AntiNex Client Documentation

Release 1.0.0

Jay Johnson

Nov 02, 2018

Contents

CHAPTER 1

AntiNex Stack Status

AntiNex client is part of the AntiNex stack:

Component	Build	Docs Link	Docs Build
REST API		Docs	
Core Worker		Docs	
Network Pipeline		Docs	
AI Utils		Docs	
Client		Docs	

CHAPTER 2

Table of Contents

These are the docs for the AntiNex client repository.

2.1 AI Client Class

This is the AntiNex Python Client class

class antinex_client.ai_client.**AIClient**(*user='user-not-set'*, *password='password-not-set'*, *url='http://localhost:8010'*, *email='email-not-set'*, *verbose=True*, *ca_dir=None*, *cert_file=None*, *key_file=None*, *debug=False*)

AntiNex Python AI Client

This can use either environment variables or keyword arguments to create a valid client.

build_response(*status=3*, *error=''*, *data=None*)

Parameters

- **status** – status code
- **error** – error message
- **data** – dictionary to send back

get_auth_header()

get_job_by_id(*job_id=None*)

Parameters **job_id** – MLJob.id in the database

get_prepare_by_id(*prepare_id=None*)

Parameters **prepare_id** – MLJob.id in the database

get_result_by_id(*result_id=None*)

Parameters **result_id** – MLJobResult.id in the database

get_token()

is_logged_in()

login()

retry_login()

run_job(*body*)

Parameters **body** – dictionary to launch job

run_prepare(*body*)

Parameters **body** – dictionary to launch prepare

wait_for_job_to_finish(*job_id*, *sec_to_sleep=5.0*, *max_retries=100000*)

Parameters

- **job_id** – MLJob.id to wait on
- **sec_to_sleep** – seconds to sleep during polling
- **max_retries** – max retires until stopping

wait_for_prepare_to_finish(*prepare_id*, *sec_to_sleep=5.0*, *max_retries=100000*)

Parameters

- **prepare_id** – MLPrepare.id to wait on
- **sec_to_sleep** – seconds to sleep during polling
- **max_retries** – max retires until stopping

2.2 Building a Client from Environment Variables

This is how to build an AntiNex Python Client object from just environment variables.

antinex_client.build_ai_client_from_env.**build_ai_client_from_env**(*verbose=True*, *debug=False*, *ca_dir=None*, *cert_file=None*, *key_file=None*)

Use environment variables to build a client

Parameters

- **verbose** – verbose logging
- **debug** – debug internal client calls
- **ca_dir** – optional path to CA bundle dir
- **cert_file** – optional path to x509 ssl cert file
- **key_file** – optional path to x509 ssl key file

2.3 AntiNex Consts

Here are the environment variables and constants used by the AntiNex client.

```
LOGIN_SUCCESS = 0
LOGIN_NOT_ATTEMPTED = 1
LOGIN_FAILED = 2
SUCCESS = 0
FAILED = 1
ERROR = 2
NOT_SET = 3
DISABLED = 4
```

2.4 Client Environment Variables

These are the environment variables used by the AntiNex client.

Note: Please make sure these match up with your local, running stack:

AntiNex REST API flags, endpoint and credentials https://github.com/jay-johnson/train-ai-with-django-swagger-jwt

AntiNex is running using compose.yml file: https://github.com/jay-johnson/train-ai-with-django-swagger-jwt/blob/master/compose.yml

---

```
ANTINEX_PUBLISH_ENABLED = bool(ev(
    "ANTINEX_PUBLISH_ENABLED",
    "1") == "1")
ANTINEX_URL = ev(
    "ANTINEX_URL",
    "http://localhost:8010")
ANTINEX_CA_FILE = os.getenv(
    "ANTINEX_CA_FILE",
    None)
ANTINEX_CERT_FILE = os.getenv(
    "ANTINEX_CERT_FILE",
    None)
ANTINEX_KEY_FILE = os.getenv(
    "ANTINEX_KEY_FILE",
    None)
ANTINEX_USER = ev(
    "ANTINEX_USER",
    "root")
ANTINEX_EMAIL = ev(
    "ANTINEX_EMAIL",
    "notreal@test.com")
ANTINEX_PASSWORD = ev(
    "ANTINEX_PASSWORD",
    "123321")
# provide a template request publish file like:
# https://github.com/jay-johnson/antinex-client/blob/master/examples/predict-rows-
↪scaler-full-django.json
ANTINEX_PUBLISH_REQUEST_FILE = ev(
    "ANTINEX_PUBLISH_REQUEST_FILE",
```

(continues on next page)

(continued from previous page)

```
    ("/opt/antinex/client/examples/"
    "predict-rows-scaler-full-django.json"))
# comma-separated list
ANTINEX_FEATURES_TO_PROCESS_STR = os.getenv(
    "ANTINEX_FEATURES_TO_PROCESS",
    None)
# comma-separated list
ANTINEX_IGNORE_FEATURES_STR = os.getenv(
    "ANTINEX_IGNORE_FEATURES",
    None)
# comma-separated list
ANTINEX_SORT_VALUES_STR = os.getenv(
    "ANTINEX_SORT_VALUES",
    None)
# comma-separated list
ANTINEX_METRICS_STR = os.getenv(
    "ANTINEX_METRICS",
    None)
# comma-separated list
ANTINEX_HISTORIES_STR = os.getenv(
    "ANTINEX_HISTORIES",
    None)
ANTINEX_ML_TYPE = ev(
    "ANTINEX_ML_TYPE",
    "classification")
ANTINEX_USE_MODEL_NAME = ev(
    "ANTINEX_USE_MODEL_NAME",
    "Full-Django-AntiNex-Simple-Scaler-DNN")
ANTINEX_PREDICT_FEATURE = ev(
    "ANTINEX_PREDICT_FEATURE",
    "label_value")
ANTINEX_SEED = int(ev(
    "ANTINEX_SEED",
    "42"))
ANTINEX_TEST_SIZE = float(ev(
    "ANTINEX_TEST_SIZE",
    "0.2"))
ANTINEX_BATCH_SIZE = int(ev(
    "ANTINEX_BATCH_SIZE",
    "32"))
ANTINEX_EPOCHS = int(ev(
    "ANTINEX_EPOCHS",
    "15"))
ANTINEX_NUM_SPLITS = int(ev(
    "ANTINEX_NUM_SPLITS",
    "3"))
ANTINEX_LOSS = ev(
    "ANTINEX_LOSS",
    "binary_crossentropy")
ANTINEX_OPTIMIZER = ev(
    "ANTINEX_OPTIMIZER",
    "adam")
ANTINEX_VERSION = ev(
    "ANTINEX_VERSION",
    "1")
ANTINEX_CONVERT_DATA = bool(ev(
    "ANTINEX_CONVERT_DATA",
```

(continues on next page)

(continued from previous page)

```
    "1") == "1")
ANTINEX_CONVERT_DATA_TYPE = ev(
    "ANTINEX_CONVERT_DATA_TYPE",
    "float")
ANTINEX_INCLUDE_FAILED_CONVERSIONS = bool(ev(
    "ANTINEX_INCLUDE_FAILED_CONVERSIONS",
    "0") == "1")
ANTINEX_MISSING_VALUE = ev(
"ANTINEX_MISSING_VALUE",
"-1.0")
ANTINEX_PUBLISH_TO_CORE = bool(ev(
    "ANTINEX_PUBLISH_TO_CORE",
    "1") == "1")
ANTINEX_CHECK_MISSING_PREDICT = bool(ev(
    "ANTINEX_CHECK_MISSING_PREDICT",
    "1") == "1")
ANTINEX_CLIENT_VERBOSE = bool(ev(
    "ANTINEX_CLIENT_VERBOSE",
    "1") == "1")
ANTINEX_CLIENT_DEBUG = bool(ev(
    "ANTINEX_CLIENT_DEBUG",
    "0") == "1")

# set empty defaults
ANTINEX_FEATURES_TO_PROCESS = []
ANTINEX_IGNORE_FEATURES = []
ANTINEX_SORT_VALUES = []
ANTINEX_METRICS = []
ANTINEX_HISTORIES = []
```

These environment variables are set as lists based of commas:

```
if ANTINEX_FEATURES_TO_PROCESS_STR:
    ANTINEX_FEATURES_TO_PROCESS = \
        ANTINEX_FEATURES_TO_PROCESS_STR.split(",")
if ANTINEX_IGNORE_FEATURES_STR:
    ANTINEX_IGNORE_FEATURES = \
        ANTINEX_IGNORE_FEATURES_STR.split(",")
if ANTINEX_SORT_VALUES_STR:
    ANTINEX_SORT_VALUES = \
        ANTINEX_SORT_VALUES_STR.split(",")
if ANTINEX_METRICS_STR:
    ANTINEX_METRICS = \
        ANTINEX_METRICS_STR.split(",")
if ANTINEX_HISTORIES_STR:
    ANTINEX_HISTORIES = \
        ANTINEX_HISTORIES_STR.split(",")
```

Environment variables that are set to dictionaries for faster lookups:

```
FILTER_FEATURES_DICT = {}
FILTER_FEATURES = []
for idx, f in enumerate(ANTINEX_FEATURES_TO_PROCESS):
    include_feature = True
    if f == ANTINEX_PREDICT_FEATURE:
        include_feature = False
    for i in ANTINEX_IGNORE_FEATURES:
```

(continues on next page)

(continued from previous page)

```
        if f == i:
            include_feature = False
            break
    if include_feature:
        FILTER_FEATURES.append(f)
        FILTER_FEATURES_DICT[f] = idx
# end of for all features not being ignored
```

2.5 Generate an AntiNex AI Request

This method will use the environment variables from the consts.py:

2.5.1 generate_ai_request Method

antinex_client.generate_ai_request.**generate_ai_request**(*predict_rows*, *req_dict=None*, *req_file='/opt/antinex/client/examples/predict-rows-scaler-full-django.json'*, *features=[]*, *ignore_features=[]*, *sort_values=[]*, *ml_type='classification'*, *use_model_name='Full-Django-AntiNex-Simple-Scaler-DNN'*, *predict_feature='label_value'*, *seed=42*, *test_size=0.2*, *batch_size=32*, *epochs=15*, *num_splits=3*, *loss='binary_crossentropy'*, *optimizer='adam'*, *metrics=[]*, *histories=[]*, *filter_features_dict={}*, *filter_features=[]*, *convert_enabled=True*, *convert_to_type='float'*, *include_failed_conversions=False*, *value_for_missing='-1.0'*, *version='1'*, *publish_to_core=True*, *check_missing_predict_feature=True*, *debug=False*)

Parameters

- **predict_rows** – list of predict rows to build into the request
- **req_dict** – request dictionary to update - for long-running clients
- **req_file** – file holding a request dict to update - one-off tests
- **features** – features to process in the data

- **ignore_features** – features to ignore in the data (non-numerics)
- **sort_values** – optional - order rows for scaler normalization
- **ml_type** – machine learning type - classification/regression
- **use_model_name** – use a pre-trained model by name
- **predict_feature** – predict the values of this feature
- **seed** – seed for randomness reproducability
- **test_size** – split train/test data
- **batch_size** – batch size for processing
- **epochs** – test epochs
- **num_splits** – test splits for cross validation
- **loss** – loss function
- **optimizer** – optimizer
- **metrics** – metrics to apply
- **histories** – historical values to test
- **filter_features_dict** – dictionary of features to use
- **filter_features** – list of features to use
- **convert_to_type** – convert predict_row values to scaler-ready values
- **include_failed_conversions** – should the predict rows include fails
- **value_for_missing** – set this value to any columns that are missing
- **version** – version of the API request
- **publish_to_core** – want to publish it to the core or the worker
- **debug** – log debug messages

2.6 AntiNex Client Utils

Utility methods

antinex_client.utils.**convert_to_date**(*value=None*, *format='%Y-%m-%d %H:%M:%S'*)
param: value - datetime object param: format - string format

antinex_client.utils.**ev**(*k*, *v*)

Parameters

- **k** – environment variable key
- **v** – environment variable value

antinex_client.utils.**ppj**(*json_data*)

Parameters **json_data** – dictionary to print

antinex_client.utils.**rnow**(*f='%Y-%m-%d %H:%M:%S'*)

Parameters **f** – format for the string

2.7 Make Predictions with a Pre-trained Deep Neural Network

This uses environment variables to build, train and make predictions using the AntiNex client. If the deep neural network already exists it will use it to make new predictions. If it does not exist it will train a new one.

This python script is available in the pip: `ai_env_predict.py`

It takes parameters:

```
parser = argparse.ArgumentParser(
    description=("Python client to make Predictions "
                 "using a Pre-trained Deep Neural Network "
                 "with AntiNex Django Rest Framework"))
parser.add_argument(
    "-f",
    help=("file to use default ./examples/"
          "predict-rows-scaler-full-django.json"),
    required=False,
    dest="datafile")
parser.add_argument(
    "-m",
    help="send mock data",
    required=False,
    dest="use_fake_rows",
    action="store_true")
parser.add_argument(
    "-s",
    help="silent",
    required=False,
    dest="silent",
    action="store_true")
parser.add_argument(
    "-d",
    help="debug",
    required=False,
    dest="debug",
    action="store_true")
```

2.7.1 Source Code

antinex_client.scripts.ai_env_predict.**start_predictions**()

Using environment variables, create an AntiNex AI Client. You can also use command line args if you want.

This can train a new deep neural network if it does not exist or it can use an existing pre-trained deep neural network within the AntiNex Core to make new predictions.

2.8 Train a New Deep Neural Network and Make Predictions

This uses environment variables to build, train and make predictions using the AntiNex client.

This python script is available in the pip: `ai_train_dnn.py`

It takes parameters:

```
parser = argparse.ArgumentParser(
    description=("Python client to Train a Deep Neural Network "
                 "with AntiNex Django Rest Framework"))
parser.add_argument(
        "-u",
        help="username",
        required=False,
        dest="user")
parser.add_argument(
        "-p",
        help="user password",
        required=False,
        dest="password")
parser.add_argument(
        "-e",
        help="user email",
        required=False,
        dest="email")
parser.add_argument(
        "-a",
        help="url endpoint with default http://localhost:8010",
        required=False,
        dest="url")
parser.add_argument(
        "-f",
        help="file to use default ./examples/test-keras-dnn.json",
        required=False,
        dest="datafile")
parser.add_argument(
        "-s",
        help="silent",
        required=False,
        dest="silent",
        action="store_true")
parser.add_argument(
        "-d",
        help="debug",
        required=False,
        dest="debug",
        action="store_true")
```

2.8.1 Source Code

antinex_client.scripts.ai_train_dnn.**train_new_deep_neural_network**()

Train a new deep neural network and store the results as a new: `MLJob` and `MLJobResult` database records.

2.9 Get an ML Job

This python script is available in the pip: `ai_get_job.py`

It takes parameters:

```
parser = argparse.ArgumentParser(
        description=("Python client get AI Job by ID"))
```

(continues on next page)

(continued from previous page)

```
parser.add_argument(
        "-u",
        help="username",
        required=False,
        dest="user")
parser.add_argument(
        "-p",
        help="user password",
        required=False,
        dest="password")
parser.add_argument(
        "-e",
        help="user email",
        required=False,
        dest="email")
parser.add_argument(
        "-a",
        help="url endpoint with default http://localhost:8010",
        required=False,
        dest="url")
parser.add_argument(
        "-i",
        help="User's MLJob.id to look up",
        required=False,
        dest="job_id")
parser.add_argument(
        "-s",
        help="silent",
        required=False,
        dest="silent",
        action="store_true")
parser.add_argument(
        "-d",
        help="debug",
        required=False,
        dest="debug",
        action="store_true")
```

2.9.1 Source Code

antinex_client.scripts.ai_get_job.**get_ml_job**()
Get an MLJob by database id.

2.10 Get ML Job Results

This python script is available in the pip: ai_get_results.py

It takes parameters:

```
parser = argparse.ArgumentParser(
        description=("Python client get AI Results by ID"))
parser.add_argument(
        "-u",
```

(continues on next page)

(continued from previous page)

```
        help="username",
        required=False,
        dest="user")
parser.add_argument(
        "-p",
        help="user password",
        required=False,
        dest="password")
parser.add_argument(
        "-e",
        help="user email",
        required=False,
        dest="email")
parser.add_argument(
        "-a",
        help="url endpoint with default http://localhost:8010",
        required=False,
        dest="url")
parser.add_argument(
        "-i",
        help="User's MLJobResult.id to look up",
        required=False,
        dest="result_id")
parser.add_argument(
        "-s",
        help="silent",
        required=False,
        dest="silent",
        action="store_true")
parser.add_argument(
        "-d",
        help="debug",
        required=False,
        dest="debug",
        action="store_true")
```

2.10.1 Source Code

antinex_client.scripts.ai_get_results.**get_ml_job_results**()
Get an MLJobResult by database id.

2.11 Prepare a New Dataset

This python script is available in the pip: ai_prepare_dataset.py

It takes parameters:

```
parser = argparse.ArgumentParser(
        description=("Python client to Prepare a dataset"))
parser.add_argument(
        "-u",
        help="username",
        required=False,
```

(continues on next page)

(continued from previous page)

```
        dest="user")
parser.add_argument(
        "-p",
        help="user password",
        required=False,
        dest="password")
parser.add_argument(
        "-e",
        help="user email",
        required=False,
        dest="email")
parser.add_argument(
        "-a",
        help="url endpoint with default http://localhost:8010",
        required=False,
        dest="url")
parser.add_argument(
        "-f",
        help="file to use default ./examples/test-keras-dnn.json",
        required=False,
        dest="prepare_file")
parser.add_argument(
        "-s",
        help="silent",
        required=False,
        dest="silent",
        action="store_true")
parser.add_argument(
        "-d",
        help="debug",
        required=False,
        dest="debug",
        action="store_true")
```

2.11.1 Source Code

antinex_client.scripts.ai_prepare_dataset.**prepare_new_dataset**()
Prepare a new MLPrepare record and dataset files on disk.

2.12 Get a Prepared Dataset from the Database

This python script is available in the pip: ai_get_prepared_dataset.py

It takes parameters:

```
parser.add_argument(
        "-u",
        help="username",
        required=False,
        dest="user")
parser.add_argument(
        "-p",
        help="user password",
```

(continues on next page)

(continued from previous page)

```
        required=False,
        dest="password")
parser.add_argument(
        "-e",
        help="user email",
        required=False,
        dest="email")
parser.add_argument(
        "-a",
        help="url endpoint with default http://localhost:8010",
        required=False,
        dest="url")
parser.add_argument(
        "-i",
        help="User's MLPrepare.id to look up",
        required=False,
        dest="prepare_id")
parser.add_argument(
        "-s",
        help="silent",
        required=False,
        dest="silent",
        action="store_true")
parser.add_argument(
        "-d",
        help="debug",
        required=False,
        dest="debug",
        action="store_true")
```

2.12.1 Source Code

antinex_client.scripts.ai_get_prepared_dataset.**get_prepared_dataset**()
Get an MLPrepare by database id.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index

W