Lazy Predict Documentation

Release 0.2.9

Shankar Rao Pandala

Feb 17, 2021

Contents:

CHAPTER 1

Lazy Predict

Lazy Predict helps build a lot of basic models without much code and helps understand which models works better without any parameter tuning.

- Free software: MIT license
- Documentation: https://lazypredict.readthedocs.io.

CHAPTER 2

Installation

To install Lazy Predict:

```
pip install lazypredict
```

CHAPTER 3

Usage

To use Lazy Predict in a project:

```
import lazypredict
```

CHAPTER 4

Classification

Example

```
from lazypredict.Supervised import LazyClassifier
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split

data = load_breast_cancer()
X = data.data
y= data.target

X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=.5,random_state␣
↪=123)

clf = LazyClassifier(verbose=0,ignore_warnings=True, custom_metric=None)
models,predictions = clf.fit(X_train, X_test, y_train, y_test)

print(models)


| Model                          |   Accuracy |   Balanced Accuracy |   ROC AUC |  ␣
↪F1 Score |   Time Taken |
|:-------------------------------|-----------:|--------------------:|----------:|-----
↪------:|-------------:|
| LinearSVC                      |   0.989474 |            0.987544 |  0.987544 |   0.
↪989462 |    0.0150008 |
| SGDClassifier                  |   0.989474 |            0.987544 |  0.987544 |   0.
↪989462 |    0.0109992 |
| MLPClassifier                  |   0.985965 |            0.986904 |  0.986904 |   0.
↪985994 |    0.426     |
| Perceptron                     |   0.985965 |            0.984797 |  0.984797 |   0.
↪985965 |    0.0120046 |
| LogisticRegression             |   0.985965 |            0.98269  |  0.98269  |   0.
↪985934 |    0.0200036 |
| LogisticRegressionCV           |   0.985965 |            0.98269  |  0.98269  |   0.
↪985934 |    0.262997  |
```

(continues on next page)

(continued from previous page)

```
| SVC                            |   0.982456 |           0.979942 |  0.979942 |   0.
↪982437 |    0.0140011 |
| CalibratedClassifierCV         |   0.982456 |           0.975728 |  0.975728 |   0.
↪982357 |    0.0350015 |
| PassiveAggressiveClassifier    |   0.975439 |           0.974448 |  0.974448 |   0.
↪975464 |    0.0130005 |
| LabelPropagation               |   0.975439 |           0.974448 |  0.974448 |   0.
↪975464 |    0.0429988 |
| LabelSpreading                 |   0.975439 |           0.974448 |  0.974448 |   0.
↪975464 |    0.0310006 |
| RandomForestClassifier         |   0.97193  |           0.969594 |  0.969594 |   0.
↪97193  |    0.033     |
| GradientBoostingClassifier     |   0.97193  |           0.967486 |  0.967486 |   0.
↪971869 |    0.166998  |
| QuadraticDiscriminantAnalysis  |   0.964912 |           0.966206 |  0.966206 |   0.
↪965052 |    0.0119994 |
| HistGradientBoostingClassifier |   0.968421 |           0.964739 |  0.964739 |   0.
↪968387 |    0.682003  |
| RidgeClassifierCV              |   0.97193  |           0.963272 |  0.963272 |   0.
↪971736 |    0.0130029 |
| RidgeClassifier                |   0.968421 |           0.960525 |  0.960525 |   0.
↪968242 |    0.0119977 |
| AdaBoostClassifier             |   0.961404 |           0.959245 |  0.959245 |   0.
↪961444 |    0.204998  |
| ExtraTreesClassifier           |   0.961404 |           0.957138 |  0.957138 |   0.
↪961362 |    0.0270066 |
| KNeighborsClassifier           |   0.961404 |           0.95503  |  0.95503  |   0.
↪961276 |    0.0560005 |
| BaggingClassifier              |   0.947368 |           0.954577 |  0.954577 |   0.
↪947882 |    0.0559971 |
| BernoulliNB                    |   0.950877 |           0.951003 |  0.951003 |   0.
↪951072 |    0.0169988 |
| LinearDiscriminantAnalysis     |   0.961404 |           0.950816 |  0.950816 |   0.
↪961089 |    0.0199995 |
| GaussianNB                     |   0.954386 |           0.949536 |  0.949536 |   0.
↪954337 |    0.0139935 |
| NuSVC                          |   0.954386 |           0.943215 |  0.943215 |   0.
↪954014 |    0.019989  |
| DecisionTreeClassifier         |   0.936842 |           0.933693 |  0.933693 |   0.
↪936971 |    0.0170023 |
| NearestCentroid                |   0.947368 |           0.933506 |  0.933506 |   0.
↪946801 |    0.0160074 |
| ExtraTreeClassifier            |   0.922807 |           0.912168 |  0.912168 |   0.
↪922462 |    0.0109999 |
| CheckingClassifier             |   0.361404 |           0.5      |  0.5      |   0.
↪191879 |    0.0170043 |
| DummyClassifier                |   0.512281 |           0.489598 |  0.489598 |   0.
↪518924 |    0.0119965 |
```

CHAPTER 5

Regression

Example

```
from lazypredict.Supervised import LazyRegressor
from sklearn import datasets
from sklearn.utils import shuffle
import numpy as np

boston = datasets.load_boston()
X, y = shuffle(boston.data, boston.target, random_state=13)
X = X.astype(np.float32)

offset = int(X.shape[0] * 0.9)

X_train, y_train = X[:offset], y[:offset]
X_test, y_test = X[offset:], y[offset:]

reg = LazyRegressor(verbose=0, ignore_warnings=False, custom_metric=None)
models, predictions = reg.fit(X_train, X_test, y_train, y_test)

print(models)


| Model                         | Adjusted R-Squared | R-Squared |  RMSE | Time Taken␣
↪|
|:------------------------------|-------------------:|----------:|------:|-----------
↪:|
| SVR                           |               0.83 |      0.88 |  2.62 |       0.01␣
↪|
| BaggingRegressor              |               0.83 |      0.88 |  2.63 |       0.03␣
↪|
| NuSVR                         |               0.82 |      0.86 |  2.76 |       0.03␣
↪|
| RandomForestRegressor         |               0.81 |      0.86 |  2.78 |       0.21␣
↪|
```

(continues on next page)

(continued from previous page)

```
| XGBRegressor                  |                0.81 |      0.86 |  2.79 |        0.06␣
↪|
| GradientBoostingRegressor     |                0.81 |      0.86 |  2.84 |        0.11␣
↪|
| ExtraTreesRegressor           |                0.79 |      0.84 |  2.98 |        0.12␣
↪|
| AdaBoostRegressor             |                0.78 |      0.83 |  3.04 |        0.07␣
↪|
| HistGradientBoostingRegressor |                0.77 |      0.83 |  3.06 |        0.17␣
↪|
| PoissonRegressor              |                0.77 |      0.83 |  3.11 |        0.01␣
↪|
| LGBMRegressor                 |                0.77 |      0.83 |  3.11 |        0.07␣
↪|
| KNeighborsRegressor           |                0.77 |      0.83 |  3.12 |        0.01␣
↪|
| DecisionTreeRegressor         |                0.65 |      0.74 |  3.79 |        0.01␣
↪|
| MLPRegressor                  |                0.65 |      0.74 |  3.80 |        1.63␣
↪|
| HuberRegressor                |                0.64 |      0.74 |  3.84 |        0.01␣
↪|
| GammaRegressor                |                0.64 |      0.73 |  3.88 |        0.01␣
↪|
| LinearSVR                     |                0.62 |      0.72 |  3.96 |        0.01␣
↪|
| RidgeCV                       |                0.62 |      0.72 |  3.97 |        0.01␣
↪|
| BayesianRidge                 |                0.62 |      0.72 |  3.97 |        0.01␣
↪|
| Ridge                         |                0.62 |      0.72 |  3.97 |        0.01␣
↪|
| TransformedTargetRegressor    |                0.62 |      0.72 |  3.97 |        0.01␣
↪|
| LinearRegression              |                0.62 |      0.72 |  3.97 |        0.01␣
↪|
| ElasticNetCV                  |                0.62 |      0.72 |  3.98 |        0.04␣
↪|
| LassoCV                       |                0.62 |      0.72 |  3.98 |        0.06␣
↪|
| LassoLarsIC                   |                0.62 |      0.72 |  3.98 |        0.01␣
↪|
| LassoLarsCV                   |                0.62 |      0.72 |  3.98 |        0.02␣
↪|
| Lars                          |                0.61 |      0.72 |  3.99 |        0.01␣
↪|
| LarsCV                        |                0.61 |      0.71 |  4.02 |        0.04␣
↪|
| SGDRegressor                  |                0.60 |      0.70 |  4.07 |        0.01␣
↪|
| TweedieRegressor              |                0.59 |      0.70 |  4.12 |        0.01␣
↪|
| GeneralizedLinearRegressor    |                0.59 |      0.70 |  4.12 |        0.01␣
↪|
| ElasticNet                    |                0.58 |      0.69 |  4.16 |        0.01␣
↪|
| Lasso                         |                0.54 |      0.66 |  4.35 |        0.02␣
↪|
```

(continues on next page)

(continued from previous page)

```
| RANSACRegressor                |                0.53 |       0.65 |  4.41 |        0.04␣
↪|
| OrthogonalMatchingPursuitCV    |                0.45 |       0.59 |  4.78 |        0.02␣
↪|
| PassiveAggressiveRegressor     |                0.37 |       0.54 |  5.09 |        0.01␣
↪|
| GaussianProcessRegressor       |                0.23 |       0.43 |  5.65 |        0.03␣
↪|
| OrthogonalMatchingPursuit      |                0.16 |       0.38 |  5.89 |        0.01␣
↪|
| ExtraTreeRegressor             |                0.08 |       0.32 |  6.17 |        0.01␣
↪|
| DummyRegressor                 |               -0.38 |      -0.02 |  7.56 |        0.01␣
↪|
| LassoLars                      |               -0.38 |      -0.02 |  7.56 |        0.01␣
↪|
| KernelRidge                    |              -11.50 |      -8.25 | 22.74 |        0.01␣
↪|
```

Warning: Regression and Classification are replaced with LazyRegressor and LazyClassifier. Regression and Classification classes will be removed in next release

CHAPTER 6

Installation

6.1 Stable release

To install Lazy Predict, run this command in your terminal:

```
$ pip install lazypredict
```

This is the preferred method to install Lazy Predict, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

6.2 From sources

The sources for Lazy Predict can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/shankarpandala/lazypredict
```

Or download the tarball:

```
$ curl -OJL https://github.com/shankarpandala/lazypredict/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER 7

Usage

To use Lazy Predict in a project:

```
import lazypredict
```

CHAPTER 8

Classification

Example

```
from lazypredict.Supervised import LazyClassifier
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
data = load_breast_cancer()
X = data.data
y= data.target
X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=.5,random_state␣
↪=123)
clf = LazyClassifier(verbose=0,ignore_warnings=True, custom_metric=None)
models,predictions = clf.fit(X_train, X_test, y_train, y_test)
models


| Model                          |   Accuracy |   Balanced Accuracy |   ROC AUC |␣
↪F1 Score |   Time Taken |
|:-------------------------------|-----------:|--------------------:|----------:|-----
↪------:|-------------:|
| LinearSVC                      |   0.989474 |            0.987544 |  0.987544 |   0.
↪989462 |    0.0150008 |
| SGDClassifier                  |   0.989474 |            0.987544 |  0.987544 |   0.
↪989462 |    0.0109992 |
| MLPClassifier                  |   0.985965 |            0.986904 |  0.986904 |   0.
↪985994 |    0.426     |
| Perceptron                     |   0.985965 |            0.984797 |  0.984797 |   0.
↪985965 |    0.0120046 |
| LogisticRegression             |   0.985965 |            0.98269  |  0.98269  |   0.
↪985934 |    0.0200036 |
| LogisticRegressionCV           |   0.985965 |            0.98269  |  0.98269  |   0.
↪985934 |    0.262997  |
| SVC                            |   0.982456 |            0.979942 |  0.979942 |   0.
↪982437 |    0.0140011 |
| CalibratedClassifierCV         |   0.982456 |            0.975728 |  0.975728 |   0.
↪982357 |    0.0350015 |
```

(continues on next page)

(continued from previous page)

```
| PassiveAggressiveClassifier     |   0.975439 |           0.974448 |  0.974448 |   0.
↪975464 |    0.0130005 |
| LabelPropagation                |   0.975439 |           0.974448 |  0.974448 |   0.
↪975464 |    0.0429988 |
| LabelSpreading                  |   0.975439 |           0.974448 |  0.974448 |   0.
↪975464 |    0.0310006 |
| RandomForestClassifier          |   0.97193  |           0.969594 |  0.969594 |   0.
↪97193  |    0.033     |
| GradientBoostingClassifier      |   0.97193  |           0.967486 |  0.967486 |   0.
↪971869 |    0.166998  |
| QuadraticDiscriminantAnalysis   |   0.964912 |           0.966206 |  0.966206 |   0.
↪965052 |    0.0119994 |
| HistGradientBoostingClassifier |   0.968421 |           0.964739 |  0.964739 |   0.
↪968387 |    0.682003  |
| RidgeClassifierCV               |   0.97193  |           0.963272 |  0.963272 |   0.
↪971736 |    0.0130029 |
| RidgeClassifier                 |   0.968421 |           0.960525 |  0.960525 |   0.
↪968242 |    0.0119977 |
| AdaBoostClassifier              |   0.961404 |           0.959245 |  0.959245 |   0.
↪961444 |    0.204998  |
| ExtraTreesClassifier            |   0.961404 |           0.957138 |  0.957138 |   0.
↪961362 |    0.0270066 |
| KNeighborsClassifier            |   0.961404 |           0.95503  |  0.95503  |   0.
↪961276 |    0.0560005 |
| BaggingClassifier               |   0.947368 |           0.954577 |  0.954577 |   0.
↪947882 |    0.0559971 |
| BernoulliNB                     |   0.950877 |           0.951003 |  0.951003 |   0.
↪951072 |    0.0169988 |
| LinearDiscriminantAnalysis      |   0.961404 |           0.950816 |  0.950816 |   0.
↪961089 |    0.0199995 |
| GaussianNB                      |   0.954386 |           0.949536 |  0.949536 |   0.
↪954337 |    0.0139935 |
| NuSVC                           |   0.954386 |           0.943215 |  0.943215 |   0.
↪954014 |    0.019989  |
| DecisionTreeClassifier          |   0.936842 |           0.933693 |  0.933693 |   0.
↪936971 |    0.0170023 |
| NearestCentroid                 |   0.947368 |           0.933506 |  0.933506 |   0.
↪946801 |    0.0160074 |
| ExtraTreeClassifier             |   0.922807 |           0.912168 |  0.912168 |   0.
↪922462 |    0.0109999 |
| CheckingClassifier              |   0.361404 |           0.5      |  0.5      |   0.
↪191879 |    0.0170043 |
| DummyClassifier                 |   0.512281 |           0.489598 |  0.489598 |   0.
↪518924 |    0.0119965 |
```

CHAPTER 9

Regression

Example

```
from lazypredict.Supervised import LazyRegressor
from sklearn import datasets
from sklearn.utils import shuffle
import numpy as np
boston = datasets.load_boston()
X, y = shuffle(boston.data, boston.target, random_state=13)
X = X.astype(np.float32)
offset = int(X.shape[0] * 0.9)
X_train, y_train = X[:offset], y[:offset]
X_test, y_test = X[offset:], y[offset:]
reg = LazyRegressor(verbose=0,ignore_warnings=False, custom_metric=None )
models,predictions = reg.fit(X_train, X_test, y_train, y_test)


| Model                         |   R-Squared |     RMSE |   Time Taken |
|:------------------------------|------------:|---------:|-------------:|
| SVR                           |    0.877199 |  2.62054 |    0.0330021 |
| RandomForestRegressor         |    0.874429 |  2.64993 |    0.0659981 |
| ExtraTreesRegressor           |    0.867566 |  2.72138 |    0.0570002 |
| AdaBoostRegressor             |    0.865851 |  2.73895 |    0.144999  |
| NuSVR                         |    0.863712 |  2.7607  |    0.0340044 |
| GradientBoostingRegressor     |    0.858693 |  2.81107 |    0.13      |
| KNeighborsRegressor           |    0.826307 |  3.1166  |    0.0179954 |
| HistGradientBoostingRegressor |    0.810479 |  3.25551 |    0.820995  |
| BaggingRegressor              |    0.800056 |  3.34383 |    0.0579946 |
| MLPRegressor                  |    0.750536 |  3.73503 |    0.725997  |
| HuberRegressor                |    0.736973 |  3.83522 |    0.0370018 |
| LinearSVR                     |    0.71914  |  3.9631  |    0.0179989 |
| RidgeCV                       |    0.718402 |  3.9683  |    0.018003  |
| BayesianRidge                 |    0.718102 |  3.97041 |    0.0159984 |
| Ridge                         |    0.71765  |  3.9736  |    0.0149941 |
| LinearRegression              |    0.71753  |  3.97444 |    0.0190051 |
```

(continues on next page)

(continued from previous page)

TransformedTargetRegressor	0.71753	3.97444	0.012001
LassoCV	0.717337	3.9758	0.0960066
ElasticNetCV	0.717104	3.97744	0.0860076
LassoLarsCV	0.717045	3.97786	0.0490005
LassoLarsIC	0.716636	3.98073	0.0210001
LarsCV	0.715031	3.99199	0.0450008
Lars	0.715031	3.99199	0.0269964
SGDRegressor	0.714362	3.99667	0.0210009
RANSACRegressor	0.707849	4.04198	0.111998
ElasticNet	0.690408	4.16088	0.0190012
Lasso	0.662141	4.34668	0.0180018
OrthogonalMatchingPursuitCV	0.591632	4.77877	0.0180008
ExtraTreeRegressor	0.583314	4.82719	0.0129974
PassiveAggressiveRegressor	0.556668	4.97914	0.0150032
GaussianProcessRegressor	0.428298	5.65425	0.0580051
OrthogonalMatchingPursuit	0.379295	5.89159	0.0180039
DecisionTreeRegressor	0.318767	6.17217	0.0230272
DummyRegressor	-0.0215752	7.55832	0.0140116
LassoLars	-0.0215752	7.55832	0.0180008
KernelRidge	-8.24669	22.7396	0.0309792

Warning: Regression and Classification are replaced with LazyRegressor and LazyClassifier. Regression and Classification classes will be removed in next release

CHAPTER 10

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

10.1 Types of Contributions

10.1.1 Report Bugs

Report bugs at https://github.com/shankarpandala/lazypredict/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

10.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

10.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

10.1.4 Write Documentation

Lazy Predict could always use more documentation, whether as part of the official Lazy Predict docs, in docstrings, or even on the web in blog posts, articles, and such.

10.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/shankarpandala/lazypredict/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

10.2 Get Started!

Ready to contribute? Here's how to set up *lazypredict* for local development.

1. Fork the *lazypredict* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/lazypredict.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv lazypredict
$ cd lazypredict/
$ python setup.py develop
$ pip install -r requirements_dev.txt
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 lazypredict tests
$ python setup.py test or pytest
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

10.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.7, 3.5, 3.6, 3.7 and 3.8, and for PyPy. Check https://travis-ci.org/shankarpandala/lazypredict/pull_requests and make sure that the tests pass for all supported Python versions.

10.4 Tips

To run a subset of tests:

```
$ pytest tests.test_lazypredict
```

10.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bump2version patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

CHAPTER 11

Credits

11.1 Development Lead

- Shankar Rao Pandala <shankar.pandala@live.com>

11.2 Contributors

- Breno Batista da Silva <brenophp@gmail.com>

CHAPTER 12

History

12.1 0.2.8 (2021-02-06)

- Removed StackingRegressor and CheckingClassifier.
- Added provided_models method.
- Added adjusted r-squared metric.
- Added cardinality check to split categorical columns into low and high cardinality features.
- Added different transformation pipeline for low and high cardinality features.
- Included all number dtypes as inputs.
- Fixed dependencies.
- Improved documentation.

12.2 0.2.7 (2020-07-09)

- Removed catboost regressor and classifier

12.3 0.2.6 (2020-01-22)

- Added xgboost, lightgbm, catboost regressors and classifiers

12.4 0.2.5 (2020-01-20)

- Removed troublesome regressors from list of CLASSIFIERS

12.5 0.2.4 (2020-01-19)

- Removed troublesome regressors from list of REGRESSORS
- Added feature to input custom metric for evaluation
- Added feature to return predictions as dataframe
- Added model training time for each model

12.6 0.2.3 (2019-11-22)

- Removed TheilSenRegressor from list of REGRESSORS
- Removed GaussianProcessClassifier from list of CLASSIFIERS

12.7 0.2.2 (2019-11-18)

- Fixed automatic deployment issue.

12.8 0.2.1 (2019-11-18)

- Release of Regression feature.

12.9 0.2.0 (2019-11-17)

- Release of Classification feature.

12.10 0.1.0 (2019-11-16)

- First release on PyPI.

CHAPTER 13

Indices and tables

- genindex
- modindex
- search