Real-World Machine Learning [1]

Eric Bailey

June 30, 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/yurrriq/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

Port the example to Idris

Chunks

Index