aisquared

The AI Squared Team

DOCUMENTATION

1	Installation															3
1.1.1 aisquared package										3						
		1.1.1.1	Module cont	ents .						 	 	 				3
		1.1.1.2	Subpackages							 	 	 				3
2	Changelog															61
Ру	thon Module	Index														65
In	dex															67



This package contains utilities to interact with the AI Squared technology stack, particularly with developing and deploying models to the AI Squared Platform or other applications developed through the AI Squared JavaScript SDK.

Current Production Version: 0.3.3

View this Documentation in PDF Format.

DOCUMENTATION 1

2 DOCUMENTATION

CHAPTER

ONE

INSTALLATION

This package is available through Pypi and can be installed by running the following command:

pip install aisquared

Alternatively, the latest version of the software can be installed directly from GitHub using the following command:

pip install git+https://github.com/AISquaredInc/aisquared

1.1 aisquared

1.1.1 aisquared package

1.1.1.1 Module contents

This package contains utilities to interact with the AI Squared technology stack, particularly with developing and deploying models to the AI Squared Browser Extension or other applications developed through the AI Squared JavaScript SDK.

1.1.1.2 Subpackages

aisquared.base package

Module contents

The aisquared base package contains both some basic objects that are used across the aisquared package backend and some objects which are designed to facilitate simple use cases of the technology.

Submodules

aisquared.base.BaseObject module

class aisquared.base.BaseObject.BaseObject

Bases: object

Base class used for all other classes within the aisquared package. This class is not meant to be used by any end user of this package, but is rather used throughout this package as a parent class.

```
to_dict() → dict
Get the object as a dictionary
to_json() → str
Return the object as a json string
```

aisquared.base.CustomObject module

Bases: BaseObject

Custom class that allows the user to define custom classes for configuration

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.base.CustomObject(
    'MyClass',
    key1 = 'foo',
    key2 = 'bar'
    )
>>> my_obj.to_dict()
{'className': 'MyClass', 'params': {'key1': 'foo', 'key2': 'bar'}}
)
```

```
\textbf{to\_dict()} \rightarrow dict
```

Get the object as a dictionary

aisquared.base.rendering module

Some allowed configuration parameters - not meant to be directly called by the end user

aisquared.base.stages module

Some allowed configuration parameters - not designed to be directly called by the user

aisquared.config package

Module contents

The aisquared.config subpackage contains utilities and objects for packaging aisquared configuration steps and models.

For in-depth examples of how to build out .air files using the utilities and classes in this library, please visit our GitHub repository at https://github.com/AISquaredInc/airFiles

Subpackages

aisquared.config.analytic package

Module contents

The aisquared.config.analytic subpackage contains objects for packaging individual analytics.

Submodules

aisquared.config.analytic.DeployedAnalytic module

Bases: BaseObject

Interaction with a remote analytic

Example usage:

```
>>> import aisquared
>>> analytic = aisquared.config.analytic.DeployedAnalytic(
    'analytic_url',
    'text'
)
>>> analytic.to_dict()
{'className': 'DeployedAnalytic',
'params': {'url': 'analytic_url',
'inputType': 'text',
'secret': 'request',
'header': None}}
```

```
property header
property input_type
property secret
to_dict() → dict
    Get the object as a dictionary
property url
```

aisquared.config.analytic.DeployedModel module

Bases: BaseObject

Interaction with a remote model

Example usage:

```
property header
property input_type
property secret
to_dict() → dict
    Get the config object as a dictionary
property url
```

aisquared.config.analytic.LocalAnalytic module

Bases: BaseObject

Interaction with an analytic (lookup table) saved to the local file system

Example usage:

(continues on next page)

```
property all
     property input_type
     property path
     to_dict() \rightarrow dict
          Get the configuration object as a dictionary
aisquared.config.analytic.LocalModel module
class aisquared.config.analytic.LocalModel.LocalModel(path: str, input_type: str)
     Bases: BaseObject
     Interaction with a model currently saved to the local file system
     Example usage:
     >>> import aisquared
     >>> analytic = aisquared.config.analytic.LocalModel(
          'model_path',
          'text'
     >>> analytic.to_dict()
     {'className': 'LocalModel',
     'params': {'path': 'model_path',
     'inputType': 'text'}}
     property input_type
     property path
     to_dict() \rightarrow dict
          Get the configuration object as a dictionary
aisquared.config.analytic.ReverseMLWorkflow module
class aisquared.config.analytic.ReverseMLWorkflow.ReverseMLWorkflow(bucket: str, filename: str,
                                                                             column: str, input_type:
                                                                             str, period: Optional[int] =
                                                                             None, secret: str = ")
     Bases: BaseObject
     Interaction with a ReverseML CSV stored in S3
     Example usage:
     >>> import aisquared
     >>> analytic = aisquared.config.analytic.ReverseMLWorkflow(
          'bucket_name',
          'file_name',
          'column_name',
```

1.1. aisquared 7

'text'

)

(continued from previous page)

```
>>> analytic.to_dict()
{'className': 'ReverseMLWorkflow',
'params': {'bucket': 'bucket_name',
'fileName': 'file_name',
'inputType': 'text',
'column': 'column_name',
'period': None,
'secret': ''}}
```

```
property bucket

property column

property filename

property input_type

property period

property secret

to_dict() → dict

Get the configuration object as a dictionary
```

aisquared.config.feedback package

Module contents

The aisquared.config.feedback subpackage contains objects for configuring feedback in aisquared models.

Submodules

aisquared.config.feedback.BinaryFeedback module

```
{\bf class} \ a is quared. config. feedback. Binary Feedback. {\bf Binary Feedback} ({\it label\_map: list}) \\ Bases: {\it BaseObject}
```

Example usage:

Feedback for binary classification

```
>>> import aisquared
>>> my_obj = aisquared.config.feedback.BinaryFeedback(['class1', 'class2'])
>>> my_obj.to_dict()
{'className': 'BinaryFeedback', 'params': {'labelMap': ['class1', 'class2']}}
```

```
property label_map

to_dict() \rightarrow dict

Return the object as a dictionary
```

aisquared.config.feedback.ModelFeedback module

class aisquared.config.feedback.ModelFeedback

Bases: BaseObject

Feedback object for questions and answers for an individual model.

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.feedback.ModelFeedback()
>>> my_obj.add_question(
    'How is the model performing?',
    choices = ['very poorly', 'poorly', 'neutral', 'well', 'very well']
>>> my_obj.add_question(
    'Any additional feedback?',
    'text'
>>> my_obj.to_dict()
{'className': 'ModelFeedback',
'params': {'questions': [{'question': 'How is the model performing?',
'answerType': 'singleChoice',
'choices': ['very poorly', 'poorly', 'neutral', 'well', 'very well']},
{'question': 'Any additional feedback?', 'answerType': 'text'}]}}
```

 $add_question(question: str, answer type: str = 'singleChoice', choices: list = [])$

Add a question to be asked.

Parameters

- **question** (*str*) The question to be asked.
- answer_type (str (default 'singleChoice')) One of either 'singleChoice', 'multiChoice', or 'text'
- **choices** (list (default [])) The choices to be provided, if answer_type is 'single-Choice' or 'multiChoice'

```
to_dict() \rightarrow dict
```

Return the object as a dictionary

aisquared.config.feedback.MulticlassFeedback module

class aisquared.config.feedback.MulticlassFeedback.MulticlassFeedback(label_map: list)

Bases: BaseObject

Feedback for multiclass classification

Example Usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.feedback.MulticlassFeedback(['class1', 'class2',
>>> my_obj.to_dict()
```

1.1. aisquared 9

(continues on next page)

(continued from previous page)

```
{'className': 'MulticlassFeedback',
'params': {'labelMap': ['class1', 'class2', 'class3']}}
```

property label_map

```
to_dict() \rightarrow dict
```

Return the object as a dictionary

aisquared.config.feedback.QualitativeFeedback module

${\bf class} \ {\bf aisquared.config.feedback.Qualitative Feedback.Qualitative Feedback}$

Bases: BaseObject

Feedback object for questions and answers for individual predictions.

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.feedback.QualitativeFeedback()
>>> my_obj.add_question('Any additional feedback?', 'text')
>>> my_obj.to_dict()
{'className': 'QualitativeFeedback',
'params': {'questions': [{'question': 'Any additional feedback?',
'answerType': 'text'}]}}
```

add_question(question: str, answer_type: str = 'singleChoice', choices: list = [])

Add a question to be asked.

Parameters

- **question** (*str*) The question to be asked.
- answer_type (str (default 'singleChoice')) One of either 'singleChoice', 'multiChoice', or 'text'
- **choices** (*list* (*default* [])) The choices to be provided, if *answer_type* is 'single-Choice' or 'multiChoice'

```
\textbf{to\_dict()} \rightarrow dict
```

Return the object as a dictionary

aisquared.config.feedback.RegressionFeedback module

class aisquared.config.feedback.RegressionFeedback.RegressionFeedback

```
Bases: BaseObject
```

Feedback for regression

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.feedback.RegressionFeedback()
>>> my_obj.to_dict()
{'className': 'RegressionFeedback', 'params': {}}
```

```
to\_dict() \rightarrow dict
```

Return the object as a dictionary

aisquared.config.feedback.SimpleFeedback module

class aisquared.config.feedback.SimpleFeedback

```
Bases: BaseObject
```

Simple thumbs-up/thumbs-down feedback for predictions

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.feedback.SimpleFeedback()
>>> my_obj.to_dict()
{'className': 'SimpleFeedback', 'params': {}}
```

```
\textbf{to\_dict()} \rightarrow dict
```

Return the object as a dictionary

aisquared.config.harvesting package

Module contents

The aisquared.config.harvesting subpackage contains objects for configuring harvesting of data.

Submodules

aisquared.config.harvesting.lmageHarvester module

```
class aisquared.config.harvesting.ImageHarvester.ImageHarvester(how: str = 'all')
```

```
Bases: BaseObject
```

Object to harvest images

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.harvesting.ImageHarvester()
>>> my_obj.to_dict()
{'className': 'ImageHarvester', 'params': {'how' : 'all'}}
```

property how

```
\textbf{to\_dict()} \rightarrow dict
```

Get the configuration object as a dictionary

aisquared.config.harvesting.InputHarvester module

aisquared.config.harvesting.QueryParameterHarvester module

Get the configuration object as a dictionary

class aisquared.config.harvesting.QueryParameterHarvester.QueryParameterHarvester(query_keys:

Union[str, list], url_locations: Union[str, list], attributes: Union[str, list])

Bases: BaseObject

 $to_dict() \rightarrow dict$

Harvester for Query Parameters

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.harvesting.QueryParameterHarvester(
    'test_key',
    'test_url',
    'test_attribute'
)
>>> my_obj.to_dict()
{'className': 'QueryParameterHarvester',
'params': {'queryKeys': ['test_key'],
'urlLocations': ['test_url'],
'attributes': ['test_attribute']}}
```

```
property attributes
     property query_keys
     to\_dict() \rightarrow dict
          Get the configuration object as a dictionary
     property url_locations
aisquared.config.harvesting.TextHarvester module
class aisquared.config.harvesting.TextHarvester.TextHarvester(how: str = 'all', regex:
                                                                       Optional[str] = None, flags: str =
                                                                       'gu', body\_only: bool = False,
                                                                       keywords: Optional[Union[str,
                                                                       list]] = None, limit: Optional[int]
                                                                       = None)
     Bases: BaseObject
     Object to harvest text
     Example usage:
     >>> import aisquared
     >>> my_obj = aisquared.config.harvesting.TextHarvester(
         how = 'all',
         body_only = True
     >>> my_obj.to_dict()
     {'className': 'TextHarvester',
     'params': {'how': 'all',
     'regex': None,
     'flags': 'gu',
     'bodyOnly': True,
     'limit': None}}
     property body_only
     property flags
     property how
     property limit
     property regex
     to\_dict() \rightarrow dict
          Get the configuration object as a dictionary
```

aisquared.config.postprocessing package

Module contents

The aisquared.config.postprocessing subpackage contains objects for configuring how predictions are postprocessed.

Submodules

aisquared.config.postprocessing.BinaryClassification module

class aisquared.config.postprocessing.BinaryClassification.BinaryClassification(label_map: list, threshold: float = 0.5)

Bases: BaseObject

Postprocesssing configuration object for binary classification

Example usage

```
>>> import aisquared
>>> my_obj = aisquared.config.postprocessing.BinaryClassification(
    ['class1', 'class2']
)
>>> my_obj.to_dict()
{'className': 'BinaryClassification',
'params': {'labelMap': ['class1', 'class2'], 'threshold': 0.5}}
```

```
property label_map
```

property threshold

```
to\_dict() \rightarrow dict
```

Get the configuration object as a dictionary

aisquared.config.postprocessing.MulticlassClassification module

Bases: BaseObject

Postprocessing configuration object for multiclass classification

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.postprocessing.MulticlassClassification(
    ['class1', 'class2', 'class3']
)
>>> my_obj.to_dict()
{'className': 'MulticlassClassification',
    'params': {'labelMap': ['class1', 'class2', 'class3']}}
```

```
property label_map  \begin{tabular}{ll} to\_dict() \rightarrow dict \\ Get the configuration object as a dictionary \\ \end{tabular}
```

aisquared.config.postprocessing.ObjectDetection module

class aisquared.config.postprocessing.ObjectDetection.ObjectDetection($label_map: list, threshold: float = 0.5$)

Bases: BaseObject

Postprocessing configuration object for object detection

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.postprocessing.ObjectDetection(
    ['class1', 'class2', 'class3']
)
>>> my_obj.to_dict()
{'className': 'ObjectDetection',
    'params': {'labelMap': ['class1', 'class2', 'class3'], 'threshold': 0.5}}
```

```
property label_map
```

property threshold

 $to_dict() \rightarrow dict$

Get the configuration object as a dictionary

aisquared.config.postprocessing.Regression module

```
 \textbf{class} \  \, \text{aisquared.config.postprocessing.Regression.Regression}(\textit{min: Optional[Union[int, float]]} = \\ \textit{None, max: Optional[Union[int, float]]} = \\ \textit{None, max: Optional[Union[int, float]]} = \\ \textit{None, round: bool} = \textit{False})
```

Bases: BaseObject

Postprocessing configuration object for Regression

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.postprocessing.Regression(
        10,
        100
)
>>> my_obj.to_dict()
{'className': 'Regression', 'params': {'min': 10, 'max': 100, 'round': False}}
```

property max
property min
property round

```
to\_dict() \rightarrow dict
```

Get the configuration object as a dictionary

aisquared.config.preprocessing package

Module contents

The aisquared.config.preprocessing subpackage contains utilities to configure the preprocessing of data in the data pipeline. It contains

three separate subpackages, aisquared.config.preprocessing.text, aisquared.config.preprocessing.image, and aisquared.config.preprocessing.tabular, which configure the preprocessing of different types of data.

Subpackages

aisquared.config.preprocessing.image package

Module contents

The aisquared.config.preprocessing.image subpackage contains objects for configuring image preprocessing.

Submodules

aisquared.config.preprocessing.image.lmagePreprocessing module

Bases: BaseObject

Preprocessor object for image data

Example usage:

```
>>> import aisquared
>>> preprocesser = aisquared.config.preprocessing.image.ImagePreprocessor()
>>> preprocesser.add_step(
    aisquared.config.preprocessing.image.AddValue(255.0)
)
```

```
add_step(step)
```

Add a step to the preprocessor object

```
property step_dict
```

```
to_dict() \rightarrow dict
```

Get the configuration object as a dictionary

aisquared.config.preprocessing.image.Steps module

```
class aisquared.config.preprocessing.image.Steps.AddValue(value: Union[int, float])
     Bases: BaseObject
     Preprocessing step to add a value to all pixels in an image
     Example usage:
     >>> import aisquared
     >>> preprocesser = aisquared.config.preprocessing.image.ImagePreprocessor()
     >>> preprocesser.add_step(
         aisquared.config.preprocessing.image.AddValue(255.0)
     )
     to\_dict() \rightarrow dict
          Get the configuration object as a dictionary
     property value
class aisquared.config.preprocessing.image.Steps.ConvertToColor(color: str)
     Bases: BaseObject
     Preprocessing step to convert images to a color scheme
     Example usage:
     >>> import aisquared
     >>> preprocesser = aisquared.config.preprocessing.image.ImagePreprocessor()
     >>> preprocesser.add_step(
         aisquared.config.preprocessing.image.ConvertToColor('RGB')
     )
     property color
     to_dict() \rightarrow dict
          Get the configuration object as a dictionary
class aisquared.config.preprocessing.image.Steps.DivideValue(value: Union[int, float])
     Bases: BaseObject
     Preprocessing step to divide all pixels in an image by a value
     Example usage:
     >>> import aisquared
     >>> preprocesser = aisquared.config.preprocessing.image.ImagePreprocessor()
     >>> preprocesser.add_step(
         aisquared.config.preprocessing.image.DivideValue(255.0)
     to_dict() \rightarrow dict
          Get the configuration object as a dictionary
     property value
```

```
class aisquared.config.preprocessing.image.Steps.MultiplyValue(value: Union[int, float])
     Bases: BaseObject
     Preprocessing step to multiply all pixels in an image by a value
     Example usage:
     >>> import aisquared
     >>> preprocesser = aisquared.config.preprocessing.image.ImagePreprocessor()
     >>> preprocesser.add_step(
         aisquared.config.preprocessing.image.MultiplyValue(2.0)
     to\_dict() \rightarrow dict
          Get the configuration object as a dictionary
     property value
class aisquared.config.preprocessing.image.Steps.Resize(size: list, method: str = 'bilinear',
                                                               preserve_aspect_ratio: bool = False)
     Bases: BaseObject
     Preprocessing step to resize an image
     >>> import aisquared
     >>> preprocesser = aisquared.config.preprocessing.image.ImagePreprocessor()
     >>> preprocesser.add_step(
         aisquared.config.preprocessing.image.Resize([100, 100])
     property method
     property preserve_aspect_ratio
     property size
     \textbf{to\_dict()} \rightarrow dict
          Get the configuration object as a dictionary
class aisquared.config.preprocessing.image.Steps.SubtractValue(value: Union[int, float])
     Bases: BaseObject
     Preprocessing step to subtract a value from all pixels in an image
     Example usage:
     >>> import aisquared
     >>> preprocesser = aisquared.config.preprocessing.image.ImagePreprocessor()
     >>> preprocesser.add_step(
         aisquared.config.preprocessing.image.SubtractValue(255.0)
     to_dict() \rightarrow dict
          Get the configuration object as a dictionary
     property value
```

aisquared.config.preprocessing.tabular package

Module contents

The aisquared.config.preprocessing.tabular subpackage contains objects for preprocessing tabular data.

Submodules

aisquared.config.preprocessing.tabular.Steps module

```
class aisquared.config.preprocessing.tabular.Steps.DropColumn(column: int)
    Bases: BaseObject
```

Drop a column from tabular data

Example usage:

property column

```
to_dict() \rightarrow dict
```

Get the configuration object as a dictionary

class aisquared.config.preprocessing.tabular.Steps.MinMax(mins: list, maxs: list, columns: Optional[list] = None)

Bases: BaseObject

Min-Max Scaling preprocessing step

Min-Max Scaling takes all associated columns and maps values relative to the minimum and maximum values of the training data.

Example usage:

```
>>> import aisquared
>>> preprocesser = aisquared.config.preprocessing.tabular.TabularPreprocessor()
>>> preprocesser.add_step(
    aisquared.config.preprocessing.tabular.MinMax(
        [0, 1.1, 2],
        [0.2, 14, 18.3]
    )
)
```

property columns

property maxs

```
property mins
     to\_dict() \rightarrow dict
          Get the configuration object as a dictionary
class aisquared.config.preprocessing.tabular.Steps.OneHot(column: int, values: list)
     Bases: BaseObject
     One Hot encoding preprocessing step
     Example usage:
     >>> import aisquared
     >>> preprocesser = aisquared.config.preprocessing.tabular.TabularPreprocessor()
     >>> preprocesser.add_step(
          aisquared.config.preprocessing.tabular.OneHot(
              ['one', 'two', 'three']
          )
     )
     property column
     to\_dict() \rightarrow dict
          Get the configuration object as a dictionary
     property values
class aisquared.config.preprocessing.tabular.Steps.ZScore(means: list, stds: list, columns:
                                                                  Optional[Union[int, list]] = None
     Bases: BaseObject
     Z-Score normalization preprocessing step
     Z-Score normalization takes each supplied column value, subtracts that column's provided mean, and divides by
     the provided standard deviation.
     Example usage:
     >>> import aisquared
     >>> preprocesser = aisquared.config.preprocessing.tabular.TabularPreprocessor()
     >>> preprocesser.add_step(
          aisquared.config.preprocessing.tabular.ZScore(
              [0, 1, 2],
              [0.2, 0.4, 0.6]
          )
     )
     property columns
     property means
     property stds
     to\_dict() \rightarrow dict
```

Get the configuration object as a dictionary

aisquared.config.preprocessing.tabular.TabularPreprocessing module

 $\textbf{class} \ \, \textbf{aisquared.config.preprocessing.tabular.TabularPreprocessing.TabularPreprocessor} (\textit{steps:} \\$

Optional[list]
=
None)

Bases: BaseObject

Preprocessor object for tabular data

Example usage:

Example usage:

```
>>> import aisquared
>>> preprocesser = aisquared.config.preprocessing.tabular.TabularPreprocessor()
>>> preprocesser.add_step(
    aisquared.config.preprocessing.tabular.ZScore(
        [0, 1, 2],
        [0.2, 0.4, 0.6]
    )
)
```

add_step(step)

Add a step to the preprocessor object

to_dict()

Get the configuration object as a dictionary

aisquared.config.preprocessing.text package

Module contents

The aisquared.config.preprocessing.text subpackage contains objects for preprocessing text data.

Submodules

aisquared.config.preprocessing.text.Steps module

class aisquared.config.preprocessing.text.Steps.ConvertToCase(lowercase: bool = True)

Bases: BaseObject

Text preprocessing object to convert inputs to all lowercase or all uppercase

Example usage:

```
>>> import aisquared
>>> preprocesser = aisquared.config.preprocessing.text.TextPreprocessor()
>>> preprocesser.add_step(
    aisquared.config.preprocessing.text.ConvertToCase()
)
```

```
property lowercase
     to\_dict() \rightarrow dict
          Get the configuration object as a dictionary
class aisquared.config.preprocessing.text.Steps.ConvertToVocabulary(vocabulary: dict,
                                                                             start character: int = 1,
                                                                             oov\_character: int = 2,
                                                                             max_vocab: Optional[int]
                                                                             = None)
     Bases: BaseObject
     Text preprocessing object to convert tokens to integer vocabularies
     Example usage:
     >>> import aisquared
     >>> preprocesser = aisquared.config.preprocessing.text.TextPreprocessor()
     >>> preprocesser.add_step(
         aisquared.config.preprocessing.text.ConvertToVocabulary(
              {
                   'test' : 3,
                   'vocabulary': 4
              }
         )
     property max_vocab
     property oov_character
     property start_character
     to\_dict() \rightarrow dict
          Get the configuration object as a dictionary
     property vocabulary
class aisquared.config.preprocessing.text.Steps.PadSequences(pad_character: int = 0, length: int =
                                                                     128, pad\_location: str = 'post',
                                                                     truncate\_location: str = 'post')
     Bases: BaseObject
     Text preprocessing object to pad sequences
     Example usage:
     >>> import aisquared
     >>> preprocesser = aisquared.config.preprocessing.text.TextPreprocessor()
     >>> preprocesser.add_step(
         aisquared.config.preprocessing.text.PadSequences()
     property length
     property pad_character
```

```
property pad_location
     to\_dict() \rightarrow dict
          Get the configuration object as a dictionary
     property truncate_location
class aisquared.config.preprocessing.text.Steps.RemoveCharacters(remove_digits: bool = True,
                                                                            remove_punctuation: bool =
                                                                            True)
     Bases: BaseObject
     Preprocessing step to remove characters from text
     Example usage:
     >>> import aisquared
     >>> preprocesser = aisquared.config.preprocessing.text.TextPreprocessor()
     >>> preprocesser.add_step(
          aisquared.config.preprocessing.text.RemoveCharacters()
     property remove_digits
     property remove_punctuation
     to\_dict() \rightarrow dict
          Get the configuration object as a dictionary
class aisquared.config.preprocessing.text.Steps.Tokenize(split_sentences: bool = False,
                                                                   split \ words: bool = True, token \ pattern:
                                                                   str = \langle x08 \rangle \langle w \rangle + \langle x08' \rangle
     Bases: BaseObject
     Preprocessing Step to tokenize text
     Example usage:
     >>> import aisquared
     >>> preprocesser = aisquared.config.preprocessing.text.TextPreprocessor()
     >>> preprocesser.add_step(
          aisquared.config.preprocessing.text.Tokenize()
     )
     property split_sentences
     property split_words
     to_dict() \rightarrow dict
          Get the configuration object as a dictionary
     property token_pattern
class aisquared.config.preprocessing.text.Steps.Trim
     Bases: BaseObject
     Text preprocessing class to trim whitespace from text
     Example usage:
```

```
>>> import aisquared
>>> preprocesser = aisquared.config.preprocessing.text.TextPreprocessor()
>>> preprocesser.add_step(
    aisquared.config.preprocessing.text.Trim()
)
```

```
to_dict() \rightarrow dict
```

Get the configuration object as a dictionary

aisquared.config.preprocessing.text.TextPreprocessing module

Bases: BaseObject

Preprocessor object for natural language

Example usage:

```
>>> import aisquared
>>> preprocesser = aisquared.config.preprocessing.text.TextPreprocessor()
>>> preprocesser.add_step(
    aisquared.config.preprocessing.text.Tokenize()
)
```

add_step(step)

Add a step to the preprocessor object

```
property step_dict
```

```
to\_dict() \rightarrow dict
```

Get the configuration object as a dictionary

aisquared.config.rendering package

Module contents

The aisquared.config.rendering subpackage contains objects for configuring how rendering of predictions is to occur.

Submodules

aisquared.config.rendering.BarChartRendering module

class aisquared.config.rendering.BarChartRendering.BarChartRendering(label: str, id: str,

chart_name: str,
container_id: str,
prediction_name_key: str,
prediction_value_key: str,
prediction_name_value:
str, display_legend: bool,
legend_icon: str,
labels_key: Optional[str]
= None, width: str =
'auto', height: str = 'auto',
xOffset: str = '0', yOffset:
str = '0', labels:
Optional[list] = None,
consolidate_rows: bool =
True)

Bases: BaseObject

Rendering class for rendering a Bar Chart

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.BarChartRendering(
    'my_label',
    'my_id',
    'my_bar_chart',
    'my_container_id',
    'name',
    'value',
    'name_value',
    True,
    'circle'
>>> my_obj.to_dict()
{'className': 'BarChartRendering',
    'label': 'my_label',
    'params': {'id': 'my_id',
    'chartName': 'my_bar_chart',
    'containerId': 'my_container_id',
    'displayLegend': True,
    'legendIcon': 'circle',
    'width': 'auto',
    'height': 'auto',
    'xOffset': '0',
    'yOffset': '0',
    'datasource': [{'labels': None,
        'labelsKey': None,
        'consolidateRows': True,
        'predictionNameKey': 'name',
        'predictionValueKey': 'value',
        'predictionNameValue': 'name_value'}]}}
```

 $\textbf{to_dict()} \rightarrow dict$

Get the configuration object as a dictionary

aisquared.config.rendering.ContainerRendering module

```
class aisquared.config.rendering.ContainerRendering.ContainerRendering(label: str, id: str, query\_selector: str, position: str = 'absolute', static\_position: Optional[str] = None, width: str = 'auto', height: str = 'auto', display: str = 'flex', xOffset: str = '0', yOffset: str = '0', orientation: str = 'column', css\_params: Optional[dict] = None)
```

Bases: BaseObject

Rendering for a container

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.ContainerRendering(
    'my container',
    'myContainerID',
    "[data-id='tabpanel-general']"
>>> my_obj.to_dict()
{'className': 'ContainerRendering',
'label': 'my container',
'params': {'id': 'myContainerID',
'width': 'auto',
'height': 'auto',
'display': 'flex',
'xOffset': '0',
'yOffset': '0',
'position': 'absolute',
'orientation': 'column',
'querySelector': "[data-id='tabpanel-general']",
'staticPosition': None}}
```

```
property display
property height
property id
property label
property orientation
property position
property query_selector
```

```
property static_position

to_dict() → dict
    Get the configuration object as a dictionary
property width
property xOffset
property yOffset
```

aisquared.config.rendering.DashboardReplacementRendering module

 $\textbf{class} \ a is quared. config. rendering. Dashboard Replacement Rendering. Dashboard Rendering.$

```
str,
where_replace
str
=
",
la-
bel:
str
=
")
```

Bases: BaseObject

Rendering for dashboard replacement

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.DashboardReplacementRendering(
    'test_anchor_selector'
)
>>> my_obj.to_dict()
{'className': 'DashboardReplacementRendering',
    'label': '',
    'params': {'anchorSelector': 'test_anchor_selector', 'whereReplace': ''}}
```

```
property anchor_selector
```

```
property label

to_dict() → dict
    Get the configuration object as a dictionary
property where_replace
```

aisquared.config.rendering.DocumentRendering module

Bases: BaseObject

Object which dictates how to render predictions on entire documents

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.DocumentRendering()
>>> my_obj.to_dict()
{'className': 'DocumentRendering',
'params': {'predictionKey': 'className',
'words': None,
'documents': None,
'includeProbability': False,
'probabilityKey': 'probability',
'underlineColor': 'blue',
'classes': None,
'thresholdKey': None,
'thresholdValue': None}}
```

```
property classes
property documents
property include_probability
property prediction_key
property probability_key
property threshold_key
property threshold_value
to_dict() → dict
Get the configuration object as a dictionary
```

Optional[Union[int,
float]] = None)

```
property underline_color
property words
```

aisquared.config.rendering.DoughnutChartRendering module

```
class aisquared.config.rendering.DoughnutChartRendering.DoughnutChartRendering(label: str, id:
```

```
str,
chart_name:
str,
container_id:
str, predic-
tion_name_key:
str, predic-
tion_value_key:
str, predic-
tion_name_value:
str, dis-
play_legend:
bool,
legend_icon:
str,
labels key:
Optional[str]
= None.
width: str =
'auto', height:
str = 'auto',
xOffset: str =
'0', yOffset:
str = '0',
labels: Op-
tional[list] =
None,
consoli-
date_rows:
bool = True)
```

Bases: BaseObject

Rendering class for rendering a Doughnut Chart

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.DoughnutChartRendering(
    'my_label',
    'my_id',
    'my_doughnut_chart',
    'my_container_id',
    'name',
    'value',
    'name_value',
    True,
```

(continues on next page)

(continued from previous page)

```
'circle'
)
>>> my_obj.to_dict()
{'className': 'DoughnutChartRendering',
'label': 'my_label',
'params': {'id': 'my_id',
'chartName': 'my_doughnut_chart',
'containerId': 'my_container_id',
'displayLegend': True,
'legendIcon': 'circle',
'width': 'auto',
'height': 'auto',
'xOffset': '0',
'yOffset': '0',
'datasource': [{'labels': None,
    'labelsKey': None,
    'consolidateRows': True,
    'predictionNameKey': 'name',
    'predictionValueKey': 'value',
    'predictionNameValue': 'name_value'}]}}
```

 $\textbf{to_dict()} \rightarrow dict$

Get the configuration object as a dictionary

aisquared.config.rendering.FilterRendering module

Bases: BaseObject

Object which dictates how predictions are to be passed to downstream analytics

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.FilterRendering(
    'inputs',
    'key',
    'gt',
    0.2
)
>>> my_obj.to_dict()
{'className': 'FilterRendering',
    'params': {'source': 'inputs', 'key': 'key', 'qualifier': 'gt', 'value': 0.2}}
```

```
property key
property qualifier
property source
```

```
to\_dict() \rightarrow dict Get the configuration object as a dictionary property value
```

aisquared.config.rendering.HTMLTagRendering module

Bases: BaseObject

Rendering for HTML tags

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.HTMLTagRendering(
    'my HTML tag',
    'MyHTMLTagRenderingID',
    'MyContainerID',
    'Example Text',
    'extra_tag',
    'append',
    'name_key'
    'value_key',
    'name_value'
>>> my_obj.to_dict()
{'className': 'HTMLTagRendering',
'label': 'my HTML tag',
'params': {'id': 'MyHTMLTagRenderingID',
'containerId': 'MyContainerID',
'htmlContent': 'Example Text',
'extraContentTag': 'extra_tag',
'injectionAction': 'append',
'predictionNameKey': 'name_key',
'predictionValueKey': 'value_key'
'predictionNameValue': 'name_value',
'content': ''}}
```

 $\textbf{to_dict()} \rightarrow dict$

Return the configuration object as a dictionary

aisquared.config.rendering.lmageRendering module

Get the configuration object as a dictionary

```
class aisquared.config.rendering.ImageRendering.ImageRendering(color: str = 'blue', thickness: str =
                                                                       '5', placement: str = 'bottomleft',
                                                                       include\_probability: bool = False,
                                                                       badge\_color: str = 'white',
                                                                       font_color: str = 'black', font_size:
                                                                       str = '5', classes: Optional[list] =
                                                                       None, threshold_key:
                                                                       Optional[str] = None,
                                                                       threshold_value:
                                                                       Optional[Union[int, float]] =
                                                                       None)
     Bases: BaseObject
     Object which dictates how to render images
     Example usage:
     >>> import aisquared
     >>> my_obj = aisquared.config.rendering.ImageRendering()
     >>> my_obj.to_dict()
     {'className': 'ImageRendering',
     'params': {'color': 'blue',
     'thickness': '5',
     'placement': 'bottomleft',
     'includeProbability': False,
     'badgeColor': 'white',
     'fontColor': 'black',
     'fontSize': '5',
     'classes': None,
     'thresholdKey': None,
     'thresholdValue': None}}
     property badge_color
     property classes
     property color
     property font_color
     property font_size
     property include_probability
     property placement
     property thickness
     property threshold_key
     property threshold_value
     to\_dict() \rightarrow dict
```

'0', yOffset: str = '0', labels: Optional[list] =

consolidate_rows: bool

(continues on next page)

None,

= True)

aisquared.config.rendering.LineChartRendering module

Bases: BaseObject

Rendering class for rendering a Line Chart

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.LineChartRendering(
    'my_label',
    'my_id',
    'my_line_chart',
    'my_container_id',
    'name',
    'value',
    'name_value',
    True,
    'circle',
    'labels'
>>> my_obj.to_dict()
{'className': 'LineChartRendering',
'label': 'my_label',
'params': {'id': 'my_id',
'chartName': 'my_line_chart',
'containerId': 'my_container_id',
'displayLegend': True,
'legendIcon': 'circle',
'width': 'auto',
'height': 'auto',
'x0ffset': '0',
'yOffset': '0',
'datasource': [{'labels': None,
    'labelsKey': 'labels',
    'consolidateRows': True,
```

(continued from previous page)

```
'predictionNameKey': 'name',
'predictionValueKey': 'value',
'predictionNameValue': 'name_value'}]}}
```

```
to\_dict() \rightarrow dict
```

Get the configuration object as a dictionary

aisquared.config.rendering.ObjectRendering module

Bases: BaseObject

Object which dictates how to render object detection in images

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.ObjectRendering()
>>> my_obj.to_dict()
{'className': 'ObjectRendering',
'params': {'color': 'blue',
'thickness': '5',
'placement': 'bottomleft',
'includeProbability': False,
'badgeColor': 'white',
'fontColor': 'black',
'fontSize': '5'}}
```

```
property badge_color
property color
property font_color
property font_size
property include_probability
property placement
property thickness
to_dict() → dict
Get the configuration object as a dictionary
```

Optional[list] = None, consolidate_rows: bool =

True)

aisquared.config.rendering.PieChartRendering module

class aisquared.config.rendering.PieChartRendering.PieChartRendering(label: str, id: str, $chart_name: str$, $container_id: str$, $prediction_name_key: str$, $prediction_value_key: str$, $prediction_value_key: str$, $prediction_name_value: str$, $display_legend: bool$, $legend_icon: str$, $labels_key: Optional[str] = None, width: <math>str = 'auto'$, height: str = 'auto', xOffset: str = '0', yOffset: str = '0'

Bases: BaseObject

Rendering class for rendering a Pie Chart

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.PieChartRendering(
        'my_label',
        'my_id',
        'my_doughnut_chart',
        'my_container_id',
        'name',
        'value',
        'name_value',
        True.
        'circle'
    )
>>> my_obj.to_dict()
{'className': 'PieChartRendering',
'label': 'my_label',
'params': {'id': 'my_id',
'chartName': 'my_doughnut_chart',
'containerId': 'my_container_id',
'displayLegend': True,
'legendIcon': 'circle',
'width': 'auto',
'height': 'auto',
'x0ffset': '0',
'yOffset': '0',
'datasource': [{'labels': None,
    'labelsKey': None,
    'consolidateRows': True,
    'predictionNameKey': 'name',
    'predictionValueKey': 'value',
    'predictionNameValue': 'name_value'}]}}
```

```
to\_dict() \rightarrow dict
Get the configuration object as a dictionary
```

aisquared.config.rendering.SOSRendering module

```
class aisquared.config.rendering.SOSRendering.SOSRendering(can_toggle: bool, label: str = ")
    Bases: BaseObject
    Rendering of an SOS dashboard
    Example usage:
```

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.SOSRendering(True)
>>> my_obj.to_dict()
{'className': 'SOSRendering', 'label': '', 'params': {'canToggle': True}}
```

```
property can_toggle
property label
to_dict() → dict
Get the configuration object as a dictionary
```

aisquared.config.rendering.TableRendering module

Bases: BaseObject

Class for rendering tables

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.TableRendering(
    'my table',
    'MyTableID',
    'MyContainerID',
    'name_key',
    'value_key',
    'name_values'
)
>>> my_obj.to_dict()
{'className': 'TableRendering',
    'label': 'my table',
    'params': {'id': 'MyTableID',
    'containerId': 'MyContainerID',
    'predictionNameKey': 'name_key',
```

(continues on next page)

(continued from previous page)

```
'predictionValueKey': 'value_key',
'predictionNameValues': 'name_values',
'tableName': ''}}
```

```
to\_dict() \rightarrow dict
```

Get the configuration object as a dictionary

aisquared.config.rendering.WordRendering module

Bases: BaseObject

Object for rendering badges on individual words

Example usage:

```
>>> import aisquared
>>> my_obj = aisquared.config.rendering.WordRendering()
>>> my_obj.to_dict()
{'className': 'WordRendering',
'params': {'wordList': 'input',
'resultKey': None,
'contentKey': None,
'badgeShape': 'star',
'badgeColor': 'blue',
'classes': None,
'thresholdKey': None,
'thresholdValue': None}}
```

```
property badge_color
property badge_shape
property classes
property content_key
property result_key
property threshold_key
property threshold_value
to_dict() → dict
Get the configuration object as a dictionary
property word_list
```

Submodules

aisquared.config.GraphConfiguration module

```
class aisquared.config.GraphConfiguration.GraphConfiguration(name: str, stage: str = 'experimental', version:

Optional[int] = None, description:

str = ", mlflow\_uri: Optional[str] =

None, mlflow\_user: Optional[str] =

None, mlflow\_token: Optional[str] =

None, owner: Optional[str] = None,

url: str = '*', auto\_run: bool =

False)
```

Bases: BaseObject

Configuration object for deploying a set of processing steps and/or analytics as a dependency graph

```
add_node(step: BaseObject, dependencies: Optional[Union[int, list]] = None) \rightarrow int Add a node to the configuration graph
```

Parameters

- **step** (aisquared configuration step) The step to add
- dependencies (int, list of int, or None) The ids of nodes which must be run before the added node

Returns

node_id - The integer id of the node that is added

Return type

int

property auto_run

```
compile(filename: Optional[str] = None, dtype: Optional[str] = None) <math>\rightarrow None
```

Compile the object into a '.air' file, which can then be dragged and dropped into applications using the AI Squared JavaScript SDK

Parameters

- **filename** (path-like or None (default None)) Filename to compile to. If None, defaults to '{NAME}.air', where {NAME} is the name of the analytic
- **dtype** (*str or None* (*default None*)) The datatype to use for the model weights when using a Keras model. If None, defaults to 'float32'

```
property description
```

```
get_filenames() \rightarrow list
```

Get filenames for all models in the configuration

```
property mlflow_token
property mlflow_uri
property mlflow_user
property name
```

```
property owner
property stage
to_dict() → dict
    Get the object as a dictionary
property url
property version
```

aisquared.config.ModelConfiguration module

class aisquared.config.ModelConfiguration.ModelConfiguration(name: str, harvesting_steps:

```
Optional[Union[BaseObject, list]] =
None, preprocessing_steps:
Optional[Union[BaseObject, list]] =
None, analytic:
Optional[Union[BaseObject, list]] =
None, postprocessing_steps:
Optional[Union[BaseObject, list]] =
None, rendering steps:
Optional[Union[BaseObject, list]] =
None, feedback_steps:
Optional[Union[BaseObject, list]] =
None, stage: str = 'experimental',
version: Optional[int] = None,
description: str = ", mlflow\_uri:
Optional[str] = None, mlflow\_user:
Optional[str] = None, mlflow\_token:
Optional[str] = None, owner:
Optional[str] = None, url: str = '*',
auto\_run: bool = False)
```

Bases: BaseObject

Configuration object for deploying a model or analytic

```
property analytic
property analytic_dict
property auto_run
```

compile($filename: Optional[str] = None, dtype: Optional[str] = None) <math>\rightarrow$ None

Compile the object into a '.air' file, which can then be dragged and dropped into applications using the AI Squared JavaScript SDK

Parameters

- **filename** (path-like or None (default None)) Filename to compile to. If None, defaults to '{NAME}.air', where {NAME} is the name of the analytic
- **dtype** (*str or None* (*default None*)) The datatype to use for the model weights. If None, defaults to 'float32'

property description

```
property feedback_dict
property feedback_steps
get_model_filenames() \rightarrow list
    Get filenames for all models in the configuration
property harvester_dict
property harvesting_steps
property mlflow_token
property mlflow_uri
property mlflow_user
property name
property owner
property postprocesser_dict
property postprocessing_steps
property preprocesser_dict
property preprocessing_steps
property render_dict
property rendering_steps
property stage
\textbf{to\_dict()} \rightarrow dict
    Get the object as a dictionary
property url
property version
```

aisquared.logging package

Module contents

The aisquared.logging subpackage contains utilities for performing experiments within aisquared.

This functionality is inhereted from MLFlow. Please see the MFLow documentatation at https://mlflow.org.

aisquared.platform package

Module contents

Utilities for interacting with the AI Squared Platform.

The primary class within this subpackage is the *AISquaredPlatformClient* class, which has the capabilities to interact with much of the functionality in the AI Squared platform. For more information about this class, please see its documentation.

Submodules

aisquared.platform.AISquaredPlatformClient module

Bases: object

Client for interacting with the AI Squared platform programmatically

When using the client for the first time, it is important to run the *client.login()* method. When doing so, the client will ask for any required information interactively.

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> # If you have never logged in before, run the following code:
>>> client.login()
>>> # Test connection
>>> client.test_connection()
True
```

Add users to a group

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.add_users_to_group('group_id', ['user_id_1', 'user_id_2'])
True
```

Parameters

- **group_id** (str) The group to add the users to
- user_ids (list of str) The IDs of the users to add
- **port** (*int* (*default* 8086)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

success - Returns True if operation was successful

Return type

bool

property base_url: str

The base URL associated with the client

 $create_group(display_name: str, role_id: str, port: int = 8086, use_port: Optional[bool] = None) \rightarrow dict$ Create a group in the platform

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.create_group(
    'group display name',
    'role_id'
)
*dictionary containing group information*
```

Parameters

- **display_name** (str) The display name of the group
- **role_id** (*str*) The role ID for the group
- **port** (*int* (*default* 8086)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

group_info - Metadata about the created group

Return type

dict

create_user($user_name$: str, $given_name$: str, $family_name$: str, email: str, $role_id$: str, active: bool = True, $middle_name$: Optional[str] = None, $company_id$: Optional[str] = None, password: Optional[str] = None, port: int = 8085, use_port : Optional[bool] = None) \rightarrow dict

Create a user within the platform

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.create_user(
    'user name',
    'given_name',
    'family_name',
    'user_email',
    'role_id'
)
*Dictionary with user information*
```

Parameters

- **user_name** (*str*) The display name of the user
- **given_name** (*str*) The user's first name
- **family_name** (*str*) The user's last name

- **email** (str) The user's email
- role_id (str) The ID of the role to be given to the user
- active (bool (default True)) Whether the user is active
- middle_name (str or None (default None)) The user's middle name
- company_id (str or None (default None)) The user's company ID
- password (str or None (default None)) The user's password
- **port** (*int* (*default* 8085)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

user_data - Metadata about the user

Return type

dict

 $delete_group(group_id, port=8086, use_port: Optional[bool] = None) \rightarrow bool$

Delete a group from the platform

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.delete_group('group_id')
True
```

Parameters

- **group_id** (*str*) The ID of the group to delete
- **port** (*int* (*default* 8086)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

result - Returns True if successful

Return type

bool

 $delete_model(id: str, port: int = 8080, use_port: Optional[bool] = None) \rightarrow bool$

Delete a model

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.delete_model('model_id')
True
```

Parameters

• **id** (*str*) – The ID for the model

- **port** (*int* (*default* 8080)) The API port for the model. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

success – Whether the action was successful

Return type

bool

 $delete_user(user_id: str, port: int = 8085, use_port: Optional[bool] = None) \rightarrow bool$

Delete a user from the system

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.delete_user('user_id')
True
```

Parameters

- user_id (str) The user's ID
- **port** (*int* (*default* 8085)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

result – Returns True if the call is successful

Return type

bool

get_group($group_id: str, port: int = 8086, use_port: Optional[bool] = None) <math>\rightarrow$ dict Retrieve information about a group

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.get_group('group_id')
*dictionary containing group data*
```

Parameters

- $group_id(str)$ The ID of the group requested
- **port** (*int* (*default* 8086)) The API port for the call. This can be handled automatically by the platform ALB
- **use_port** (*bool or None* (*default None*)) Whether to use port in URL formatting. If None, defaults to class value

Returns

group_info – The information about the group

Return type

dict

get_group_id_by_name($group_name: str, port: int = 8083, use_port: Optional[bool] = None) <math>\rightarrow$ str Get the ID of a group by searching for its display name

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.get_group_id_by_name('Group Name')
*group_id*
```

Parameters

- **group_name** (str) The display name of the group
- **port** (*int* (*default* 8083)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

```
group_id – The ID of the group
```

Return type

str

 $get_model(id: str, port: int = 8080, use_port: Optional[bool] = None) \rightarrow dict$ Retrieve a model configuration

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.get_model('model_id')
*JSON Response including model data and metadata*
```

Parameters

- **id** (*str*) The ID for the model
- port (int (default 8080)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

model – Metadata about the model coupled with the model's configuration information

Return type

dictionary

 $get_model_id_by_name(model_name: str, port: int = 8080, use_port: Optional[bool] = None) \rightarrow str$ Retrieve a model's ID using the name of the model

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.get_model_id_by_name('my_awesome_model')
*model_id*
```

Parameters

- model_name (str) The name of the model
- **port** (*int* (*default* 8080)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

```
model id – The model's ID
```

Return type

str

get_role_id_by_role_name($role_name$: str, port: $int = 8086, use_port$: Optional[bool] = None) \rightarrow str Get the ID of a role by searching for its display name

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.get_role_id_by_role_name('Role Name')
*role_id*
```

Parameters

- role_name (str) The name of the role
- **port** (*int* (*default* 8086)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

```
role_id - The ID of the role
```

Return type

str

get_user($user_id: str, port: int = 8085, use_port: Optional[bool] = None) <math>\rightarrow$ dict Retrieve a user's information from the platform

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.get_user('user_id')
*dictionary with results*
```

Parameters

- user_id (str) The ID of the user
- **port** (*int* (*default* 8085)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

user_info - The information about the user

Return type

dict

get_user_id_by_name($name: str, port: int = 8080, use_port: Optional[bool] = None) <math>\rightarrow$ str Get a user's ID from their display name

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.get_user_id_by_name('User Name')
*user_id*
```

Parameters

- **name** (*str*) The display name of the user
- **port** (*int* (*default* 8080)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

id – The ID of the user

Return type

str

property headers

Headers used for authentication with the AI Squared Platform

```
\label{list_group_users} \textbf{list\_group\_} id: \textit{str}, \textit{as\_} df: \textit{bool} = \textit{True}, \textit{port: int} = 8083, \textit{use\_} port: \textit{Optional[bool]} = \textit{None}) \\ \rightarrow \textbf{Union[DataFrame, dict]}
```

List users in a group

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.list_group_users('group_id')
*DataFrame with results*
```

Parameters

- **group_id** (str) The ID for the group
- as_df (bool (default True)) Whether to return the response as a pandas DataFrame
- **port** (*int* (*default* 8083)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

users – The response from the API

Return type

pandas DataFrame or dictionary

list_groups(max_count : int = 100, as_df : bool = True, port: int = 8083, use_port : Optional[bool] = None) \rightarrow Union[DataFrame, dict]

List all groups

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.list_groups()
*DataFrame with results*
```

Parameters

- max_count (int (default 100)) The maximum number of groups to return
- as_df (bool (default True)) Whether to return the result as a pandas DataFrame
- **port** (*int* (*default* 8083)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

groups - The response from the API

Return type

pandas DataFrame or dictionary

list_model_feedback($model_id: str, limit: int = 10, as_df: bool = True, port: int = 8080, use_port: Optional[bool] = None) <math>\rightarrow$ Union[dict, DataFrame]

List feedback on a model

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.list_model_feedback('model_id')
*DataFrame with Results*
```

Parameters

- model_id (str) The ID of the model
- limit (int (default 10)) The maximum number of feedback items to return
- port (int (default 8080)) The API port to use. This can be handled automatically by the platform ALB
- use_port(bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

feedback – The feedback

Return type

dict or pandas DataFrame

list_model_prediction_feedback($model_id: str, as_df: bool = True, port: int = 8080, use_port: Optional[bool] = None) <math>\rightarrow$ Union[dict, DataFrame]

List all feedback for a model

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.list_model_prediction_feedback('model_id')
*DataFrame with Results*
```

- model_id (str) The ID of the model requested
- as_df (bool (default True)) Whether to return the results as a pandas DataFrame
- **port** (*int* (*default* 8080)) The API port to use. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

results – The results from the platform

Return type

dict or pandas DataFrame

list_model_usage_metrics($model_id$: str, period: str = 'hourly', as_df : bool = True, port: int = 8080, use_port : Optional[bool] = None) \rightarrow Union[dict, DataFrame]

Get usage metrics for a model

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.get_model_usage_metrics('model_id')
*DataFrame with results*
```

Parameters

- model_id (str) The ID of the model
- **period** (str (default 'hourly')) The period to group metrics into
- as_df (bool (default True)) Whether to return results as a pandas DataFrame
- **port** (*int* (*default* 8080)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

results – The results from the platform

Return type

pandas DataFrame or dict

list_model_users($id: str, as_df: bool = True, port: int = 8080, use_port: Optional[bool] = None) <math>\rightarrow$ Union[DataFrame, dict]

List users for a model

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.list_model_users('model_id')
*DataFrame with results*
```

- **id** (*str*) The ID for the model
- as_df (bool (default True)) Whether to return the response as a Pandas DataFrame
- **port** (*int* (*default* 8080)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

model users – The users for the model

Return type

pandas DataFrame or dictionary

 $\label{list_models} \textbf{list_models}(\textit{as_df: bool} = \textit{True}, \textit{port: int} = 8080, \textit{use_port: Optional[bool]} = \textit{None}) \rightarrow \text{Union[DataFrame, dict]}$

List models within the platform

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.list_models()
*DataFrame with results*
```

Parameters

- as_df (bool (default True)) Whether to return the response as a pandas DataFrame
- **port** (*default None*) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

models – The models

Return type

pandas DataFrame or dictionary

list_prediction_feedback($prediction_id: str, as_df: bool = True, port: int = 8080, use_port: Optional[bool] = None) <math>\rightarrow$ Union[DataFrame, dict]

List prediction feedback given a prediction ID

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.list_prediction_feedback('prediction_id')
*DataFrame with results*
```

- prediction_id (str) The prediction ID
- as_df (bool (default True)) Whether to return the results as a pandas DataFrame
- **port** (*int* (*default* 8080)) The API port to use. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

results – The results from the platform

Return type

pandas DataFrame or dict

 $list_roles(as_df: bool = True, port: int = 8086, use_port: Optional[bool] = None) \rightarrow Union[DataFrame, dict]$

List the roles available in the platform

Example usage:

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.list_roles()
*DataFrame with results*
```

Parameters

- as_df (bool (default True)) Whether to return the results as a pandas DataFrame
- **port** (*int* (*default* 8086)) The API port for the call. This can be handled automatically by the platform ALB
- **use_port** (*bool or None* (*default None*)) Whether to use port in URL formatting. If None, defaults to class value

Returns

roles - The roles

Return type

pandas DataFrame or dict

 $\label{list_user_usage_metrics} \textbf{(user_id: str, period: str = 'hourly', as_df: bool = True, port: int = 8080, \\ use_port: Optional[bool] = None) \rightarrow \textbf{Union[dict, DataFrame]}$

Get usage metrics for a user

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.get_user_usage_metrics('user_id')
*DataFrame with results*
```

Parameters

- **user_id** (*str*) The ID of the user
- **period** (*str* (*default* '*hourly*')) The period to group metrics into

- as_df (bool (default True)) Whether to return results as a pandas DataFrame
- **port** (*int* (*default* 8080)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

results – The results from the platform

Return type

pandas DataFrame or dict

```
\label{list_users} \textbf{(max\_count: int = 100, as\_df: bool = True, port: int = 8080, use\_port: Optional[bool] = None)} \\ \rightarrow \textbf{Union[DataFrame, dict]}
```

List all users

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.list_users()
*DataFrame with results*
```

Parameters

- max_count (int (default 100)) The maximum number of users to return
- as_df (bool (default True)) Whether to return the data as a Pandas DataFrame
- **port** (*int* (*default* 8080)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

users – The response from the API

Return type

pandas DataFrame or dictionary

```
login(url: Optional[str] = None, port: int = 8080, username: Optional[str] = None, password: Optional[str] = None, use_port: Optional[bool] = None) <math>\rightarrow None
```

Log in to the platform programmatically. If no url, username, or password are provided, logs in interactively

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.login()
Enter URL: https://platform.squared.ai
Enter Username: your.email@your_domain.com
Enter Password: <hidden>
```

Parameters

- url (str or None (default None)) The URL for the platform API
- port(int or None (default 8080)) The API port for the call. This can be handled automatically by the platform ALB

- username (str or None (default None)) The username
- password (str or None (default None)) The password
- use_port(bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

property password: str

The password associated with the client

 $remove_users_from_group(group_id: str, user_ids: list, port: int = 8086, use_port: Optional[bool] = None) \rightarrow bool$

Remove users from a group

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.remove_users_from_group('group_id', ['user_id_1', 'user_id_2'])
True
```

Parameters

- **group_id** (str) The ID of the group
- user_ids (list of str) The IDs of the users to remove
- **port** (*int* (*default* = 8086)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

success - Returns True if successful

Return type

bool

```
share_model_with_group(model_id: str, group_id: str, port: int = 8080, use\_port: Optional[bool] = None) \rightarrow bool
```

Share a model with a group

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.share_model_with_group('model_id', 'group_id')
True
```

Parameters

- $model_id(str)$ The ID for the model to be shared
- **group_id** (*str*) The ID for the group to be shared with. This can be handled automatically by the platform ALB
- **port** (*int* (*default* 8080)) The API port to use. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

success - Returns True if successful

Return type

bool

 $share_model_with_user(model_id: str, user_id: str, port: int = 8080, use_port: Optional[bool] = None)$ $\rightarrow bool$

Share a model with a user

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.share_model_with_user('model_id', 'user_id')
True
```

Parameters

- model_id (str) The ID for the model
- user_id (str) The ID for the user
- **port** (*int* (*default* 8080)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

success - Whether the action was successful

Return type

bool

 $test_connection(port: int = 8080, use_port: Optional[bool] = None) \rightarrow bool$

Test whether there is a healthy connection to the platform

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.test_connection()
True
```

Parameters

- **port** (*int* (*default* 8080)) The API port for the call. This can be handled automatically by the platform ALB
- use_port(bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

success - True if connection was successful

Return type

bool

property token: str

The token associated with the client

```
unshare_model_with_group(model_id: str, group_id: str, port: int = 8080, use\_port: Optional[bool] = None) <math>\rightarrow bool
```

Unshare a model with a group

```
>>> import aisquared
>>> client = aisquared.client.AISquaredPlatformClient()
>>> client.unshare_model_with_group('model_id', 'group_id')
True
```

Parameters

- $model_id(str)$ The ID of the model
- **group_id** (str) The ID of the group
- **port** (*int* (*default* 8080)) The API port to use. This can be handled automatically by the platform ALB
- **use_port** (*bool* or *None* (*default None*)) Whether to use port in URL formatting. If None, defaults to class value

Returns

success - Returns True if successful

Return type

bool

```
unshare\_model\_with\_user(model\_id: str, user\_id: str, port: int = 8080, use\_port: Optional[bool] = None) \rightarrow bool
```

Unshare a model with a user

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.unshare_model_with_user('model_id', 'user_id')
True
```

Parameters

- model_id (str) The ID for the model
- **user_id** (*str*) The ID for the user
- **port** (*int* (*default* 8080)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

success - Whether the action was successful

Return type

bool

update_group($group_id$: str, $display_name$: str, $role_id$: str, port: int = 8086, use_port : Optional[bool] = None) \rightarrow bool

Update information about a group

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.update_group(
    'group_id',
    'group display name',
    'role_id'
)
True
```

- **group_id** (str) The ID of the group to update
- **display_name** (str) The display name of the group
- role_id (str) The ID of the role for the group
- **port** (*int* (*default* 8086)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

success - Returns True if successful

Return type

bool

update_user($user_id: str, user_name: str, given_name: str, family_name: str, email: str, role_id: str, active: bool = True, middle_name: Optional[str] = None, company_id: Optional[str] = None, password: Optional[str] = None, port: int = 8085, use_port: Optional[bool] = None) <math>\rightarrow$ bool

Update information about a user

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.update_user(
    'user_id',
    'user name',
    'given_name',
    'family_name',
    'user_email',
    'role_id'
)
True
```

Parameters

- **user_id** (*str*) The ID of the user to update
- **user_name** (*str*) The display name of the user
- **given_name** (str) The first name of the user
- family_name (str) The last name of the user
- **email** (str) The user's email
- **role_id** (*str*) The ID of the user's role

- active (bool (default True)) Whether the user is active
- middle_name (str or None (default None)) The user's middle name
- company_id(str or None (default None)) The user's company ID
- password (str or None (default None)) The user's password
- **port** (*int* (*default* 8085)) The API port for the call. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

success - Returns True if update is successful

Return type

bool

upload_model($model_file: str, port: int = 8081, use_port: Optional[bool] = None) <math>\rightarrow$ str Upload a model to the platform

```
>>> import aisquared
>>> client = aisquared.platform.AISquaredPlatformClient()
>>> client.upload_model('my_model_filename.air')
True
```

Parameters

- model_file (path or path-like) The path to the model file
- **port** (*int* (*default* 8081)) The API port to use. This can be handled automatically by the platform ALB
- use_port (bool or None (default None)) Whether to use port in URL formatting. If None, defaults to class value

Returns

successful – Whether the action was successful

Return type

bool

property use_port

property username: str

The username associated with the client

aisquared.serving package

Module contents

The aisquared serving package contains utilities to serve models to a local REST endpoint.

Here is an example of how to serve a simple keras model using these utilities:

And to retrieve predictions from the model:

```
>>> # From a separate terminal, assume data is already loaded
>>> from aisquared import serving
>>> serving.get_remote_predictions(data) # Do not need to change host or port if_
--predicting from the same machine
*predictions*
```

Submodules

aisquared.serving.deploy_model module

```
aisquared.serving.deploy_model.deploy_model(saved_model: str, model_type: str, host: str = '127.0.0.1', port: int = 2244, custom_objects: Optional[dict] = None, additional functions file: Optional[str] = None)
```

Deploy a model to a Flask server on the specified host

Parameters

- **saved_model** (*Path-like*) The path to the saved model directory or model file
- model_type (str) The type of model
- host (str (default '127.0.0.1')) The host to deploy to
- port (int (default 2244)) The port to deploy to
- **custom_objects**(*dict or None* (*default None*)) Any custom objects to load when using a BeyondML model
- additional_functions_file (file-like or None (default None)) File name containing additional functions (which have to be named preprocess and postprocess, if created) that are used during the prediction process

aisquared.serving.deploy_model.load_beyondml_model(model: str, custom_objects: dict)

Load a BeyondML model with custom objects

aisquared.serving.get_remote_prediction module

```
aisquared.serving.get_remote_prediction.get_remote_prediction(data: Union[dict, str, ndarray, list], host: str = '127.0.0.1', port: int = 2244) \rightarrow list
```

Send data to use for prediction

Parameters

- data (dict, str, np.ndarray, or list) The data to be predicted on
- host (str (default '127.0.0.1')) The host to use
- port (int (default '2244')) The port to use

Notes

- If data is a dictionary, it is expected to already be correctly formatted
- If data is a string, it is expected to already be correctly formatted

Returns

predictions – The predictions from the deployed model

Return type

list

aisquared.utils package

Module contents

Additional utilities to use with the *aisquared* package. These utilities currently consist of two functions, the *mimic_model* and *get_model* functions. They utilize functionality that exists in our open source package BeyondML to train teacher-student models

To see in-depth examples of how to use these functions, please visit our GitHub repository at https://github.com/AISquaredInc/MimicModelExamples

Submodules

aisquared.utils.utils module

```
aisquared.utils.utils.get_model(model\_type: str, input\_shape: Union[int, tuple], num\_outputs: int, output\_activation: <math>str, size: str = 'small', vocab\_size: Union[None, int] = None)
```

Get a pre-configured model for different use cases

Parameters

- model_type (str) Either 'cv', 'nlp_embedding', or 'fc', defining the model type
- input_shape (int or tuple of int) The input shape to the model
- **num_outputs** (*int*) The output shape of the model

- output_activation (str or keras activation function) The activation of the final layer of the model
- size (str (default 'small')) One of either 'small', 'medium', or 'large'
- **vocab_size** (*str or None* (*default None*)) Size of the vocab, if model_type is 'nlp_embedding'

Returns

model – The model

Return type

TensorFlow Keras model

aisquared.utils.utils.mimic_model (trained_model: BaseEstimator, nnet: Model, training_data: ndarray, test_data: ndarray, test_labels: ndarray, problem_type: str, loss: str, metrics: Union[str, list], optimizer: str, mimic_proba: bool = False, retention: float = 0.9, batch_size: int = 32, epochs: int = 100, starting_sparsification: int = 0, max_sparsification: int = 99, sparsification_rate: int = 5) \rightarrow Model

Train a sparse neural network to mimic a scikit-learn model

Parameters

- **trained_model** (*sklearn model*) The model that is already trained
- nnet (TensorFlow keras Model) The neural network to train to mimic the trained model
- training_data (array or array-like) The input data that was used to train the trained model
- **test_data** (*array or array-like*) The input data to be used for testing
- test_labels (array or array-like) The output data used in testing
- **problem_type** (*str*) The type of problem, either 'classification' or 'regression'
- loss (str or keras loss function) The loss to use
- metrics (str, function or list of str, function) Metrics to measure
- **optimizer** (*str or keras optimizer*) The optimizer to use
- mimic_proba (bool (default False)) For classification, mimic the probability outputs
- **retention** (*float* (*default* 0.9)) The retention of performance to allow further pruning
- batch_size (int (default 32)) The batch size to use while training
- **epochs** (*int* (*default* 100)) The number of epochs (if early stopping is not met beforehand)
- starting_sparsification (int (default 0)) The starting model sparsification
- max_sparsification (int (default 99)) The maximum sparsification to allow
- sparsification_rate (int (default 5)) The sparsification rate when invoked

Returns

nnet – The trained model

Return type

TensorFlow keras Model

TWO

CHANGELOG

• Version 0.1.3

- Added *flags* parameter to *TextHarvester* using regular expression harvesting
- Deleted model_feedback parameter in ModelConfiguration object and included functionality in feedback_steps parameter
- Changed format parameter to header for both deployed analytics
- Added feedback and stages to *DocumentPredictor* and *ImagePredictor* objects
- Non-API changes for ALLOWED_STAGES
- Fixed bugs preventing Windows users from importing the package
- Updated ModelConfiguration to include url parameter
- Changed default tokenization string

• Version 0.2.0

- Moved preprocessing steps under subpackages for specific kinds of preprocessing steps
- Cleaned up documentation to render within programmatic access environments
- Added aisquared.logging subpackage

- Created InputHarvester

- * Allows for harvesting of input text, images, and tabular data
- Created the aisquared.serving subpackage, specifically the deploy_model and get_remote_prediction functions
- Created the GraphConfiguration class
- Added auto-run parameter to ModelConfiguration and GraphConfiguration classes

- Created the aisquared CLI with the following commands:

- * aisquared deploy, which deploys a model locally
- * aisquared predict, which predicts using a local JSON file
- * aisquared airfiles, which contains the subcommands list, delete, download, and upload
- Changed all classes within aisquared.config.analytic to accept 'tabular' as an input type
- Removed aisquared.logging and aisquared.remote from top-level imports
- Added round parameter to Regression postprocesser
- Removed *DocumentPredictor* and *ImagePredictor* classes

- Removed *ChainRendering* class
- Created FilterRendering class
- Altered QUALIFIERS
- Added advanced rendering parameters to rendering objects
- Removed *logging* and *remote* subpackages from top-level *aisquared* import

Version 0.2.1

- Added the S3Connector class to the analytics subpackage, which allows download of an analytic directly from S3
- Updated the documentation and added the docs subdirectory for hosting the documentation on GitHub Pages

• Version 0.2.2

- Fixed bug in to_dict method within ObjectRendering class
- Fixed bug in name of MultiplyValue step
- Fixed bug in datatype checking for text harvester
- Added body_only parameter to TextHarvester
- Added 'underline' to possible badges
- Added threshold_key and threshold_values to relevant rendering classes
- Added *Trim* text preprocessing class
- Added CustomObject in the base package to allow for creation of custom classes
- Added keyword harvesting capabilities
- Added utils subpackage with capabilities to mimic a trained sklearn model
- Small documentation changes
- Changed the required imports for the package to streamline installation process, and created two installation options *aisquared* and *aisquared*[full]

• Version 0.2.3

- Added functionality to add custom preprocessing and postprocessing functions to the model deployment pipeline
- Added all parameter to LocalAnalytic class
- Changed under-the-hood functionality of mimic model function in line with updates to BeyondML
- Altered the ReverseMLWorkflow analytic
- Added the BarChartRendering, ContainerRendering, DashboardReplacementRendering, DoughnutChartRendering, HTMLTagRendering, LineChartRendering, PieChartRendering, SOSRendering, and TableRendering rendering classes
- Added the ${\it QueryParameterHarvester}$ harvester class
- Added the *limit* parameter to the TextHarvester class

• Version 0.3.0

- Added type hinting to documentation strings
- Revamped documentation to use Sphinx

• Version 0.3.1

- Changed Python type hints to allow for backwards compatibility with older versions of Python

• Version 0.3.2

- Added functionality to the AISquaredPlatformClient
- Added top_level_kwargs parameter to the CustomObject class
- Added DashboardRendering class
- Removed 'px' from default values in ImageRendering and ObjectRendering classes
- Added functionality for creating, updating, and deleting users to AISquaredPlatformClient
- Added functionality for creating, updating, and delting groups to AISquaredPlatformClient
- Fixed bug related to requiring *auto_run* parameter to be string (fix involves casting as string)
- Altered schemas for different "Chart" Rendering classes to conform to JavaScript standards
- Streamlined the ModelConfiguration class to allow a more functional interface to build .air files
- Updated ContainerRendering class with parameters for position and static_position
- Updated across-the-board functionality of the AISquaredPlatformClient

• Version 0.3.3

- Updated functionality of the AISquaredPlatformClient to interact directly with the platform ALB
- Changed function names in support of change from MANN to BeyondML
- Added documentation surrounding global configuration objects
- Removed redundant additional dependencies

• Version 0.3.4

- Added support for custom CSS strings to appropriate rendering classes
- Refactored AISquaredPlatformClient to import functions from support files
- Fixed documentation errors for the documentation site
- Checked whether responses returned OK status code rather than 200
- Moved *CustomObject* to *aisquared.config* from *aisquared.base*
- Changed endpoint used to list platform users
- Fixed response behaviors where no data was returned from AISquaredPlatformClient

PYTHON MODULE INDEX

```
а
                                                                                                                              aisquared.config.postprocessing.Regression,
aisquared, 3
                                                                                                                              aisquared.config.preprocessing, 16
aisquared.base, 3
                                                                                                                              aisquared.config.preprocessing.image, 16
aisquared.base.BaseObject, 3
                                                                                                                              aisquared.config.preprocessing.image.ImagePreprocessing.
aisquared.base.CustomObject, 4
aisquared.base.rendering, 4
                                                                                                                              aisquared.config.preprocessing.image.Steps,
aisquared.base.stages, 4
                                                                                                                                                    17
aisquared.config, 4
                                                                                                                              aisquared.config.preprocessing.tabular, 19
aisquared.config.analytic, 5
                                                                                                                              aisquared.config.preprocessing.tabular.Steps,
aisquared.config.analytic.DeployedAnalytic,5
aisquared.config.analytic.DeployedModel, 6
                                                                                                                              aisquared.config.preprocessing.tabular.TabularPreprocessing.
aisquared.config.analytic.LocalAnalytic, 6
                                                                                                                                                   21
aisquared.config.analytic.LocalModel, 7
                                                                                                                              aisquared.config.preprocessing.text, 21
aisquared.config.analytic.ReverseMLWorkflow,
                                                                                                                              aisquared.config.preprocessing.text.Steps, 21
                                                                                                                              aisquared.config.preprocessing.text.TextPreprocessing,
aisquared.config.feedback, 8
aisquared.config.feedback.BinaryFeedback, 8
                                                                                                                              aisquared.config.rendering, 24
aisquared.config.feedback.ModelFeedback,9
                                                                                                                              aisquared.config.rendering.BarChartRendering,
aisquared.config.feedback.MulticlassFeedback,
a is quared. config. feedback. Qualitative Feedback, \\a is quared. config. rendering. Container Rendering, \\a is quared. \\a is
                                                                                                                              aisquared.config.rendering.DashboardReplacementRendering,
aisquared.config.feedback.RegressionFeedback,
                                                                                                                              aisquared.config.rendering.DocumentRendering,
aisquared.config.feedback.SimpleFeedback, 11
aisquared.config.GraphConfiguration, 38
                                                                                                                              aisquared.config.rendering.DoughnutChartRendering,
aisquared.config.harvesting, 11
aisquared.config.harvesting.ImageHarvester,
                                                                                                                              aisquared.config.rendering.FilterRendering,
aisquared.config.harvesting.InputHarvester,
                                                                                                                              aisquared.config.rendering.HTMLTagRendering,
aisquared.config.harvesting.QueryParameterHarvester, 31
                                                                                                                              aisquared.config.rendering.ImageRendering, 32
                                                                                                                              aisquared.config.rendering.LineChartRendering,
aisquared.config.harvesting.TextHarvester, 13
aisquared.config.ModelConfiguration, 39
                                                                                                                              aisquared.config.rendering.ObjectRendering,
aisquared.config.postprocessing, 14
aisquared.config.postprocessing.BinaryClassification, 34
                                                                                                                              aisquared.config.rendering.PieChartRendering,
aisquared.config.postprocessing.MulticlassClassification,
                                                                                                                              aisquared.config.rendering.SOSRendering,36
a is quared. config. postprocessing. Object Detection, squared. config. rendering. Table Rendering, 36 and squared. The squared config. Table Rendering and squared config. Table Rendering config. Table Rendering config. Table Rendering config. Table Rendering config. Table Render
                                                                                                                              aisquared.config.rendering.WordRendering, 37
                     15
```

66 Python Module Index

INDEX

A	aisquared.config.analytic.ReverseMLWorkflow
add_node() (aisquared.config.GraphConfiguration.Graph method), 38	Configuration 7 aisquared.config.feedback
<pre>add_question() (aisquared.config.feedback.ModelFeedback</pre>	ack.Mmaduleed8ack aisquared.config.feedback.BinaryFeedback
add_question() (aisquared.config.feedback.QualitativeFinethod), 10	eedbath du the stative feedback aisquared.config.feedback.ModelFeedback
add_step() (aisquared.config.preprocessing.image.Image method), 16	Preprocessing ImagePreprocessor aisquared.config.feedback.MulticlassFeedback
add_step() (aisquared.config.preprocessing.tabular.Tabu method), 21	larPreprodulesing.TabularPreprocessor aisquared.config.feedback.QualitativeFeedback
add_step() (aisquared.config.preprocessing.text.TextPrepmethod), 24	processing. Text Preprocessor aisquared.config.feedback.RegressionFeedback
add users to group()	module, 10
(aisquared.platform.AISquaredPlatformClient.AI	saisquared contignfeedback.SimpleFeedback module,11
AddValue (class in aisquared.config.preprocessing.image.S	_{tals} quared.config.GraphConfiguration
17	module, 38
aisquared	aisquared.config.harvesting
module, 3	module, 11
aisquared.base	aisquared.config.harvesting.ImageHarvester
module, 3	module, 11
aisquared.base.BaseObject	aisquared.config.harvesting.InputHarvester
module, 3	module, 12
aisquared.base.CustomObject	aisquared.config.harvesting.QueryParameterHarvester
module, 4	module, 12
aisquared.base.rendering	aisquared.config.harvesting.TextHarvester
module, 4	module, 13
aisquared.base.stages	aisquared.config.ModelConfiguration
module, 4	module, 39
aisquared.config	aisquared.config.postprocessing
module, 4	module, 14
<pre>aisquared.config.analytic module, 5</pre>	aisquared.config.postprocessing.BinaryClassification module, 14
<pre>aisquared.config.analytic.DeployedAnalytic module, 5</pre>	aisquared.config.postprocessing.MulticlassClassification module, 14
aisquared.config.analytic.DeployedModel module, 6	<pre>aisquared.config.postprocessing.ObjectDetection module, 15</pre>
aisquared.config.analytic.LocalAnalytic	aisquared.config.postprocessing.Regression
module, 6	module, 15
aisquared.config.analytic.LocalModel	aisquared.config.preprocessing
module, 7	module, 16

module, 41

aisquared.config.preprocessing.image module, 16	aisquared.serving module, 57
aisquared.config.preprocessing.image.ImagePre $$ module, 16	parissamessained.serving.deploy_model module,58
aisquared.config.preprocessing.image.Steps	aisquared.serving.get_remote_prediction
module, 17	module, 59
aisquared.config.preprocessing.tabular	aisquared.utils
module, 19	module, 59
aisquared.config.preprocessing.tabular.Steps module, 19	aisquared.utils.utils module, 59
aisquared.config.preprocessing.tabular.Tabula	
module, 21	aisquared.platform.AISquaredPlatformClient),
aisquared.config.preprocessing.text	41
module, 21	all (aisquared.config.analytic.LocalAnalytic.LocalAnalytic
aisquared.config.preprocessing.text.Steps	property), 6
module, 21	analytic (aisquared.config.ModelConfiguration.ModelConfiguration
aisquared.config.preprocessing.text.TextPrepr	
	-1 1 1/
module, 24	analytic_dict (aisquared.config.ModelConfiguration.ModelConfiguration)
aisquared.config.rendering	property), 39
module, 24	anchor_selector(aisquared.config.rendering.DashboardReplacementRe
aisquared.config.rendering.BarChartRendering	property), 27
module, 24	${\tt attributes} \ (a is quared. config. harvesting. Query Parameter Harvester. Query Parameter Harvester. Query Parameter Harvester (All Parameter) and the property of the p$
aisquared.config.rendering.ContainerRendering	
module, 26	auto_run(aisquared.config.GraphConfiguration.GraphConfiguration
$\verb aisquared.config.rendering.DashboardReplaceme \\$	* -* *·
module, 27	auto_run(aisquared.config.ModelConfiguration.ModelConfiguration
<pre>aisquared.config.rendering.DocumentRendering</pre>	property), 39
module, 28	
$\verb aisquared.config.rendering.DoughnutChartRende \\$	r ti ng
module, 29	badge_color (aisquared.config.rendering.ImageRendering.ImageRendering.
aisquared.config.rendering.FilterRendering	property), 32
module, 30	badge_color (aisquared.config.rendering.ObjectRendering.Object
aisquared.config.rendering.HTMLTagRendering	property), 34
module, 31	badge_color (aisquared.config.rendering.WordRendering.WordRendering
aisquared.config.rendering.ImageRendering	property), 37
module, 32	badge_shape (aisquared.config.rendering.WordRendering.WordRendering
aisquared.config.rendering.LineChartRendering	property), 37
module, 33	property), sr
aisquared.config.rendering.ObjectRendering	· · · · · · · · · · · · · · · · · · ·
module, 34	aisquared.config.rendering.BarChartRendering), 24
aisquared.config.rendering.PieChartRendering	- ·
module, 35	base_url (aisquared.platform.AISquaredPlatformClient.AISquaredPlatforproperty), 42
aisquared.config.rendering.SOSRendering	BaseObject (class in aisquared.base.BaseObject), 3
module, 36	BinaryClassification (class in
aisquared.config.rendering.TableRendering module, 36	aisquared.config.postprocessing.BinaryClassification),
aisquared.config.rendering.WordRendering	
module, 37	BinaryFeedback (class in
aisquared.logging	aisquared.config.feedback.BinaryFeedback), 8
module, 40	body_only (aisquared.config.harvesting.TextHarvester.TextHarvester
aisquared.platform	property), 13
module, 41	bucket (aisquared.config.analytic.ReverseMLWorkflow.ReverseMLWorkflow
	property), 8
aisquared.platform.AISquaredPlatformClient	

C	${\tt delete_group()}\ (a is quared. platform. A IS quared Platform Client. A IS quared Platform Client.$
<pre>can_toggle(aisquared.config.rendering.SOSRendering.S</pre>	SOSRenderifigethod), 43 delete_model() (aisquared.platform.AISquaredPlatformClient.AISquared
classes (aisquared.config.rendering.DocumentRendering property), 28	delete_user() (aisquarea.piatjorm.AIsquareaPiatjormCiient.AIsquareaI
classes (aisquared.config.rendering.ImageRendering.Imagernerty) 32	deploy_model() (in module
classes (aisquared.config.rendering.WordRendering.Worproperty), 37	Deployeumaly cic (class in
nronerty) 17	ertToColor aisquared.config.analytic.DeployedAnalytic), 5 DeployedModel (class in
color (aisquared.config.rendering.ImageRendering.Image property), 32	0
nuan autu) 21	cthesaringion (aisquared.config.GraphConfiguration.GraphConfiguration property), 38
nranarty) Q	everserriestion (qisquared.config.ModelConfiguration.ModelConfiguration property), 39
property). 19	opeisplay (aisquared.config.rendering.ContainerRendering.ContainerRende property), 26
column (aisquared.config.preprocessing.tabular.Steps.One property), 20	uisquarea.conjig.preprocessing.image.sieps),
columns (aisquared.config.preprocessing.tabular.Steps.Maproperty), 19	DocumentRendering (class in
columns (aisquared.config.preprocessing.tabular.Steps.ZS property), 20	28
method) 38	Consuments (aisquared.config.rendering.DocumentRendering.DocumentRen property), 28
compile() (aisquared.config.ModelConfiguration.ModelConfiguration), 39	aisquarea.conjig.renaering.DoughnuiChariKenaering),
ContainerRendering (class in aisquared.config.rendering.ContainerRendering)), DropColumn (class in aisquared.config.preprocessing.tabular.Steps),
26 content_key (aisquared.config.rendering.WordRendering	19
property), 37 ConvertToCase (class in	features (aisquared.config.harvesting.InputHarvester.InputHarvester
aisquared.config.preprocessing.text.Steps),	<pre>property), 12 feedback_dict(aisquared.config.ModelConfiguration.ModelConfiguratio</pre>
ConvertToColor (class in aisquared.config.preprocessing.image.Steps),	property), 39 feedback_steps (aisquared.config.ModelConfiguration.ModelConfiguration)
17 ConvertToVocabulary (class in	property), 40 filename (aisquared.config.analytic.ReverseMLWorkflow.ReverseMLWorkf
aisquared.config.preprocessing.text.Steps), 22	<pre>property), 8 FilterRendering (class in</pre>
<pre>create_group() (aisquared.platform.AISquaredPlatform</pre>	30
create_user() (aisquared.platform.AISquaredPlatformC method), 42	Cli <mark>ena99SqisarearPdagorfinCharn</mark> esting.TextHarvester.TextHarvester property), 13
CustomObject (class in aisquared.base.CustomObject), 4	<pre>font_color (aisquared.config.rendering.ImageRendering.ImageRendering</pre>
D	<pre>font_color (aisquared.config.rendering.ObjectRendering.ObjectRendering property), 34</pre>
DashboardReplacementRendering (class in aisquared.config.rendering.DashboardReplacem 27	font_size (aisquared.config.rendering.ImageRendering.ImageRendering nentRendering), 32

```
font_size(aisquared.config.rendering.ObjectRendering.ObjectRendering
                                              property), 34
                                                                                                                                                                                                                                                                                           id (aisquared.config.rendering.ContainerRendering.ContainerRendering
                                                                                                                                                                                                                                                                                                                                         property), 26
G
                                                                                                                                                                                                                                                                                           ImageHarvester
                                                                                                                                                                                                                                                                                                                                                                                                                                                            (class
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                in
get_filenames() (aisquared.config.GraphConfiguration.GraphConfiguration.GraphConfiguration.GraphConfiguration.GraphConfiguration.GraphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfiguration.graphConfigu
                                               method), 38
\verb|get_group()| (a is quared. platform. A IS quared Platform Clien \verb|Indog | \verb|Part | | \textit{Platform} Client|) |
                                                                                                                                                                                                                                                                                                                                                                                                                                                                   (class
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                in
                                              method), 44
                                                                                                                                                                                                                                                                                                                                           aisquared.config.preprocessing.image.ImagePreprocessing),
get_group_id_by_name()
                                               (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformClient
                                                                                                                                                                                                                                                                                                                                                                                                                                                           (class
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               in
                                                                                                                                                                                                                                                                                                                                          aisquared.config.rendering.ImageRendering),
\verb|get_model()| (a is quared. platform. A IS quared Platform Client. A IS quared Platform Client)|
                                              method), 45
                                                                                                                                                                                                                                                                                           include_probability
get_model() (in module aisquared.utils.utils), 59
                                                                                                                                                                                                                                                                                                                                           (aisquared.config.rendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.DocumentRendering.Do
 get_model_filenames()
                                                                                                                                                                                                                                                                                                                                         property), 28
                                               (aisquared.config.ModelConfiguration.ModelConfiguration_probability
                                              method), 40
                                                                                                                                                                                                                                                                                                                                           (aisquared.config.rendering.ImageRendering.ImageRendering
 get_model_id_by_name()
                                                                                                                                                                                                                                                                                                                                          property), 32
                                               (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredP
                                              method), 45
                                                                                                                                                                                                                                                                                                                                          (aisquared.config.rendering.ObjectRendering.ObjectRendering
get_remote_prediction()
                                                                                                                                                                                                                                            module
                                                                                                                                                                                       (in
                                                                                                                                                                                                                                                                                                                                         property), 34
                                              aisquared.serving.get remote prediction),
                                                                                                                                                                                                                                                                                           input_type (aisquared.config.analytic.DeployedAnalytic.DeployedAnalyti
                                                                                                                                                                                                                                                                                                                                         property), 5
get_role_id_by_role_name()
                                                                                                                                                                                                                                                                                           \verb"input_type" (a is quared. config. analytic. Deployed Model. Deployed Model) \\
                                                (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformGlient
                                               method), 46
                                                                                                                                                                                                                                                                                          input_type (aisquared.config.analytic.LocalAnalytic.LocalAnalytic
get_user() (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformClient
                                              method), 46
                                                                                                                                                                                                                                                                                           input\_type (aisquared.config.analytic.LocalModel.LocalModel
get_user_id_by_name()
                                                                                                                                                                                                                                                                                                                                         property), 7
                                               (a is quared. platform. A IS {\it quaredPlatformClient.} A IS 
                                              method), 47
                                                                                                                                                                                                                                                                                                                                         property), 8
GraphConfiguration
                                                                                                                                                                             (class
                                                                                                                                                                                                                                                                                          input_type(aisquared.config.harvesting.InputHarvester.InputHarvester
                                               aisquared.config.GraphConfiguration), 38
                                                                                                                                                                                                                                                                                                                                         property), 12
                                                                                                                                                                                                                                                                                          InputHarvester
                                                                                                                                                                                                                                                                                                                                                                                                                                                            (class
Н
                                                                                                                                                                                                                                                                                                                                           aisquared.config.harvesting.InputHarvester),
harvester_dict(aisquared.config.ModelConfiguration.ModelConfiguration
                                              property), 40
harvesting_steps(aisquared.config.ModelConfiguration.ModelConfiguration
                                                                                                                                                                                                                                                                                          {\bf key}\,(a is quared. config. rendering. Filter Rendering. Filter Rendering
                                              property), 40
header (aisquared.config.analytic.DeployedAnalytic.DeployedAnalytieroperty), 30
                                              property), 5
{\bf header} \ (a is quared. config. analytic. Deployed Model. Deployed {\it Model}
                                                                                                                                                                                                                                                                                           label (aisquared.config.rendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRendering.ContainerRenderin
                                              property), 6
headers (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AI
                                                                                                                                                                                                                                                                                          label (aisquared.config.rendering.DashboardReplacementRendering.Dash
                                              property), 47
height (aisquared.config.rendering.ContainerRendering.ContainerRenderitg), 27
                                              property), 26
                                                                                                                                                                                                                                                                                          label (aisquared.config.rendering.SOSRendering.SOSRendering
how (aisquared.config.harvesting.ImageHarvester.ImageHarvester
                                                                                                                                                                                                                                                                                                                                         property), 36
                                              property), 11
                                                                                                                                                                                                                                                                                           label_map (aisquared.config.feedback.BinaryFeedback.BinaryFeedback
how (aisquared.config.harvesting.TextHarvester.TextHarvester
                                                                                                                                                                                                                                                                                                                                         property), 8
                                              property), 13
                                                                                                                                                                                                                                                                                           label_map (aisquared.config.feedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeedback.MulticlassFeed
HTMLTagRendering
                                                                                                                                                                       (class
                                                                                                                                                                                                                                                                       in
                                                                                                                                                                                                                                                                                                                                         property), 10
                                              aisquared.config.rendering.HTMLTagRendering),label_map(aisquared.config.postprocessing.BinaryClassification.BinaryC
                                               31
                                                                                                                                                                                                                                                                                                                                         property), 14
```

```
label_map (aisquared.config.postprocessing.MulticlassClamaificAtions). What laisquast ClamaificAtions (Alaxififigther vesting.InputHarvester.InputHarvester).
                                  property), 14
                                                                                                                                                                                                                                                    property), 12
label_map (aisquared.config.postprocessing.ObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObjectDetectimatObj
                                  property), 15
                                                                                                                                                                                                                                                    property), 22
length (aisquared.config.preprocessing.text.Steps.PadSequared.config.preprocessing.tabular.Steps.MinMax
                                  property), 22
                                                                                                                                                                                                                                                    property), 19
limit (aisquared.config.harvesting.TextHarvester.TextHarvmetans (aisquared.config.preprocessing.tabular.Steps.ZScore
                                  property), 13
                                                                                                                                                                                                                                                    property), 20
LineChartRendering
                                                                                                                                 (class
                                                                                                                                                                                                  in method(aisquared.config.preprocessing.image.Steps.Resize
                                  a is quared. config. rendering. Line Chart Rendering),\\
                                                                                                                                                                                                                                                    property), 18
                                                                                                                                                                                                                 mimic_model() (in module aisquared.utils.utils), 60
list_group_users() (aisquared.platform.AISquaredPlatform@disquetN&quanftPplatfopmocelisintg.Regression.Regression
                                  method), 47
                                                                                                                                                                                                                                                    property), 15
list_groups() (aisquared.platform.AlSquaredPlatformClNaimMdSquarediPlatsqvmCellentnfig.preprocessing.tabular.Steps),
                                  method), 47
                                                                                                                                                                                                                 {\tt mins} (aisquared.config.preprocessing.tabular.Steps.MinMax
list_model_feedback()
                                  (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredP
                                  method), 48
                                                                                                                                                                                                                 mlflow_token(aisquared.config.GraphConfiguration.GraphConfiguration
list_model_prediction_feedback()
                                                                                                                                                                                                                                                    property), 38
                                   (aisquared.platform.AISquaredPlatformClient.AIStqlutilxettPlatficent@diixqutared.config.ModelConfiguration.ModelConfiguration
                                  method), 48
                                                                                                                                                                                                                                                    property), 40
list_model_usage_metrics()
                                                                                                                                                                                                                 mlflow_uri(aisquared.config.GraphConfiguration.GraphConfiguration
                                  (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredP
                                  method), 49
                                                                                                                                                                                                                 mlflow_uri(aisquared.config.ModelConfiguration.ModelConfiguration
list_model_users() (aisquared.platform.AISquaredPlatformClientpAbSqutxp;dPlatformClient
                                  method), 49
                                                                                                                                                                                                                 mlflow_user(aisquared.config.GraphConfiguration.GraphConfiguration
list_models() (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformClient
                                  method), 50
                                                                                                                                                                                                                 \verb|mlflow_user| (a is quared. config. Model Configuration. Model Configuration)|
list_prediction_feedback()
                                                                                                                                                                                                                                                    property), 40
                                  (aisquared.platform.AISquaredPlatformClient.AISMoodeddBlatffogunCdteron
                                                                                                                                                                                                                                                                                                                                                  (class
                                                                                                                                                                                                                                                                                                                                                                                                                   in
                                  method), 50
                                                                                                                                                                                                                                                     aisquared.config.ModelConfiguration), 39
list_roles() (aisquared.platform.AISquaredPlatformClieModdSipeedbldtktformClient
                                                                                                                                                                                                                                                                                                                                      (class
                                                                                                                                                                                                                                                                                                                                                                                                                    in
                                                                                                                                                                                                                                                     aisquared.config.feedback.ModelFeedback), 9
                                  method), 51
                                                                                                                                                                                                                 module
list_user_usage_metrics()
                                   (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredP
                                  method), 51
                                                                                                                                                                                                                                   aisquared.base, 3
list_users() (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClient.AISquaredPlatformClie
                                  method), 52
                                                                                                                                                                                                                                   aisquared.base.CustomObject, 4
load_beyondml_model()
                                                                                                                                   (in
                                                                                                                                                                               module
                                                                                                                                                                                                                                   aisquared.base.rendering, 4
                                  aisquared.serving.deploy_model), 58
                                                                                                                                                                                                                                   aisquared.base.stages, 4
LocalAnalytic
                                                                                                                                                                                                                                   aisquared.config, 4
                                                                                                                    (class
                                                                                                                                                                                                  in
                                  aisquared.config.analytic.LocalAnalytic),
                                                                                                                                                                                                                                   aisquared.config.analytic, 5
                                                                                                                                                                                                                                   aisquared.config.analytic.DeployedAnalytic,
LocalModel (class in aisquared.config.analytic.LocalModel),
                                                                                                                                                                                                                                   aisquared.config.analytic.DeployedModel,
login() (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformClient
                                                                                                                                                                                                                                   aisquared.config.analytic.LocalAnalytic,
                                  method), 52
lowercase (aisquared.config.preprocessing.text.Steps.ConvertToCas&
                                  property), 21
                                                                                                                                                                                                                                   aisquared.config.analytic.LocalModel, 7
                                                                                                                                                                                                                                   aisquared.config.analytic.ReverseMLWorkflow,
M
                                                                                                                                                                                                                                   aisquared.config.feedback, 8
max (aisquared.config.postprocessing.Regression.Regression
                                                                                                                                                                                                                                   aisquared.config.feedback.BinaryFeedback,
                                  property), 15
```

```
aisquared.config.feedback.ModelFeedback,
                                                aisquared.config.rendering.DocumentRendering.
aisquared.config.feedback.MulticlassFeedback, aisquared.config.rendering.DoughnutChartRendering,
aisquared.config.feedback.QualitativeFeedback,aisquared.config.rendering.FilterRendering,
aisquared.config.feedback.RegressionFeedback, aisquared.config.rendering.HTMLTagRendering,
aisquared.config.feedback.SimpleFeedback,
                                                aisquared.config.rendering.ImageRendering,
aisquared.config.GraphConfiguration, 38
                                                aisquared.config.rendering.LineChartRendering,
aisquared.config.harvesting, 11
                                                aisquared.config.rendering.ObjectRendering,
aisquared.config.harvesting.ImageHarvester,
aisquared.config.harvesting.InputHarvester,
                                                aisquared.config.rendering.PieChartRendering,
aisquared.config.harvesting.QueryParameterHarvæistærared.config.rendering.SOSRendering,
aisquared.config.harvesting.TextHarvester,
                                                aisquared.config.rendering.TableRendering,
                                                aisquared.config.rendering.WordRendering,
aisquared.config.ModelConfiguration, 39
aisquared.config.postprocessing, 14
aisquared.config.postprocessing.BinaryClassifiacistqioanced.logging, 40
                                                aisquared.platform, 41
aisquared.config.postprocessing.MulticlassClassisficantendonplatform.AISquaredPlatformClient,
a is quared. config. postprocessing. Object {\tt Detectiom}; squared. serving, 57
                                                aisquared.serving.deploy_model, 58
aisquared.config.postprocessing.Regression,
                                                aisquared.serving.get_remote_prediction,
                                                     59
aisquared.config.preprocessing, 16
                                                aisquared.utils, 59
aisquared.config.preprocessing.image, 16
                                                aisquared.utils.utils, 59
aisquared.config.preprocessing.image.ImagePwprocessification
                                                                             (class
                                                     aisquared.config.postprocessing.MulticlassClassification),
aisquared.config.preprocessing.image.Steps,
                                            MulticlassFeedback
                                                                                         in
                                                                          (class
aisquared.config.preprocessing.tabular,
                                                    aisquared.config.feedback.MulticlassFeedback),
aisquared.config.preprocessing.tabular.StelfsltiplyValue
                                                                       (class
                                                                                         in
                                                    aisquared.config.preprocessing.image.Steps),
aisquared.config.preprocessing.tabular.TabularPreprocessing,
                                            Ν
aisquared.config.preprocessing.text, 21
aisquared.config.preprocessing.text.Steps, name (aisquared.config.GraphConfiguration.GraphConfiguration
                                                    property), 38
aisquared.config.preprocessing.text.TextPrapme(esqinesd.config.ModelConfiguration.ModelConfiguration
                                                    property), 40
aisquared.config.rendering, 24
aisquared.config.rendering.BarChartRendering,
                                            ObjectDetection
                                                                        (class
aisquared.config.rendering.ContainerRendering,
                                                     aisquared.config.postprocessing.ObjectDetection),
aisquared.config.rendering.DashboardReplacementRendering
                                                                        (class
                                                                                         in
   27
                                                     aisquared.config.rendering.ObjectRendering),
```

34	Q
OneHot (class in aisquared.config.preprocessing.tabular.Sta	tengualifier(aisquared.config.rendering.FilterRendering.FilterRendering
oov_character(aisquared.config.preprocessing.text.Steps	property), 30 os Guarret TOYOGF Bularyack (class in
property), 22 orientation(aisquared.config.rendering.ContainerRenderin	aisquared.config.feedback.QualitativeFeedback), dering.ContainerRendering
property), 26 owner (aisquared.config.GraphConfiguration.GraphConfig	query keys (aisquared.config.harvesting, Ouery Parameter Harvester, Ouer
<pre>property), 38 owner (aisquared.config.ModelConfiguration.ModelConfig</pre>	query selector (aisquared.config.rendering.ContainerRendering.Contai
property), 40	QueryParameterHarvester (class in
P	aisquared.config.harvesting.QueryParameterHarvester), 12
<pre>pad_character(aisquared.config.preprocessing.text.Steps</pre>	ps.PadSequences R
	s.PadSequensquared.config.harvesting.TextHarvester.TextHarvester property), 13
PadSequences (class in	${\tt Regression} ({\it class in a is quared. config. postprocessing. Regression}),$
aisquared.config.preprocessing.text.Steps), 22	15 RegressionFeedback (class in
password (aisquared.platform.AISquaredPlatformClient.A property), 53	AISquaredPlatfqtimCli.evinfig.feedback.RegressionFeedback),
path (aisquared.config.analytic.LocalAnalytic.LocalAnalyticproperty), 7	ytigemove_digits (aisquared.config.preprocessing.text.Steps.RemoveCharac property), 23
${\tt period} (a is quared. config. analytic. Reverse MLW or k flow. Reverse MLW or k flow.$	everence Waster from group ()
<pre>property), 8 PieChartRendering (class in</pre>	memou j, 55
aisquared.config.rendering.PieChartRendering), 35	RemoveCharacters (class in aisquared.config.preprocessing.text.Steps),
placement (aisquared.config.rendering.ImageRendering.In	ImageRende <u>ri</u> ng
property), 32 placement (aisquared.config.rendering.ObjectRendering.O	1 1 2//
property), 34 position (aisquared.config.rendering.ContainerRendering	rendering_steps(aisquared.config.ModelConfiguration.ModelConfigura ng.Containerppgggggggg
<pre>property), 26 postprocesser_dict(aisquared.config.ModelConfigurar</pre>	Resize (class in aisquared.config.preprocessing.image.Steps), ation.ModelGonfiguration
property), 40	$\verb"result_key" (a is quared. config. rendering. Word Rendering. Word Rendering) and the property of the prope$
$postprocessing_steps\\ (a is quared. config. Model Configuration. Model Configuration) and the configuration of t$	property), 37 On freverise MLWorkflow (class in
<pre>property), 40 prediction_key (aisquared.config.rendering.DocumentR</pre>	aisquared.config.analytic.ReverseMLWorkflow), Rendering.DocumentRendering
<pre>property), 28 preprocesser_dict(aisquared.config.ModelConfiguration)</pre>	round (aisquared.config.postprocessing.Regression.Regression
property), 40	1 1 7//
preprocessing_steps (aisquared.config.ModelConfiguration.ModelCon	S onfiguration secret (aisquared.config.analytic.DeployedAnalytic.DeployedAnalytic
<pre>property), 40 preserve_aspect_ratio</pre>	property), 5
(aisquared.config.preprocessing.image.Steps.Res	DIODELLY I. U
property), 18 probability_key (aisquared.config.rendering.Document.property), 28	secret (append), secret

share_model_with_gro (aisquared.plat)	oup() form.AISquaredPlatformC	Client.AI	thickness(aisquared.config.rendering.ObjectRendering.ObjectRendering SquaredPl anfopenClje Bt4
method), 53			$\verb threshold (a is quared. config. postprocessing. Binary Classification. Binary Classifi$
share_model_with_use	er()		property), 14
(aisquared.plat)	form.AISquaredPlatformC	Client.AI	S tames:hOladforinGlime tl.config.postprocessing.ObjectDetection.ObjectDetec
method), 54			property), 15
SimpleFeedback	(class	in	$threshold_key~(a is quared. config. rendering. Document Renderin$
aisquared.config	g.feedback.SimpleFeedbac	ck),	property), 28
11			$threshold_key (\it ais quared.config.rendering.Image Rendering.Image Rendering$
<pre>size (aisquared.config.pr</pre>	reprocessing.image.Steps.	Resize	property), 32
property), 18			$threshold_key~(a is quared. config. rendering. Word Renderin$
SOSRendering	(class	in	property), 37
			$threshold_value (\it ais quared.config. rendering. Document Render$
$\verb"source" (a is quared. config$.rendering.FilterRenderin		
property), 30			${\tt threshold_value} \ (a is quared. config. rendering. Image Rendering. $
<pre>split_sentences(aisqu</pre>	ıared.config.preprocessin		
property), 23			$threshold_value (\it ais quared.config. rendering. Word Render$
split_words(aisquared	.config.preprocessing.text		
property), 23			to_dict() (aisquared.base.BaseObject.BaseObject
$\verb stage (a is quared. config. 0$	${\it Graph Configuration. Grap}$		
property), 39			to_dict() (aisquared.base.CustomObject.CustomObject
$\verb stage (a is quared. config. l$	ModelConfiguration.Mode		
property), 40			${\tt to_dict()}\ (a is quared. config. analytic. Deployed Analytic. De$
start_character(aisqu	ıared.config.preprocessin		
property), 22			to_dict() (aisquared.config.analytic.DeployedModel.DeployedModel
_	iared.config.rendering.Co		Rendering. Grathoith; rRendering
property), 26			to_dict() (aisquared.config.analytic.LocalAnalytic
stds (aisquared.config.pr	eprocessing.tabular.Steps		
property), 20			to_dict() (aisquared.config.analytic.LocalModel.LocalModel
	nfig.preprocessing.image.		reprocessingethate. Preprocessor
property), 16	T		to_dict() (aisquared.config.analytic.ReverseMLWorkflow.ReverseMLWorfd.Re
_	плд.preprocessing.text.1e		ocessing. Tenter hope consistency of the last th
property), 24	(1		to_dict() (aisquared.config.feedback.BinaryFeedback.BinaryFeedback
SubtractValue	(class	in	method), 8
aisquarea.confi 18	g.preprocessing.image.Ste	eps),	<pre>to_dict() (aisquared.config.feedback.ModelFeedback.ModelFeedback</pre>
Т			to_dict() (aisquared.config.feedback.MulticlassFeed
TableRendering	(class	in	$\verb"to_dict(")" (a is quared. config. feedback. Qualitative Feedba$
aisquared.config	g.rendering.TableRenderi	ng),	method), 10
36			$\verb"to_dict(")" (a is quared. config. feedback. Regression Feedback. Regression Feedback. Regression Feedback. The property of the property of$
TabularPreprocessor	(class	in	method), 10
aisquared.confiz 21	g.preprocessing.tabular.To	abularPi	rtprdiestifig (aisquared.config.feedback.SimpleFeedback.SimpleFeedback method), 11
test_connection()(ai	squared.platform.AISquai	redPlatfo	taCdient. Al Sqiquate the config. Graph Configuration. Graph Configuration
method), 54			method), 39
TextHarvester	(class	in	$\verb"to_dict()" (a is quared. config. harvesting. Image Harvester. Image Harvester) \\$
aisquared.config	g.harvesting.TextHarveste	(r), 13	method), 11
TextPreprocessor	(class		$\verb"to_dict()" (a is quared. config. harvesting. Input Harvester. Input Harvester) \\$
aisquared.config	g.preprocessing.text.TextF	Preproce	ssing), method), 12
24			$\verb"to_dict()" (a is quared. config. harvesting. Query Parameter Harvester. Query Parameter Harvester) and the property of the$
thickness (aisquared.co	nfig.rendering.ImageRend	dering.In	nageRenderinghod), 13
property), 32	-	-	to_dict() (aisquared.config.harvesting.TextHarvester.TextHarvester

method), 13

- to_dict() (aisquared.config.ModelConfiguration.ModelConfiguration) (aisquared.config.rendering.DocumentRendering.Documen
- to_dict() (aisquared.config.postprocessing.BinaryClassificatidin.Bihin.kwCskqussificationfig.rendering.DoughnutChartRendering.DoughnutCha
- to_dict() (aisquared.config.postprocessing.MulticlassClaxxificativar(.MultisqlussAlassifigarindering.FilterRendering.FilterRendering method), 15 method), 30
- to_dict() (aisquared.config.postprocessing.ObjectDetectionsObj
- to_dict() (aisquared.config.postprocessing.Regression.R
- to_dict() (aisquared.config.preprocessing.image.ImagePreprocessing.(InixgedPreprocessing.LineChartRendering.
- to_dict() (aisquared.config.preprocessing.image.Steps.Addlevaltict() (aisquared.config.rendering.ObjectRendering.ObjectRendering.objectRenderi
- to_dict() (aisquared.config.preprocessing.image.Steps.ConvertToCO) (aisquared.config.rendering.PieChartRendering.PieChar
- to_dict() (aisquared.config.preprocessing.image.Steps.Di**tioleNate**() (aisquared.config.rendering.SOSRendering.SOSRendering method), 17
- to_dict() (aisquared.config.preprocessing.image.Steps.MnHoipdyNxtHQ) (aisquared.config.rendering.TableRendering.TableRendering method), 18 method), 37
- to_dict() (aisquared.config.preprocessing.image.Steps.Rexigedict() (aisquared.config.rendering.WordRendering.WordRendering method), 18 method), 37
- to_dict() (aisquared.config.preprocessing.image.Steps.SubtrajtSod() (aisquared.base.BaseObject.BaseObject method), 18 method), 4
- to_dict() (aisquared.config.preprocessing.tabular.Steps.Drojx@n(unisquared.platform.AISquaredPlatformClient.AISquaredPlatformC
- to_dict() (aisquared.config.preprocessing.tabular.Steps.MiokAnc_pattern (aisquared.config.preprocessing.text.Steps.Tokenize method), 20 property), 23
- to_dict() (aisquared.config.preprocessing.tabular.Steps.Of nekkenize (class in aisquared.config.preprocessing.text.Steps), method), 20 23
- to_dict() (aisquared.config.preprocessing.tabular.Steps.**ZSrone**(class in aisquared.config.preprocessing.text.Steps), method), 20 23
- method), 20
 to_dict() (aisquared.config.preprocessing.tabular.TabularHrupcactes_slugcExtbioln(Risapnoved.samfig.preprocessing.text.Steps.PadSeque.
- to_dict() (aisquared.config.preprocessing.text.Steps.ConvertToCase method), 22

method), 21

to_dict() (aisquared.config.preprocessing.text.Steps.ConvertdeVortedulerYor (aisquared.config.rendering.DocumentRendering.DocumentNod), 22 property), 28

property), 23

- to_dict() (aisquared.config.preprocessing.text.Steps.PadSequences_model_with_group()
 method), 23
 (aisquared.platform AIS
- method), 23 (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformto_dict() (aisquared.config.preprocessing.text.Steps.RemoveCharactering), 54
- method), 23 unshare_model_with_user()
- to_dict() (aisquared.config.preprocessing.text.Steps.Tokenize (aisquared.platform.AISquaredPlatformClient.AISquaredPlatform
- method), 23 method), 55 to_dict() (aisquared.config.preprocessing.text.Steps.Trim_update_group() (aisquared.platform.AISquaredPlatformClient.AISquared
- method), 24 method), 55
 to_dict() (aisquared.config.preprocessing.text.TextPreprograms_TextPre
- method), 24 method), 56
 to_dict() (aisquared.config.rendering.BarChartRendering.BurChartReveleringsquared.platform.AISquaredPlatformClient.AISquared
- method), 25
 method), 57
 to_dict() (aisquared.config.rendering.ContainerRendering.GontainerRendering.GontainerRendering.GontainerRendering.analytic.DeployedAnalytic.DeployedAnalytic method), 27
 property), 5
- to_dict() (aisquared.config.rendering.DashboardReplacement@splacement@splacement@splacement@splacement@splacement@splacement@splacement@splacement@splacement@splacement@splacement@splacement@splacement@splacement@splacement.

```
url (aisquared.config.GraphConfiguration.GraphConfiguration
                   property), 39
                                                                                                                     ZScore (class in aisquared.config.preprocessing.tabular.Steps),
url (aisquared.config.ModelConfiguration.ModelConfiguration
                   property), 40
url_locations (aisquared.config.harvesting.QueryParameterHarvester.QueryParameterHarvester
                   property), 13
{\tt use\_port} \ (a is quared. plat form. AIS quared Plat form Client. AIS
                   property), 57
username (aisquared.platform.AISquaredPlatformClient.AISquaredPlatformClient
                   property), 57
V
value (aisquared.config.preprocessing.image.Steps.AddValue
                   property), 17
value (aisquared.config.preprocessing.image.Steps.DivideValue
                   property), 17
value (aisquared.config.preprocessing.image.Steps.MultiplyValue
                   property), 18
{\tt value} \ (a is quared. config. preprocessing. image. Steps. Subtract Value
                   property), 18
value (aisquared.config.rendering.FilterRendering.FilterRendering
                   property), 31
values (aisquared.config.preprocessing.tabular.Steps.OneHot
                   property), 20
version (aisquared.config.GraphConfiguration.GraphConfiguration
                   property), 39
version (aisquared.config.ModelConfiguration.ModelConfiguration
                   property), 40
vocabulary (aisquared.config.preprocessing.text.Steps.ConvertToVocabulary
                   property), 22
W
where_replace (aisquared.config.rendering.DashboardReplacementRendering.DashboardReplacementRendering
                   property), 27
width (aisquared.config.rendering.ContainerRendering.ContainerRendering
                   property), 27
{\tt word\_list} (\it a is quared. config. rendering. Word Rendering. Word Rendering) \\
                   property), 37
WordRendering
                                                                 (class
                   aisquared.config.rendering.WordRendering),
words (aisquared.config.rendering.DocumentRendering.DocumentRendering
                   property), 29
Χ
xOffset (aisquared.config.rendering.ContainerRendering.ContainerRendering
                   property), 27
Υ
yOffset (aisquared.config.rendering.ContainerRendering.ContainerRendering
                   property), 27
```