D214: Data Analytics Graduate Capstone Task 2

Justin Huynh

Student ID: 012229514

M.S. Data Analytics

A. Research Question

My research question for this project is "How does the marketing channel used in digital campaigns impact customer conversion rates, and which channel is the most effective in driving conversions?”

In today’s highly competitive digital marketing landscape, organizations allocate substantial budgets to various marketing channels in an effort to reach their target audience and drive conversions. It is essential to understand which channels yield the highest returns on investment (ROI) in terms of customer conversions to maximize profits. The chosen research question is significant for businesses seeking to optimize their marketing strategies and ensure they are focusing their resources on the most profitable platforms. So by answering this question, companies can reallocate marketing budgets more effectively, improve their ROI and enhance the impact of their digital campaigns. As a professional marketer, this question is particularly relevant for someone like me who’s worked in digital marketing agencies for over 7 years helping businesses maximize their revenue and profits.

The digital marketing industry continues to grow constantly and companies are investing in numerous platforms such as social media, email marketing, and pay-per-click (PPC) advertising to reach new and existing customers. Each marketing channel requires significant investments in terms of advertising spend and campaign management. So companies need to ensure that they are investing in the channels that provide the highest conversion rates, which usually means higher revenue.

The data used for this project comes from a publicly available dataset from Kaggle. The data includes customer demographics, campaign types, ad spending, and

key performance metrics (KPIs) like click-through rate and conversion rate. Using this data, we can perform statistical tests to analyze the effectiveness of each marketing channel and identify the most successful ones to determine whether we reject or fail to reject the null hypothesis. The null hypothesis (H_0): There is no significant difference in the conversion rates across different marketing channels. The alternate hypothesis (H_1): There is a significant difference in the conversion rates across different marketing channels.

B. Data Collection

For this project, I used an existing dataset sourced from Kaggle, a publicly available data science platform. The dataset includes 8,000 customer records and 20 variables that provide detailed information about customer demographics, campaign metrics, and conversion data. The key variables we'll be using include:

- CustomerID: A unique identifier for each customer.
- Age: The age of the customer.
- Gender: The gender of the customer (Male/Female).
- Income: The annual income of the customer.
- CampaignChannel: The channel used for the marketing campaign (i.e. social media, Email, PPC).
- CampaignType: The type of campaign (i.e. Awareness, Retention, Conversion).
- AdSpend: The amount spent on the advertising campaign.
- ClickThroughRate: The ratio of users who clicked on an ad compared to the total who viewed it.

- ConversionRate: The percentage of visitors who took the desired action (i.e. purchase).
- Conversion: A binary indicator showing whether a customer converted (1) or did not convert (0).

One key advantage of using an existing dataset from Kaggle is the availability of pre-collected, cleaned, and structured data, which significantly reduces the time required for data collection. This enabled me to focus more on analyzing and interpreting the data rather than gathering and processing it from scratch. The dataset was also comprehensive and contained relevant metrics needed to answer the research question.

A disadvantage of using publicly available datasets is the lack of control over data quality and relevance. While the dataset is comprehensive, it may not perfectly align with every nuance of the research question. For example, some of the campaign types or demographics might not be as granular as desired. The dataset may also lack context regarding how the data was collected, which could impact its interpretation.

The main challenge in using an existing dataset was making sure that the variables aligned with the research objectives. In this case, the dataset provided sufficient variables related to customer demographics and campaign performance, but it required careful selection of data to align with the research hypothesis. For example, grouping conversion rates by campaign channels and preparing the data for hypothesis testing required some preprocessing steps. To overcome this challenge, I conducted a thorough exploratory data analysis (EDA) to better understand the dataset and ensure it was suitable for the research question. EDA helped identify any inconsistencies or

irrelevant data points that needed to be filtered out. From there, we can use Python and its libraries to perform further cleaning, aggregation, and data transformation needed for analysis.

C: Data Extraction and Preparation

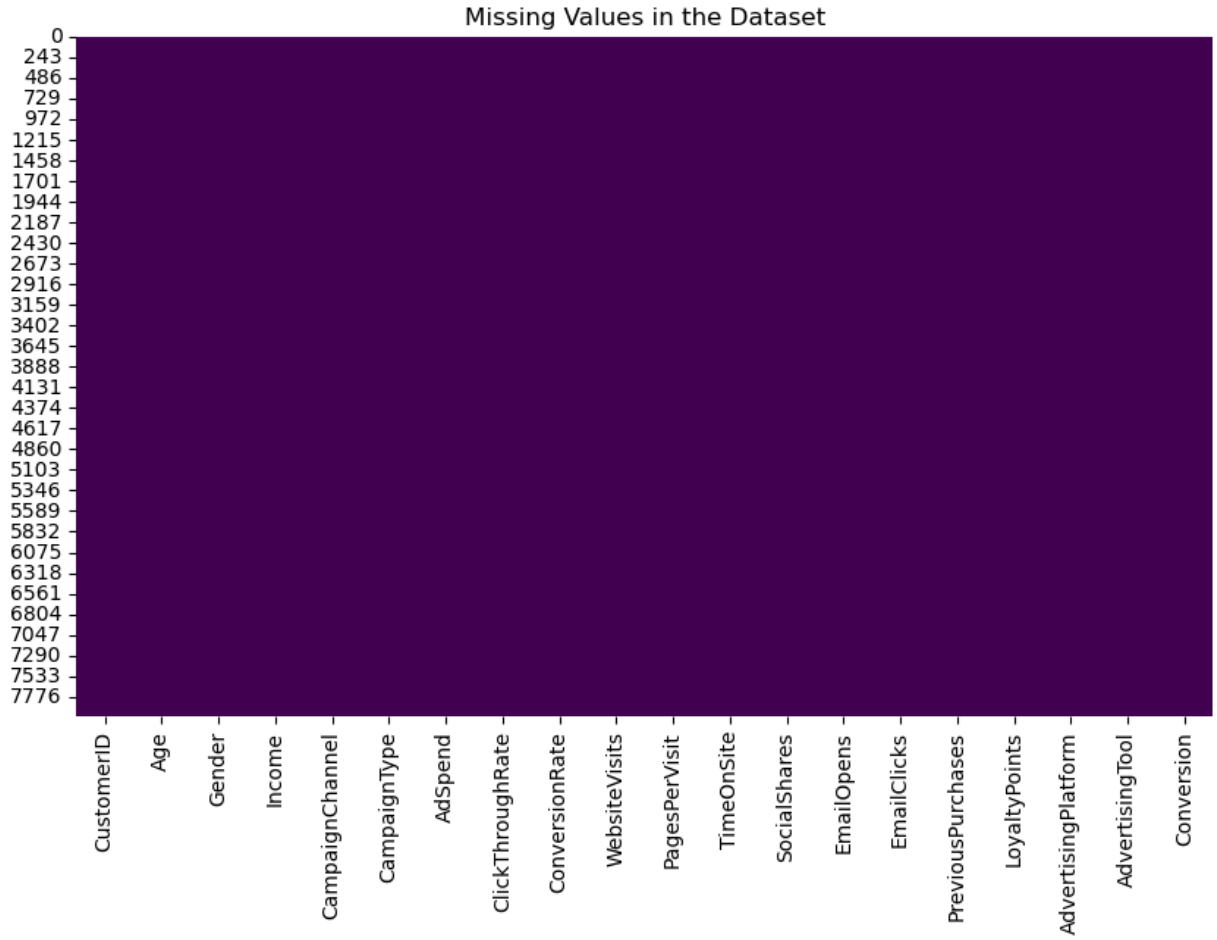
For the data extraction and preparation process, the dataset was first imported using Python's Pandas library, which provides robust data manipulation capabilities. The dataset was loaded in CSV format into a Pandas DataFrame for easy access and transformation.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix
from imblearn.over_sampling import SMOTE

# import kaggle dataset through the file path
df = pd.read_csv("C:/Users/justi/OneDrive/Desktop/digital_marketing_campaign_dataset.csv")
# display info about this file
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8000 entries, 0 to 7999
Data columns (total 20 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   CustomerID           8000 non-null   int64
 1   Age                  8000 non-null   int64
 2   Gender               8000 non-null   object
 3   Income               8000 non-null   int64
 4   CampaignChannel      8000 non-null   object
 5   CampaignType         8000 non-null   object
 6   AdSpend              8000 non-null   float64
 7   ClickThroughRate     8000 non-null   float64
 8   ConversionRate       8000 non-null   float64
 9   WebsiteVisits        8000 non-null   int64
 10  PagesPerVisit        8000 non-null   float64
 11  TimeOnSite           8000 non-null   float64
 12  SocialShares         8000 non-null   int64
 13  EmailOpens           8000 non-null   int64
 14  EmailClicks          8000 non-null   int64
 15  PreviousPurchases    8000 non-null   int64
 16  LoyaltyPoints        8000 non-null   int64
 17  AdvertisingPlatform  8000 non-null   object
 18  AdvertisingTool      8000 non-null   object
 19  Conversion           8000 non-null   int64
dtypes: float64(5), int64(10), object(5)
memory usage: 1.2+ MB
```

Next, we checked for missing values using the .isnull() method, which was visualized through a heatmap generated with Seaborn to quickly identify any gaps in the data. There were no missing values as expected.

```
# check and visualize any missing values
plt.figure(figsize=(10, 6))
sns.heatmap(df.isnull(), cbar=False, cmap='viridis')
plt.title("Missing Values in the Dataset")
plt.show()
```



We then dropped any unnecessary columns like CustomerID from the dataset to focus on relevant variables for analysis. Categorical variables like CampaignChannel, Gender, and CampaignType were converted into the categorical data type using Pandas' .astype() function, which optimizes memory usage and prepares the data for analysis.

```
# drop unnecessary columns
df_cleaned = df.drop(columns=['CustomerID'])
```

```
# convert categorical columns to 'category' dtype
df_cleaned['Gender'] = df_cleaned['Gender'].astype('category')
df_cleaned['CampaignChannel'] = df_cleaned['CampaignChannel'].astype('category')
df_cleaned['CampaignType'] = df_cleaned['CampaignType'].astype('category')
df_cleaned['AdvertisingPlatform'] = df_cleaned['AdvertisingPlatform'].astype('category')

# review cleaned data
df_cleaned.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8000 entries, 0 to 7999
Data columns (total 19 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Age                  8000 non-null   int64
 1   Gender               8000 non-null   category
 2   Income               8000 non-null   int64
 3   CampaignChannel      8000 non-null   category
 4   CampaignType         8000 non-null   category
 5   AdSpend              8000 non-null   float64
 6   ClickThroughRate     8000 non-null   float64
 7   ConversionRate       8000 non-null   float64
 8   WebsiteVisits        8000 non-null   int64
 9   PagesPerVisit        8000 non-null   float64
 10  TimeOnSite           8000 non-null   float64
 11  SocialShares         8000 non-null   int64
 12  EmailOpens           8000 non-null   int64
 13  EmailClicks          8000 non-null   int64
 14  PreviousPurchases    8000 non-null   int64
 15  LoyaltyPoints        8000 non-null   int64
 16  AdvertisingPlatform  8000 non-null   category
 17  AdvertisingTool      8000 non-null   object
 18  Conversion           8000 non-null   int64
dtypes: category(4), float64(5), int64(9), object(1)
memory usage: 969.5+ KB
```

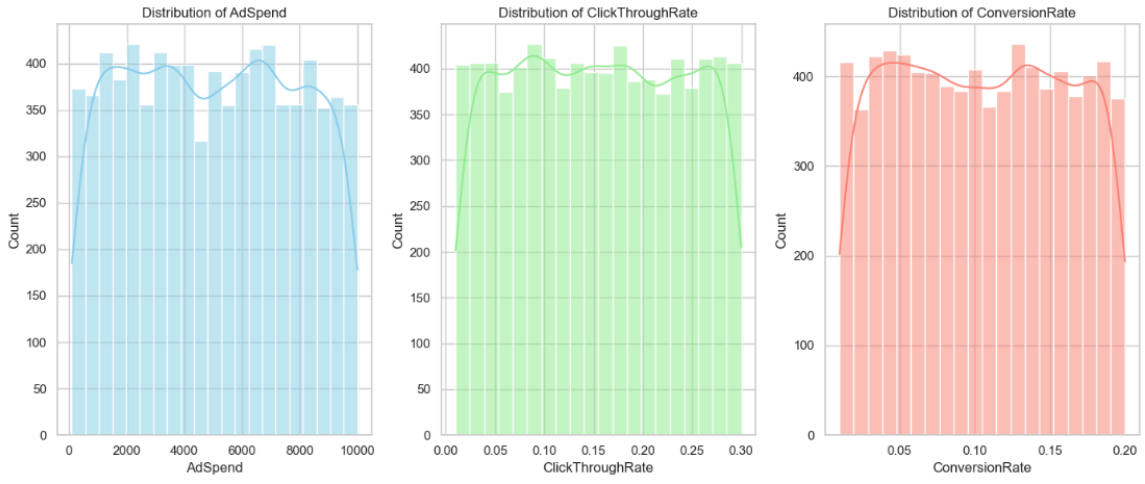
The advantage of using Pandas is its ease of use and efficiency in handling large datasets, which allows rapid transformation and cleaning. However, a potential disadvantage is that Pandas operations can become memory-intensive with larger datasets, which can slow down the process. Despite the disadvantage, Pandas is still an effective tool for this dataset, streamlining the data preparation process.

See code attached in WGU_D214_Task_2.ipynb.

D: Analysis

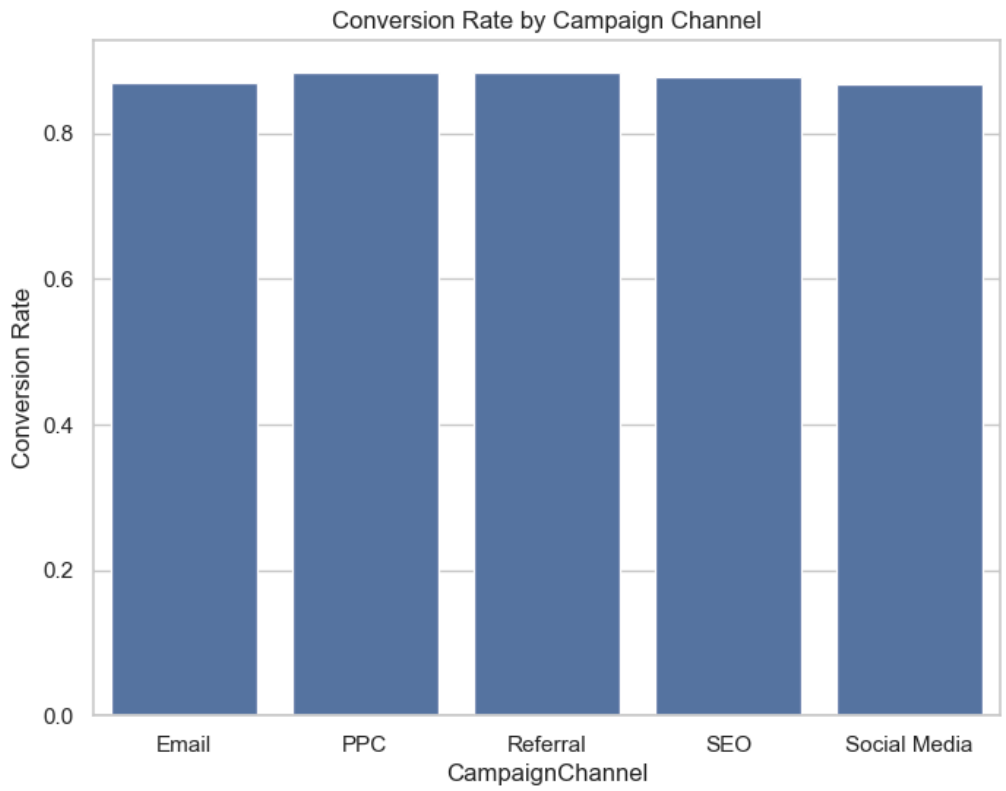
We first started off with Exploratory Data Analysis (EDA) to uncover patterns and relationships in the dataset. We began by examining the distribution of key features like AdSpend, ClickThroughRate, and ConversionRate. We observed that AdSpend was relatively uniform across the dataset while ClickThroughRate and ConversionRate had moderate engagement values, mostly concentrated between 0.05 and 0.25. A bar plot of conversion rates across different marketing channels showed relatively similar conversion rates across the channels. This initial observation prompted us to conduct a formal statistical test to determine if these differences were statistically significant.

```
# exploratory data analysis (EDA)

# set the style for visualizations
sns.set(style="whitegrid")

# distribution of 'AdSpend', 'ClickThroughRate', and 'ConversionRate'
plt.figure(figsize=(14, 6))

# subplot 1: distribution of 'AdSpend'
plt.subplot(1, 3, 1)
sns.histplot(df_cleaned['AdSpend'], kde=True, color='skyblue')
plt.title('Distribution of AdSpend')

# subplot 2: distribution of 'ClickThroughRate'
plt.subplot(1, 3, 2)
sns.histplot(df_cleaned['ClickThroughRate'], kde=True, color='lightgreen')
plt.title('Distribution of ClickThroughRate')

# subplot 3: distribution of 'ConversionRate'
plt.subplot(1, 3, 3)
sns.histplot(df_cleaned['ConversionRate'], kde=True, color='salmon')
plt.title('Distribution of ConversionRate')

plt.tight_layout()
plt.show()
```



To test whether there were statistically significant differences in conversion rates across marketing channels, we performed a One-Way ANOVA (Analysis of Variance). The null hypothesis for the ANOVA tests was that there were no significant differences in conversion rates across the groups. After testing the conversion rates across the different marketing channels, we found that the p-value was 0.595, which is greater than the typical significance level of 0.05. This meant we failed to reject the null hypothesis since there are no statistically significant differences in conversion rates across the different marketing channels. The results led us to shift our focus to building a predictive model that could predict individual conversions based on other factors.

```
# conversion rate per 'CampaignChannel'
plt.figure(figsize=(8, 6))
sns.barplot(x='CampaignChannel', y='Conversion', data=df_cleaned, estimator=lambda x: sum(x) / len(x), errorbar=None)
plt.title('Conversion Rate by Campaign Channel')
plt.ylabel('Conversion Rate')
plt.show()
```



```python
# prepare the data for ANOVA test by campaign channel
channels = df_cleaned['CampaignChannel'].unique()
conversion_by_channel = [df_cleaned[df_cleaned['CampaignChannel'] == channel]['Conversion'] for channel in channels]

# perform one-way ANOVA test
f_statistic, p_value = stats.f_oneway(*conversion_by_channel)

# display the results
print(f"F-statistic: {f_statistic}")
print(f"P-value: {p_value}")
```

```
F-statistic: 0.6960206721427569
P-value: 0.5946070531108417
```

The primary analysis technique used was logistic regression, which is a classification method suitable for binary outcomes (0 or 1). In our case, the target variable was customer conversion, where we wanted to predict whether a customer would convert (1) or not convert (0) based on features such as ad spend, click-through rate, and conversion rate. Logistic regression was chosen for its simplicity and interpretability, especially given the structured nature of the dataset. We first select the features like AdSpend, ClickThroughRate, ConversionRate, Income and Age to predict the binary outcome of conversion. The first logistic regression model achieved an overall accuracy of 88% but it was heavily biased toward predicting conversions, failing to adequately predict non-conversions due to class imbalance. To address this, we improved the model's ability to predict both conversions and non-conversions by applying Synthetic Minority Over-sampling Technique (SMOTE). This technique generated synthetic examples of non-conversions to balance the dataset, which resulted in a more fair and balanced model.

```python
# logistic regression with SMOTE to balance the dataset and improve the model

# select relevant features for prediction
features = ['AdSpend', 'ClickThroughRate', 'ConversionRate', 'Income', 'Age']
X = df_cleaned[features]

# encode the target variable (Conversion)
y = df_cleaned['Conversion']

# apply SMOTE to balance the dataset
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X, y)

# split the resampled data into training and test sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled, test_size=0.2, random_state=42)

# fit the logistic regression model
log_model_smote = LogisticRegression(max_iter=1000)
log_model_smote.fit(X_train, y_train)

# make predictions
y_pred_smote = log_model_smote.predict(X_test)

# evaluate the model
print("Classification Report (SMOTE):\n", classification_report(y_test, y_pred_smote))
print("Confusion Matrix (SMOTE):\n", confusion_matrix(y_test, y_pred_smote))
```

```
Classification Report (SMOTE):
              precision    recall  f1-score   support

           0       0.61      0.63      0.62      1377
           1       0.63      0.61      0.62      1428

    accuracy                           0.62      2805
   macro avg       0.62      0.62      0.62      2805
weighted avg       0.62      0.62      0.62      2805

Confusion Matrix (SMOTE):
 [[874 503]
 [558 870]]
```

Afterwards, we performed hyperparameter tuning using GridSearchCV to optimize the regularization strength (c) and solver (lbfgs). The best parameters were C=100 and solver=lbfgs. However, the overall accuracy remained at 62%. The final logistic regression model with SMOTE and hyperparameter tuning achieved a balance performance with an accuracy of 62%. Both classes of conversions and non-conversions were predicted fairly with a precision recall around 0.61-0.64 for each class.

```python
# tuning the model

# define the parameter grid for tuning
# regularization strength
# different solvers for logistic regression
param_grid = {
    'C': [0.01, 0.1, 1, 10, 100],
    'solver': ['liblinear', 'lbfgs']
}

# initialize the logistic regression model
log_model = LogisticRegression(max_iter=1000)

# perform grid search with cross-validation (5-fold)
grid_search = GridSearchCV(log_model, param_grid, cv=5, scoring='accuracy', verbose=1)

# fit the model on the training data
grid_search.fit(X_train, y_train)

# display the best parameters found by the grid search
best_params = grid_search.best_params_
print(f"Best Parameters: {best_params}")

# evaluate the best model on the test set
best_model = grid_search.best_estimator_
y_pred_best = best_model.predict(X_test)

# show classification report and confusion matrix for the tuned model
print("Tuned Classification Report:\n", classification_report(y_test, y_pred_best))
print("Tuned Confusion Matrix:\n", confusion_matrix(y_test, y_pred_best))
```

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
Best Parameters: {'C': 100, 'solver': 'lbfgs'}
Tuned Classification Report:
              precision    recall  f1-score   support

           0       0.61      0.64      0.62      1377
           1       0.64      0.61      0.62      1428

    accuracy                           0.62      2805
   macro avg       0.62      0.62      0.62      2805
weighted avg       0.62      0.62      0.62      2805

Tuned Confusion Matrix:
 [[877 500]
 [556 872]]
```

We selected logistic regression due to its simplicity and interpretability, which made it easier to understand how each feature influenced the outcome. One advantage

of logistic regression is its transparency, which allows us to see the direct impact of variables like AdSpend and ClickThroughRate on conversion. However, a disadvantage is that logistic regression assumes a linear relationship between the features and the target variable, which may limit its ability to capture more complex, non-linear interactions. The use of ANOVA was appropriate for reducing the risk of false positive errors while logistic regression was suitable for predicting individual customer conversions based on several features.

See code attached in WGU_D214_Task_2.ipynb.

E: Data Summary and Implications

The primary goal of the analysis was to determine the impact of marketing channels on conversion rates, and to predict customer conversions based on key features like ad spend and click-through rates. After performing ANOVA tests, we found that there were no statistically significant differences in conversion rates across marketing channels, so we failed to reject the null hypothesis. This suggested that the differences observed between these groups were likely due to random variation rather than meaningful effects.

To move beyond these differences, we applied a logistic regression model to predict customer conversion. The initial model was biased toward predicting conversions due to class imbalance, so we applied SMOTE to balance the dataset. After tuning the model's hyperparameters, the finalized logistic regression model achieved an overall accuracy of 62%, with balanced precision and recall for both conversions and non-conversions. This indicates that the model is equally capable of

predicting both outcomes but may not capture all the complexity of customer conversion behavior.

One limitation of the analysis is the assumption of linearity that is inherent in logistic regression. Logistic regression assumes a linear relationship between the independent variables and the log-odds of the target variable (conversion). However, customer behavior is often driven by more complex, non-linear relationships that this model may not fully capture. Because of this limitation, the model may miss important interactions between variables, which leads to a modest accuracy score of 62%.

Given that no significant differences in conversion rates were found between marketing channels, the organization should focus on individual-level prediction models, like the logistic regression used in this analysis, to understand the likelihood of conversion for each customer. The model's prediction can be used to inform targeted marketing strategies, allocating resources toward customers who are more likely to convert based on their behavior (i.e. high ad spend, high click-through rates, etc.). This data-driven approach can help optimize marketing efforts and improve overall conversion rates without relying on broad assumptions about specific channels.

As for directions for future studies of the dataset, here are 2 options that can be explored:

- Explore Non-Linear Models
    - One next step would be to explore non-linear models like Random Forest or Gradient Boosting, which can capture complex interactions and non-linear relationships between variables. These models may be able to improve predictive accuracy by identifying patterns missed by logistic

regression. The organization can determine whether more complex models offer a significant advantage for conversion prediction by comparing their performance to logistic regression.

- Feature Engineering
    - Another area for future study involves feature engineering to create new variables that may improve model performance. For example, we could create interaction terms between AdSpend and ClickThroughRate, or generate time-based features to track customer engagement over specific periods of time. The organization can also gather new data related to customer preferences or behaviors (i.e. social media engagement metrics) to further enhance the model's predictive power.

By exploring these directions, the organization could build a more accurate and comprehensive model for predicting customer conversions, which leads to better marketing decisions and resource allocation and ultimately, increases profits for the company.

F: Sources

1. "Digital Marketing Analytics Dataset" Retrieved from https://www.kaggle.com/datasets/arpit2712/digital-marketing-company
2. "Data Visualization with Seaborn - Python" Retrieved from https://www.geeksforgeeks.org/data-visualization-with-python-seaborn/
3. "Python for Data 26: ANOVA" Retrieved from https://www.kaggle.com/code/hamelg/python-for-data-26-anova

4. “Logistic Regression - Class Imbalance SMOTE” Retrieved from https://www.kaggle.com/code/ghanender/logistic-regression-class-imbalance-smote
5. “Hyperparameter Tuning with GridSearchCV” Retrieved from https://www.mygreatlearning.com/blog/gridsearchcv/