

## 補足：自分のデータで分析するには

Wolfram言語でデータを分析するには、以下のような方法で手軽にWolframノートブック上のプログラムに挿入することができます。

- ・画像ファイルは、ファイルを「ドラッグ&ドロップ」で
- ・テキストファイルは、テキストエディタで開いてからテキスト部分を「コピー&ペースト」で

これらのいわゆる「ファイルの貼り付け」の操作は、Wolfram Cloud の無料アカウントで利用できます。そのため、本書で紹介しているデータを使った各種の分析は、基本的に無料アカウントで実行可能だと考えてよいでしょう。

ただし、音声データファイルに対しては、「ファイルの貼り付け」の方法が使えません。さらに、画像やテキストファイルについても、Wolfram言語を使い込んでいくにつれて、ファイルの数が増えたり、データのサイズが大きくなると、「ファイルの貼り付け」の操作に時間がかかり、面倒になってきます。

そういうときはプログラミングで、**Import** 関数を使ってファイルを読み込みましょう。Wolfram言語でファイルを使ってデータ分析するためには、用意した画像や音声やテキストのファイルをプログラムの実行環境 (Wolfram Cloud) にアップロードし、**Import**関数を使ってファイルを読み込みます。アップロードの操作には、Wolfram Cloud の有料アカウントが必要ですが、まずは『Wolfram 評価版』を利用して、アップロードの操作を体験してみてください。

この資料では、以下の内容を説明します。

1. Wolfram Cloudの有料アカウントについて
2. ファイルやフォルダをWolfram Cloudにアップロードする
3. ファイルのパスを指定する
4. **Import**関数でファイルを読み込む
5. 複数のファイルを **Import**関数でまとめて読み込みたいときは

### 1. Wolfram Cloudの有料アカウントについて

Wolfram Cloudの利用には、無料のBasicプランと有料のプランがあります。

Basic プランでは「できない」こと(制約)のひとつとして、『Wolfram Cloudのストレージには、お手元にあるファイルをアップロードすることができない(機能上の制約 -1)』があります。

このため、**Import**関数を使ってファイルを読み込むには、Wolfram Cloud の有料アカウントが必要です。

有料アカウントをお持ちでない方は、『Wolfram 評価版』をご利用ください。

本書のviiiページに以下の内容を記載しています。

- Wolfram 評価版のご案内 ●
- Wolfram Cloudのプランについて ●

評価版の期間 (15日間) 終了後に、Wolframのライセンス購入等についてご質問のある場合は、以下のサイトの一番下に記載されている連絡先 (ウルフラムリサーチアジアリミティッドまたは販売代理店) へお問い合わせください。

<https://www.wolfram.com/japan/>

## 2. ファイルやフォルダをWolfram Cloudにアップロードする

ここでは、本書サポートページで、サンプルデータとして提供している画像データファイルの中から、cats フォルダの猫の画像ファイルを例として、以下に手順を示します。


まず、本書サポートページからSampleDataFiles.zipをダウンロードし、解凍しておきましょう。

また、本書サポートページで提供しているWolframノートブックを[Make Your Own Copy]して作成したファイルは Copied Files フォルダに入っています。このノートブックでプログラムを実行する、として、以下の説明を行います。(\*1)

＜ファイルアップロードの手順＞（図1を参照）

①Cloud Files をクリックし、

②[Copied Files] をダブルクリックします。

③  Upload File(s)のボタンを押します。すると図1の右のようなウィンドウが開きます。

そのウィンドウに、ダウンロードしたChapter\_1\_Dataフォルダを表示させます。そのChapter\_1\_Data フォルダごと、Copied Filesにドラッグ&ドロップします。（Chapter\_1\_Dataフォルダをマウスでクリックしたまま、Copied Filesの場所に移動してクリックを外します）

すると、Copied FilesにChapter\_1\_Dataフォルダがアップロードされます。（アップロードが終わったら開いたウィンドウは「キャンセル」ボタンで閉じてください）

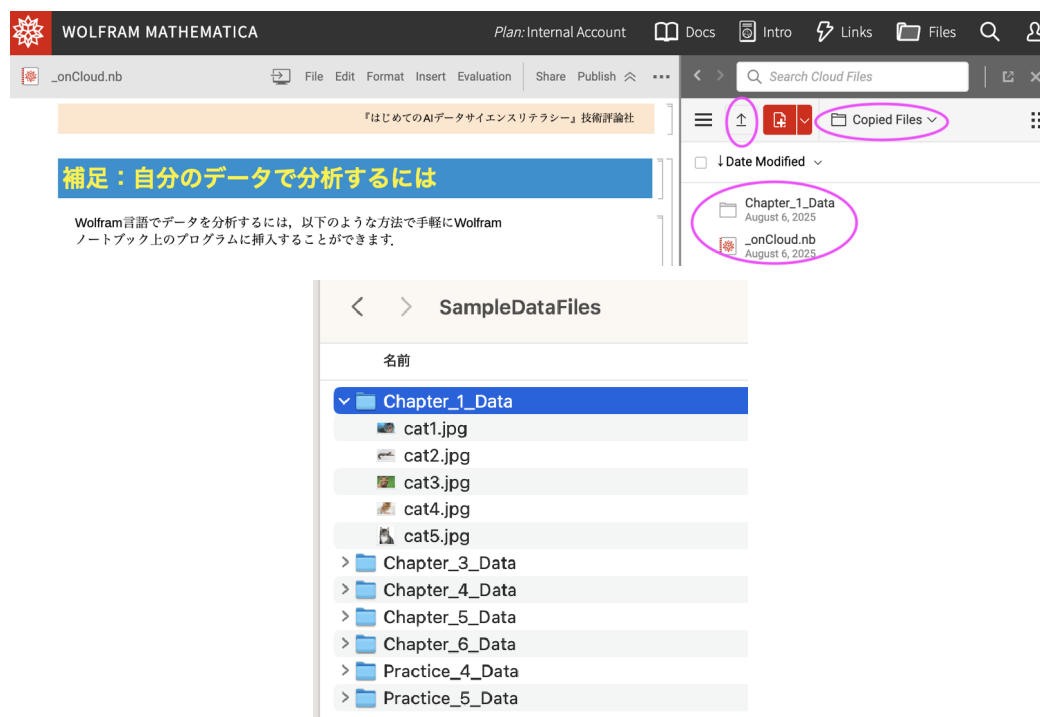


図1：Wolfram Cloudへのアップロードの手順

(\*1) ここではImport関数を実行したいノートブックが[Copied Files]フォルダにあることを想定して、以降の説明をします。もし、フォルダの場所が違う場合は、**Import**関数を実行するノートブックのある場所と、データファイルのある場所を確認してください。

### 3. ファイルのパスを指定する

ファイルをプログラムで読み込むには、そのファイルがどこにあるかをWolfram言語が知る必要があります。パスを指定すると、Wolfram言語は、指定されたパスの場所を起点としてファイルを探し、読み込みます。まず、**NotebookDirectory[]**を実行して、今自分が作業している教材のファイルがどこにあるかを確認します。

In[1]= **NotebookDirectory[]**  
ノートブックのディレクトリ

Out[1]=  
/wolframcloud/userfiles/8f2/8f2ce0ce-6edf-440d-a4e8-4ad0aa62697d/Copied Files/

このファイルは今「Copied Files」というフォルダの中にあることが確認できます。次に、**SetDirectory**関数で、この今いるフォルダをパスに指定します。

In[2]= **SetDirectory[NotebookDirectory[]]**  
ディレクトリ... ノートブックのディレクトリ

Out[2]=  
/wolframcloud/userfiles/8f2/8f2ce0ce-6edf-440d-a4e8-4ad0aa62697d/Copied Files

上記の「1.ファイルやフォルダをWolfram Cloudにアップロードする」で、catsフォルダを Copied Files フォルダにアップロードしました。**FileNames**関数を使って、catsフォルダの中にあるファイルを見てみましょう。

アスタリスク(\*)はcatsフォルダの中にあるもの全て(なんでも)という意味です。

以下のプログラムを実行して、5枚のファイル名が正しく出力されたらOKです。

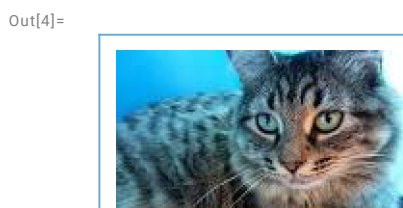
In[3]= **FileNames["Chapter\_1\_Data/\*"]**  
ファイル名

Out[3]=  
{Chapter\_1\_Data/cat1.jpg, Chapter\_1\_Data/cat2.jpg, Chapter\_1\_Data/cat3.jpg, Chapter\_1\_Data/cat4.jpg, Chapter\_1\_Data/cat5.jpg}

### 4. Import関数でファイルを読み込む

それでは、画像ファイルcat1.jpgを読み込んでみましょう。**Import**関数でファイル名を与えて実行します。ファイル名はダブルクォーテーションで囲みます。このとき、neko1=Import[...]のように、変数名に代入すると、その後の分析がしやすくなります。

In[4]= **neko1 = Import["Chapter\_1\_Data/cat1.jpg"]**  
インポート



猫の画像が表示され、正しく読み込んだことがわかります。

### 🔊音声ファイルやテキストファイルの読み込みの例

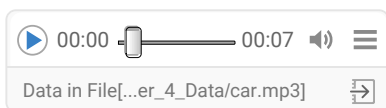
ここまでの手順では、画像ファイルを例に具体的な方法を説明しましたが、画像ファイルのほか、音声ファイル、テキストファイル、あるいは、エクセルファイルなどの数値や文字が混じったファイルを読み込む場合も、同じ方法で行うことができます。

以下では、上記1～3の手順でそれぞれのフォルダをインポートできる状態になっていることを前提に、ファイルを読み込む手順を示します。

■ 音声ファイル：Import関数でファイル名を与え、変数car に代入します。

```
In[5]= car = Import["Chapter_4_Data/car.mp3"]
      インポート
```

Out[5]=



■ テキストファイル：Import関数でファイル名を与え、変数hashirewakachi に代入します。

```
In[6]= hashirewakachi = Import["Chapter_6_Data/hashire_wakachi.txt"]
      インポート
```

Out[6]=

メロス は 激怒 し た 。 必ず 、 かの 邪智 暴虐 の 王 を 除か なけれ ば なら ぬ と 決意 し た 。 メロス に は 政治 が わから ぬ 。 メロス は 、 村 の 牧人 で ある 。 笛 を 吹き 、 羊 と 遊ん で 暮し て 来 た 。 けれど も 邪悪 に 対し て は 、 人 一 倍 に 敏感 で あっ た 。 きょう 未明 メロス は 村 を 出発 し 、 野 を 越え 山越え 、 十 里 は なれ た 此 この シラクス の 市 に やっ て 来 た 。 メロス に は 父 も 、 母 も 無い 。 女房 も 無い 。 十 六 の 、 内気 な 妹 と 二 人 暮し だ 。

■ エクセルファイル：Import関数でファイル名を与え、変数test に代入します。

```
In[7]= test = Import["Chapter_7_Data/testsampledata.xls"]
      インポート
```

Out[7]=

```
{{ {学籍番号, 数学, 国語, 英語}, {10 001., 67., 55., 42.},
  {10 002., 42., 50., 33.}, {10 003., 55., 42., 57.}, {10 004., 87., 82., 35.},
  {10 005., 92., 84., 30.}, {10 006., 60., 70., 54.}, {10 007., 35., 40., 70.},
  {10 008., 46., 32., 82.}, {10 009., 72., 68., 42.}, {10 010., 95., 98., 30.}}}
```

## 複数のファイルをImport関数でまとめて読み込みたいときは

Import関数を使って、複数枚のファイルを一度に読み込んでリストにしておくと、機械学習のデータとして扱うなど、分析時にも複数のデータをまとめて扱うことができ便利です。ここでは、ラベル（名前）なしでファイルを読み込む方法と、本書の第5章で学ぶ教師あり機械学習の訓練データ用に、ラベル（名前）付きで読み込む方法を説明します。ここでは、例として画像ファイルで説明していますが、音声データファイルでも手順は同じ(\*2)です。

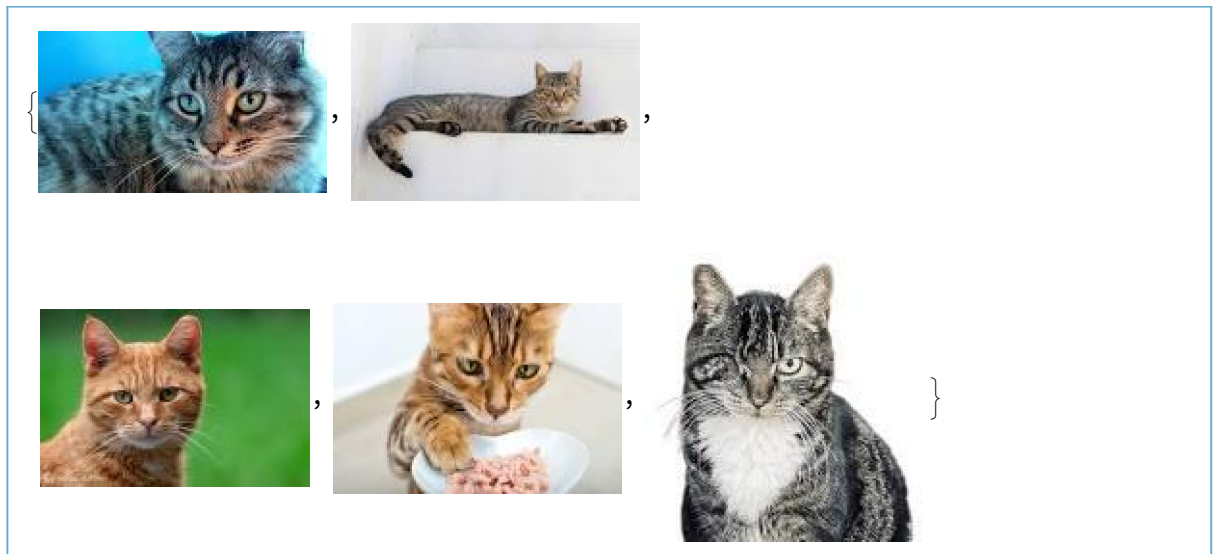
### (1) ラベルなしで読み込む方法

以下のプログラムは、Chapter\_1\_Dataフォルダの中にあるすべての画像ファイルを読み込みます。このプログラムを実行すると、5枚の画像データのリストが **cats** という変数に代入されました。

```
In[8]= cats = Map[Import[#] &, FileNames["Chapter_1_Data/*"]]
```

適用    インポート    ファイル名

Out[8]=



ここで使われている **Import[#]&** は「純関数(\*3)」という方法で書かれたプログラムです。  
**#**の中に**FileNames["Chapter\_1\_Data/\*"]**を実行して得られるリストの要素ひとつひとつが入ります。  
 (上記3を参照)  
**\***は「ワイルドカード」といい、**"Chapter\_1\_Data/\*"**とすることで、「Chapter\_1\_Dataフォルダに入っている全てのもの」という意味になります。  
**Map**関数(\*2)は、第2引数のリストの要素全てに対して、第1引数の関数を適用するという関数です。

(\*2) 「機械学習」や「ラベル」についての説明は、「第5章 AIのしくみ」を参照してください。

(\*3) 純関数やMap関数の詳細については、本書のコラムで紹介しているオンラインチュートリアル「Wolfram言語：プログラミングの基本」の対象ページを参照してください。

・純関数について    ・Map、関数の適用

## (2) ラベル付きで読み込む方法

上のラベルなしのときのプログラムに、ラベルを付ける部分だけを追加したプログラムです。  
 上のプログラムでは、`Import[#]&`となっていた部分が、`Import[#]->"ねこ"`に変わっているだけです。これで、画像ファイルひとつひとつに“ねこ”というラベルが付いたリストを 変数 `trainingcats` に代入することができます。

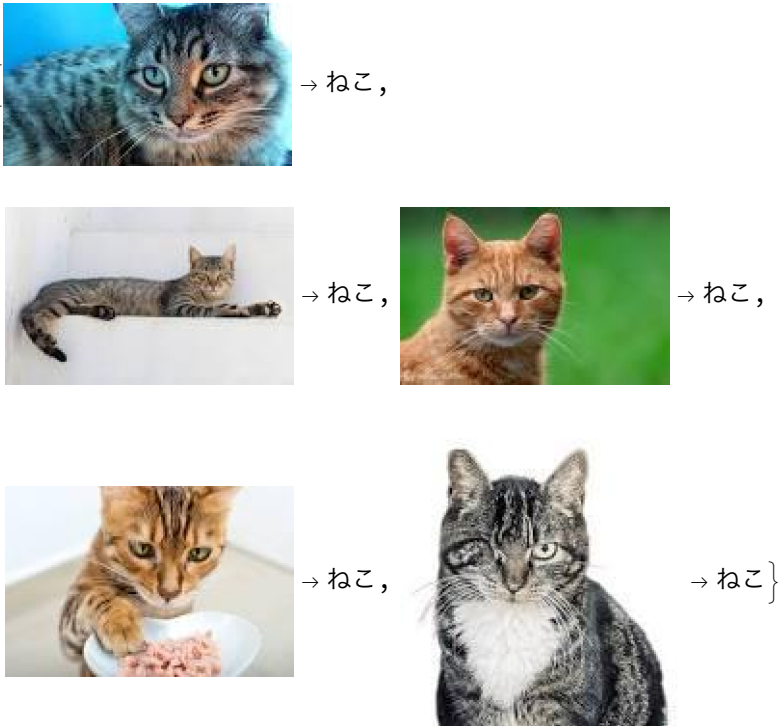
In[9]=

```
trainingcats = Map[Import[#] -> "ねこ" &, FileNames["Chapter_1_Data/*"]]
```

適用 インポート

ファイル名

Out[9]=



→ ねこ,

→ ねこ,

→ ねこ,

→ ねこ,

→ ねこ }

### 💡 読み込むデータ数が多い場合の注意点

ここでは、わかりやすくするために、読み込んだ画像やラベルの付いたリストをすべて表示していますが、データ数が多い場合は、プログラムの最後に `;` (セミコロン) をつけて、画面には読み込んだデータを表示しないようにしたほうがよいです。たくさんの画像や音声のデータを画面上に表示させるだけでもパワーが必要なので、プログラムがスムーズに動かなくなってしまうからです。