```
import Foundation

/**
 * MLModel
 *
 * Construct a model and evaluate on a specific set of input features.
 * Inputs and outputs are accessed via the MLFeatureProvider protocol.
 * Returns a model or nil if there is an error.
 */
@available macOS 10.13
open class MLModel   NSObject

    /// A model holds a description of its required inputs and expected outputs.
    open var modelDescription  MLModelDescription   get

    /// The load-time parameters used to instantiate this MLModel object.
    @available macOS 10.14
    open var configuration  MLModelConfiguration   get

    /// Construct a model with a default MLModelConfiguration object
    public convenience init                 URL  throws

    /// Construct a model given the location of its on-disk representation. Returns
nil on error.
    @available macOS 10.14
    public convenience init                 URL
                MLModelConfiguration  throws

    /// Run a prediction on a model synchronously.
    ///
    /// This is a convenience overload method of
`prediction(from:options:)` that uses the default prediction options.
    ///
    /// - Parameters
    ///   - input: The input features to make a prediction from.
    ///   - error: The output parameter to be filled with error information on
failure.
    /// - Returns: The output features from the prediction.
    open func prediction             any MLFeatureProvider
throws    any MLFeatureProvider

    /// Run a prediction on a model synchronously
    ///
    /// - Parameters
    ///   - input: The input features to make a prediction from.
    ///   - options: Prediction options to modify how the prediction is run.
    ///   - error: The output parameter to be filled with error information on
failure.
    /// - Returns: The output features from the prediction.
    open func prediction             any MLFeatureProvider
```

```
        MLPredictionOptions  throws    any MLFeatureProvider

    /// Batch prediction without explicit options
    @available macOS 10.14
    open func predictions                          any
MLBatchProvider  throws    any MLBatchProvider

    /// Batch prediction with explicit options
    @available macOS 10.14
    open func predictions                    any
MLBatchProvider          MLPredictionOptions  throws    any
MLBatchProvider

    /// Provides value for the given parameter. Returns nil on error.
    @available macOS 10.15
    open func parameterValue          MLParameterKey  throws
   Any

    /**
       Construct a model asynchronously from a compiled model asset.

       @param asset Compiled model asset derived from in-memory or on-disk
Core ML model
       @param configuration Model configuration that hold options for loading a
model
       @param handler When the model load completes successfully or
unsuccessfully, the completion handler is invoked with a valid MLModel instance or
NSError object.
    */
    @available macOS 13.0
    open class func load _        MLModelAsset
MLModelConfiguration                                @escaping
 MLModel    any Error       Void

    /**
       Construct a model asynchronously from a compiled model asset.

       @param asset Compiled model asset derived from in-memory or on-disk
Core ML model
       @param configuration Model configuration that hold options for loading a
model
       @param handler When the model load completes successfully or
unsuccessfully, the completion handler is invoked with a valid MLModel instance or
NSError object.
    */
    @available macOS 13.0
    open class func load        MLModelAsset
MLModelConfiguration  async throws    MLModel


extension MLModel
```

```
    /// Run a prediction on a model.
    ///
    /// This method requires all inputs and outputs to be multidimensional arrays. If your model doesn't satisfy
    /// this requirement, materialize the tensor inputs to `MLShapedArray` values to create feature
    /// values for each, for example:
    /// ```swift
    /// let shapedArray = await tensor.shapedArray(of: Float.self)
    /// let inputFeatures = try MLDictionaryFeatureProvider(dictionary: [
    ///     "x": MLFeatureValue(shapedArray: shapedArray),
    ///     // Other non-multidimensional array inputs
    /// ])
    /// let prediction = try await model.prediction(from: inputFeatures)
    /// ```
    ///
    /// - Parameter inputs: The named input, or inputs, to make a prediction from.
    /// - Returns: The output, or outputs, from the prediction.
    @available macOS 15.0  iOS 18.0  tvOS 18.0  watchOS 11.0  visionOS 2.0
    public func prediction                String   MLTensor  async throws     String   MLTensor

    /// Run a stateful prediction on a model.
    ///
    /// This method requires all inputs and outputs to be multidimensional arrays. If your model doesn't satisfy
    /// this requirement, materialize the tensor inputs to `MLShapedArray` values to create feature
    /// values for each, for example:
    /// ```swift
    /// let shapedArray = await tensor.shapedArray(of: Float.self)
    /// let inputFeatures = try MLDictionaryFeatureProvider(dictionary: [
    ///     "x": MLFeatureValue(shapedArray: shapedArray),
    ///     // Other non-multidimensional array inputs
    /// ])
    /// let state = model.newState()
    /// let prediction = try await model.prediction(from: inputFeatures, using: state)
    /// ```
    ///
    /// - Parameters:
    ///   - inputs: The named input, or inputs, to make a prediction from.
    ///   - state: The state.
```

```
    /// - Returns: The output, or outputs, from the prediction.
    @available macOS 15.0  iOS 18.0  tvOS 18.0  watchOS 11.0
visionOS 2.0
    public func prediction               String   MLTensor
             MLState  async throws     String   MLTensor


@available macOS 11.0  iOS 14.0  watchOS 7.0  tvOS 14.0
extension MLModel

    /// Construct a model asynchronously given the location of its on-disk
representation and configuration.
    ///
    /// Model loading may take time when the model content is not immediately
available (e.g. encrypted model).
    /// Use this factory method especially when the caller is on the main thread.
    ///
    /// - Parameter url:           Location of its on-disk representation
(.mlmodelc directory).
    /// - Parameter configuration: Configuration The model
configuration
    /// - Parameter handler:       When the model load completes
successfully or unsuccessfully,
    ///                            the completion handler is invoked
with a valid MLModel instance or NSError object.
    public class func load                  URL
MLModelConfiguration
                           @escaping  Result MLModel  any
Error      Void

    /// Construct a model asynchronously given the location of its on-disk
representation and configuration.
    ///
    /// Model loading may take time when the model content is not immediately
available (e.g. encrypted model).
    /// Use this factory method especially when the caller is on the main thread.
    ///
    /// - Parameter url:           Location of its on-disk representation
(.mlmodelc directory).
    /// - Parameter configuration: Configuration The model
configuration
    /// - Returns: A constructed MLModel object.
    @available macOS 12.0  iOS 15.0  watchOS 8.0  tvOS 15.0

    public class func load                  URL
MLModelConfiguration                        async throws
MLModel

    /// Compile a .mlmodel asynchronously given its on-disk location.
    ///
    /// Model compilation may take a considerable amount of time.
```

```
    /// Use this factory method especially when the caller is on the main thread.
    ///
    /// - Parameter url:            Location of the .mlmodel file.
    /// - Parameter handler:        When the model compilation
completes successfully the completion handler is invoked
    ///                             with a valid URL to the
compiled .mlmodelc directory.
    ///                             On failure the completion handler is
invoked with an NSError object.
    ///
    /// The model is compiled to a temporary location on disk. You must move
the compiled model to a permanent location if you wish to keep it.
    @available macOS 13.0  iOS 16.0  tvOS 16.0
    @available
    public class func compileModel          URL
                            @escaping  Result URL  any Error
   Void

    /// Using Swift 'async' idiom.
    @available macOS 13.0  iOS 16.0  tvOS 16.0
    @available
    public class func compileModel          URL  async throws
   URL

    /// Creates a new state object.
    ///
    /// Core ML framework will allocate the state buffers declared in the model.
    ///
    /// The allocated state buffers are initialized to zeros. To initialize with
different values, use `.withMultiArray(for:)` to get the mutable
`MLMultiArray`-view to the state buffer.
    ///
    /// ```swift
    /// // Create state that contains two state buffers: s1
and s2.
    /// // Then, initialize s1 to 1.0 and s2 to 2.0.
    /// let state = model.makeState()
    /// state.withMultiArray(for: "s1") { stateMultiArray in
    ///     stateMultiArray[0] = 1.0
    /// }
    /// state.withMultiArray(for: "s2") { stateMultiArray in
    ///     stateMultiArray[0] = 2.0
    /// }
    /// ```
    @available macOS 15.0  iOS 18.0  watchOS 11.0  tvOS 18.0
visionOS 2.0
    public func makeState      MLState

    /// Run a prediction on a model asynchronously.
    ///
```

```
    /// - Parameter input:    The input features to make a prediction from.
    /// - Parameter options: Optional prediction options to modify how the
prediction is run
    /// - Returns: The output from the prediction.
    @available macOS 14.0  iOS 17.0  watchOS 10.0  tvOS 17.0

    public func prediction             any MLFeatureProvider
         MLPredictionOptions                              async
throws    any MLFeatureProvider

    /// Run a stateful prediction on a model asynchronously.
    ///
    /// ```swift
    /// let state = model.newState()
    /// let prediction = try await model.prediction(from:
inputFeatures, using: state)
    /// ```
    ///
    /// - Parameters:
    ///    - input:    The input features to make a prediction from.
    ///    - state:    The state.
    ///    - options: Optional prediction options to modify how the prediction
is run
    /// - Returns: The output from the prediction.
    @available macOS 15.0  iOS 18.0  watchOS 11.0  tvOS 18.0
visionOS 2.0
    public func prediction             any MLFeatureProvider
             MLState           MLPredictionOptions
                        async throws    any MLFeatureProvider

    /// The list of available compute devices for CoreML.
    ///
    /// Use the method to get the list of compute devices that MLModel's
prediction can use.
    ///
    /// The hardware may have other compute devices that are not available to
CoreML. Use
    /// `MLComputeDevice.allComputeDevices` to get the complete list.
    @available macOS 14.0  iOS 17.0  watchOS 10.0  tvOS 17.0

    public static var availableComputeDevices
 MLComputeDevice    get
```