```
import Foundation

@available macOS 10.13
public let MLModelErrorDomain  String

@available macOS 10.13
public struct MLModelError   CustomNSError  Hashable  Error

    public init           NSError

    public static var errorDomain  String   get

    /**
      MLModelError defines the set of MLModel related error codes.

      The framework communicates the error code to the application through
NSError's code property. The
      application could use the error code to present an helpful error message to
the user or to
      diagnose the problem.

      See also NSError's localizedDescription property, which often contains more
detailed information.
     */
    @available macOS 10.13
    public enum Code   Int  @unchecked Sendable  Equatable

        /**
          MLModelError defines the set of MLModel related error codes.

          The framework communicates the error code to the application through
NSError's code property. The
          application could use the error code to present an helpful error
message to the user or to
          diagnose the problem.

          See also NSError's localizedDescription property, which often contains
more detailed information.
         */
        public typealias _ErrorType   MLModelError

        /**    Core ML throws/returns this error when the framework encounters
an generic error.

              The typical cause for this error is an unexpected framework level
problem.
           */
        case generic   0

        /**   Core ML throws/returns this error when the model client, typically
an application, sends
```

```
                        the wrong feature type to a model's input.

                        The typical cause for this error is a programming mistake.

                        For example, a prediction method will throw/return the error when
the caller passes an image
                        to a model's input that expects an `MLMultiArray`.
                    */
                case featureType    1

                /**   Core ML throws/returns this error when the framework encounters
some I/O problem, most
                        likely a file I/O problem.

                        For example, a model loading will throw/return the error when the
caller requests a
                        non-existing model URL.
                    */
                case io    3

                /**   Core ML throws/returns this error when the framework encounters
an error in the custom
                        layer subsystem.

                        The typical cause for this error is a programming mistake.

                        For example, a prediction method will throw/return the error when
it fails to find the custom
                        layer implementation.
                    */
                @available macOS 10.13 2
                case customLayer    4

                /**   Core ML throws/returns this error when the framework encounters
an error in the custom
                        model subsystem.

                        The typical cause for this error is a programming mistake.

                        For example, a prediction method will throw/return the error when
it fails to find the custom
                        model implementation.
                    */
                @available macOS 10.14
                case customModel    5

                /**   Core ML throws/returns this error when the framework encounters
an error while performing
                        the on-device model update.

                        For example, the framework will throw/return the error when it
fails to save the updated model.
                    */
```

```
        @available macOS 10.15
        case update   6

        /**   Core ML throws/returns this error when the model client, typically
an application, queries
              an unsupported model parameter (see MLParameterKey).

              The typical cause for this error is a programming mistake.
            */
        @available macOS 10.15
        case parameters   7

        /**   Core ML throws/returns this error when the framework fails to
download the model decryption
              key.

              The typical cause for this error is a network connection issue to
the key server.
            */
        @available macOS 11.0
        case modelDecryptionKeyFetch   8

        /**   Core ML throws/returns this error when the framework encounters
an error in the model
              decryption subsystem.

              The typical cause for this error is in the key server configuration
and the client application
              cannot do much about it.

              For example, a model loading method will throw/return the error
when it uses incorrect model
              decryption key.
            */
        @available macOS 11.0
        case modelDecryption   9

        /**   Core ML throws/returns this error when the framework encounters
an error in the model
              collection deployment subsystem.

              The typical cause for this error is the network connectability issue
to the model deployment
              server.
            */
        @available macOS 11.0
        case modelCollection   10

        ///  Core ML throws/returns this error when the prediction is cancelled
prior to completing.
        @available macOS 14.0
        case predictionCancelled   11
```

```
    /**
              Core ML throws/returns this error when the framework encounters an
generic error.

              The typical cause for this error is an unexpected framework level
problem.
        */
    public static var generic  MLModelError Code   get

    /**   Core ML throws/returns this error when the model client, typically an
application, sends
              the wrong feature type to a model's input.

              The typical cause for this error is a programming mistake.

              For example, a prediction method will throw/return the error when the
caller passes an image
              to a model's input that expects an `MLMultiArray`.
        */
    public static var featureType  MLModelError Code   get

    /**   Core ML throws/returns this error when the framework encounters some
I/O problem, most
              likely a file I/O problem.

              For example, a model loading will throw/return the error when the caller
requests a
              non-existing model URL.
        */
    public static var io  MLModelError Code   get

    /**   Core ML throws/returns this error when the framework encounters an
error in the custom
              layer subsystem.

              The typical cause for this error is a programming mistake.

              For example, a prediction method will throw/return the error when it
fails to find the custom
              layer implementation.
        */
    @available macOS 10.13 2
    public static var customLayer  MLModelError Code   get

    /**   Core ML throws/returns this error when the framework encounters an
error in the custom
              model subsystem.

              The typical cause for this error is a programming mistake.

              For example, a prediction method will throw/return the error when it
```

```
fails to find the custom
                model implementation.
            */
    @available macOS 10.14
    public static var customModel  MLModelError Code    get

    /**    Core ML throws/returns this error when the framework encounters an
error while performing
                the on-device model update.

                For example, the framework will throw/return the error when it fails to
save the updated model.
            */
    @available macOS 10.15
    public static var update  MLModelError Code    get

    /**    Core ML throws/returns this error when the model client, typically an
application, queries
                an unsupported model parameter (see MLParameterKey).

                The typical cause for this error is a programming mistake.
            */
    @available macOS 10.15
    public static var parameters  MLModelError Code    get

    /**    Core ML throws/returns this error when the framework fails to download
the model decryption
                key.

                The typical cause for this error is a network connection issue to the key
server.
            */
    @available macOS 11.0
    public static var modelDecryptionKeyFetch
MLModelError Code    get

    /**    Core ML throws/returns this error when the framework encounters an
error in the model
                decryption subsystem.

                The typical cause for this error is in the key server configuration and
the client application
                cannot do much about it.

                For example, a model loading method will throw/return the error when it
uses incorrect model
                decryption key.
            */
    @available macOS 11.0
    public static var modelDecryption  MLModelError Code    get
    /**   Core ML throws/returns this error when the framework encounters an
error in the model
          collection deployment subsystem.

          The typical cause for this error is the network connectability issue to
the model deployment
          server.
        */
    @available macOS 11.0
    public static var modelCollection  MLModelError Code   get


    ///  Core ML throws/returns this error when the prediction is cancelled prior to
completing.
    @available macOS 14.0
    public static var predictionCancelled  MLModelError Code
get
```