

2025年度_採用課題

 日付

@July 7, 2025

題目：表情推定によるマーケティング調査

【背景と目的】

近年、ライブ配信プラットフォームや Instagram / TikTok 広告など**視聴者とコンテンツがリアルタイムで相互作用する場面**が爆発的に増えています。クリエイターや広告主は「いいね数」や「コメント量」といった従来の "言語的" 指標だけでなく、

| "画面の向こう側で、視聴者がどんな表情をしているのか？"

を知りたいというニーズを強く抱えています。とりわけ **「笑顔の瞬間」** は一

なぜ重要？	ビジネスインパクト
純粋なポジティブ感情をダイレクトに示す	・クリエイティブ最適化（どの演者の台詞や演出が刺さった？）・広告入稿タイミングの自動最適化
コメントしづらいライト層の反応も回収	・サイレントマジョリティのエンゲージメントを可視化
言語・文化に依存しづらい	・グローバル配信でのUX指標として横展開しやすい

近年、**モバイル端末やブラウザ上で動作する FaceMesh / MediaPipe** などのライブラリが普及し、**顔のランドマーク（Keypoints）座標**を高速取得できるようになりました。

この **幾何情報だけ** をクラウドに送信し、"笑顔フラグ" を判定すれば、

- ・ **処理コスト**：複雑な推論 → ミリ秒級の推論
- ・ **プライバシー**：顔画像を送らないためリーガルリスクを最小化

という一石二鳥のシステムが実装できます。

本課題の狙い

そこで本テストでは、**顔の特徴点 (Keypoints)** という**最小限の情報**だけを用いて、

「**笑顔 = True / False**」を返す関数 `smile_predict()` を実装していただきます。

重要：予測性能は選考における評価対象に含まれます。

【データ構造の解説】

本課題では、Keypoints の意味づけがなされていないデータが提供されます。実データにおいて、各点がどの顔の部位に対応するかを探索的に特定する能力も評価対象となります。

データの構造は以下のとおりです：

```
[
  "サンプルID",
  [x1, y1], # 特徴点1
  [x2, y2], # 特徴点2
  ...
  [x15, y15], # 特徴点15
  {
    "smile": true/false # 笑顔フラグ
  }
]
```

つまり、各サンプルは：

- サンプルID（文字列）
- 15組の座標点 `[x, y]` の配列
- 最後に笑顔かどうかを示すオブジェクト

となっています。

注意： データ中の各特徴点が顔のどの部分（目、鼻、口など）に対応するかは明示されていませんが、実際の顔のデータから抽出されたものとなります。

課題 1：Notebook 実装

ファイル名：`coding_test_smile_detection.ipynb`

1. 関数実装

- 関数名：smile_predict(facial_data: list) → bool
- 入力：下記フォーマットのリスト

```
sample_face = [  
    "サンプルID",  
    [x1, y1], # 特徴点1  
    [x2, y2], # 特徴点2  
    ...  
    [x15, y15] # 特徴点15  
]
```

- **出力**：True（笑顔） または False（非笑顔）

1. 動作確認

- a. 動作検証として下記のコード（提出用コードに元からセルが入っています）を回して出力を確認してください。

```
# --- 動作確認: pred vs ground truth を文字列で表示（編集不要） -----  
-----  
import json  
from pathlib import Path  
  
# データセットのロード  
data_path = Path('data') / 'facial_keypoints.json'  
with open(data_path, 'r', encoding='utf-8') as fp:  
    data = json.load(fp)  
  
test = data['test']  
  
print("=== Test Data Predictions vs Ground Truth ===")  
for idx, sample in enumerate(test, start=1):  
    face_input = sample[:-1] # 最後のsmileオブジェクトを除いたデータ
```

```
gt_bool = sample[-1]['smile']
pred_bool = smile_predict(face_input)

# ブールを文字列に変換
gt = 'smile' if gt_bool else 'not smile'
pred = 'smile' if pred_bool else 'not smile'

print(f"[{idx:03d}] Input ID: {face_input[0]} → Pred: {pred} , GT: {gt}")
```

課題 2：検証レポート

ファイル名：smile_detection_report.pdf

Notebookでの検証についてレポートを作成してください。レポートの構成・内容全て選考の評価対象となります。

用意済みのデータについて

提出フォルダ内のdataフォルダ内に存在しているfacial_keypoints.json は、下記構造によって、実際にテストに使用するデータと、それに対して実装上で活用するためのデータが存在しております。

※詳細は適宜データをご確認ください。

```
{
  "train": [
    [
      "サンプルID",
      [x1, y1], # 特徴点1
      [x2, y2], # 特徴点2
      ...
      [x15, y15], # 特徴点15
      {
        "smile": true/false
      }
    ]
  ]
}
```

```
    ],  
    # 以下同様のデータが続く  
  ],  
  "test": [  
    # 同様の構造  
  ]  
}
```

提出物

1. coding_test_smile_detection.ipynb
2. smile_detection_report.pdf