コンピュータリテラシ発展 ～Pythonを学ぶ～

第11回：　計算やデータ　析をやってみよう

情　学 情　学科 情　　ディア　攻
清水 [illegible]也 (shimizu@info.shonan-it.ac.jp)

今回の授業内容

今回の授業内容

- 前回の課題解説
- データの分析
- データの可視化
- 課題

前回の課題解説

前回の課題解説

- 前回の課題の解答を示します
- 解答について質問があればご連絡ください

解答

https://colab.research.google.com/drive/1BN8_-3cvnWOEKBMtSmVQ-gVyd1qjLbTL?usp=sharing

データの分析

データの準備

- これから行う分析のためにデータを準備します
- Moodleにある「**data_analysis.zip**」をダウンロードして解凍してください
- Google Driveの作業場所に「**data_analysis**」フォルダごとアップロードします
- 「**data_analysis**」の中身は次のとおりです
  - 「customer.csv」
  - 「item.csv」
  - 「transaction_1.csv」
  - 「transaction_2.csv」

データの読み込み

目標：商品の購買データと顧客データを組み合わせ，購買分析を行います

- pandasを使ってデー　を　み込み，顧客ID上位5名 の顧客情　を　示さ　ます
- `read_csv()` 関数：CSVフ　イ　をpandasにデー　を　み込ま　ます
- 　み込ん　デー　はDataFrame　　ェ　トとして扱うことが　きます
- DataFrame　　ェ　トの `head()`　　ト　デー　の　ル5行を　示　きます

```
import pandas as pd  # pandasモジュールをインポート

# CSVファイルのパスを指定
path = '/content/drive/MyDrive/???/data_analysis/customer.csv'

# 指定されたCSVファイルを読み込み，データフレームに格納
customer = pd.read_csv(path)

# データフレームの最初の5行を表示
customer.head()
```

データの読み込み

- デー　セ　トの　み込み関数
- pandas　は各種デー　セ　トを　み込む関数が用意されています

データ形式	関数	解説
CSV	`read_csv()`	区切り文　字　区切られたデー　を　み込む
excel	`read_excel()`	Excel　式のデー　（　　子.xls, .xlsx）を　み込む
json	`read_json()`	JSONデー　を　み込む
html	`read_html()`	HTMLフ　イ　内のテー　　式のデー　を　み込む

データの読み込み

- デー　の基　情　を確　します
- pandasには，デー　の基　情　を簡単に　示する　プ　ョ　も　供されています
- DataFrame　　ェ　トの　プ　ョ　を　下の　にまとめます

オプション	解説
`shape`	行数と列数を　示する（　:15行6列の場合，(15,6)と出力）
`columns`	列名　を　示する
`dtypes`	列名　と，列のデー　型を　示

データの結合

- 下の2つの結合方法について解説します
  - データを縦方向に結合する処理
  - キーのカラムを1つにして横方向にデータを結合する処理

データの結合

データを縦　向に結合

- `concat()`関数を使ってデー　を縦に　結します
  - **transaction_1.csv**と**transaction_2.csv**は■じ種■のデー　を扱っています
  - デー　分析をする際に1つのデー　となっている■が扱いやすい
  - この2つのデー　を縦に　結します

データの結合

- `concat()` 関数　デー　を縦に　結する
  - `ignore_index` を `True` に　　すること　，　結　のデー　がもつイ　デ　　ラ　を　視し，新しくイ　デ　　ラ　　を作成します
  - `ignore_index` を `True` に　　しない場合（省略），元デー　のイ　デ　　ラ　をそのまま使　します

```
path = '/content/drive/MyDrive/???/data_analysis/'

# 指定されたCSVファイルを読み込み、データフレームに格納
transaction_1 = pd.read_csv(path + 'transaction_1.csv')
transaction_2 = pd.read_csv(path + 'transaction_2.csv')

# 2つのデータフレームを結合し、新しいデータフレームに格納
# ignore_index=True は、結合後のデータフレームのインデックスを再設定することを示す
transaction = pd.concat([transaction_1, transaction_2], ignore_index=True)

transaction
```

データの結合

特定の　をキーにして横　向にデータを結合

- `merge()` 関数　　の　ーをもとにデー　を横に結合する
  - transactionデー　は顧客IDがある
  - customerデー　は顧客IDに　　く顧客情　（顧客名，性別，　）がある
  - 顧客IDを　ーとして結合処理を行う

データの結合

- `merge()`関数　　の　ーをもとにデー　を横に結合します
  - 引数`on`に　　ー（顧客ID）を　　します
  - `concat()`関数とデー　の　方 法 は異なるの　　意

```
# transactionデータフレームとcustomerデータフレームを '顧客ID' 列でマージ
# '顧客ID' 列の値が一致する行を結合して、新しいデータフレームを作成
sales_data = pd.merge(transaction, customer, on='顧客ID')

# マージされたデータフレームの内容を表示
sales_data
```

データの集計

- 合計を求める
  - デー　の集計作業を行う
  - 顧客IDごとの合計購　金額と合計購　数を求める
  - `merge` した `sales_data` デー　を使う
  - `groupby()`　　トを使って　　の　ーを　　する
  - 合計購　金額 → 「購　金額」
  - 合計購　数 → 「購　数」
  - 合計 → `.sum()`

データの集計

- 合計を求める
  - デー　の集計作業を行う
  - 顧客IDごとの合計購　金額と合計購　数を求める

```
# sales_data データフレームを '顧客ID' 列でグループ化し、
# '購買金額' と '購入数' 列の合計を計算
sales_per_customer = sales_data.groupby('顧客ID')[['購買金額', '購入数']].sum()

# 顧客ごとの合計購買金額と合計購入数を含むデータフレームの内容を表示
sales_per_customer
```

データの集計

- 合計を求める - デー　の集計作業を行う
  - `groupby()`　　ト　　ープ化された結果を`GroupBy`　　ェ　トと呼ふ
  - `GroupBy`　　ェ　トの　の　　トを　下にまとめる

メソッド名	解説
`count()`	ープ化されたデー　の個数を　示
`mean()`	ープ化されたデー　の均　を　示
`sum()`	ープ化されたデー　の総 を　示
`describe()`	ープ化されたデー　の統計情　を　示（　：最 大 ，標準 差）

データの集計

- 均を求める
  - デー　の集計作業を行う
  - `mean()`　　トを使って「購　　」ごとの均　　上を求める

```
# sales_data データフレームを '購買日' 列でグループ化し、
# '購買金額' 列の平均を計算
sales_per_day = sales_data.groupby('購買日').購買金額.mean()

# 日ごとの平均購買金額を含むデータフレームの内容を表示
sales_per_day
```

データの可視化

データを可視化するためのライブラリ

- 集計デー　を使って　ラフを作成します
- 使用するライ　ラ
  - Matplotlib
    - Python　　ラフを描くための基　的なライ　ラ
    -　　ラフの基　的な設　を行う
  - seaborn
    - Matplotlibを　ー　にしたより複雑な可視化を簡単に行うことが　きるライラ

データを可視化するためのライブラリ

- 集計デー　を使って　ラフを作成します
- **Matplotlib**も**seaborn**もColabにはイ　　トー　済み　す
- イ　ポートする必　があります

```
# matplotlibモジュールのpyplotサブモジュールをインポート（グラフ描画のためのツール）
from matplotlib import pyplot as plt

# seabornモジュールをインポート（データの可視化を行うための高レベルなインターフェース）
import seaborn as sns
```

データを可視化するためのライブラリ

- 集計デー　を使って　ラフを作成します
- 日　語フォ　トの使用について
  - Matplotlibは日　語に対応していないの　日　語を　示するためには日　語フォトを指定　する必　があります
  - japanize_matplotlibライ　ラ　をイ　　トー　してイ　ポートします

```
# pipコマンドを使って japanize_matplotlib パッケージをインストール
!pip install japanize_matplotlib

# japanize_matplotlib パッケージをインポート
import japanize_matplotlib
```

データを可視化するためのライブラリ

- 集計デー　を使って　ラフを作成する
- **sales_per_customer**デー　を棒　ラフ　　示します
- `barplot()`関数を使用します
- 引数「**data**」に対象のDataFrame　　ェ　トを指　する

```
# 顧客ごとの合計購買金額を棒グラフで表示
# x軸に顧客ID、y軸に購買金額を指定
ax = sns.barplot(x=sales_per_customer.index, y='購買金額', data=sales_per_customer)
```

データを可視化するためのライブラリ

- 集計デー　を使って　ラフを作成する
- **sales_per_day**デー　を折れ　　ラフ　　示する
- `lineplot()`関数を使用する
- 引数「**data**」に対象のDataFrame　　ェ　トを渡　する

```
# 日ごとの平均購買金額を折れ線グラフで表示
# data引数にsales_per_dayデータフレームを指定
ax = sns.lineplot(data=sales_per_day)
```

データを可視化するためのライブラリ

- ラフのサイ　を　整する
  - x軸のデー　　示が重なってしまい見　らいの　　ラフを横に広くします
  - **Matplotlib**の設　　修正可
  - `figure()`関数の`figsize`引数を　する

```
# グラフのサイズを指定（幅18、高さ4）
plt.figure(figsize=(18, 4))

# 日ごとの平均購買金額を折れ線グラフで表示
# data引数にsales_per_dayデータフレームを指定
ax = sns.lineplot(data=sales_per_day)
```

データを可視化するためのライブラリ

- ラフの　示を設
  - `set_title()`　　ト：　ラフ　イト　を
  - `set()`　　ト：引数`xlabel`，`ylabel`を　して x軸，y軸のラ　　を作成

```
# グラフのサイズを指定（幅18、高さ4）
plt.figure(figsize=(18, 4))

# 日ごとの平均購買金額を折れ線グラフで表示
# data引数にsales_per_dayデータフレームを指定
ax = sns.lineplot(data=sales_per_day)

# グラフのタイトルを設定
ax.set_title('購買日ごとの平均売上')

# x軸とy軸のラベルを設定
ax.set(xlabel='購買日', ylabel='平均売上（円）')
```

課題

課題

- Moodleにある「SCfCL-11-prac.ipynb」フ　イ　を　ウ　ロードしてColabにア　プロードしてく　さい
- 課　が完了したら「File」>「Download」>「Download .ipynb」　「.ipynb」　式ウ　ロードしてく　さい
- 　ウ　ロードした **.ipynbファイル** をMoodleに　出してく　さい
- 　出期 は **7月4日(木) 20時まで**　す