Projeto de Regressão Logística

Neste projeto trabalharemos com um conjunto de dados falso de publicidade, indicando se um usuário de internet específico clicou ou não em uma propaganda. Vamos tentar criar um modelo que preveja se clicará ou não em um anúncio baseado nos recursos desse usuário.

Este conjunto de dados contém os seguintes recursos:

- 'Daily Time Spent on Site': tempo no site em minutos.
- 'Age': idade do consumidor.
- 'Area Income': Média da renda do consumidor na região.
- 'Daily Internet Usage': Média em minutos por dia que o consumidor está na internet.
- 'Ad Topic Line': Título do anúncio.
- 'City': Cidade do consumidor.
- 'Male': Se o consumidor era ou não masculino.
- 'Country': País do consumidor.
- 'Timestamp': hora em que o consumidor clicou no anúncio ou janela fechada.
- 'Clicked on Ad': 0 ou 1 indicam se clicou ou não no anúncio.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:
```
ad_data = pd.read_csv('advertising.csv')
```

In [3]:
```
ad_data.head()
```

Out[3]:

	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Ad Topic Line	City	Male	Country	Timestamp	Clicked on Ad
0	68.95	35	61833.90	256.09	Cloned 5thgeneration orchestration	Wrightburgh	0	Tunisia	2016-03-27 00:53:11	0
1	80.23	31	68441.85	193.77	Monitored national standardization	West Jodi	1	Nauru	2016-04-04 01:39:02	0
2	69.47	26	59785.94	236.50	Organic bottom-line service-desk	Davidton	0	San Marino	2016-03-13 20:35:42	0
3	74.15	29	54806.18	245.89	Triple-buffered reciprocal time-frame	West Terrifurt	1	Italy	2016-01-10 02:31:19	0
4	68.37	35	73889.99	225.58	Robust logistical utilization	South Manuel	0	Iceland	2016-06-03 03:36:18	0

In [4]:
```
ad_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 10 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Daily Time Spent on Site  1000 non-null   float64
 1   Age                       1000 non-null   int64  
 2   Area Income               1000 non-null   float64
 3   Daily Internet Usage      1000 non-null   float64
 4   Ad Topic Line             1000 non-null   object 
 5   City                      1000 non-null   object 
 6   Male                      1000 non-null   int64  
 7   Country                   1000 non-null   object 
 8   Timestamp                 1000 non-null   object 
 9   Clicked on Ad             1000 non-null   int64  
dtypes: float64(3), int64(3), object(4)
memory usage: 78.2+ KB
```

In [5]:
```
ad_data.describe()
```

Out[5]:

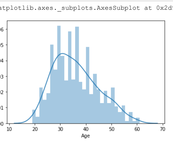
	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Male	Clicked on Ad
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000
mean	65.000200	36.009000	55000.000080	180.000100	0.481000	0.50000
std	15.853615	8.785562	13414.634022	43.902339	0.499889	0.50025
min	32.600000	19.000000	13996.500000	104.780000	0.000000	0.00000
25%	51.360000	29.000000	47031.802500	138.830000	0.000000	0.00000
50%	68.215000	35.000000	57012.300000	183.130000	0.000000	0.50000
75%	78.547500	42.000000	65470.635000	218.792500	1.000000	1.00000
max	91.430000	61.000000	79484.800000	269.960000	1.000000	1.00000

Análise exploratória de dados

Observando a distribuição de idade em um histograma

In [18]:
```
sns.distplot(ad_data['Age'], bins = 30)
#ad_data['Age'].hist(bins=30)
```

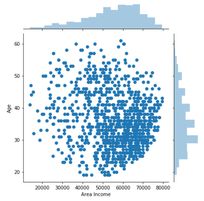
Out[18]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2d9baabd688>
```



Observando distribuição renda por idade

In [7]:
```
sns.jointplot(x = ad_data['Area Income'], y = ad_data['Age'])
```

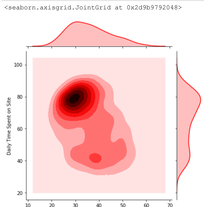
Out[7]:
```
<seaborn.axisgrid.JointGrid at 0x2d9b95e7408>
```



Observando a distribuição de tempo gasto no site por idade

In [8]:
```
sns.jointplot(y = ad_data['Daily Time Spent on Site'], x = ad_data['Age'], kind = 'kde', color='red')
```

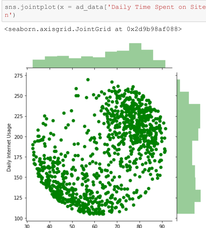
Out[8]:
```
<seaborn.axisgrid.JointGrid at 0x2d9b9792048>
```



Obseervando a distribuição entre "tempo gasto na internet por dia" em relação ai "tempo médio gasto no site"

In [9]:
```
sns.jointplot(x = ad_data['Daily Time Spent on Site'], y = ad_data['Daily Internet Usage'], color='green')
```

Out[9]:
```
<seaborn.axisgrid.JointGrid at 0x2d9b98af088>
```



Diversas distribuições diferenciadas por "Cliked on Ad" (se clicou ou não no anúncio)

In [10]:
```
sns.pairplot(ad_data, hue ='Clicked on Ad', palette = 'bwr',diag_kind = 'hist')
```

Out[10]:
```
<seaborn.axisgrid.PairGrid at 0x2d9b9a05d88>
```

Regressão Logística

Treinando o modelo

In [11]:
```
from sklearn.model_selection import train_test_split
```

In [12]:
```
X = ad_data.drop(['Clicked on Ad', 'Timestamp', 'City', 'Ad Topic Line', 'Country'], axis=1)
y = ad_data['Clicked on Ad']
#y = y[y.columns]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state = 42)
```

In [13]:
```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X_train, y_train)
```

Out[13]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

Previsões e avaliações

Prevendo

In [14]:
```
predictions = model.predict(X_test)
```

Relatório de classificação do modelo

In [21]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test, predictions))
```

```
              precision    recall  f1-score   support

           0       0.86      0.96      0.91       162
           1       0.96      0.85      0.90       168

    accuracy                           0.91       330
   macro avg       0.91      0.91      0.91       330
weighted avg       0.91      0.91      0.91       330
```