Data Quality Quick Start with EazyML APIs

1 Overview

Machine Learning (ML) is an involved science, its models often complex, not easy to understand. Transparent ML explains itself – its working, its prediction, its insights – so that the user understands it. Data Quality Check – before proceeding with the exhaustive AI/ML exercise, it's worthwhile to check if your data is good enough. EazyML's functions – Augmented Intelligence and Overlap Factor – derive the metric from data to alert you of data shortfalls for various measures – from data and model drift to completeness and bias. The python package is called "eazyml_data_quality.py". This package is offered as a quick start to dive into the comprehensive set of data quality checks described on eazyml.com.

2 Authenticate

EazyML authenticates you with username and API token. You can obtain your API token by logging into EazyML service portal and then navigating to "My Accounts" → "API Key".

Usage:

```
python eazyml_data_quality.py --username <username> --api_key
<api_key>
```

Example:

```
python eazyml_data_quality.py --username vikr.nunia@gmail.com --
api_key <api_key>
```

Authentication successful.

Authentication information is stored in authentication.json

Please note that your authentication information gets stored in a local file and then gets used for all subsequent calls described in the sequel below. Authentication is the mandatory first step in any experiment.

3 Data Shape Assessment

Post authentication, you can then perform data shape quality check.

Usage:

```
python eazyml_data_quality.py  --prefix_name <file_prefix> --
train_file <train file> --outcome <outcome_col> --data_shape
```

Example:

```
python eazyml_data_quality.py --prefix_name BPCL --train_file
Blood_Pressure_classification.csv --outcome "Systolic pressure" --
data_shape
```

--prefix_name BPCL

--train_file ../eazyml_quick_start/Blood_Pressure_classification.csv

--outcome Systolic pressure

--data_shape

Authentication successful

Authentication information is stored in authentication.json

The response for data quality assessment is stored in BPCL_data_quality.json

4 Data Balance Assessment

Post authentication, you can then perform data balance quality check.

Usage:

```
python eazyml_data_quality.py  --prefix_name <file_prefix> --
train_file <train file> --outcome <outcome_col> --data_balance
```

Example:

```
python eazyml_data_quality.py --prefix_name BPCL --train_file
Blood_Pressure_classification.csv --outcome "Systolic pressure" --
data_balance
```

--prefix_name BPCL

--train_file ../eazyml_quick_start/Blood_Pressure_classification.csv

--outcome Systolic pressure

--data_balance

Authentication successful

Authentication information is stored in authentication.json

The response for data quality assessment is stored in BPCL_data_quality.json

5 Data Emptiness Assessment

Post authentication, you can then perform data emptiness quality check.

Usage:

```
python eazyml_data_quality.py  --prefix_name <file_prefix> --
train_file <train file> --outcome <outcome_col> --data_emptiness
```

Example:

```
python eazyml_data_quality.py --prefix_name BPCL --train_file
Blood_Pressure_classification.csv --outcome "Systolic pressure" --
data_emptiness
```

--prefix_name BPCL

--train_file ../eazyml_quick_start/Blood_Pressure_classification.csv

--outcome Systolic pressure

--data_emptiness

Authentication successful

Authentication information is stored in authentication.json

The response for data quality assessment is stored in BPCL_data_quality.json

6 Data Completeness Assessment

Post authentication, you can then perform data completeness quality check.

Usage:

```
python eazyml_data_quality.py  --prefix_name <file_prefix> --
train_file <train file> --outcome <outcome_col> --data_completeness
```

Example:

```
python eazyml_data_quality.py --prefix_name BPCL --train_file
Blood_Pressure_classification.csv --outcome "Systolic pressure" --
data_completeness
```

--prefix_name BPCL

--train_file ../eazyml_quick_start/Blood_Pressure_classification.csv

--outcome Systolic pressure

--data_completeness

Authentication successful

Authentication information is stored in authentication.json

The response for data quality assessment is stored in BPCL_data_quality.json

7 Data Correctness Assessment

Post authentication, you can then perform data corretness quality check.

Usage:

```
python eazyml_data_quality.py  --prefix_name <file_prefix> --
train_file <train file> --outcome <outcome_col> --data_correctness
```

Example:

```
python eazyml_data_quality.py --prefix_name BPCL --train_file
Blood_Pressure_classification.csv --outcome "Systolic pressure" --
data_correctness
```

--prefix_name BPCL

--train_file ../eazyml_quick_start/Blood_Pressure_classification.csv

--outcome Systolic pressure

--data_correctness

Authentication successful

Authentication information is stored in authentication.json

The response for data quality assessment is stored in BPCL_data_quality.json

8 Data Drift Assessment

Post authentication, you can then perform data drift quality check.

Usage:

```
python eazyml_data_quality.py  --prefix_name <file_prefix> --
train_file <train file> --outcome <outcome_col> --data_drift
```

Example:

```
python eazyml_data_quality.py --prefix_name BPCL --train_file
Blood_Pressure_classification.csv --outcome "Systolic pressure" --
data_drift
```

--prefix_name BPCL

--train_file ../eazyml_quick_start/Blood_Pressure_classification.csv

--outcome Systolic pressure

--data_drift

Authentication successful

Authentication information is stored in authentication.json

The response for data quality assessment is stored in BPCL_data_quality.json

9 Data Model Drift Assessment

Post authentication, you can then perform data model drift quality check.

Usage:

```
python eazyml_data_quality.py  --prefix_name <file_prefix> --
train_file <train file> --outcome <outcome_col> --model_drft --
test_file <test_file>
```

Example:

```
python eazyml_data_quality.py --prefix_name BPCL --train_file
Blood_Pressure_classification.csv --outcome "Systolic pressure" --
model_drift --test_file BP_test.csv
```

--prefix_name BPCL

--train_file ../eazyml_quick_start/Blood_Pressure_classification.csv

--outcome Systolic pressure

--model_drift

--test_file BP_test.csv

Authentication successful

Authentication information is stored in authentication.json

The response for data quality assessment is stored in BPCL_data_quality.json

10 Feature Importance Assessment

Post authentication, you can then get the important features.

Usage:

```
python eazyml_data_quality.py  --prefix_name <file_prefix> --
train_file <train file> --outcome <outcome_col> --feature_importance
```

Example:

```
python eazyml_data_quality.py --prefix_name BPCL --train_file
Blood_Pressure_classification.csv --outcome "Systolic pressure" –
feature_importance
```

--prefix_name BPCL

--train_file ../eazyml_quick_start/Blood_Pressure_classification.csv

--outcome Systolic pressure

--feature_importance

Authentication successful

Authentication information is stored in authentication.json

The response for data quality assessment is stored in BPCL_data_quality.json

11 Data Outcome Correlation Assessment

Post authentication, you can then get the important features.

Usage:

```
python eazyml_data_quality.py  --prefix_name <file_prefix> --
train_file <train file> --outcome <outcome_col> --data_correlation
```

Example:

```
python eazyml_data_quality.py --prefix_name BPCL --train_file
Blood_Pressure_classification.csv --outcome "Systolic pressure" --
data_correlation
```

--prefix_name BPCL

--train_file ../eazyml_quick_start/Blood_Pressure_classification.csv

--outcome Systolic pressure

--data_correlation

Authentication successful

Authentication information is stored in authentication.json

The response for data quality assessment is stored in BPCL_data_quality.json

12 All Data Quality Assessment

Here's how you can combine multiple of previously explained steps from start to finish. In the process you perform all data quality checks with one command.

Usage:

```
python eazyml_data_quality.py  --prefix_name <file_prefix> --
train_file <train file> --outcome <outcome_col> --id_col <ID Col> --
discard_col_list <comma separated list if more than 1> --impute --
remove_outliers --data_shape --data_balance --data_emptiness --
data_outliers --data_completeness --data_correctness --data_drift --
model_drift --feature_importance --data_correlation --test_file
<predict file>
```

Example:

```
python eazyml_data_quality.py  --prefix_name "BPCL" --train_file
Blood_Pressure_classification.csv --outcome "Systolic pressure" --
impute --remove_outliers --data_shape --data_balance --data_emptiness
--data_outliers --data_completeness --data_correctness --data_drift -
-model_drift --feature_importance --data_correlation --test_file
BP_test.csv
```

--prefix_name BPCL

--train_file Blood_Pressure_classification.csv

--outcome Systolic pressure

--impute

--remove_outliers

--data_shape

--data_balance

--data_emptiness

--data_outliers

--data_completeness

--data_correctness

--data_drift

--model_drift

--feature_importance

--data_correlation

--test_file BP_test.csv

Authentication successful

Authentication information is stored in authentication.json

The response for data quality assessment is stored in BPCL_data_quality.json

Proprietary and Confidential