```
import Foundation

/**
 * An immutable variant holding a data value of a supported MLFeatureType
 *
 * MLFeatureValue does not support type conversion in its accessor properties. It
 * can also have a missing or undefined value of a well defined type.
 */
@available(macOS 10.13, *)
open class MLFeatureValue : NSObject, NSCopying, NSSecureCoding {

    /// Type of the value for which the corresponding property below is held
    open var type: MLFeatureType { get }

    /// True if the value represents a missing or undefined value
    open var isUndefined: Bool { get }

    /// Populated value if the type is MLFeatureTypeInt64
    open var int64Value: Int64 { get }

    /// Populated value if the type is MLFeatureTypeDouble
    open var doubleValue: Double { get }

    /// Populated value if the type is MLFeatureTypeString
```

```
    open var stringValue: String { get }

    /// Populated value if the type is
MLFeatureTypeMultiArray
    open var multiArrayValue:
MLMultiArray? { get }

    /// Populated value if the type is
MLFeatureTypeDictionary
    open var dictionaryValue:
[AnyHashable : NSNumber] { get }

    /// Populated value if the type is
MLFeatureTypeImage
    open var imageBufferValue:
CVPixelBuffer? { get }

    /// Populated value if the type is
MLFeatureTypeSequence
    @available(macOS 10.14, *)
    open var sequenceValue: MLSequence? {
get }

    /// Hold an object with the specified
value
    public convenience init(int64 value:
Int64)

    public convenience init(double value:
Double)

    public convenience init(string value:
String)
```

```
    public convenience init(multiArray
value: MLMultiArray)

    public convenience init(pixelBuffer
value: CVPixelBuffer)

    @available(macOS 10.14, *)
    public convenience init(sequence:
MLSequence)

    /// Represent an undefined value of a
specified type
    public convenience init(undefined
type: MLFeatureType)

    /**
     * For encoding a sparse feature set
or for encoding probabilities. Input keys
that are not
     * NSNumber * or NSString * are
rejected on construction and return a
MLModelErrorFeatureTypeMismatch
     * error. Further validation for
consistency occurs on evaluation
     */
    public convenience init(dictionary
value: [AnyHashable : NSNumber]) throws

    /**
     * @abstract Returns a Boolean value
that indicates whether a feature value is
equal to another.
```

```
     *
     * @discussion If the types of the
MLFeatureValue objects "self" and "value"
are integer in one case and
     * double in the other (in either
order) then those mixed mode numeric
values are compared as NSNumbers.
     * Otherwise if the types of the
MLFeatureValue objects are different NO
is returned.
     * When "self" and "value" are both
PixelBuffer MLFeatureValue types, only
their CVPixelBufferRef values are
compared for equality,
     * the underlying arrays of
pixelValues are not examined.
     * [So, distinct PixelBuffer
MLFeatureValue objects with distinct
CVPixelBufferRef values which encapsulate
the same array of pixels will compare
*not* equal.]
     * For all other (matching)
MLFeatureValue types, the BOOL value
returned is the result of comparing
"self" with "value" via
     * isEqualToNumber:,
isEqualToString:, isEqualtoDictionary:,
isEqualToMultiArray:, isEqualToArray: as
chosen by the MLFeatureValue types.
     */
    open func isEqual(to value:
MLFeatureValue) -> Bool
}
```

```
@available(macOS 12.0, iOS 15.0, watchOS
8.0, tvOS 15.0, *)
extension MLFeatureValue {

    /// Constructs from MLShapedArray.
    ///
    /// The value will be semantically
copied (i.e. Copy-on-Write) to
MLFeatureValue object. A
    /// mutation to MLFeatureValue won't
affect the source MLShapedArray and vise
versa.
    ///
    /// - Parameter shapedArray The
MLShapedArray object.
    public convenience
init<Scalar>(shapedArray:
MLShapedArray<Scalar>) where Scalar :
MLShapedArrayScalar

    /// Returns MLShapedArray of the
specified scalar type.
    ///
    /// The function returns non-nil
value when the feature value has
MLMultiArray with the
    /// specified type.
    ///
    /// - Parameter type: The scalar type
    public func
shapedArrayValue<Scalar>(of type:
Scalar.Type) -> MLShapedArray<Scalar>?
```

```
where Scalar : MLShapedArrayScalar
}

extension MLFeatureValue {

    /// Creates a feature value from a
sendable feature value.
    @available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
    public convenience init(_ value:
MLSendableFeatureValue)
}
```