

Projekt rekomendacji gier

Sara Fijołek i Miron Kwiatkowski

Opis technologiczny projektu

Projekt to aplikacja webowa zbudowana w oparciu o framework Flask, która umożliwia użytkownikowi znalezienie gry odpowiadającej jego preferencjom. Struktura aplikacji obejmuje interfejs użytkownika, logikę backendową oraz model uczenia maszynowego. Opisane poniżej komponenty tworzą integralny system rekomendacji.

1 Backend (Flask i Python)

Aplikacja korzysta z Flask jako frameworka webowego, który umożliwia zarządzanie routingiem i przetwarzaniem zapytań HTTP. Główne endpointy aplikacji to:

- `/` (strona główna): wyświetla formularz z pytaniami dotyczącymi preferencji użytkownika.
- `/result`: przetwarza odpowiedzi użytkownika, używa modelu do przewidywania gry, a następnie zwraca rekomendacje na stronie wynikowej.

2 Model uczenia maszynowego (Decision Tree)

Do przewidywania gry na podstawie odpowiedzi użytkownika użyto modelu drzewa decyzyjnego zaimplementowanego z wykorzystaniem `scikit-learn`. Model jest trenowany na danych zawierających cechy gier oraz ich klasyfikacje:

- Dane wejściowe (X) to macierz cech (np. tryb gry, perspektywa, gatunek, platforma).
- Dane wyjściowe (y) to nazwy gier.

Model został przeszkolony na niewielkim zbiorze danych zawierającym popularne gry. W czasie rzeczywistym, na podstawie preferencji użytkownika, model dokonuje predykcji najbardziej odpowiedniej gry.

3 Interfejs użytkownika (HTML i CSS)

Interfejs użytkownika jest prosty i intuicyjny, zaprojektowany z użyciem HTML i CSS. Składa się z dwóch głównych stron:

- **Strona główna:** formularz zawierający pytania dotyczące preferencji gier (tryb jednoosobowy/wielooosobowy, perspektywa pierwszo/trzecioosobowa, gatunek, platforma).
- **Strona wynikowa:** wyświetla rekomendowaną grę wraz z jej okładką na podstawie predykcji modelu.

4 Przechowywanie i wyświetlanie obrazów

Aplikacja przechowuje okładki gier w katalogu `static/images` i wyświetla je na stronie wynikowej na podstawie rekomendacji modelu. Słownik `cover` mapuje nazwy gier na odpowiednie pliki graficzne, co umożliwia dynamiczne ładowanie obrazów dla każdej rekomendacji.

5 Logika działania

Po otrzymaniu odpowiedzi użytkownika, aplikacja przetwarza dane i przekazuje je do modelu, który generuje odpowiedź w postaci tytułu gry. Następnie aplikacja renderuje stronę wynikową z nazwą gry i jej okładką, dając użytkownikowi wrażenie spersonalizowanej rekomendacji.

Podsumowanie

Aplikacja stanowi przykład integracji prostego modelu uczenia maszynowego z aplikacją webową, umożliwiając użytkownikowi interaktywne dopasowanie gier w oparciu o wybrane preferencje.

6 app.py

Listing 1: Kod źródłowy `app.py`

```
from flask import Flask, render_template, request
from sklearn.tree import DecisionTreeClassifier
import numpy as np

app = Flask(__name__)

# Decision tree data (X) and corresponding games (y)
X = np.array([
    [0, 0, 0, 0], # singleplayer, first person, survival, pc
    [1, 0, 0, 0], # multiplayer, first person, survival, pc
```

```

[0, 1, 0, 0], # singleplayer, third person, survival, pc
[0, 0, 1, 0], # singleplayer, first person, action, pc
[0, 0, 0, 1], # singleplayer, first person, survival, console
[1, 1, 0, 0], # multiplayer, third person, survival, pc
[1, 0, 1, 0], # multiplayer, first person, action, pc
[1, 1, 1, 0], # multiplayer, third person, action, pc
[0, 1, 1, 0], # singleplayer, third person, action, pc
[1, 1, 0, 1], # multiplayer, third person, survival, console
[0, 1, 0, 1], # singleplayer, third person, survival, console
[1, 0, 0, 1], # multiplayer, first person, survival, console
[1, 0, 1, 1], # multiplayer, first person, action, console
[0, 0, 1, 1], # singleplayer, first person, action, console
[0, 1, 1, 1], # singleplayer, third person, action, console
[1, 1, 1, 1], # multiplayer, third person, action, console
])

y = np.array([
    "SonsOfTheForest", # singleplayer, first person, survival,
        pc
    "Rust", # multiplayer, first person, survival, pc
    "Valheim", # singleplayer, third person, survival, pc
    "DOOM_Eternal", # singleplayer, first person, action, pc
    "Subnautica", # singleplayer, first person, survival, console
    "Astroneer", # multiplayer, third person, survival, pc
    "RainbowSixSiege", # multiplayer, first person, action, pc
    "Gears5", # multiplayer, third person, action, pc
    "The_Witcher_3", # singleplayer, third person, action, pc
    "ConanExiles", # multiplayer, third person, survival, console
    "HorizonZeroDawn", # singleplayer, third person, survival,
        console
    "DayZ", # multiplayer, first person, survival, console
    "Borderlands3", # multiplayer, first person, action, console
    "MetroExodus", # singleplayer, first person, action, console
    "Ghost_of_Tsushima", # singleplayer, third person, action,
        console
    "Fortnite", # multiplayer, third person, action, console
])

# Game cover images
cover = {
    "SonsOfTheForest": "sotf.jpg",
    "Rust": "rust.jpg",
    "Valheim": "valheim.png",
    "DOOM_Eternal": "doom.jpeg",
    "Subnautica": "sub.png",
    "Astroneer": "astro.jpg",
    "RainbowSixSiege": "siege.png",
    "Gears5": "gears.png",
    "ConanExiles": "conan.jpg",
    "HorizonZeroDawn": "horizon.jpg",
    "DayZ": "dayz.jpg",
    "Borderlands3": "bl3.jpg",
    "MetroExodus": "metro.png",
    "Ghost_of_Tsushima": "tsushima.jpg",
    "The_Witcher_3": "witcher3.jpg",
    "Fortnite": "fortnite.png",
}

```

```

# Train the decision tree model
clf = DecisionTreeClassifier()
clf.fit(X, y)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/result', methods=['POST'])
def result():
    # Collect user answers
    answers = [
        int(request.form['question1']),
        int(request.form['question2']),
        int(request.form['question3']),
        int(request.form['question4']),
    ]

    # Predict the game
    predicted_game = clf.predict([answers])[0]

    # Display result
    return render_template('result.html', game_title=predicted_game,
        image_file=cover[predicted_game])

if __name__ == '__main__':
    app.run(debug=True)

```

7 Kod HTML - Strona wyboru gry

Listing 2: Kod HTML strony wyboru gry

```

<!DOCTYPE html>
<html lang="pl">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
        scale=1.0">
    <link rel = "stylesheet" href="{url_for('static', filename='
        css/style.css')}">
    <title>Wyb r Gry</title>
</head>
<body>
    <h1>Wybierz odpowiedzi na pytania, a dopasujemy gr do Ciebie<
        /h1>
    <form action="/result" method="POST">
        <p><strong>Czy wolisz gry jednoosobowe czy gry wieloosobowe
            ?</strong></p>
        <input type="radio" id="single" name="question1" value="0">
        <label for="single">Jednoosobowe</label><br>
        <input type="radio" id="multi" name="question1" value="1">
        <label for="multi">Wieloosobowe</label><br><br>

```

```

<p><strong>Czy wolisz widok pierwszo czy trzecioosobowy?</strong></p>
<input type="radio" id="first" name="question2" value="0">
<label for="first">Pierwszoosobowy</label><br>
<input type="radio" id="third" name="question2" value="1">
<label for="third">Trzecioosobowy</label><br><br>

<p><strong>Czy wolisz gry survivalowe czy gry akcji?</strong></p>
<input type="radio" id="survival" name="question3" value="0">
<label for="survival">Survivalowe</label><br>
<input type="radio" id="action" name="question3" value="1">
<label for="action">Akcji</label><br><br>

<p><strong>Czy wolisz gry na PC czy na konsole?</strong></p>
<input type="radio" id="pc" name="question4" value="0">
<label for="pc">PC</label><br>
<input type="radio" id="console" name="question4" value="1">
<label for="console">Konsole</label><br><br>

<input type="submit" value="Zatwierd Ź odpowiedzi">
</form>
</body>
</html>

```

8 Kod HTML - Strona wyniku

Listing 3: Kod HTML strony wyniku

```

<!DOCTYPE html>
<html lang="pl">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, Ź initial-
    scale=1.0">
  <link rel = "stylesheet" href="{{ Źurl_for('static', Źfilename='
    css/style.css') Ź }}">
  <title>Wynik</title>
</head>
<body>
  <div class="container">
    <h1>Gra, kt ra pasuje do Ciebie to: {{ game_title }}</h1>
    
    <br><br>
    <form action="/">
      <button type="submit" class="button">Powr t do strony
        g wnej</button>
    </form>
  </div>
</body>
</html>

```

9 Styl CSS

Listing 4: Styl CSS dla aplikacji

```
body {
    font-family: Arial, sans-serif;
    background-color: #ccdba3;
    margin: 0;
    padding: 20px;
}

h1 {
    color: #333;
    text-align: center;
}

form {
    margin: 20px auto;
    width: 300px;
    padding: 20px;
    background-color: white;
    border: 1px solid #ccc;
    border-radius: 10px;
}

label {
    display: block;
    margin-bottom: 5px;
}

input[type="submit"] {
    background-color: #4CAF50;
    color: white;
    padding: 10px 15px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}

input[type="submit"]:hover {
    background-color: #921227;
}

/* Wyrodkowanie tekstu i obrazka */
.container {
    text-align: center;
    margin: 0 auto;
    padding: 20px;
}

.game-cover {
    display: block;
    margin: 0 auto;
    width: 400px;
    height: 500px;
}
```

```
.button {
  display: inline-block;
  padding: 10px 20px;
  background-color: #4CAF50;
  color: white;
  text-align: center;
  text-decoration: none;
  font-size: 16px;
  border-radius: 5px;
  transition: background-color 0.3s;
  margin-top: 20px;
}

.button:hover {
  background-color: #45a049;
}
```

10 Zrzuty ekranu z aplikacji

Wybierz odpowiedzi na pytania, a dopasujemy grę do Ciebie

Czy wolisz gry jednoosobowe czy gry wieloosobowe?

☐ Jednoosobowe

☐ Wieloosobowe

Czy wolisz widok pierwszo czy trzecioosobowy?

☐ Pierwszoosobowy

☐ Trzecioosobowy

Czy wolisz gry survivalowe czy gry akcji?

☐ Survivalowe

☐ Akcji

Czy wolisz gry na PC czy na konsole?

☐ PC

☐ Konsole

Zatwierdź odpowiedzi

Gra, która pasuje do Ciebie to: DayZ



[Powrót do strony głównej](#)