ДОДАТОК А

Відомість кваліфікаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	92 с.
	Інші документи	
2.	Презентаційні матеріали	15 плакатів

.	.	Прізвище та ініціали.	Підп.	Дата			
Розробив		Литвиненко А.			Аналіз ефективності рекламних кампаній з використанням методів A/B тестування та статистичного моделювання	Шифр групи	Код напр./спец.
Перевірив		Павленко Є.П.				ДСзм-23-1	122
Н.контр.		Малєєва І.А.				ХНУРЕ кафедра ШІ	
Затв.		Золотухін О.В.					

ДОДАТОК Б

Для оптимізації рекламних креативів у Meta Ads за допомогою Python, передбачаються такі кроки:

1. Отримати доступ до Facebook API через Facebook Graph API.
2. Використовувати дані рекламних кампаній для аналізу їх ефективності (A/B тестування).
3. Побудувати просту модель для оптимізації рекламних креативів.

Необхідні бібліотеки:
requests - для взаємодії з API.
pandas - для обробки та аналізу даних.
matplotlib / seaborn - для візуалізації результатів.

Крок 1: Налаштування доступу до Facebook API
Для взаємодії з API вам потрібно отримати токен доступу для Facebook Marketing API. Це можна зробити через Facebook Developer Console.
pip install requests pandas matplotlib seaborn

Крок 2: Завантаження даних рекламних кампаній
import requests import pandas as pd

# Ваш токен доступу до API access_token = 'your_access_token'
ad_account_id = 'your_ad_account_id' # Наприклад, 'act_1234567890'

# URL для запиту статистики рекламних оголошень
url = f'https://graph.facebook.com/v14.0/{ad_account_id}/ads' params = {
'access_token': access_token,
'fields':
'id,name,adcreatives,insights{date_preset,impressions,clicks,spend,cost_per_click,cost_per_impressi on}',
}

# Отримуємо дані
response = requests.get(url, params=params) data = response.json()

# Перевіримо отриману структуру даних
print(data)

Крок 3: Обробка та аналіз даних
Перетворимо отримані дані в DataFrame для зручного аналізу ads_data = []

# Перебираємо всі оголошення і збираємо їх статистику for ad in data.get('data', []):
ad_id = ad['id'] ad_name = ad['name']
# Перевіряємо, чи є статистика if 'insights' in ad:
for insight in ad['insights']['data']: ads_data.append({
'ad_id': ad_id, 'ad_name': ad_name,
'impressions': insight['impressions'], 'clicks': insight['clicks'],
'spend': insight['spend'],
'cpc': insight.get('cost_per_click', 0),

```
'cpi': insight.get('cost_per_impression', 0)
})

# Створимо DataFrame для аналізу df = pd.DataFrame(ads_data)
# Переглянемо перші рядки даних print(df.head())
```

Крок 4: Візуалізація та оптимізація креативів

Зараз ми можемо порівняти ефективність різних рекламних креативів на основі CPC (вартість за клік) та CPI (вартість за показ).

```
import matplotlib.pyplot as plt
import seaborn as sns

# Візуалізуємо порівняння ефективності рекламних оголошень за показниками CPC та
CPI
plt.figure(figsize=(12, 6))
sns.barplot(data=df, x='ad_name', y='cpc', palette='viridis')
plt.title('Вартість за клік (CPC) для різних рекламних креативів')
plt.xlabel('Назва оголошення')
plt.ylabel('CPC')
plt.xticks(rotation=90)
plt.show()
plt.figure(figsize=(12, 6))
sns.barplot(data=df, x='ad_name', y='cpi', palette='viridis')
plt.title('Вартість за показ (CPI) для різних рекламних креативів')
plt.xlabel('Назва оголошення')
plt.ylabel('CPI')
plt.xticks(rotation=90)
plt.show()
```

Крок 5: Оптимізація на основі результатів

Вибір найбільш ефективного рекламного креативу можна зробити, аналізуючи отримані

метрики, такі як CPC та CPI. Для більш глибокої оптимізації можна використовувати

статистичні методи, такі як A/B тестування (порівняння варіантів креативів) або машинне

навчання для прогнозування найкращих креативів на основі минулих даних. Отже побудова моделі для прогнозування, який креатив принесе найкращі результати.

```
from sklearn.linear_model import LinearRegression

# Створимо модель для прогнозування CPC
X = df[['impressions', 'spend']] # Особливості: кількість показів і витрати
y = df['cpc'] # Ціль: CPC

# Розділяємо на тренувальний та тестовий набори
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Навчання моделі
model = LinearRegression()
model.fit(X_train, y_train)

# Оцінка моделі
print(f"Точність моделі: {model.score(X_test, y_test)}")

# Прогнозування результатів
y_pred = model.predict(X_test)

# Виведемо прогнозовані та реальні значення CPC
results = pd.DataFrame({'Real CPC': y_test, 'Predicted CPC': y_pred})
print(results.head())
```

ДОДАТОК В

Для оптимізації рекламних креативів у Google Ads за допомогою Python, передбачаються такі кроки:

1. **Налаштування Google Ads API**: Отримання доступу до Google Ads API через Google Cloud Console.
2. **Завантаження даних з Google Ads**: Використання Google Ads API для отримання статистики кампаній.
3. **Аналіз та оптимізація креативів**: Аналіз ефективності рекламних оголошень за метриками та оптимізація на основі результатів.

Необхідні бібліотеки:

- `google-ads` - офіційна бібліотека для роботи з Google Ads API.
- `pandas` - для аналізу даних.
- `matplotlib` / `seaborn` — для візуалізації результатів.

Крок 1: Налаштування Google Ads API

Щоб працювати з Google Ads API, вам необхідно:

- Створити проект у [Google Cloud Console](https://console.cloud.google.com/).
- Отримати доступ до Google Ads API, створивши OAuth2 клієнт.
- Завантажити конфігураційний файл `google-ads.yaml`. Встановіть бібліотеку для роботи з Google Ads API:

```bash
pip install google-ads
```

Крок 2: Завантаження даних з Google Ads API

```
# Визначення запиту для отримання статистики по кампаніям
query = '''
 SELECT
 campaign.id,
 campaign.name,
 ad_group.id,
 ad_group.name,
 ad.id,
 ad.headline,
 ad.description,
 metrics.impressions,
 metrics.clicks,
 metrics.average_cpc
 FROM
 ad
 WHERE
 segments.date DURING LAST_30_DAYS
'''
# Запуск запиту
ga_service = client.get_service('GoogleAdsService')
response = ga_service.search_stream(customer_id=customer_id, query=query)

# Збір даних у список
ads_data = []
```

```
for batch in response:
 for row in batch.results:
 ads_data.append({
 'ad_id': row.ad.id,
 'ad_name': row.ad.headline,
 'description': row.ad.description,
 'impressions': row.metrics.impressions,
 'clicks': row.metrics.clicks,
 'average_cpc': row.metrics.average_cpc,
 })

# Перетворимо на DataFrame для подальшого аналізу
df = pd.DataFrame(ads_data)

# Перевіримо отримані дані
print(df.head())
```

Крок 3: Аналіз результатів рекламних оголошень

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Візуалізація ефективності рекламних оголошень за середнім CPC
plt.figure(figsize=(12, 6))
sns.barplot(data=df, x='ad_name', y='average_cpc', palette='viridis')
plt.title('Середній CPC для різних рекламних оголошень')
plt.xlabel('Назва оголошення')
plt.ylabel('Середній CPC')
plt.xticks(rotation=90)
plt.show()

# Візуалізація кількості кліків та показів для кожного оголошення
plt.figure(figsize=(12, 6))
sns.scatterplot(data=df, x='impressions', y='clicks', hue='ad_name', palette='tab20')
plt.title('Кліки vs Покази для різних рекламних оголошень')
plt.xlabel('Кількість показів')
plt.ylabel('Кількість кліків')
plt.show()
```

Крок 4: Оптимізація на основі результатів

На основі отриманих даних можна вибрати рекламні оголошення, які мають найнижчий **CPC** (вартість за клік) та найвищі **CTR** (коефіцієнт клікабельності).

Потім можна здійснити корекцію креативів або стратегії, щоб покращити ці метрики.

```python

# Вибір найкращих оголошень за середнім CPC
best_ads = df[df['average_cpc'] == df['average_cpc'].min()]
print("Найбільш ефективні оголошення за середнім CPC:")
```

```
print(best_ads)

# Вибір оголошень з найвищим числом кліків
top_ads_by_clicks = df[df['clicks'] == df['clicks'].max()]
print("Оголошення з найбільшим числом кліків:")
print(top_ads_by_clicks)
```

Крок 5: Оптимізація креативів (простий підхід)
На основі статистики, ви можете приймати рішення про зміни в креативі, як-от:
- Змінити заголовок або опис оголошення.
- Тестувати нові креативи через **A/B тестування**.
- Змінити налаштування таргетингу