令和７年度「地方やデジタル分野における専修学校理系転換等推進事業」

ビッグデータ技術応用教材資料１

本ビッグデータ技術応用教材資料は、文部科学省の教育政策推進事業委託費による委託事業として、一般社団法人全国専門学校情報教育協会が実施した令和７年度「地方やデジタル分野における専修学校理系転換等推進事業」の成果物です。

情報成長分野の教育プログラム整備と教員育成による学科転換・新設推進事業

目　次

ビッグデータ技術応用 3
はじめに 3

ビッグデータ技術応用__実習１ 9
導入編 9
Part1 15
Part2 24
Part3 34
Part4 43

ビッグデータ技術応用_実習２ 51
導入編 51
Part1 56
Part2 66
Part3 75
Part4 85

ビッグデータ技術応用_実習３ 94
導入編 94
Part1 99
Part2 107
Part3 116
Part4 124

ビッグデータ技術応用_実習まとめ編 132

ビッグデータ技術応用

はじめに



ビッグデータ技術応用 実習ガイド

ビッグデータ技術応用

この教材は、『ビッグデータ技術基礎』で学んだ理論を、実際に手を動かして確認するための**実践教材**です。基礎教材では全7章で構成された内容で、ビッグデータ分析の理論や仕組みを体系的に学習しましたが、**『ビッグデータ技術応用』**では、それらの理論を３つの実習で、**「理解」を「実践」に変える**ことを目指します。

『ビッグデータ技術基礎』（理論・知識の学習）
- 第1章 ビッグデータとその活用
- 第2章 データ分析基礎
- 第3章 スクリプト言語による分析
- 第4章 ツールとモニタリング
- 第5章 列指向ストレージによる高速化
- 第6章 データマートの基本構造
- 第7章 大規模分散処理

『ビッグデータ技術応用』
（実践・体験による理解の深化）
- 実習１ 分析実践
- 実習２ 業務活用
- 実習３ 大規模処理

学習のゴール

ビッグデータ技術応用

- ✓ 『ビッグデータ技術基礎』で学んだ知識を**「使えるスキル」**に変える
- ✓ ビジネス課題をデータで解決する**「実践的な流れ」**を体験する
- ✓ 大規模データを扱うための**「基盤技術」**の基礎を習得する



実習構成

ビッグデータ技術応用

実習 1 分析実践
Pythonによる
顧客データ分析と
レポーティング

→

実習 2 業務活用
アクセスログETL処理
とBIツールでの可視化

→

実習 3 大規模処理
Sparkによる
大規模データ集計と
高速化

ビッグデータ技術基礎		ビッグデータ技術応用		
第1章 ビッグデータとその活用	→	実習 1		
第2章 データ分析基礎	→	実習 1		
第3章 スクリプト言語による分析	→	実習 1	実習 2	
第4章 ツールとモニタリング	→		実習 2	
第5章 列指向ストレージによる高速化	→			実習 3
第6章 データマートの基本構造	→		実習 2	
第7章 大規模分散処理	→			実習 3

実習で使用する主なツール・技術

ビッグデータ技術応用

Python (pandas, matplotlib)：データ分析のコア言語・ライブラリ

Google Colaboratory：環境構築不要のPython実行環境

Tableau / Power BI：データを可視化するBIツール

Apache Spark　：大規模データを高速処理する分散処理フレームワーク

Docker　：Spark環境をPC上で動かすための基盤技術

ポイント
すべて無料で利用できるツール・環境を選定しています

実習の進め方

ビッグデータ技術応用

導入ガイド　：実習の概要、環境準備、データの入手方法等

オンライン講義（動画）　：各Partの解説とデモ

ワークブック　：実際にコードや考察を記入する教材

発展課題　：学んだ知識を試す追加課題

実習ステップ

ビッグデータ技術応用

準備	**導入ガイド**を読み、環境と機材を準備する
視聴	**オンライン講義**（動画）を視聴する
実践	**ワークブック**のタスクに沿って、自分で配線やコード実行する
記録	**ワークブック**に実行結果や気づきを記録する
考察	**考察問題**に挑戦し、自分の言葉でまとめる
挑戦	**発展課題**でさらにスキルを磨く

ビッグデータ技術応用

「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」

実習１ 分析実践
実習２ 業務活用
実習３ 大規模処理

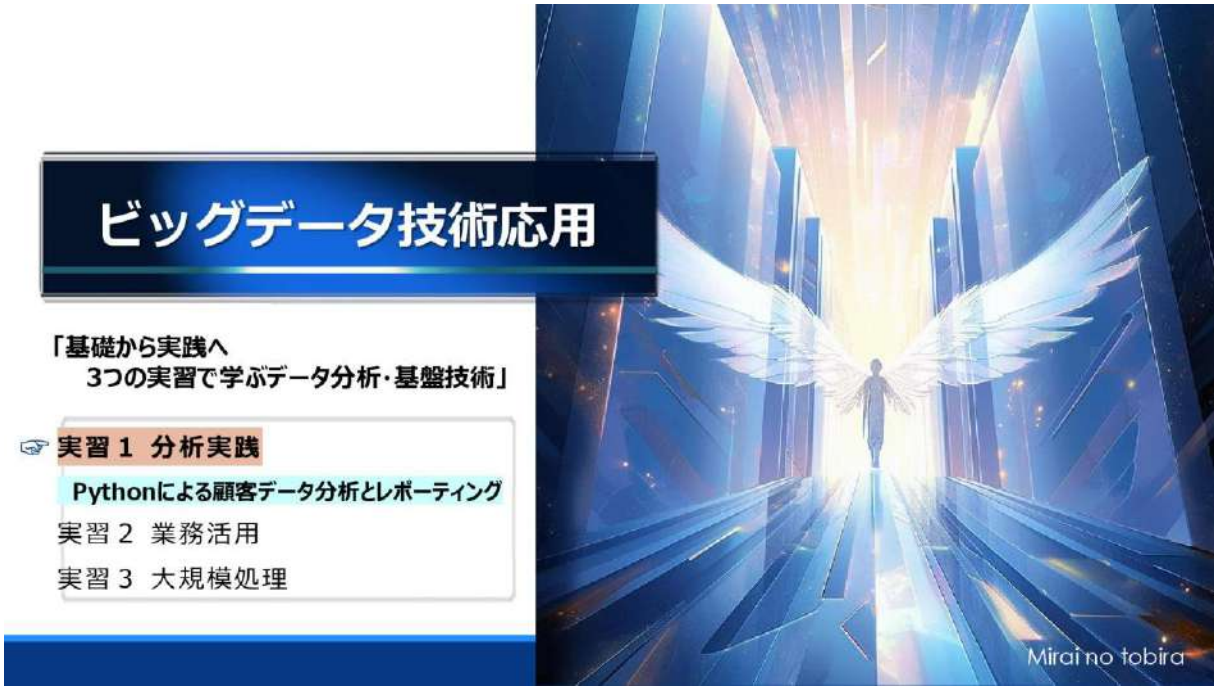



NEXT
ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
実習２ 業務活用
実習３ 大規模処理
Mirai no tobira

ビッグデータ技術応用__実習１

導入編


ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
実習２ 業務活用
実習３ 大規模処理
Mirai no tobira


ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
Pythonによる顧客データ分析とレポーティング
実習２ 業務活用
実習３ 大規模処理
Mirai no tobira

実習構成

ビッグデータ技術応用

実習 1 分析実践
Pythonによる
顧客データ分析と
レポーティング

実習 2 業務活用
アクセスログETL処理と
BIツールによる可視化

実習 3 大規模処理
Spark による
大規模データ処理

	ビッグデータ技術基礎		ビッグデータ技術応用		
第1章	ビッグデータとその活用	→	実習 1		
第2章	データ分析基礎	→	実習 1		
第3章	スクリプト言語による分析	→	実習 1	実習 2	
第4章	ツールとモニタリング	→		実習 2	
第5章	列指向ストレージによる高速化	→			実習 3
第6章	データマートの基本構造	→		実習 2	
第7章	大規模分散処理	→			実習 3

学習のゴール

ビッグデータ技術応用

「データ分析の一連のプロセスを、すべて自分の手で体験する」

- ✓ **データを読み込み** （Part 1）
- ✓ **データをきれいにし** （Part 2）
- ✓ **集計・可視化して特徴を探す** （Part 3）
- ✓ **レポートにまとめる** （Part 4）

… この**全工程を体験**します



利用するデータ

ビッグデータ技術応用

KAGGLE:データ分析コンペのサイトで公開されている Eコマースデータ

- オンラインショップの 約54万件に及ぶ実際の購買履歴データ
- 「誰が」「いつ」「何を」「いくつ」「いくらで」買ったのかが記録されている

☞ このデータから、「優良顧客は誰か？」というテーマで分析を進めます

利用するツール

ビッグデータ技術応用

分析には、**Python** と **pandasライブラリ** を使う

実行環境は、**Google Colaboratory** を推奨

- 環境構築なしに Python を実行できる無料のサービス
- 自分のPCに Python をインストール不要



実習 1 分析実践

ビッグデータ技術応用

実習 1 分析実践
Pythonによる顧客データ分析とレポーティング

実習 2 業務活用

実習 3 大規模処理

Part 1：環境構築とデータ理解
Part 2：データクレンジングと前処理
Part 3：探索的データ分析と可視化
Part 4：統合分析（RFM）とレポート作成



実習の進め方

ビッグデータ技術応用

導入ガイド	： 実習の概要、環境準備、データの入手方法等
オンライン講義（動画）	： 各Partの解説とデモ
ワークブック	： 実際にコードや考察を記入する教材
発展課題	： 学んだ知識を試す追加課題


ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
実習２ 業務活用
実習３ 大規模処理
Mirai no tobira


NEXT
ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
実習２ 業務活用
実習３ 大規模処理
Mirai no tobira

ビッグデータ技術応用__実習１

Part1


ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
Pythonによる顧客データ分析とレポーティング
実習２ 業務活用
実習３ 大規模処理
Mirai no tobira


ビッグデータ技術応用
実習１ 分析実践
Pythonによる顧客データ分析とレポーティング
Part １：環境構築とデータ理解
Part ２：データクレンジングと前処理
Part ３：探索的データ分析と可視化
Part ４：統合分析（RFM）とレポート作成
Mirai no tobira

実習 1 分析実践

ビッグデータ技術応用

実習 1 分析実践
Pythonによる
顧客データ分析と
レポーティング

実習 2 業務活用

実習 3 大規模処理

Part 1：環境構築とデータ理解

Part 2：データクレンジングと前処理

Part 3：探索的データ分析と可視化

Part 4：統合分析（RFM）とレポート作成



Part 1 ：環境構築とデータ理解

実習 1 分析実践

データ分析の第一歩：まずは敵を知る





Part１：環境構築とデータ理解 の学習目標

実習１ 分析実践

- ✓ Google Colab で Python環境を準備できる
- ✓ pandas で CSVデータを読み込める
- ✓ データの全体像（行数、列数、型）を把握できる
- ✓ 基本統計量（平均、最小値など）の意味を理解できる
- ✓ データに潜む「問題点」（欠損値、異常値）を発見できる

実習１ Part１の進め方

実習１ 分析実践

４つのタスク

- ➢ **タスク1：環境構築**　（分析の「作業場」準備）
- ➢ **タスク2：データ読み込み**　（分析対象をセット）
- ➢ **タスク3：データ構造の把握**　（「戸籍謄本」を見る）
- ➢ **タスク4：基本統計量の算出**　（「健康診断」をする）

環境の選択

実習 1 分析実践 Part 1

タスク1：Python環境の選択

	Google Colaboratory (推奨)	ローカル環境（Jupyter）
インストール	不要（ブラウザのみ）	必要 (Anaconda等)
必要なもの	Googleアカウント	Python, 各種ライブラリ
メリット	環境構築でつまずかない	オフライン作業可能

結論

この実習では Google Colaboratory を使います
環境構築の時間を節約し、分析に集中しましょう

Google Colaboratory を始めよう

実習 1 分析実践 Part 1

Google Colaboratory の始め方／4つのステップ

1. **https://colab.research.google.com** にアクセス
2. **Googleアカウントでログイン**
3. **「ファイル」→「ノートブックを新規作成」**
4. **ノートブック名を変更**（例：bigdata_ex1_part1.ipynb）

分析の『道具』を準備しよう

実習 1 分析実践 Part 1

ライブラリのインポート

```
# データ操作・分析
import pandas as pd
# 数値計算
import numpy as np
# グラフ描画
import matplotlib.pyplot as plt
import seaborn as sns
```


データの読み込み

実習 1 分析実践 Part 1

タスク2：Kaggle の CSVデータを読み込む

ステップ1：ファイルアップロード

```
from google.colab import files
uploaded = files.upload()
```

ステップ2：CSV を pandas で読み込む

```
# 読み込み（文字化け対策の encoding を指定）
df = pd.read_csv('data.csv', encoding='ISO-8859-1')
```

ポイント

encoding='ISO-8859-1' は、海外のデータ（特にヨーロッパ系）を扱う際の文字化けを防ぐ「おまじない」です

データの「チラ見」

実習 1 分析実践 Part 1

まずは『チラ見』する： .head()

✓ 54万行のデータのまずは先頭5行を見てどんな列があるか確認する

```
df.head()
```


データの「健康診断」

実習 1 分析実践 Part 1

タスク3：データの『健康診断』

① 形状確認 (.shape)：何行、何列？

② 基本情報 (.info())：データ型は？ 欠損値は？

数値データの統計量

実習 1 分析実践 Part 1

タスク4：数値データの『健康診断』

.describe() で、数値列の統計量（平均(mean)、最小(min)、最大(max)）を一覧表示します

実習 1 Part 1 のまとめ

実習 1 分析実践 Part 1

- ✓ **分析用の「生データ」は、そのままでは使えない**
- ✓ **問題①：欠損値**
  CustomerID が大量に抜けている
- ✓ **問題②：異常値（返品）**
  Quantity にマイナスの値がある
- ✓ **問題③：異常値（無料）**
  UnitPrice に 0.0 の値がある

Part 1：環境構築とデータ理解　ビッグデータ技術応用 実習 1

NEXT

実習１ Part２：データクレンジングと前処理

実習１ 分析実践

実習１ Part２では、
これらの「問題点」をすべて解決する
「データクレンジング（お掃除）」を行います

NEXT

ビッグデータ技術応用

実習１ 分析実践
Pythonによる顧客データ分析とレポーティング

Part １：環境構築とデータ理解
☞ **Part ２：データクレンジングと前処理**
Part ３：探索的データ分析と可視化
Part ４：統合分析（RFM）とレポート作成



ビッグデータ技術応用__実習１

Part2


ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
Pythonによる顧客データ分析とレポーティング
実習２ 業務活用
実習３ 大規模処理
Mirai no tobira


ビッグデータ技術応用
実習１ 分析実践
Pythonによる顧客データ分析とレポーティング
Part １：環境構築とデータ理解
Part ２：データクレンジングと前処理
Part ３：探索的データ分析と可視化
Part ４：統合分析（RFM）とレポート作成
Mirai no tobira

実習 1 分析実践

ビッグデータ技術応用

実習 1 分析実践
Pythonによる
顧客データ分析と
レポーティング

実習 2 業務活用

実習 3 大規模処理

Part 1：環境構築とデータ理解

☞ **Part 2：データクレンジングと前処理**

Part 3：探索的データ分析と可視化

Part 4：統合分析（RFM）とレポート作成



Part 2：データクレンジングと前処理

実習 1 分析実践

分析のための『お掃除』と『下ごしらえ』

Part 2：データクレンジングと前処理 の達成目標

実習 1 分析実践

- ✓ Part 1で見つけたデータの問題点を特定できる
- ✓ 欠損値（空白）を適切に処理（削除）できる
- ✓ 異常値（マイナス値など）をルールに基づいて処理できる
- ✓ データ型を分析しやすいように変換できる
- ✓ 分析に役立つ新しい列（特徴量）を作成できる

クレンジングの重要性

実習 1 分析実践

なぜデータクレンジングが必要？

➢ **GIGO** (Garbage In, Garbage Out)

「ゴミを入れたら、ゴミしか出てこない」

汚れたデータ (Garbage In)

- ▶ 欠損値　(CustomerID = ?)
- ▶ 異常値　(Quantity = -100)
- ▶ 型がバラバラ　(ID = 123.0)

間違った分析結果 (Garbage Out)

- ▶ 顧客数が正しく数えられない
- ▶ 売上合計がマイナスになる
- ▶ エラーで分析が止まる

Part 1 のおさらい

実習 1 分析実践 Part 2

Part 1で発見した問題点

- **問題①：欠損値**
  - `CustomerID` が約25%も空白 (NaN)
- **問題②：異常値** (Quantity)
  - `Quantity` の最小値がマイナス（返品データ）
- **問題③：異常値** (UnitPrice)
  - `UnitPrice` の最小値が 0.0（無料？）
- **問題④：データ型**
  - `CustomerID` が整数なのに小数 (float) になっている

今回のミッション
これら4つの問題をすべて解決し、分析できある「きれいなデータ」を作ります

データの準備

実習 1 分析実践 Part 2

タスク1：データの準備

Part 1 と同様に、ライブラリをインポートし、データを読み込みます

今回の重要ポイント：

元のデータを df_raw にコピーして保護する

```
# 元のデータをコピーして作業用DataFrameを作成
df_raw = df.copy()
```

問題①：欠損値の処理

実習 1 分析実践 Part 2

タスク2：欠損値の処理 (CustomerID)

分析目的：「優良顧客を見つける」

→ **顧客ID** (CustomerID) **が不明なデータは？**

→ 分析に使えない！ → **削除する**

```
# CustomerID が欠損している行を削除
df_cleaned = df.dropna(subset=['CustomerID'])
```


処理結果の確認

実習 1 分析実践 Part 2

必ず確認！処理前後の比較

➢ **処理を実行したら、必ず結果を確認する癖をつけましょう**

処理前 (df)
- ▶ 行数: 541,909 行
- ▶ CustomerID欠損: 135,080 件

処理後 (df_cleaned)
- ▶ 行数: 406,829 行
- ▶ CustomerID欠損: 0 件

ポイント
約13.5万行のデータが、今回の分析対象から除外されたことを確認します

問題②③：異常値の処理

実習 1 分析実践 Part 2

タスク3：異常値の処理 (Quantity, UnitPrice)

分析目的：「購入データ」の分析

→ **数量**(Quantity)**が 0 以下**（マイナス）のデータは？
→ **「返品」**データであり、**「購入」**ではない → **削除する**
→ **単価**(UnitPrice)**が 0 以下**のデータは？
→ **「売上」**に貢献しない（無料サンプル等）→ **削除する**

異常値処理のコード

実習 1 分析実践 Part 2

Pandasの『条件抽出』

条件に合う行だけを残す、という書き方を使います

```
# Quantity が 0 より大きいデータのみを抽出
df_cleaned = df_cleaned[df_cleaned['Quantity'] > 0]

# UnitPrice が 0 より大きいデータのみを抽出
df_cleaned = df_cleaned[df_cleaned['UnitPrice'] > 0]
```

問題④：データ型の変換

実習 1 分析実践 Part 2

タスク4：データ型の変換

欠損値がなくなったので、
CustomerID を「小数(float)」から「整数(int)」に変換できます

```
# .astype() を使って型を変換
df_cleaned['CustomerID'] =
df_cleaned['CustomerID'].astype('int64')
```

変換前		変換後
12345.0	➡	12345

特徴量の作成

実習 1 分析実践 Part 2

特徴量作成①：TotalPrice

分析に必要な**「合計金額」列**を、**既存の列から作成**します
これを **特徴量エンジニアリング** と呼びます

TotalPrice = Quantity × UnitPrice

```
df_cleaned['TotalPrice'] = df_cleaned['Quantity'] * df_cleaned['UnitPrice']
```

日付データの処理

実習１ 分析実践 Part２

特徴量作成②：日付データ

InvoiceDate 列（文字列）を**「日付型」に変換**し
便利な情報を抽出します

```
# 1. 文字列を日付型に変換
df_cleaned['InvoiceDate'] =
pd.to_datetime(df_cleaned['InvoiceDate'])

# 2. 日付型から「月」「曜日」「時間」などを抽出
df_cleaned['Month'] =
df_cleaned['InvoiceDate'].dt.month
df_cleaned['DayOfWeek'] =
df_cleaned['InvoiceDate'].dt.day_name()
df_cleaned['Hour'] = df_cleaned['InvoiceDate'].dt.hour
```


実習１ Part２のまとめ

実習１ 分析実践 Part２

「汚れた」生データから「きれいな」分析用データを作成しました

- ✅ **欠損値処理：.dropna() で不要な行を削除した**
- ✅ **異常値処理：条件指定 df[**（条件）**] で不要な行を削除した**
- ✅ **型変換　　：.astype() でデータ型を正しく直した**
- ✅ **特徴量作成：四則演算や .dt で新しい列を追加した**

🎯 **最も重要なポイント**
データクレンジングは「なんとなく」ではなく**「分析の目的に合わせて」**ルールを決めることが重要

NEXT

実習１ Part３：探索的データ分析と可視化

実習１ 分析実践

実習１ Part３では、
きれいになったデータを使って
分析とグラフ作成を行います

NEXT

ビッグデータ技術応用

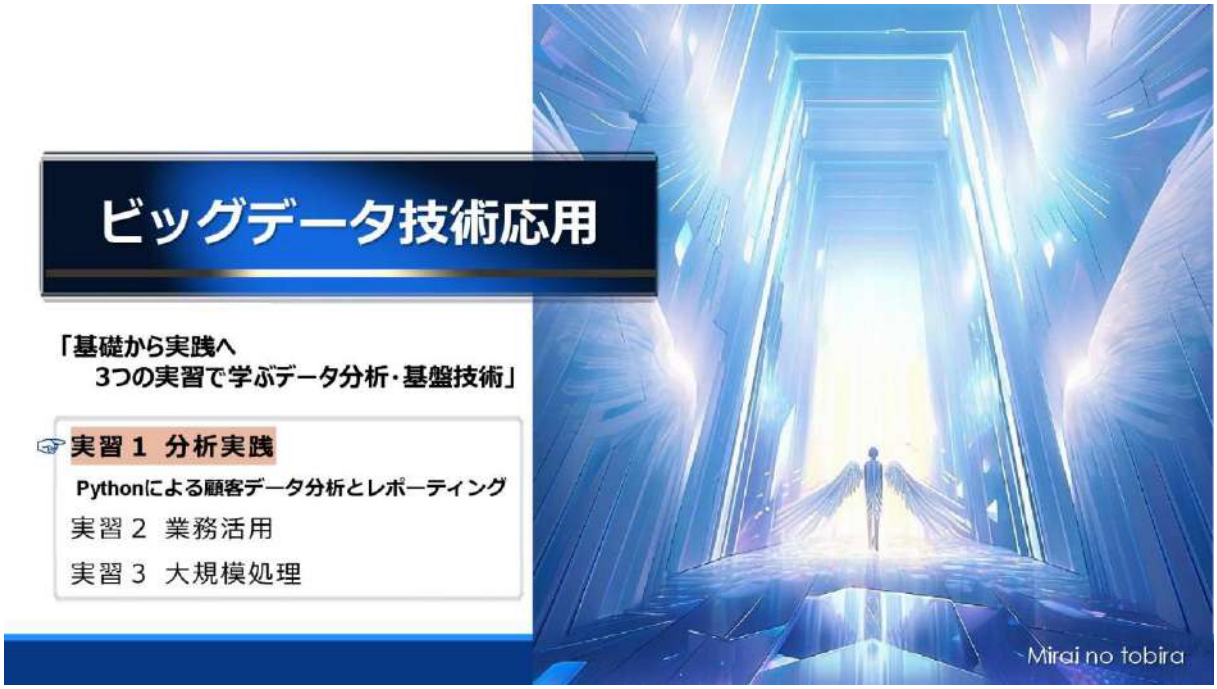
実習１ 分析実践
Pythonによる顧客データ分析とレポーティング

Part １：環境構築とデータ理解
Part ２：データクレンジングと前処理
Part ３：探索的データ分析と可視化
Part ４：統合分析（RFM）とレポート作成



ビッグデータ技術応用__実習１

Part3


ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
Pythonによる顧客データ分析とレポーティング
実習２ 業務活用
実習３ 大規模処理
Mirai no tobira


ビッグデータ技術応用
実習１ 分析実践
Pythonによる顧客データ分析とレポーティング
Part １：環境構築とデータ理解
Part ２：データクレンジングと前処理
Part ３：探索的データ分析と可視化
Part ４：統合分析（RFM）とレポート作成
Mirai no tobira

実習１ 分析実践

ビッグデータ技術応用

実習１ 分析実践
Pythonによる
顧客データ分析と
レポーティング

→ 実習２ 業務活用 → 実習３ 大規模処理

Part １：環境構築とデータ理解
Part ２：データクレンジングと前処理
☞ **Part ３：探索的データ分析と可視化**
Part ４：統合分析（RFM）とレポート作成



Part３ ：探索的データ分析と可視化

実習１ 分析実践

データと対話し、インサイト（発見）を引き出す

Part３：探索的データ分析と可視化 の達成目標

実習１ 分析実践

- ✓ データ分析の「問い」を立てる
- ✓ .groupby() でデータを自在に集計する
- ✓ matplotlib で折れ線グラフを作成し、トレンドを読む
- ✓ seaborn で棒グラフを作成し、ランキングを比較する
- ✓ グラフからビジネス的な「考察」を導き出す

分析の「問い」

実習１ 分析実践

データに尋ねる『問い』

Part２でデータはきれいになりました。しかし、ただ眺めていても答えは出ません
分析とは、データに**「具体的な問い」**を投げかけることです

Part３で答える「問い」：

- ▶ **問い①：全体の売上はどのように推移しているか？（トレンド分析）**
- ▶ **問い②：どの商品が一番売れているか？（売れ筋分析）**
- ▶ **問い③：どの国からの注文が多いのか？（顧客分布）**

最強の武器: .groupby()

実習 1 分析実践 Part 3

最強の集計武器： .groupby()

これらの「問い」に答えるため、**pandas の .groupby() を使います**

「○○ ごとに、△△ を □□ する」

- ▶ 例：「**月ごとに、売上** を **合計** する」
- ▶ 例：「**商品 ごとに、数量** を **合計** する」
- ▶ 例：「**国 ごとに、売上** を **合計** する」

問い①：売上トレンド

実習 1 分析実践 Part 3

タスク2：売上トレンドの分析

ステップ1：集計

```
# 「年月(YearMonth) ごとに、売上
(TotalPrice) を 合計(sum) する」
monthly_sales =
df_cleaned.groupby('YearMonth')
['TotalPrice'].sum()
```

ステップ2：可視化（折れ線グラフ）

```
# グラフのサイズ指定
plt.figure(figsize=(12, 6))
# 折れ線グラフを描画
plt.plot(monthly_sales.index.to_timestamp(),
monthly_sales.values, marker='o')
# タイトルやラベル
plt.title('Monthly Sales Trend')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.show()
```

Part 3 ：探索的データ分析と可視化　ビッグデータ技術応用 実習 1

考察①：グラフから読む

実習 1 分析実践 Part 3

グラフから何を読み取るか？

考察のポイント

- 傾向　：全体として売上は伸びているか？（→ 右肩上がりに見える）
- 特異点：特に目立つ月は？（→ 11月が突出して高い）
- 仮説　：なぜ11月が高い？（→ クリスマスの年末商戦が理由かも？）

分析とは、グラフを作ることではなく、グラフから「仮説」を立てることです

Seabornで可視化

実習 1 分析実践 Part 3

Seabornで棒グラフをきれいに描画

ランキングの可視化には**「棒グラフ」**が最適です

seaborn を使うと簡単に描画できます

```
plt.figure(figsize=(12, 7))
# 棒グラフ（barplot）を作成
sns.barplot(x=top_10_products.values, y=top_10_products.index)
plt.title('Top 10 Best-Selling Products')
plt.show()
```

問い③：国別売上

実習１ 分析実践 Part３

タスク4：国別売上の分析

集計コード：

```
# 「国(Country) ごとに、売上(TotalPrice) を 合計(sum) する」
country_sales =
df_cleaned.groupby('Country')['TotalPrice'].sum().sort_values(ascending=False)
```

集計結果（抜粋）：

United Kingdom	6,747,155.95
Netherlands	283,373.40
EIRE	250,001.38
Germany	208,091.21

UKを除外して可視化

実習１ 分析実践 Part３

UKを除外して比較する

```
# 'United Kingdom' を除外(.drop)して、上位10件を取得
top_10_excluding_uk = country_sales.drop('United Kingdom').head(10)

# 棒グラフで可視化
plt.figure(figsize=(12, 7))
sns.barplot(x=top_10_excluding_uk.values,
y=top_10_excluding_uk.index)
plt.title('Top 10 Countries by Sales (Excluding UK)')
plt.show()
```

実習１ Part３のまとめ

実習１ 分析実践 Part３

✓ きれいなデータに「問い」を投げかけ、
集計・可視化を行いました

- ☑ **集計** ：`.groupby()` が最強の武器
- ☑ **可視化**（トレンド） ：折れ線グラフ (`plt.plot`)
- ☑ **可視化**（ランキング）：棒グラフ (`sns.barplot`)
- ☑ **考察：** グラフから傾向や仮説を読み取ることがゴール

NEXT

Part４：統合分析（RFM）とレポート作成

実習１ 分析実践

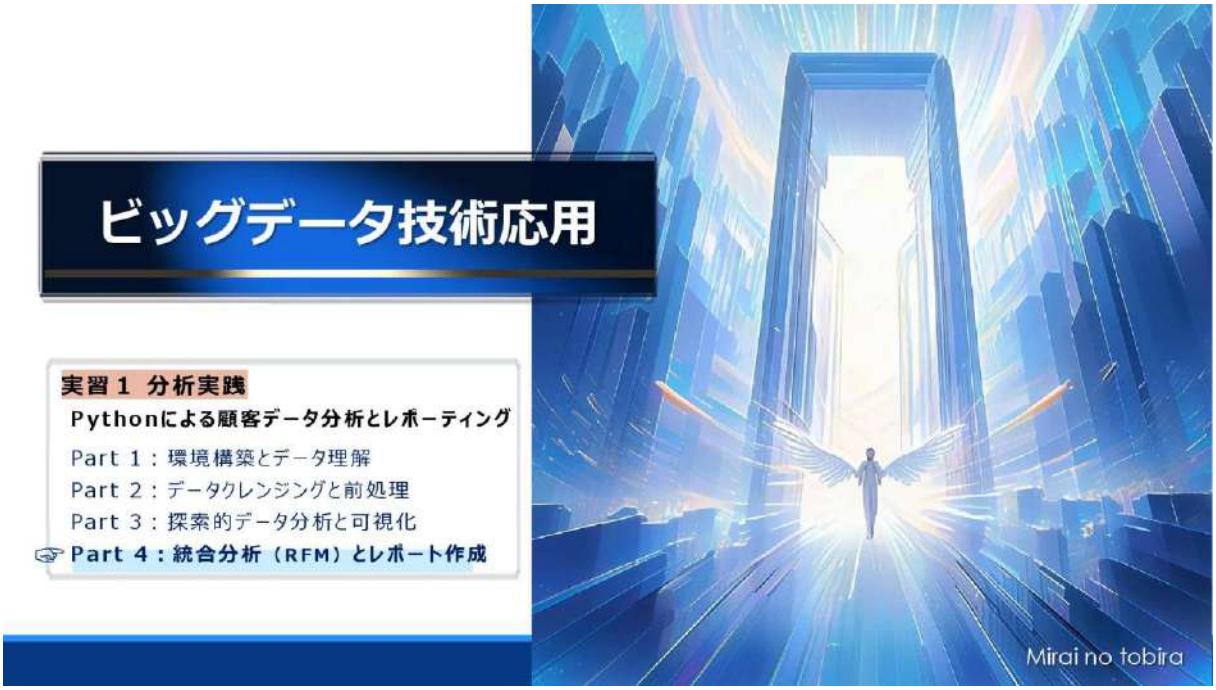
実習１ Part４では、
統合分析（RFM）とレポート作成
分析の「問い」を「顧客」に絞ります


NEXT
ビッグデータ技術応用
実習１ 分析実践
Pythonによる顧客データ分析とレポーティング
Part 1：環境構築とデータ理解
Part 2：データクレンジングと前処理
Part 3：探索的データ分析と可視化
Part 4：統合分析（RFM）とレポート作成
Mirai no tobira

ビッグデータ技術応用__実習１

Part4


ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
Pythonによる顧客データ分析とレポーティング
実習２ 業務活用
実習３ 大規模処理
Mirai no tobira


ビッグデータ技術応用
実習１ 分析実践
Pythonによる顧客データ分析とレポーティング
Part １：環境構築とデータ理解
Part ２：データクレンジングと前処理
Part ３：探索的データ分析と可視化
Part ４：統合分析（RFM）とレポート作成
Mirai no tobira

実習 1 分析実践

ビッグデータ技術応用

実習 1 分析実践
Pythonによる
顧客データ分析と
レポーティング

実習 2 業務活用

実習 3 大規模処理

Part 1：環境構築とデータ理解
Part 2：データクレンジングと前処理
Part 3：探索的データ分析と可視化
Part 4：統合分析（RFM）とレポート作成



Part 4：統合分析（RFM）とレポート作成

実習 1 分析実践

分析結果を価値に変える：顧客理解と提案

Part 4：統合分析（RFM）とレポート作成の達成目標

実習 1 分析実践

- ✓ 顧客分析のフレームワーク「RFM分析」を理解し、実装できる
- ✓ 顧客をスコアリングし「優良顧客」や「離反顧客」を特定できる
- ✓ セグメントごとの特徴を分析し、施策のヒントを得る
- ✓ 分析プロセス全体の結果を統合し、レポートとしてまとめる

RFM分析とは？

実習 1 分析実践

顧客を3つの視点で評価：RFM分析

顧客の価値を測るための、シンプルで強力なフレームワークです

R	F	M
Recency	Frequency	Monetary
最新購買日	購買頻度	購買金額
(最近来たか？)	(何回買ったか？)	(いくら使ったか？)

🎯 なぜRFMが重要？

「Rが高く、Fが高く、Mが高い顧客」＝ 最も重要な優良顧客
この顧客層を特定し、維持・育成することがビジネス成長の鍵です

RFM指標の計算

実習 1 分析実践 Part 4

タスク2-1：RFM指標の計算

顧客ごとにR, F, Mを計算します

```
# 基準日を設定（最終取引日の翌日）
snapshot_date = df_cleaned['InvoiceDate'].max() + dt.timedelta(days=1)

# 顧客ごとに集計
rfm_data = df_cleaned.groupby('CustomerID').agg({
'InvoiceDate': lambda date: (snapshot_date - date.max()).days, # Recency
'InvoiceNo': 'nunique', # Frequency
'TotalPrice': 'sum' # Monetary
})

# 列名変更
rfm_data.rename(columns={...}, inplace=True)
```


RFMスコアリング

実習 1 分析実践 Part 4

タスク2-2：RFMスコアの計算

R, F, Mの値を、それぞれ1～5点のスコアに変換します

pd.qcut() でデータを5等分します

Recency: 短いほど高スコア (5点)／**Frequency:** 多いほど高スコア (5点)／**Monetary:** 高いほど高スコア (5点)

```
# Recency（値が小さいほど良い）
r_labels = range(5, 0, -1)
rfm_data['R_Score'] = pd.qcut(rfm_data['Recency'], q=5, labels=r_labels, ...)
# Frequency, Monetary（値が大きいほど良い）
fm_labels = range(1, 6)
rfm_data['F_Score'] = pd.qcut(rfm_data['Frequency'].rank(...), q=5,
labels=fm_labels, ...)
rfm_data['M_Score'] = pd.qcut(rfm_data['Monetary'], q=5, labels=fm_labels, ...)
```

顧客セグメンテーション

実習 1 分析実践 Part 4

タスク2-3：RFMセグメントの作成

RとFのスコアを使って、顧客をグループ分け（セグメンテーション）します

セグメント例：

- ▶ Champions (R≥4, F≥4)：最優良顧客
- ▶ Loyal Customers (R≥3, F≥3)：優良顧客
- ▶ Potential Loyalists (R≥4, F<3)：育成候補
- ▶ At Risk Customers (R<3, F≥4)：離反危険
- ▶ Lost Customers (R<3, F<3)：離反顧客

セグメント分析

実習 1 分析実践 Part 4

タスク2-4：セグメント分析と可視化

作成したセグメントごとに、顧客数や平均購入額などを比較します

```
# セグメントごとに集計
segment_summary = rfm_data.groupby('Segment_Name').agg(...)

# 顧客数を棒グラフで表示
sns.countplot(y='Segment_Name', data=rfm_data, ...)
plt.show()
```

レポート作成のポイント

実習 1 分析実践 Part 4

タスク3：分析レポートを作成する

✓ 分析は「伝える」までがセット

レポート作成のポイント：

① **目的** ： なぜこの分析をしたのか？
② **データ** ： 何を使い、どう処理したか？
③ **発見** ： 何がわかったか？ (グラフとともに)
④ **提案** ： だから、どうすべきか？ (ネクストステップ)

実習 1 のまとめ

実習 1 分析実践

4つのPartを通して、データ分析の基本プロセスを完走しました

- ✅ **生データの理解と問題発見** (Part 1)
- ✅ **データのクレンジングと前処理** (Part 2)
- ✅ **集計と可視化によるインサイト発見** (Part 3)
- ✅ **RFM分析による顧客理解とレポート作成** (Part 4)

NEXT

実習２ 業務活用

ビッグデータ技術応用

実習１の経験を活かし

実習２ 業務活用（ETL処理とBIツール）

実習３ 大規模処理 （ Spark ）

へと進みましょう

NEXT

ビッグデータ技術応用

「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」

実習１ 分析実践

☞ **実習２ 業務活用**

アクセスログETL処理とBIツールによる可視化

実習３ 大規模処理



ビッグデータ技術応用_実習２

導入編


ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
実習２ 業務活用
実習３ 大規模処理
Mirai no tobira


ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
実習２ 業務活用
アクセスログETL処理とBIツールによる可視化
実習３ 大規模処理
Mirai no tobira

実習構成

ビッグデータ技術応用

実習１ 分析実践
Python による
顧客データ分析と
レポーティング

→

実習２ 業務活用
アクセスログETL処理と
BIツールによる可視化

→

実習３ 大規模処理
Spark による
大規模データ処理

ビッグデータ技術基礎		ビッグデータ技術応用		
第1章 ビッグデータとその活用	→	実習１		
第2章 データ分析基礎	→	実習１		
第3章 スクリプト言語による分析	→	実習１	**実習２**	
第4章 ツールとモニタリング	→		**実習２**	
第5章 列指向ストレージによる高速化	→			実習３
第6章 データマートの基本構造	→		**実習２**	
第7章 大規模分散処理	→			実習３

学習のステップ

ビッグデータ技術応用

実習１では、比較的整理されたCSVデータを扱ったが、実世界のデータは「生の」扱いにくい形をしていることが多く、実習２で扱う「アクセスログ」は、その代表例で、
Webサーバーが自動で記録したテキストデータ
ここから分析に必要な情報を取り出し(Extract)、使いやすい形に変換し(Transform)、最終的に分析ツールで読み込める形にする(Load)
この一連の流れを**「ETL処理」**と呼ぶ

✓ **実習２の前半では、Pythonを使ってこのETL処理を実践します**



実習で使用するツール

ビッグデータ技術応用

実習２の後半では、ETL処理で作ったデータを**「BIツール」**で可視化

Tableau (タブロー) と **Power BI** (パワービーアイ) という2つのツールを使う

これらのツールを使うと、プログラミングなしに、ドラッグ＆ドロップなどの

簡単な操作で、インタラクティブなグラフやダッシュボードを素早く作成できる

実習２ 業務活用

ビッグデータ技術応用

実習１ 分析実践 → **実習２ 業務活用**
アクセスログETL処理と
BIツールによる可視化 → 実習３ 大規模処理

Part１：アクセスログとETLの基礎

Part２：データ型の整備とデータマートの基礎

Part３：Tableau Publicによるダッシュボード作成

Part４：Power BI Desktopによるダッシュボード作成

▶ データ加工（Python）から可視化（BIツール）までを
一気通貫で体験できる

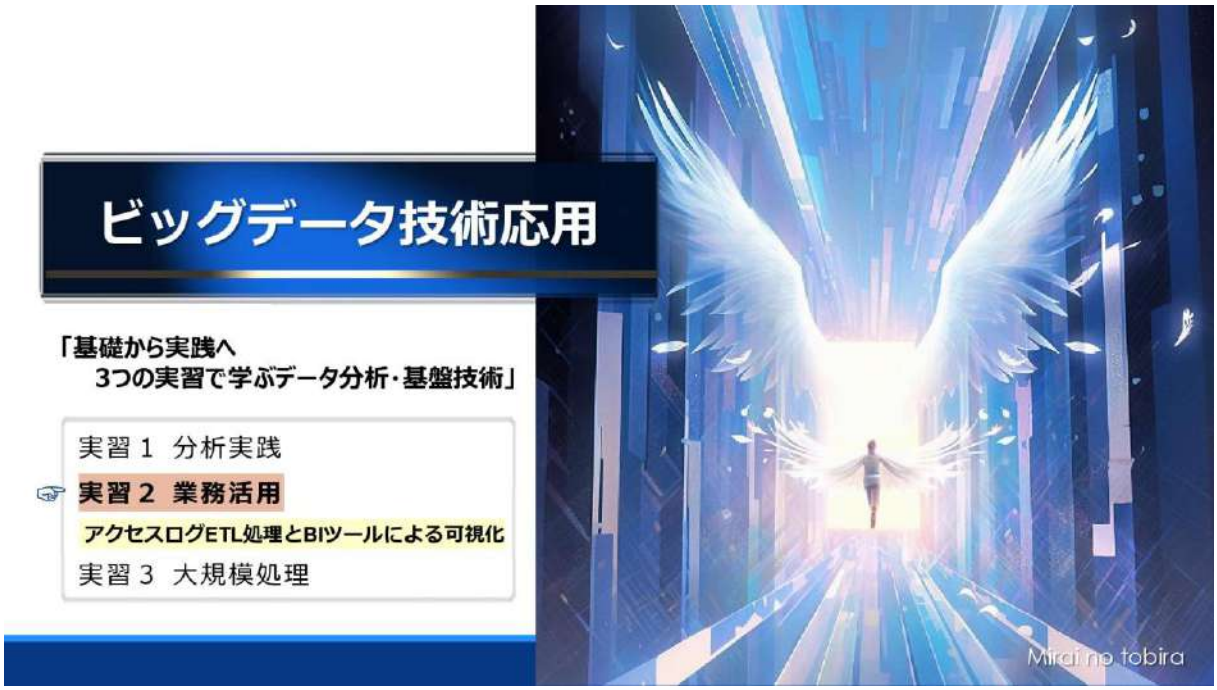


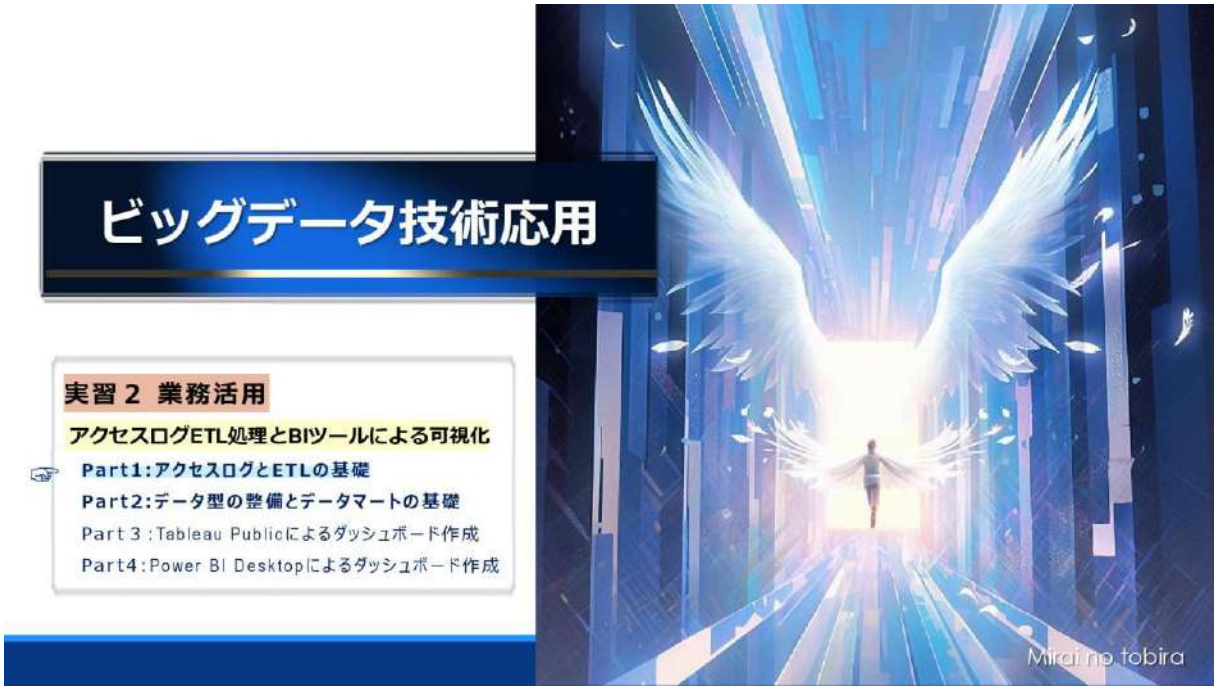

ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
実習２ 業務活用
実習３ 大規模処理
Mirai no tobira


NEXT
ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１ 分析実践
実習２ 業務活用
アクセスログETL処理とBIツールによる可視化
実習３ 大規模処理
Mirai no tobira

ビッグデータ技術応用_実習２

Part1


ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１　分析実践
実習２　業務活用
アクセスログETL処理とBIツールによる可視化
実習３　大規模処理
Mirai no tobira


ビッグデータ技術応用
実習２　業務活用
アクセスログETL処理とBIツールによる可視化
Part1:アクセスログとETLの基礎
Part2:データ型の整備とデータマートの基礎
Part３:Tableau Publicによるダッシュボード作成
Part4:Power BI Desktopによるダッシュボード作成
Mirai no tobira

実習２ 業務活用

ビッグデータ技術応用

実習１ 分析実践 → **実習２ 業務活用**（アクセスログETL処理とBIツールによる可視化） → 実習３ 大規模処理

☞ **Part１：アクセスログとETLの基礎**

Part２：データ型の整備とデータマートの基礎

Part３：Tableau Publicによるダッシュボード作成

Part４：Power BI Desktopによるダッシュボード作成



実習２ Part １：アクセスログとETLの基礎

実習２ 業務活用

Webサイトの足跡を宝に変える技術

Part 1：アクセスログとETLの基礎 の達成目標

実習 2 業務活用

- ✓ **Webアクセスログがどのような情報か理解する**
- ✓ **ETL**（Extract, Transform, Load）**の概念を理解する**
- ✓ **Pythonでテキストファイルを読み込み、行ごとに処理する方法を学ぶ**
- ✓ **正規表現を使ってログから必要な情報**（IP, 日時, URL等）**を抽出する**
- ✓ **抽出したデータを構造化**（DataFrame）**する方法を学ぶ**

Webアクセスログとは？

実習 2 業務活用

Webサーバーの『日記』：アクセスログ

Webサーバーに「誰が」「いつ」「どのページに」アクセスしたかの記録です

一般的な形式 (Common Log Format):

```
127.0.0.1 - - [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 200 2326
```

含まれる主な情報：

- ▶ IPアドレス　：アクセス元のコンピュータ
- ▶ タイムスタンプ　：アクセス日時
- ▶ リクエスト　：どのページ(URL)を見たか
- ▶ ステータスコード: 結果 (200=成功, 404=Not Found)
- ▶ 転送量　：送信したデータサイズタスク1：環境構築

ログデータの課題

実習２ 業務活用 Part 1

生のログデータは『扱いにくい』

アクセスログは宝の山ですが、そのままでは分析できません

- **非構造化データ**　：ただのテキスト。Excelで開いても意味不明
- **大量**　：人気サイトなら1日数十GB～TBになることも
- **形式が微妙に違うことも**：サーバーの種類や設定で形式が変わる

→ **分析できる「綺麗な形」に加工する必要がある**

ETLとは？

実習２ 業務活用 Part 1

データを『変身』させる魔法：ETL

扱いにくい生データを、分析しやすい綺麗なデータに加工するプロセスです

E: Extract 抽出	→	T: Transform 変換・加工	→	L: Load 書き出し
ログファイルから データを読み込む		必要な情報を取り出し 形式を整える		分析ツールで読み込める 形式で保存

実習2のゴール

このETLプロセスをPythonで実装し、最終的にBIツールで可視化できるデータ（データマート）を作成します

データの準備

実習２ 業務活用 Part１

タスク1：データの準備

実習では、サンプルアクセスログファイル (access_log.txt) を使います
（実際のログがない場合は、提供されるPythonスクリプトで生成します）

ステップ1：ファイル読み込み

```
# ファイルを開いて一行ずつ読み込む準備
file_path = 'access_log.txt'
with open(file_path, 'r') as f:
# 最初の数行を表示して確認
for i, line in enumerate(f):
if i < 5:
print(line.strip()) # strip()で改行文字を削除
else:
break
```


情報抽出の武器：正規表現

実習２ 業務活用 Part１

テキストから宝を探す：正規表現

ログのような「決まったパターン」を持つテキストから、
特定の情報（IPアドレス、日時など）を抜き出すための強力なツールです

例：IPアドレスを抜き出すパターン

```
r"(¥d{1,3}¥.¥d{1,3}¥.¥d{1,3}¥.¥d{1,3})"
```

- ¥d{1,3}：1～3桁の数字
- ¥.：ピリオド（特殊文字なので ¥ が必要）
- (...)：カッコで囲んだ部分を抽出

ポイント
正規表現は最初は難解に見えますが、ログ解析やテキスト処理には必須のスキルです。少しずつ慣れていきましょう

正規表現の実装 (Python)

実習２ 業務活用 Part１

タスク2：Pythonと正規表現で情報抽出

Pythonの re モジュールを使って、ログ1行から情報を抽出します

```
import re # 正規表現モジュール
# ログのパターンを定義 (Common Log Format用)
log_pattern = re.compile(r'(¥S+) (¥S+) (¥S+) ¥[([¥w:/]+¥s[+¥-]¥d{4})¥] "(¥S+)¥s?(¥S+)?¥s?(¥S+)?" (¥d{3}) (¥S+)')
# サンプルログ1行
line = '127.0.0.1 - - [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 200 2326'
# パターンにマッチさせる
match = log_pattern.match(line)
if match:
ip_address = match.group(1) # 1番目のカッコの中身
timestamp = match.group(4) # 4番目のカッコの中身
request_url = match.group(6) # 6番目のカッコの中身
status_code = match.group(8) # 8番目のカッコの中身
print(f"IP: {ip_address}, Time: {timestamp}, URL: {request_url}, Status: {status_code}")
```


正規表現の実装 (Python)

実習２ 業務活用 Part１

```
import re # 正規表現モジュール
# ログのパターンを定義 (Common Log Format用)
log_pattern = re.compile(r'(¥S+) (¥S+) (¥S+) ¥[([¥w:/]+¥s[+¥-]¥d{4})¥]
"(¥S+)¥s?(¥S+)?¥s?(¥S+)?" (¥d{3}) (¥S+)')
# サンプルログ1行
line = '127.0.0.1 - - [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 200 2326'
# パターンにマッチさせる
match = log_pattern.match(line)
if match:
ip_address = match.group(1) # 1番目のカッコの中身
timestamp = match.group(4) # 4番目のカッコの中身
request_url = match.group(6) # 6番目のカッコの中身
status_code = match.group(8) # 8番目のカッコの中身
print(f"IP: {ip_address}, Time: {timestamp}, URL: {request_url}, Status: {status_code}")
```

Part 1：アクセスログとETLの基礎　ビッグデータ技術応用 実習２

全ログデータの処理

実習２ 業務活用 Part１

タスク3：全ログデータをループ処理

ログファイル全体を1行ずつ読み込み、正規表現で情報を抽出し、リストに格納していきます

```
parsed_data = [] # 結果を格納するリスト
with open(file_path, 'r') as f:
for line in f:
match = log_pattern.match(line)
if match:
# 抽出した情報を辞書形式でリストに追加
data = { 'ip': match.group(1),
'time': match.group(4),
'url': match.group(6),
'status': match.group(8)
# ... 他の情報も必要なら追加 ...}
parsed_data.append(data)
print(f"処理完了！ {len(parsed_data):,} 件のログを解析しました。")
```


全ログデータの処理

実習２ 業務活用 Part１

```
parsed_data = [] # 結果を格納するリスト
with open(file_path, 'r') as f:
for line in f:
match = log_pattern.match(line)

if match:
# 抽出した情報を辞書形式でリストに追加
data = {
'ip': match.group(1),
'time': match.group(4),
'url': match.group(6),
'status': match.group(8)

# ... 他の情報も必要なら追加 ...
}
parsed_data.append(data)
print(f"処理完了！ {len(parsed_data):,} 件のログを解析しました。")
```

Part 1：アクセスログとETLの基礎 ビッグデータ技術応用 実習２

DataFrame への変換

実習２ 業務活用 Part１

タスク4：構造化データ (DataFrame) へ

解析結果（辞書のリスト）を、pandas DataFrame に変換します
これでやっと分析できる形になります

```
import pandas as pd

# リストからDataFrameを作成
log_df = pd.DataFrame(parsed_data)

# 結果を確認
log_df.info() # データ型や欠損値を確認
log_df.head() # 最初の5行を表示
```


実習２ Part１のまとめ

実習２ 業務活用 Part１

生のアクセスログを分析可能な形にする第一歩を踏み出しました

- ☑ アクセスログの内容とETLの概念を理解した
- ☑ Pythonでテキストファイルを読み込む方法を学んだ
- ☑ 正規表現を使って必要な情報を抽出するスキルを身につけた
- ☑ 抽出結果を DataFrame に変換し、構造化できた

NEXT

Part２：データ型の整備とデータマートの基礎

実習２ 業務活用

実習２ Part２では、
Part１で作成した DataFrame を
さらに使いやすくします

NEXT

ビッグデータ技術応用

実習２ 業務活用

アクセスログETL処理とBIツールによる可視化

Part1：アクセスログとETLの基礎

Part2：データ型の整備とデータマートの基礎

Part３：Tableau Publicによるダッシュボード作成

Part4：Power BI Desktopによるダッシュボード作成



ビッグデータ技術応用_実習２

Part2


ビッグデータ技術応用
「基礎から実践へ
3つの実習で学ぶデータ分析・基盤技術」
実習１　分析実践
実習２　業務活用
アクセスログETL処理とBIツールによる可視化
実習３　大規模処理
Mirai no tobira


ビッグデータ技術応用
実習２　業務活用
アクセスログETL処理とBIツールによる可視化
Part1：アクセスログとETLの基礎
Part2：データ型の整備とデータマートの基礎
Part３：Tableau Publicによるダッシュボード作成
Part4：Power BI Desktopによるダッシュボード作成
Mirai no tobira

実習 2 業務活用

ビッグデータ技術応用

実習 1 分析実践 → **実習 2 業務活用**（アクセスログETL処理とBIツールによる可視化） → 実習 3 大規模処理

Part 1 : アクセスログとETLの基礎

☞ **Part 2 : データ型の整備とデータマートの基礎**

Part 3 : Tableau Publicによるダッシュボード作成

Part 4 : Power BI Desktopによるダッシュボード作成



Part 2 : データ型の整備とデータマート基礎

実習 2 業務活用

分析の下準備 : データを『使える』形にする

Part 2：データ型の整備とデータマート基礎 の達成目標

実習 2 業務活用

- ✓ ログのタイムスタンプ（文字列）を日時型 (datetime) に変換できる
- ✓ 日時データから年・月・曜日・時間などを抽出できる
- ✓ データマートの考え方を理解し、設計できる
- ✓ 分析に必要な列を選択し、CSVファイルとして保存できる

Part 1 のおさらい

実習 2 業務活用 Part 2

Part 1 の成果：構造化データへ

Part 1 では、生のログテキストを解析し、
log_df というDataFrameを作成しました

▶ Part 2 の課題

- ➢ この `log_df` はまだ「下ごしらえ」が必要です
- ➢ 特に `time` 列が文字列のままでは、時間軸での分析ができません

日時データ処理の重要性

実習２ 業務活用 Part２

タスク2：なぜ日時データ処理が重要？

文字列のまま (object)

- ▶ 月ごとの集計ができない
- ▶ 時間帯別の分析ができない
- ▶ 曜日による違いが見れない
- ▶ 並び替えが文字順になる

➡

日時型へ変換 (datetime)

- ▶ 月・曜日・時間などを抽出可能
- ▶ 時間軸での集計・分析が可能
- ▶ 時系列グラフの作成が可能
- ▶ 日付順での並び替えが可能

🎯 結論

時間に関する分析（いつアクセスが多いかなど）を行うには、**必ず日時型への変換が必要**

pd.to_datetime

実習２ 業務活用 Part２

文字列を日時に変身：pd.to_datetime

pandasの強力な関数です

ただし、元の文字列の「形式」を正しく指定する必要があります

ログの形式: `'10/Oct/2000:13:55:36 -0700'`

対応する書式指定子 (format): `'%d/%b/%Y:%H:%M:%S %z'`

```
log_df['timestamp'] = pd.to_datetime(log_df['time'],
format='%d/%b/%Y:%H:%M:%S %z', errors='coerce')
```

日時からの特徴量抽出

実習2 業務活用 Part 2

日時データの『解剖』：.dt アクセサ

日時型に変換すると、.dt という便利な道具箱が使えます

```
# 年月日時などを抽出
log_df['year'] = log_df['timestamp'].dt.year
log_df['month'] = log_df['timestamp'].dt.month
log_df['day'] = log_df['timestamp'].dt.day
log_df['hour'] = log_df['timestamp'].dt.hour
log_df['dayofweek'] = log_df['timestamp'].dt.dayofweek # 月=0, 日=6
log_df['dayname'] = log_df['timestamp'].dt.day_name() # Monday, Tuesday...
```


URLクリーニング (補足)

実習2 業務活用 Part 2

タスク3：URLを綺麗にする (オプション)

URL末尾のパラメータ (?id=1 など) が不要な場合
.str.split('?') で除去できます

```
log_df['url_cleaned'] = log_df['url'].str.split('?').str[0]
```

データマートとは？

実習２ 業務活用 Part２

タスク4：分析用の『売店』を作る：データマート

分析目的に合わせて、必要な情報だけを選び、使いやすく整理したデータセット

元のデータ (log_df)	データマート (access_mart_df)
▶ 全情報を含む	▶ 今回の分析に必要な列だけ
▶ データ量が大きい	▶ データ量が小さい
▶ 不要な列もある	▶ BIツールで高速に動作

🎯 **今回の設計**
「いつ」「誰が」「どのページを」見たかを分析するため、
`timestamp`, `ip`, `url`, `status` 及び抽出した日時関連列を選択します

データマート作成＆保存

実習２ 業務活用 Part２

タスク5：データマート作成と保存 (Load)

ステップ1：列選択

```
mart_columns = ['timestamp', 'year', ..., 'status'] # 必要な列リスト
access_mart_df = log_df_final[mart_columns].copy()
```

ステップ2：CSVファイルへ保存

```
output_file_name = 'access_log_mart.csv'
access_mart_df.to_csv(output_file_name, index=False) # index=False が重要
```

🎯 **ETL完了！**
これでETLプロセス (Extract, Transform, Load) が完了！`access_log_mart.csv` がPart 3&4の分析データです

実習２ Part２のまとめ

実習２ 業務活用 Part２

扱いにくいログデータを、BIツールで使える綺麗なデータマートに加工しました

- ✅ `pd.to_datetime` で日時型に変換する重要性と方法
- ✅ `.dt` アクセサで日時から特徴量を抽出する技術
- ✅ データマートの考え方と設計
- ✅ `.to_csv` で処理結果を保存するETLのLoadステップ

NEXT

Part３：Tableau Public によるダッシュボード作成

実習２ 業務活用

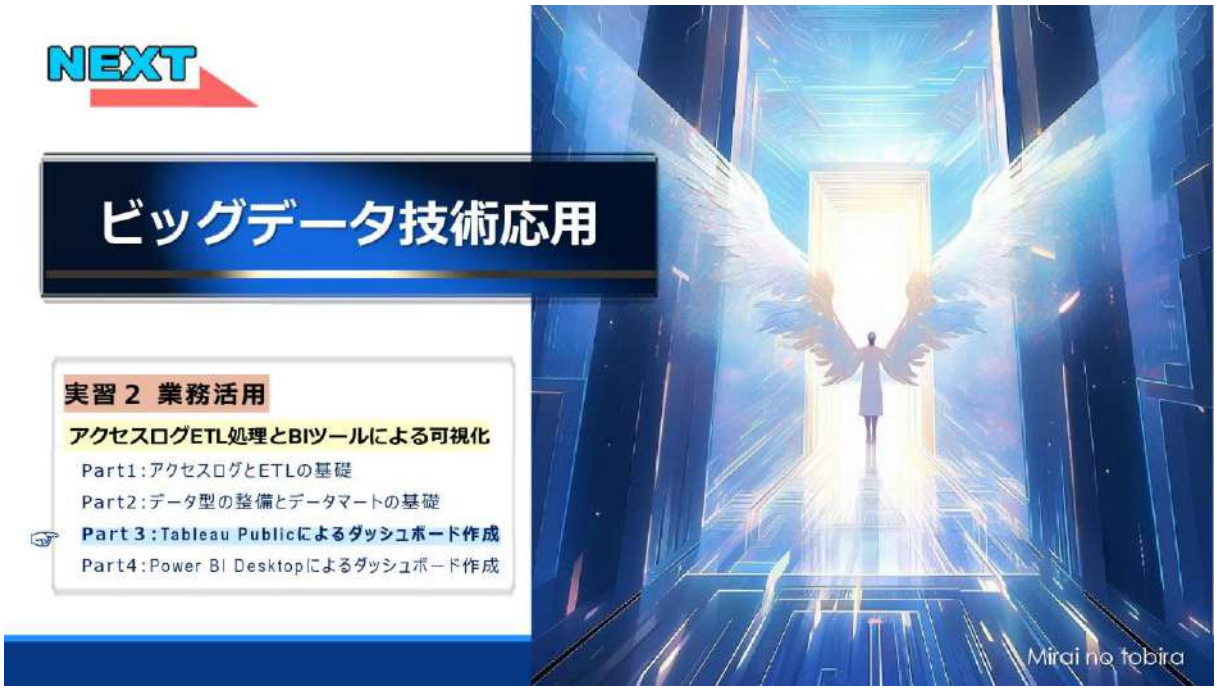
このあと実習２では、

Part２で作成した access_log_mart.csv を使って

Part３：Tableau Public でインタラクティブなダッシュボードの作成

Part４：Power BI Desktop で同様のダッシュボードの作成

をします


NEXT
ビッグデータ技術応用
実習２ 業務活用
アクセスログETL処理とBIツールによる可視化
Part1：アクセスログとETLの基礎
Part2：データ型の整備とデータマートの基礎
Part３：Tableau Publicによるダッシュボード作成
Part4：Power BI Desktopによるダッシュボード作成
Mirai no tobira

令和7年度「地方やデジタル分野における専修学校理系転換等推進事業」
情報成長分野の教育プログラム整備と教員育成による学科転換・新設推進事業

ビッグデータ技術応用教材資料 1

令和８年２月
一般社団法人全国専門学校情報教育協会
〒164-0003 東京都中野区東中野 1-57-8 辻沢ビル３F
電話：03-5332-5081 FAX.03-5332-5083

●本書の内容を無断で転記、掲載することは禁じます。