

Real-World Machine Learning ¹

Eric Bailey

July 2, 2017

¹ <https://www.manning.com/books/real-world-machine-learning>

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special contents, but the length of words should match the language.

Project Setup

```
1 <setup.py 1>≡
    import os
    from distutils.core import setup

    <Helper function to file contents to a string 2>

    setup(
        name = 'real_world_machine_learning',
        version = read('VERSION'),
        author = 'Eric Bailey',
        author_email = 'eric@ericb.me',
        description = 'Real-World Machine Learning',

        license = 'MIT',
        url = 'https://github.com/yurriq/real_world_machine_learning',
        packages = ['real_world_machine_learning'],
    )

2 <Helper function to file contents to a string 2>≡ (1)
    def read(fname):
        return open(os.path.join(os.path.dirname(__file__), fname)).read()
```

Describe this briefly and mention the reasoning behind VERSION.

Chapter 2: Real-world data

```

<ch2.py 3>≡
    <Chapter 2 imports 4>

    <Categorical feature example 5>

    <Titanic example 8>

<Chapter 2 imports 4>≡ (3)
    from numpy import (array, unique)

<Categorical feature example 5>≡ (3)
    <Categorical data 6>

    <Convert a categorical feature to a number 7>

<Categorical data 6>≡ (5)
    cat_data = array([
        'male', 'female', 'male', 'male',
        'female', 'male', 'female', 'female'
    ])

<Convert a categorical feature to a number 7>≡ (5)
    def cat_to_num(data):
        categories = unique(data)
        features = []
        for cat in categories:
            binary = (data == cat)
            features.append(binary.astype("int"))
        return features

```

Titanic Example (feature extraction)

```

<Titanic example 8>≡ (3)
    <Titanic data 9>

    <Titanic cabin feature extraction 10>

```

Import from code/data/titanic.csv

```

<Titanic data 9>≡ (8)
    cabin_data = array(["C65", "", "E36", "C54", "B57 B59 B63 B66"])

```

```

10  <Titanic cabin feature extraction 10>≡ (8)
    def _cabin_char(cabins):
        try:
            return len(cabins), cabins[0][0]
        except IndexError:
            return 0, "X"

    def _cabin_num(cabins):
        try:
            return int(cabins[0][1:])
        except:
            return -1

    def cabin_features(data):
        features = []
        for cabin in data:
            cabins = cabin.split(" ")
            n_cabins, cabin_char = _cabin_char(cabins)
            cabin_num = _cabin_num(cabins)
            features.append([cabin_char, cabin_num, n_cabins])
        return features

```

Idris Port

```

11  <Data.ML.Categorical 11>≡
    - ----- [ Categorical.idr ]

    module Data.ML.Categorical

    import public Data.ML.Util

    %access export

    catToNum : (Num n, Cast Bool n, Ord a) => List a -> List (List n)
    catToNum xs = [ cast . (== cat) <$> xs | cat <- unique xs ]

    - ----- [ EOF ]

```

```

⟨Data.ML.Util 12⟩≡
- ----- [ Util.idr ]

module Data.ML.Util

%access public export

||| Convert a Bool to an Integer, as in C99's <stdbool.h>.
||| @ bool a Bool to convert
boolToInteger : (bool : Bool) -> Integer
boolToInteger False = 0
boolToInteger True  = 1

||| Cast Booleans to Integers via boolToInteger'.
implementation Cast Bool Integer where
  cast = boolToInteger

||| Simplified clone of numpy.unique'.
|||
||| Return the sorted unique elements of a list.
|||
||| '''
||| unique = sort . nub
||| '''
unique : Ord a => List a -> List a
unique = sort . nub

- ----- [ EOF ]

```

```

13  <Data.ML.Example 13>≡
    - ----- [ Example.idr ]

    module Data.ML.Example

    import public Data.ML.Categorical

    %access public export

    - ----- [ Categorical Feature Example ]

    namespace Categorical

    exampleData : List String
    exampleData = [ "male", "female", "male", "male"
                    , "female", "male", "female", "female"
                    ]

    ||| idris example
    ||| catToNum Categorical.exampleData
    ||| '''
    example : List (List Integer)
    example = catToNum Categorical.exampleData

    - ----- [ Titanic Example ]

    namespace Titanic

    exampleData : List String
    exampleData = [ "C65", "", "E36", "C54", "B57 B59 B63 B66" ]

    - TODO: record CabinFeature where

    ||| idris example
    ||| Titanic.example
    ||| '''
    example : List (Char, Integer, Nat)
    example = go . words <$> Titanic.exampleData
    where
      go : List String -> (Char, Integer, Nat)
      go [] = ('X', -1, 0)
      go (cabin :: cabins) with (strM cabin)
      go (" " :: cabins) | StrNil = ('X', -1, 0)
      go (strCons c num :: cabins) | (StrCons c num) =
        ( c
        , cast num
        , S (length cabins)

```

```
)
```

```
- ..... [ EOF ]
```

```
<ipkg 14>≡
```

```
package real_world_ml
```

```
opts = "-total"
```

```
modules = Data.ML.Categorical  
          , Data.ML.Example  
          , Data.ML.Util
```

Chunks

<Categorical data [6](#)>
 <Categorical feature example [5](#)>
 <ch2.py [3](#)>
 <Chapter 2 imports [4](#)>
 <Convert a categorical feature to a number [7](#)>
 <Data.ML.Categorical [11](#)>
 <Data.ML.Example [13](#)>
 <Data.ML.Util [12](#)>
 <Helper function to file contents to a string [2](#)>
 <ipkg [14](#)>
 <setup.py [1](#)>
 <Titanic cabin feature extraction [10](#)>
 <Titanic data [9](#)>
 <Titanic example [8](#)>

Index

_cabin_char: [10](#)
 _cabin_num: [10](#)
 array: [4](#), [6](#), [9](#)
 boolToInteger: [12](#), [12](#)
 cabin_data: [9](#)
 cabin_features: [10](#)
 cat_data: [6](#)
 cat_to_num: [7](#)
 Categorical.example: [13](#)
 Categorical.exampleData: [13](#)
 catToNum: [11](#), [13](#)
 Data.ML.Categorical: [11](#), [13](#), [14](#)
 Data.ML.Example: [13](#), [14](#)
 Data.ML.Util: [11](#), [12](#), [14](#)
 idris: [13](#), [13](#), [13](#), [13](#), [13](#), [13](#)
 numpy: [4](#), [12](#)
 os: [1](#), [2](#)
 read: [1](#), [2](#)
 setup: [1](#)
 Titanic.example: [13](#)
 Titanic.exampleData: [13](#)
 unique: [4](#), [7](#), [11](#), [12](#)