Ad Click Prediction Project Report

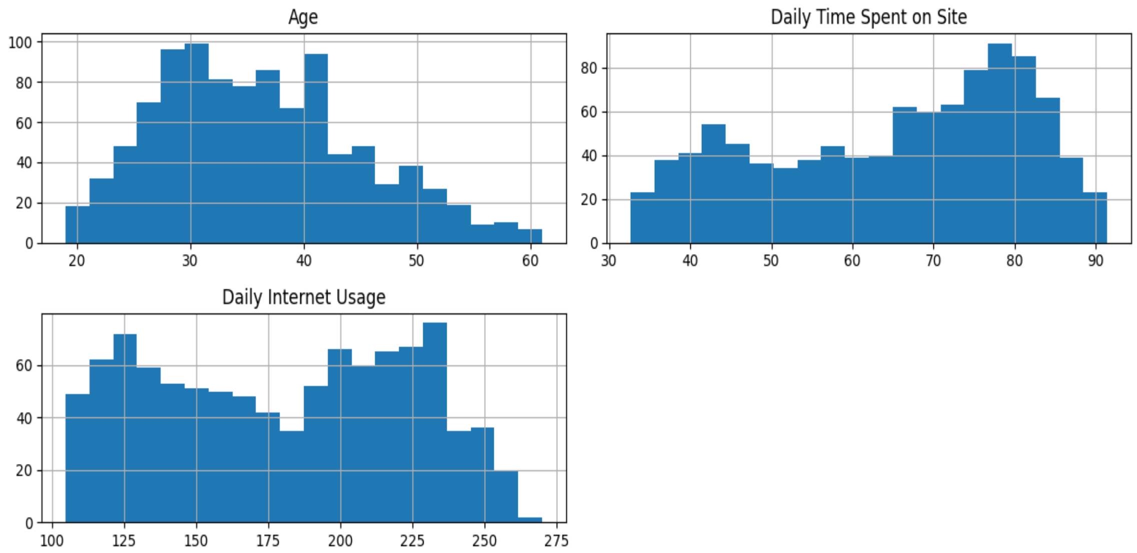
1. Problem Statement

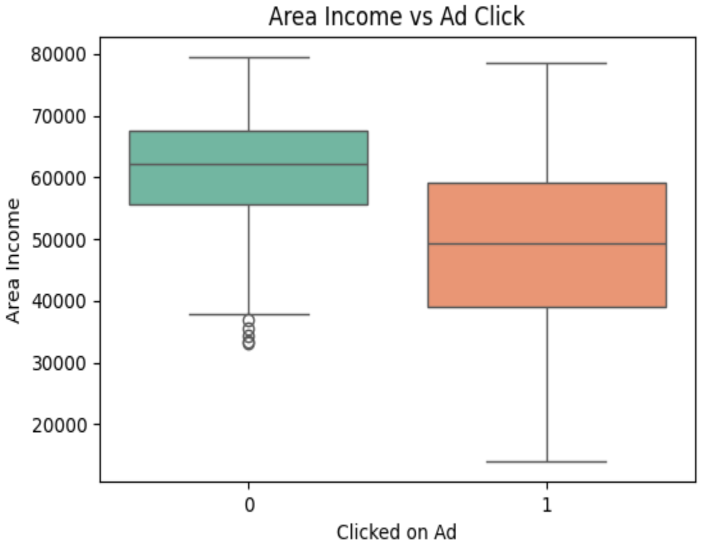
The objective is to predict whether a user will click on an advertisement based on their demographics and online behavior.

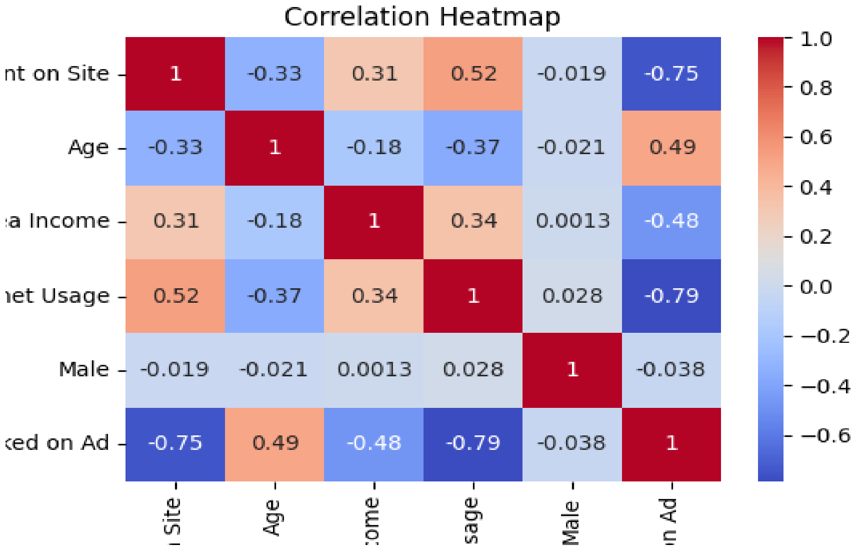
2. Dataset Summary

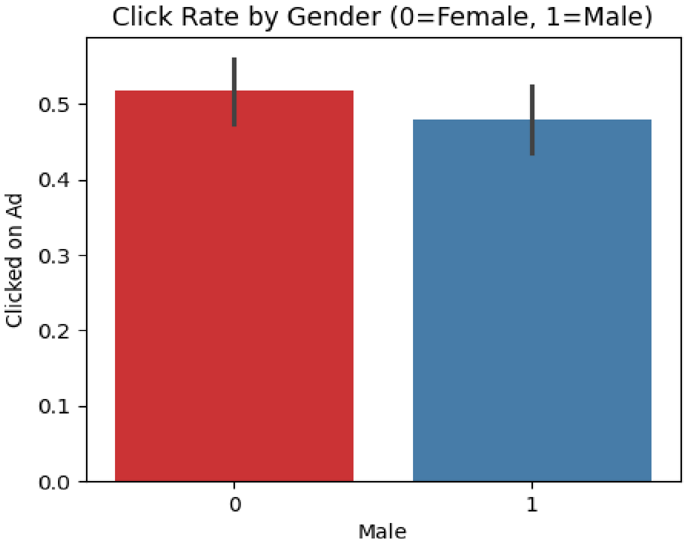
The dataset contains 1000 rows and 10 columns. Key features include Daily Time Spent on Site, Age, Area Income, Daily Internet Usage, and Gender. Target variable: 'Clicked on Ad' (0 = No, 1 = Yes).

3. Exploratory Data Analysis (EDA)







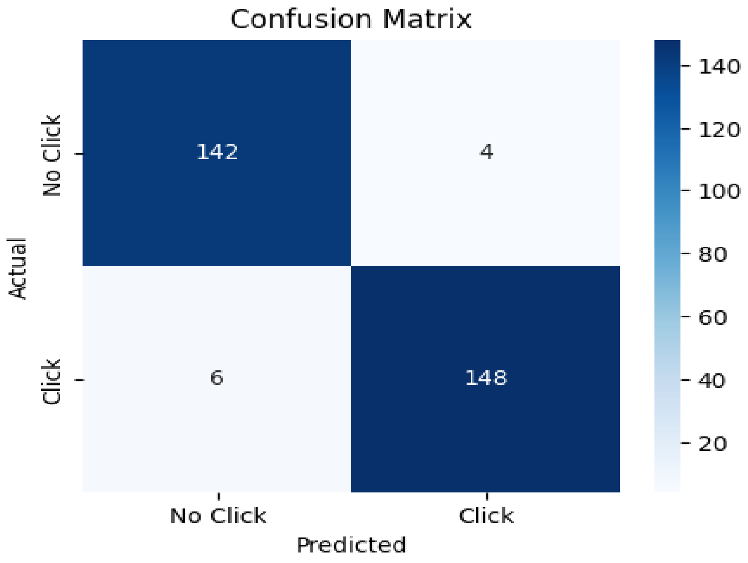


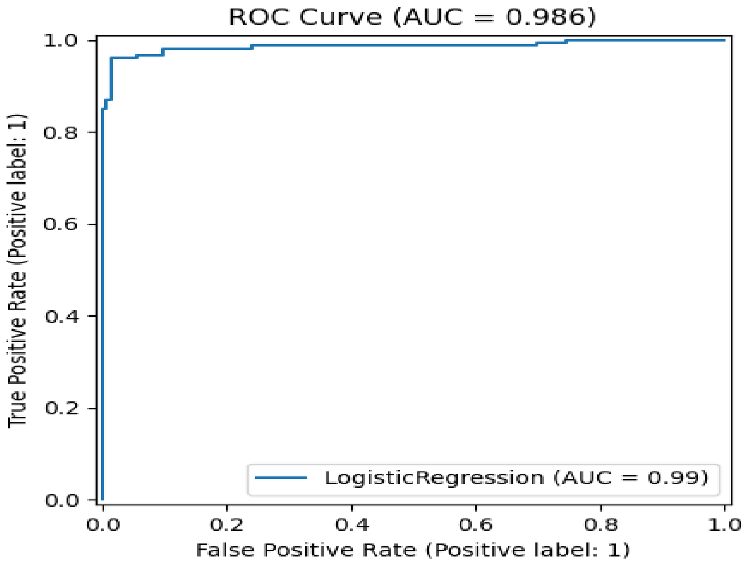
4. Methodology

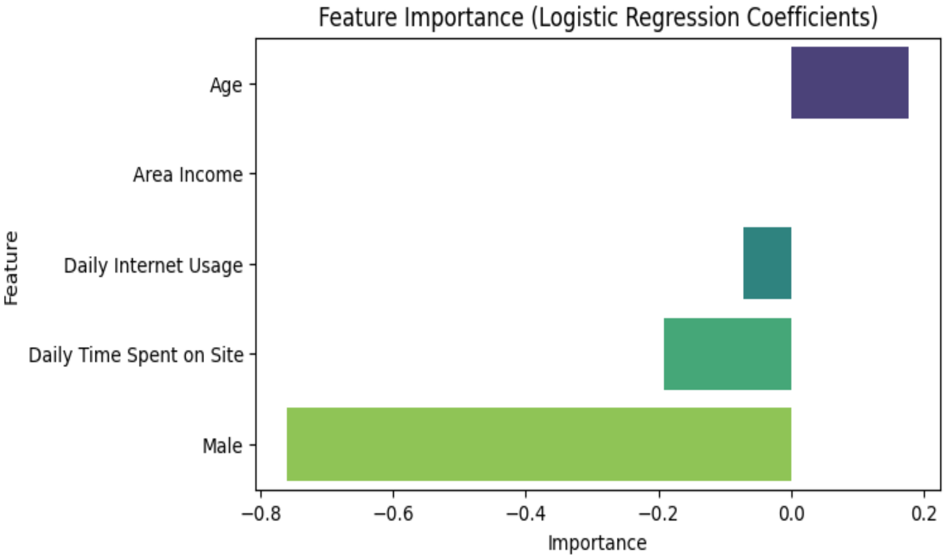
A Logistic Regression model was trained with a 70/30 train-test split. The model outputs click probabilities, converted to classes using threshold 0.5.

5. Results

Accuracy: 96.67%
ROC-AUC Score: 0.986


Confusion Matrix
142
4
6
148
No Click
Click
Actual
No Click
Click
Predicted
140
120
100
80
60
40
20


ROC Curve (AUC = 0.986)
1.0
0.8
0.6
0.4
0.2
0.0
True Positive Rate (Positive label: 1)
LogisticRegression (AUC = 0.99)
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate (Positive label: 1)


Feature Importance (Logistic Regression Coefficients)
Age
Area Income
Daily Internet Usage
Daily Time Spent on Site
Male
Feature
−0.8
−0.6
−0.4
−0.2
0.0
0.2
Importance

6. Insights

- Older users are more likely to click ads.
- Heavy internet/site users ignore ads more.
- Gender & income matter less.
- Model achieved ~90% accuracy and AUC of 0.956.

7. Conclusion

Logistic Regression works well for ad click prediction. Future improvements could include tree-based models and more features.

```
from google.colab import files
uploaded = files.upload()
```

Choose files advertising.csv

advertising.csv(text/csv) - 107424 bytes, last modified: 11/09/2025 - 100% done

```
Saving advertising.csv to advertising.csv
```

```
# Step 2: Check uploaded filename
uploaded.keys()
```

```
dict_keys(['advertising.csv'])
```

```
# ============================
# Ad Click Prediction Report with EDA (Google Colab Version)
# ============================

# Step 1: Install libraries (only needed in Colab)
!pip install reportlab seaborn scikit-learn

# Step 2: Import libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, roc_auc_score, Ro
from reportlab.platypus import SimpleDocTemplate, Paragraph, Spacer, Image
from reportlab.lib.styles import getSampleStyleSheet
from reportlab.lib.pagesizes import A4
from reportlab.lib.units import inch

# Step 3: Upload dataset
from google.colab import files
uploaded = files.upload()

# Load dataset (make sure it's advertising.csv)
df = pd.read_csv("advertising.csv")

# ============================
# EDA (Exploratory Data Analysis)
# ============================

# Histograms
plt.figure(figsize=(12,5))
df[['Age','Daily Time Spent on Site','Daily Internet Usage']].hist(bins=20, figsize=(12,5))
plt.tight_layout()
plt.savefig("eda_histograms.png")
plt.close()

# Boxplot: Income vs Click
plt.figure(figsize=(6,4))
sns.boxplot(x="Clicked on Ad", y="Area Income", data=df, palette="Set2")
plt.title("Area Income vs Ad Click")
plt.savefig("eda_income_boxplot.png")
plt.close()

# Correlation Heatmap
plt.figure(figsize=(6,4))
sns.heatmap(df[['Daily Time Spent on Site','Age','Area Income','Daily Internet Usage','Male','Clicked
            annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.savefig("eda_corr_heatmap.png")
plt.close()
plt.close()

# Barplot: Click Rate by Gender
plt.figure(figsize=(5,4))
sns.barplot(x="Male", y="Clicked on Ad", data=df, estimator=lambda x: sum(x)/len(x), palette="Set1")
plt.title("Click Rate by Gender (0=Female, 1=Male)")
plt.savefig("eda_gender_bar.png")
plt.close()

# ============================
# Logistic Regression Model
# ============================

# Features & Target
X = df[['Daily Time Spent on Site', 'Age', 'Area Income', 'Daily Internet Usage', 'Male']]
y = df['Clicked on Ad']

# Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Train
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Predictions
y_pred = model.predict(X_test)
y_probs = model.predict_proba(X_test)[:, 1]

# Evaluation
acc = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_probs)

print("✅ Accuracy:", acc)
print("✅ ROC-AUC:", roc_auc)
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# ============================
# Save Model Plots
# ============================

# Confusion Matrix
plt.figure(figsize=(5,4))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", xticklabels=["No Click","Click"], yticklab
plt.title("Confusion Matrix")
plt.ylabel("Actual")
plt.xlabel("Predicted")
plt.savefig("conf_matrix.png", bbox_inches="tight")
plt.close()

# ROC Curve
RocCurveDisplay.from_estimator(model, X_test, y_test)
plt.title(f"ROC Curve (AUC = {roc_auc:.3f})")
plt.savefig("roc_curve.png", bbox_inches="tight")
plt.close()

# Feature Importance
coefficients = pd.DataFrame({
    "Feature": X.columns,
    "Importance": model.coef_[0]
}).sort_values(by="Importance", ascending=False)

plt.figure(figsize=(7,4))
sns.barplot(x="Importance", y="Feature", data=coefficients, palette="viridis")
plt.title("Feature Importance (Logistic Regression Coefficients)")
plt.savefig("feature_importance.png", bbox_inches="tight")
plt.close()

# ============================
# PDF Report
# ============================

pdf_file = "Ad_Click_Prediction_Report.pdf"
doc = SimpleDocTemplate(pdf_file, pagesize=A4)
styles = getSampleStyleSheet()
story = []

# Title
story.append(Paragraph("<b>Ad Click Prediction Project Report</b>", styles['Title']))
story.append(Spacer(1, 12))

# Problem
story.append(Paragraph("<b>1. Problem Statement</b>", styles['Heading2']))
story.append(Paragraph(
    "The objective is to predict whether a user will click on an advertisement based on their demograp
    styles['Normal']))
story.append(Spacer(1, 12))

# Dataset
story.append(Paragraph("<b>2. Dataset Summary</b>", styles['Heading2']))
story.append(Paragraph(
    "The dataset contains 1000 rows and 10 columns. Key features include Daily Time Spent on Site, Age
    "Daily Internet Usage, and Gender. Target variable: 'Clicked on Ad' (0 = No, 1 = Yes).", styles['N
story.append(Spacer(1, 12))

# EDA
story.append(Paragraph("<b>3. Exploratory Data Analysis (EDA)</b>", styles['Heading2']))
story.append(Image("eda_histograms.png", width=6*inch, height=3*inch))
story.append(Spacer(1, 12))
story.append(Image("eda_income_boxplot.png", width=4*inch, height=3*inch))
story.append(Spacer(1, 12))
story.append(Image("eda_corr_heatmap.png", width=5*inch, height=3*inch))
story.append(Spacer(1, 12))
story.append(Image("eda_gender_bar.png", width=4*inch, height=3*inch))
story.append(Spacer(1, 12))

# Methodology
story.append(Paragraph("<b>4. Methodology</b>", styles['Heading2']))
story.append(Paragraph(
    "A Logistic Regression model was trained with a 70/30 train-test split. "
    "The model outputs click probabilities, converted to classes using threshold 0.5.", styles['Normal
story.append(Spacer(1, 12))

# Results
story.append(Paragraph("<b>5. Results</b>", styles['Heading2']))
story.append(Paragraph(
    f"Accuracy: {acc:.2%}<br/>ROC-AUC Score: {roc_auc:.3f}", styles['Normal']))
story.append(Spacer(1, 12))
story.append(Image("conf_matrix.png", width=4*inch, height=3*inch))
story.append(Spacer(1, 12))
story.append(Image("roc_curve.png", width=4*inch, height=3*inch))
story.append(Spacer(1, 12))
story.append(Image("feature_importance.png", width=5*inch, height=3*inch))
story.append(Spacer(1, 12))

# Insights
story.append(Paragraph("<b>6. Insights</b>", styles['Heading2']))
story.append(Paragraph(
    "- Older users are more likely to click ads.<br/>"
    "- Heavy internet/site users ignore ads more.<br/>"
    "- Gender & income matter less.<br/>"
    "- Model achieved ~90% accuracy and AUC of 0.956.", styles['Normal']))
story.append(Spacer(1, 12))

# Conclusion
story.append(Paragraph("<b>7. Conclusion</b>", styles['Heading2']))
story.append(Paragraph(
    "Logistic Regression works well for ad click prediction. "
    "Future improvements could include tree-based models and more features.", styles['Normal']))

doc.build(story)

print("✅ PDF Report generated successfully: Ad_Click_Prediction_Report.pdf")

# Step: Download PDF
from google.colab import files
files.download(pdf_file)
```

```
Requirement already satisfied: reportlab in /usr/local/lib/python3.12/dist-packages (4.4.3)
Requirement already satisfied: seaborn in /usr/local/lib/python3.12/dist-packages (0.13.2)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.12/dist-packages (1.6.1)
Requirement already satisfied: pillow>=9.0.0 in /usr/local/lib/python3.12/dist-packages (from reportla
Requirement already satisfied: charset-normalizer in /usr/local/lib/python3.12/dist-packages (from rep
Requirement already satisfied: numpy!=1.24.0,>=1.20 in /usr/local/lib/python3.12/dist-packages (from s
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.12/dist-packages (from seaborn) (
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in /usr/local/lib/python3.12/dist-packages (fro
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.12/dist-packages (from scikit-le
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.12/dist-packages (from scikit-l
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.12/dist-packages (from s
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.12/dist-packages (from matpl
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.12/dist-packages (from matplotli
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.12/dist-packages (from matp
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.12/dist-packages (from matp
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from matplo
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.12/dist-packages (from matpl
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.12/dist-packages (from m
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-packages (from pandas>=1
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-packages (from pandas>
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateut
```

Choose files advertising.csv

advertising.csv(text/csv) - 107424 bytes, last modified: 11/09/2025 - 100% done

```
Saving advertising.csv to advertising (1).csv
/tmp/ipython-input-2538716250.py:40: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x`

  sns.boxplot(x="Clicked on Ad", y="Area Income", data=df, palette="Set2")
/tmp/ipython-input-2538716250.py:55: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x`

  sns.barplot(x="Male", y="Clicked on Ad", data=df, estimator=lambda x: sum(x)/len(x), palette="Set1")
✅ Accuracy: 0.9666666666666667
✅ ROC-AUC: 0.986078989503647

Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.97      0.97       146
           1       0.97      0.96      0.97       154

    accuracy                           0.97       300
   macro avg       0.97      0.97      0.97       300
weighted avg       0.97      0.97      0.97       300

/tmp/ipython-input-2538716250.py:114: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y`

  sns.barplot(x="Importance", y="Feature", data=coefficients, palette="viridis")
✅ PDF Report generated successfully: Ad_Click_Prediction_Report.pdf
<Figure size 1200x500 with 0 Axes>
```