gradient
  - GradientBoostingClassifier
  - GradientBoostingRegressor

- from autooptimizer.ensemble import adaboostclassifier
  - adaboostclassifier(x, y)
- from autooptimizer.ensemble import adaboostregressor
  - adaboostregressor(x, y)
- from autooptimizer.ensemble import baggingclassifier
  - baggingclassifier(x, y)
- from autooptimizer.ensemble import baggingregressor
  - baggingregressor(x, y)
- from autooptimizer.ensemble import extratreesclassifier
  - extratreesclassifier(x, y)
- from autooptimizer.ensemble import randomforestclassifier
  - randomforestclassifier(x, y)
- from autooptimizer.ensemble import randomforestregressor
  - randomforestregressor(x, y)
- from autooptimizer.ensemble import gradientboostingclassifier
  - gradientboostingclassifier(x, y)
- from autooptimizer.ensemble import gradientboostingregressor
  - gradientboostingregressor(x, y)

'x' should be your independent variable or feature's values and 'y' is target variable or dependent variable. The output of the program is **prepared optimized model** to use in your program for prediction, with best possible accuracy.

Example:

```
from autooptimizer.ensemble import adaboostclassifier

optimized_model = adaboostclassifier(x, y)

optimized_model.fit(x, y)
```

The default metric is “roc_auc_ovo” for classification algorithms and “neg_mean_squared_error” for regression algorithms. The default metric can be changed with the scoring parameter. (x, y, scoring).

Regression Metrics

Regression refers to predictive modeling problems that involve predicting a numeric value. Metrics for regression involve calculating an error score to summarize the predictive skill of a model.
Metrics tell you accurate measurements about how the process is functioning and provide base for you to suggest improvements.

- from autooptimizer.metrics import root_mean_squared_error
- from autooptimizer.metrics import root_mean_squared_log_error
- from autooptimizer.metrics import root_mean_squared_precentage_error
- from autooptimizer.metrics import symmetric_mean_absolute_precentage_error
- from autooptimizer.metrics import mean_bias_error
- from autooptimizer.metrics import relative_squared_error
- from autooptimizer.metrics import root_relative_squared_error
- from autooptimizer.metrics import relative_absolute_error
- from autooptimizer.metrics import median_absolute_percentage_error
- from autooptimizer.metrics import mean_absolute_percentage_error

Outlier remover

Outliers are unusual values in your dataset, and they can distort statistical analyses and increase the variability in your data set, which decreases statistical power. Consequently, excluding outliers can cause your results to become statistically significant. It uses **Interquartile Range, Z-Score**, **Standard Deviation** and **Local Outlier Factor (LOF)** methods to remove outliers from array or data set.

Interquartile Range

In descriptive statistics, the interquartile range (IQR) is a measure of statistical dispersion, which is the spread of the data. The IQR may also be called the mid spread, middle 50%, fourth spread, or H-spread. It is defined as the difference between the 75th and 25th percentiles of the data. To calculate the IQR, the data set is divided into quartiles, or four rank-ordered even parts via linear interpolation. These quartiles are denoted by Q1 (also called the lower quartile), Q2 (the median), and Q3 (also called the upper quartile). The lower quartile corresponds with the 25th percentile and the upper quartile corresponds with the 75th percentile, so $IQR = Q3 - Q1$.

The IQR is an example of a trimmed estimator, defined as the 25% trimmed range, which enhances the accuracy of dataset statistics by dropping lower contribution, outlying points. It is also used as a robust measure of scale It can be clearly visualized by the box on a Box plot.

```
from autooptimizer.outlier import interquartile_removal

interquartile_removal(dataset, axis=0, output='df')
```

Parameters:

- **dataset**: The desired dataset or array for cleaning out of outliers.
- **axis**: value of 0 applies the cleanup to rows and 1 to columns. by default, it is 0.
- **output**: The output of the program can be received in two forms, array '**ar**' or dataframe '**df**'. by default, it is 'df'.

Z-Score Method

The Z-score is a way of describing a data point in terms of its relationship to the mean and standard deviation of a group of points. Taking a Z-score is simply mapping the data onto a distribution whose mean is defined as 0 and whose standard deviation is defined as 1.

The goal of taking Z-scores is to remove the effects of the location and scale of the data, allowing different datasets to be compared directly. The intuition behind the Z-score method of outlier detection is that, once we've centred and rescaled the data, anything that is too far from zero (the threshold is usually a Z-score of 3 or -3) should be considered an outlier.

```
from autooptimizer.outlier import zscore_removal

zscore_removal(dataset, axis=0, output='df')
```

Parameters:

- **dataset**: The desired dataset or array for cleaning out of outliers.
- **axis**: value of 0 applies the cleanup to rows and 1 to columns. by default, it is 0.
- **output**: The output of the program can be received in two forms, array '**ar**' or dataframe '**df**'. by default, it is 'df'.

Standard Deviation Method

Standard deviation of the residuals are calculated and compared. If a value is a certain number of standard deviations away from the mean, that data point is identified as an outlier. The specified number of standard deviations is called the threshold. The default value is 3.

```
from autooptimizer.outlier import std_removal

std_removal(dataset, axis=0, output='df', threshold=3)
```

Parameters:

- **dataset**: The desired dataset or array for cleaning out of outliers.
- **axis**: value of 0 applies the cleanup to rows and 1 to columns. by default, it is 0.
- **output**: The output of the program can be received in two forms, array '**ar**' or dataframe '**df**'. by default, it is 'df'.
- **threshold**: Number of standard deviations, which is used to create the threshold range. By default, it is 3.

Local Outlier Factor

The Local Outlier Factor (LOF) algorithm is an unsupervised anomaly detection method which computes the local density deviation of a given data point with respect to its neighbors. It considers as outliers the samples that have a substantially lower density than their neighbors.

The number of neighbors considered (parameter threshold) is typically set 1) greater than the minimum number of samples a cluster has to contain, so that other samples can be local outliers relative to this cluster, and 2) smaller than the maximum number of close by samples that can potentially be local outliers. In practice, such information is generally not available, and taking n_neighbors=10 appears to work well in general.

Note, when LOF is used for outlier detection it has no predict, decision_function and score_samples methods.

```
from autooptimizer.outlier import lof_removal

lof_removal(dataset, output='df', threshold=10)
```

Parameters:

- **dataset**: The desired dataset or array for cleaning out of outliers.
- **output**: The output of the program can be received in two forms, array '**ar**' or dataframe '**df**'. by default, it is 'df'.
- **threshold**: The number of neighbors considered. By default, it is 10.

Contact and Contributing

Everyone is welcome to contribute. If you find an error or bug in the code or documentation, do not hesitate to submit a ticket or report error. You are also welcome to request feature.

If you are reporting a bug or error, please include the following:

1. A summary of the bug.
2. A self-contained code snippet to reproduce the bug.
3. Autooptimizer version.
4. Mail to info@GenesisCube.ir

Let us know how we can improve the package to serve the community. Thanks for your participation in the project.

AutoOptimizer Package WebSite

Info@GenesisCube.ir

The project is hosted on https://github.com/mrb987/autooptimizer

Autooptimizer License

MIT License