メディアアート・プログラミング
ml5.js 実践 – 転移学習2：特徴抽出による画像の回帰分析

2019年10月29日

前橋工科大学総合デザイン工学科
田所 淳

今日の内容

- 転移学習 (Transfer Learning) のプログラミング、その2
- 転移学習による画像の特徴量抽出を利用して、画像を回帰分析 (Regression)

- ml5による機械学習に入る前に…
  - p5.jsのGUI機能について
    - スライダー
    - ボタン

- ml5.jsの特徴量抽出を活用した回帰分析 (Image Feature Extractor Regression)
  - スライダーとボタンのGUIを使用して学習
  - 円を動かしてみる

p5.jsのGUI機能について

p5.jsのGUI機能について

- p5.jsにはHTMLの要素をそのままCanvasに表示できる機能がある

- この機能を活用すると、HTMLのGUIの要素をp5.jsのプログラム内で利用できる
  - スライダー
  - ボタン
  - テキスト入力
  - HTMLテキスト表示
  - ...etc.

- UTF-8に対応しているので、日本語など多言語の文字表示も問題なし

p5.jsのGUI機能について

- p5.jsのGUIの簡単なコードをつくってみる!
  - スライダー：円の大きさを変更
  - ボタン：円の色を変更

p5.jsのGUI機能について

- sketch.js

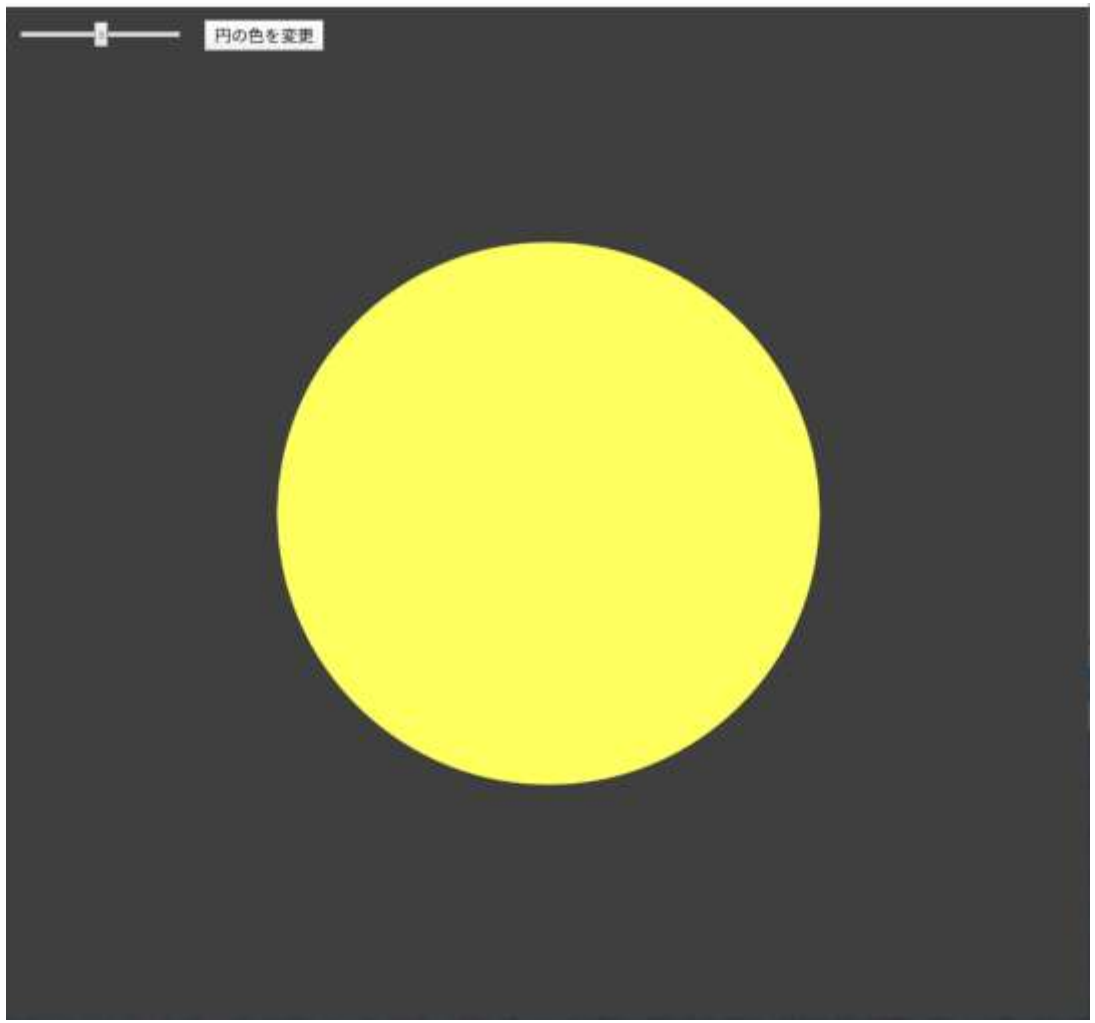
```
let slider; //スライダー
let button; //ボタン
let circleColor; //円の色
function setup() {
  createCanvas(windowWidth, windowHeight);
  //スライダー作成
  slider = createSlider(0, width, width/2);
  slider.position(10, 10);
  //ボタン作成
  button = createButton('円の色を変更');
  button.position(160, 10);
  //ボタンを押したらchangeColorを実行
  button.mousePressed(changeColor);
  //円の色初期値
  circleColor = color(255, 255, 63);
}

function draw() {
  background(63);
  noStroke();
  fill(circleColor);
  let radius = slider.value();
  circle(width/2, height/2, radius);
}

function changeColor(){
  //円の色をランダムに変化させる
  circleColor = color(random(255), random(255),
random(255));
}
```

p5.jsのGUI機能について

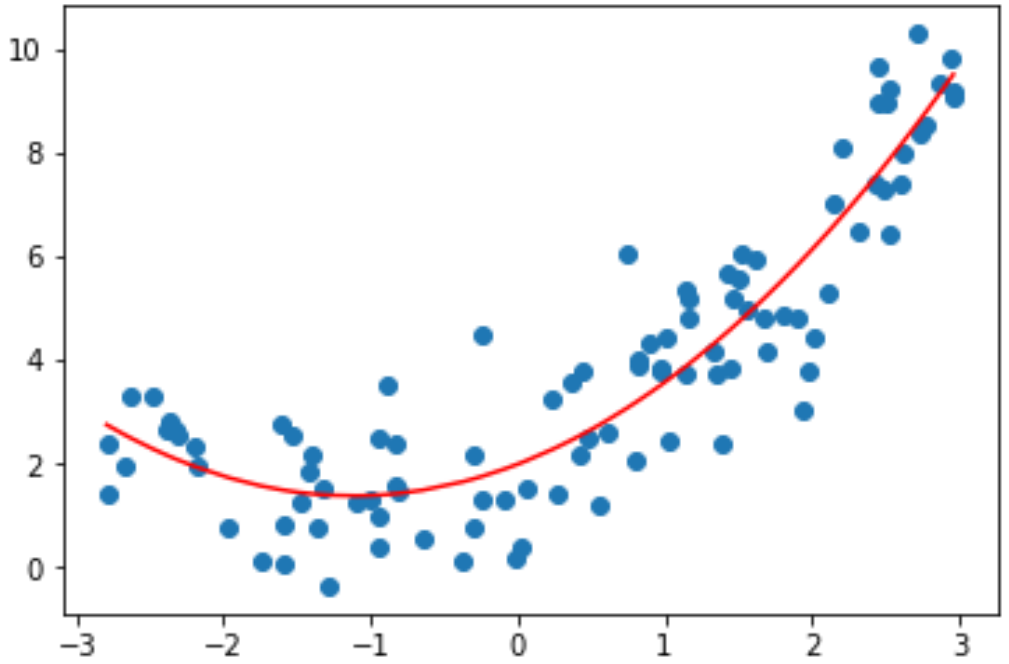
- 完成!　→ このGUI機能を後で活用します!



回帰分析 (Regression) とは?

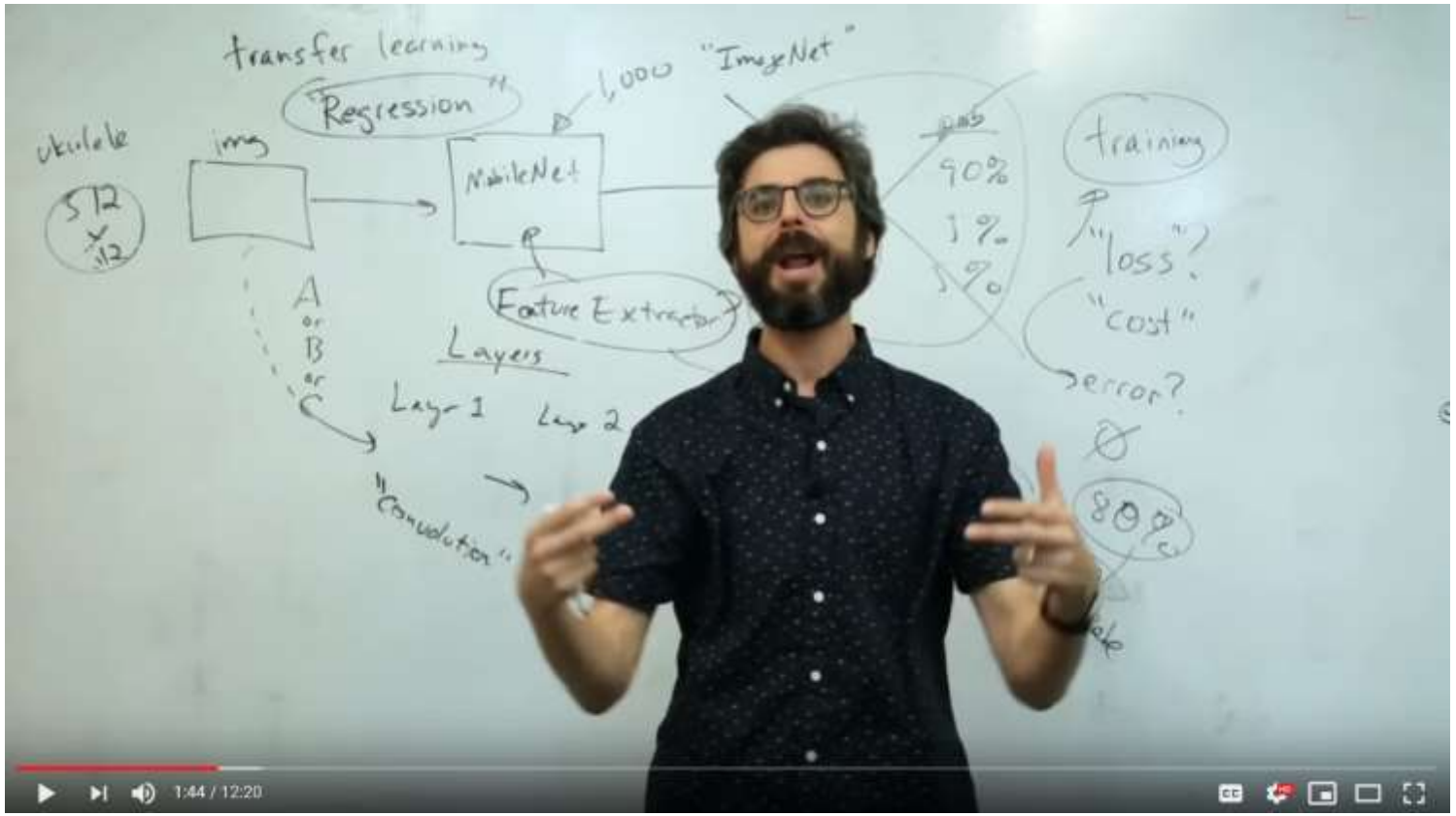
回帰分析 (Regression) とは?

- 回帰分析 (Regression) とは何か?
- 回帰分析：
  - 相関関係や因果関係があると思われる2つの変数のうち、一方の変数から将来的な値を予測するための予測式（回帰直線）を求めるための手法。



回帰分析 (Regression) とは?

- 前回の特徴量抽出によるクラス分けと回帰分析はどう違うのか
- 今回もDan Shiffman先生の解説を参考にしてみる
- https://youtu.be/aKgq0m1YjvQ



m5js 転移学習：特徴量抽出による画像の回帰分析

m5js 転移学習：特徴量抽出による画像の回帰分析

- 今回は、前回の「特徴量抽出による画像のクラス分け」のサンプルを改造します
- 以下からダウンロード (or コピペ)
- https://gist.github.com/tado/a27bf33ece7c96e1101c3ff330337c1e

sketch.js Raw

```
/*
 *  メディアアートプログラミング
 *  転移学習の特徴抽出による画像のクラス分けサンプル
 *  2019.10.23
 */

let featureExtractor;
let classifier;
let video;
let loss;
let dogImages = 0;
let catImages = 0;
let badgerImages = 0;
let status = ''; //現在の状態を左上に表示
let result = ''; //クラス分け結果を中央に表示

function setup() {
  //キャンバスを画面全体に
  createCanvas(windowWidth, windowHeight);
  video = createCapture(VIDEO);
  video.size(320, 240);
```

m5js 転移学習：特徴量抽出による画像の回帰分析

- 改造するポイント
  - キーによる操作ではなく、GUIのスライダーとボタンによる操作へ
  - クラス分けから回帰分析へ
    - featureExtractor.classification → featureExtractor.regression
  - スライダーの値と画像を組にして追加していく
  - ラベル名を表示するのではなく、回帰分析のスコアを表示するように

m5js 転移学習：特徴量抽出による画像の回帰分析

- sketch.js

```
let featureExtractor;
let regressor; //classifierをregressorに
let video;
let loss;
let status = ''; //現在の状態を左上に表示
let showResult = ''; //クラス分け結果を中央に表示
let slider; //スライダー
//ボタン
let addImageButton, trainButton, predictButton;

function setup() {
  createCanvas(windowWidth, windowHeight);
  video = createCapture(VIDEO);
  video.size(320, 240);
  video.hide();
  //特徴量抽出
  featureExtractor
  = ml5.featureExtractor('MobileNet', modelReady);
  const options = { numLabels: 3 };
  //classificationからrecressionへ
  regressor = featureExtractor.regression(video);
  //スライダーを配置
  slider = createSlider(0.0, 1.0, 0.5, 0.01);
  slider.position(20, 40);
  //ボタンを配置
  addImageButton = createButton('add image');
  addImageButton.position(slider.x+slider.width+20,40);
  addImageButton.mousePressed(addImage);
  trainButton = createButton('train');
  trainButton.position(addImageButton.x+addImageButton.
width+5,40);
  trainButton.mousePressed(train);
  predictButton = createButton('start predict');
  predictButton.position(trainButton.x+trainButton.widt
h + 5,40);
  predictButton.mousePressed(predict);
}
```

m5js 転移学習：特徴量抽出による画像の回帰分析

- sketch.js

```
function draw(){
  background(0);
  //ビデオ映像をフルクスリーン表示
  image(video, 0, 0, width, height);
  //現在の状態(status)を表示
  fill(255);
  textSize(12);
  textAlign(LEFT);
  text(status, 20, 20);
  //回帰分析の結果を表示
  fill(255, 255, 0);
  textSize(100);
  textAlign(CENTER);
  text(showResult, width/2, height/2);
}

//画像を追加
function addImage(){
  regressor.addImage(slider.value());
}
//訓練開始
function train(){
  regressor.train(function(lossValue) {
    if (lossValue) {
      loss = lossValue;
      status = 'Loss: ' + loss;
    } else {
      status = 'Done Training! Final Loss: ' + loss;
    }
  });
}
```

m5js 転移学習：特徴量抽出による画像の回帰分析

- 完成!!

特徴量抽出による画像の回帰分析 – 応用

特徴量抽出による画像の回帰分析 – 応用

- 回帰分析のコードを使って応用してみる

- 例えば…
- 回帰分析の結果で円の位置を動かしてみる

特徴量抽出による画像の回帰分析 – 応用

- sketch.js

```
// ... (前略)...

function setup() {
  createCanvas(windowWidth, windowHeight);
  video = createCapture(VIDEO);
  video.size(320, 240);
  video.hide();

  // ... (中略)...

  //円の初期位置
  circleX = width/2;
}
```

特徴量抽出による画像の回帰分析 – 応用

- sketch.js

```
function draw(){
  background(0);
  //ビデオ映像をフルクスリーン表示
  image(video, 0, 0, width, height);
  //現在の状態(status)を表示
  fill(255);
  textSize(12);
  textAlign(LEFT);
  text(status, 20, 20);
  //回帰分析の結果をもとに円を描画
  fill(255, 255, 63);
  noStroke();
  circle(circleX, height/2, 200);
}
```

特徴量抽出による画像の回帰分析 – 応用

- sketch.js

```
// ...（中略）...

function gotResults(err, result) {
  //エラー表示
  if (err) {
    console.error(err);
    status = err;
  }
  //クラス分けした結果を表示
  if (result && result.value) {
    showResult = result.value;
    circleX = map(result.value, 0.0, 1.0, width/4, width/4*3);
    predict();
  }
}
```

特徴量抽出による画像の回帰分析 – 応用

- 完成!!

今日はここまで!