In [13]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [3]:

```
ad_data = pd.read_csv('advertising.csv')
```

In [4]:

```
ad_data.head()
```

Out[4]:

	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 [5]:

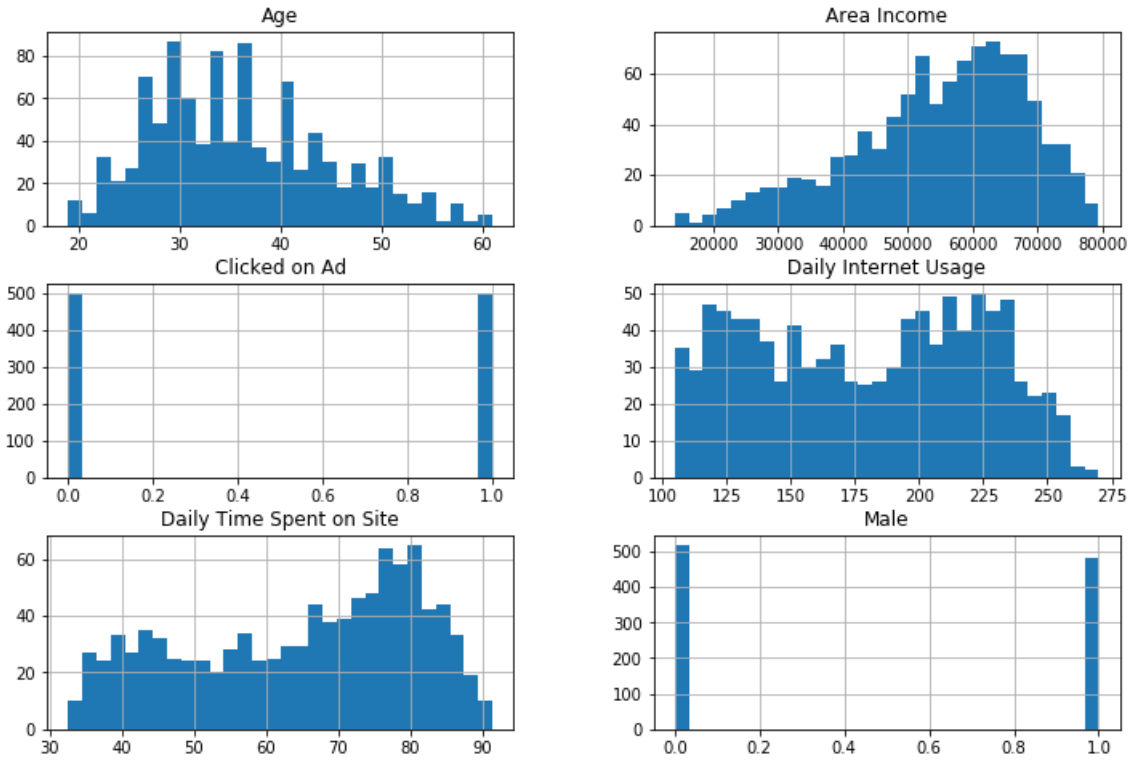
```
ad_data.shape
```

Out[5]:

```
(1000, 10)
```

In [7]:

```
ad_data.hist(bins=30, figsize=(12, 8));
```

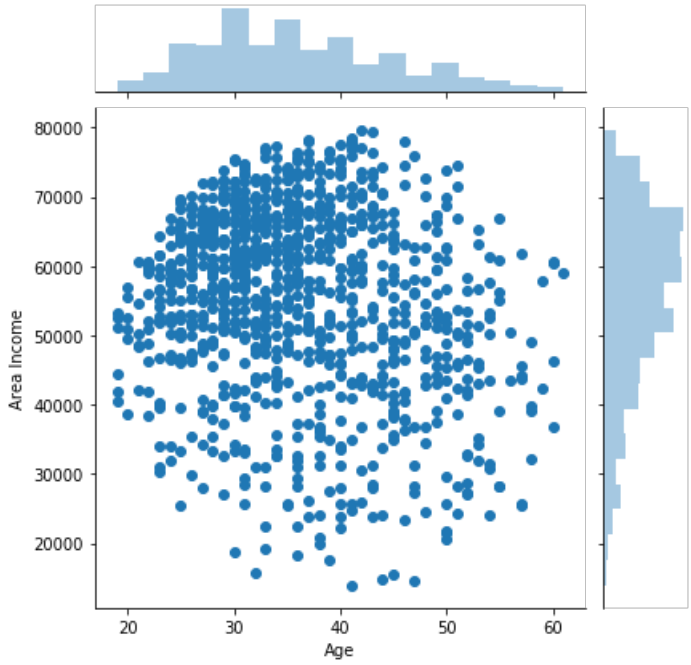


In [8]:

```
plt.figure(figsize=(12, 8))
sns.jointplot(x='Age', y='Area Income', data=ad_data);
```

```
/Users/sudeng/anaconda3/lib/python3.7/site-packages/scipy/stats/stats.py:1713: FutureWarning: Using a non-tuple sequence for multidimensional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted as an array index, `arr[np.array(seq)]`, which will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

```
<Figure size 864x576 with 0 Axes>
```

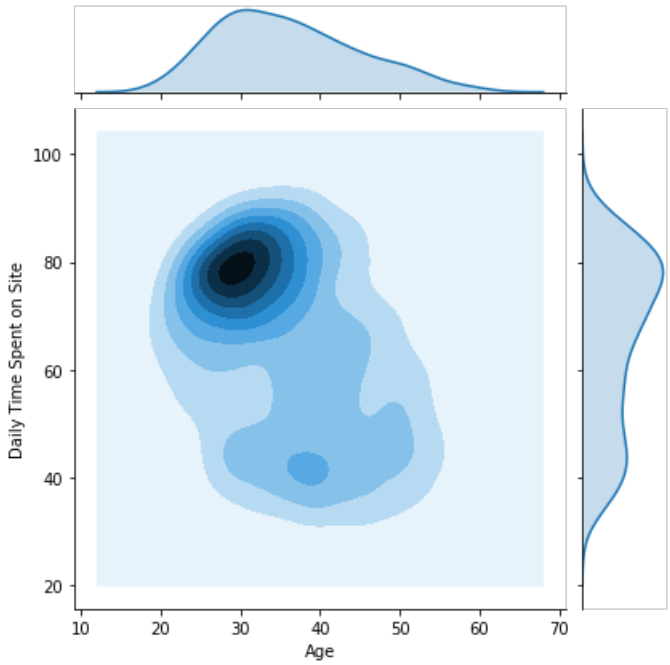


In [9]:

```
plt.figure(figsize=(12, 8))
sns.jointplot(x='Age', y='Daily Time Spent on Site', data=ad_data, kind='kde');
```

```
/Users/sudeng/anaconda3/lib/python3.7/site-packages/scipy/stats/stats.py:1713: FutureWarning: Using a non-tuple sequence for multidimensional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted as an array index, `arr[np.array(seq)]`, which will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

```
<Figure size 864x576 with 0 Axes>
```

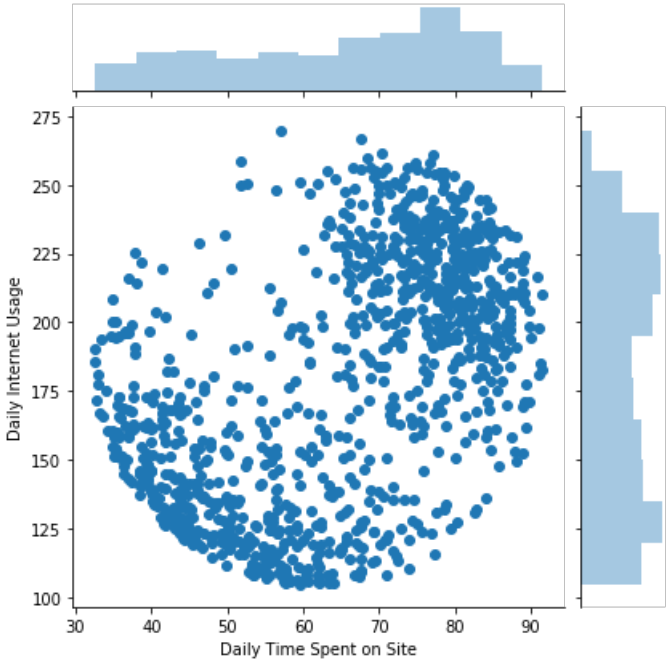


In [11]:

```
plt.figure(figsize=(12, 8))
sns.jointplot(x='Daily Time Spent on Site', y='Daily Internet Usage', data=ad_data);
```

```
/Users/sudeng/anaconda3/lib/python3.7/site-packages/scipy/stats/stats.py:1713: FutureWarning: Using a non-tuple sequence for multidimensional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted as an array index, `arr[np.array(seq)]`, which will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

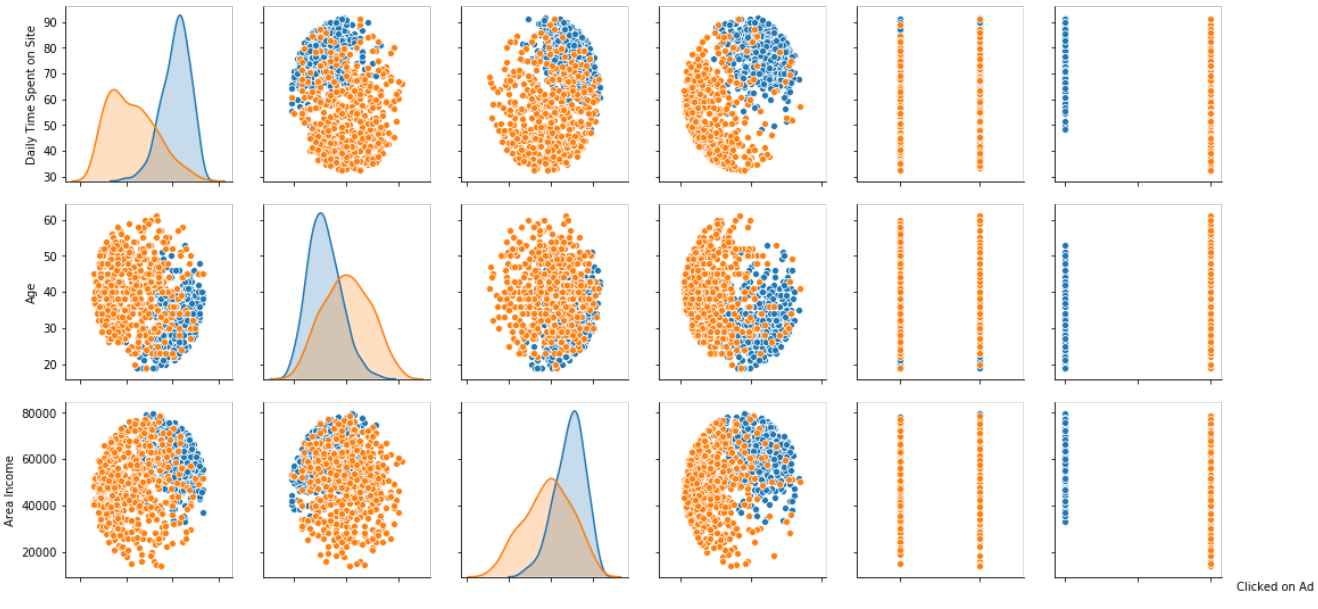
```
<Figure size 864x576 with 0 Axes>
```

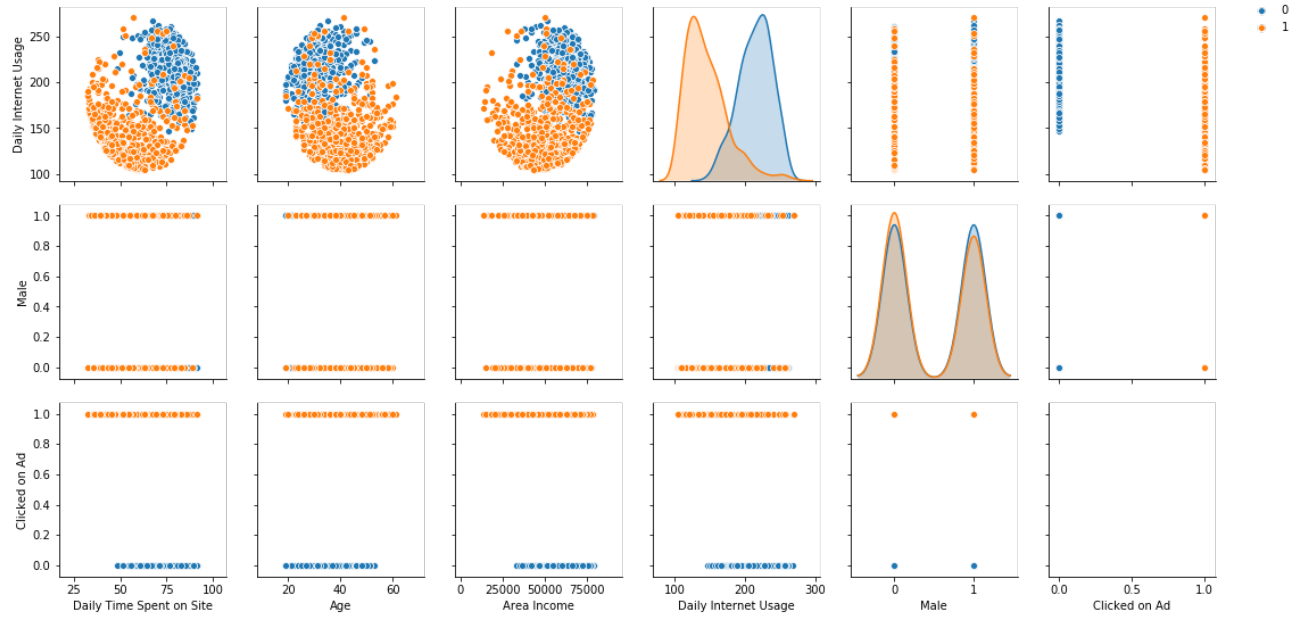


In [12]:

```
sns.pairplot(ad_data, hue='Clicked on Ad');
```

```
/Users/sudeng/anaconda3/lib/python3.7/site-packages/scipy/stats/stats.py:1713: FutureWarning: Using a non-tuple sequence for multidimensional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted as an array index, `arr[np.array(seq)]`, which will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
/Users/sudeng/anaconda3/lib/python3.7/site-packages/statsmodels/nonparametric/kde.py:488: RuntimeWarning: invalid value encountered in true_divide
  binned = fast_linbin(X, a, b, gridsize) / (delta * nobs)
/Users/sudeng/anaconda3/lib/python3.7/site-packages/statsmodels/nonparametric/kdetools.py:34: RuntimeWarning: invalid value encountered in double_scalars
  FAC1 = 2*(np.pi*bw/RANGE)**2
/Users/sudeng/anaconda3/lib/python3.7/site-packages/numpy/core/fromnumeric.py:83: RuntimeWarning: invalid value encountered in reduce
  return ufunc.reduce(obj, axis, dtype, out, **passkwargs)
```





In [14]:

```
from sklearn.cross_validation import train_test_split
```

In [16]:

```
ad_data.columns
```

Out[16]:

```
Index(['Daily Time Spent on Site', 'Age', 'Area Income',
       'Daily Internet Usage', 'Ad Topic Line', 'City', 'Male', 'Country',
       'Timestamp', 'Clicked on Ad'],
      dtype='object')
```

In [17]:

```
x = ad_data[['Daily Time Spent on Site', 'Age', 'Area Income',
       'Daily Internet Usage', 'Male']]
y = ad_data['Clicked on Ad']
```

In [18]:

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=101)
```

In [19]:

```
from sklearn.linear_model import LogisticRegression
```

In [20]:

```
logmodel = LogisticRegression()
```

In [22]:

```
logmodel.fit(x_train, y_train)
```

Out[22]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

In [23]:

```
predictions = logmodel.predict(x_test)
```

In [26]:

```
from sklearn.metrics import classification_report, confusion_matrix
```

In [25]:

```
classification_report(y_test, predictions)
```

Out[25]:

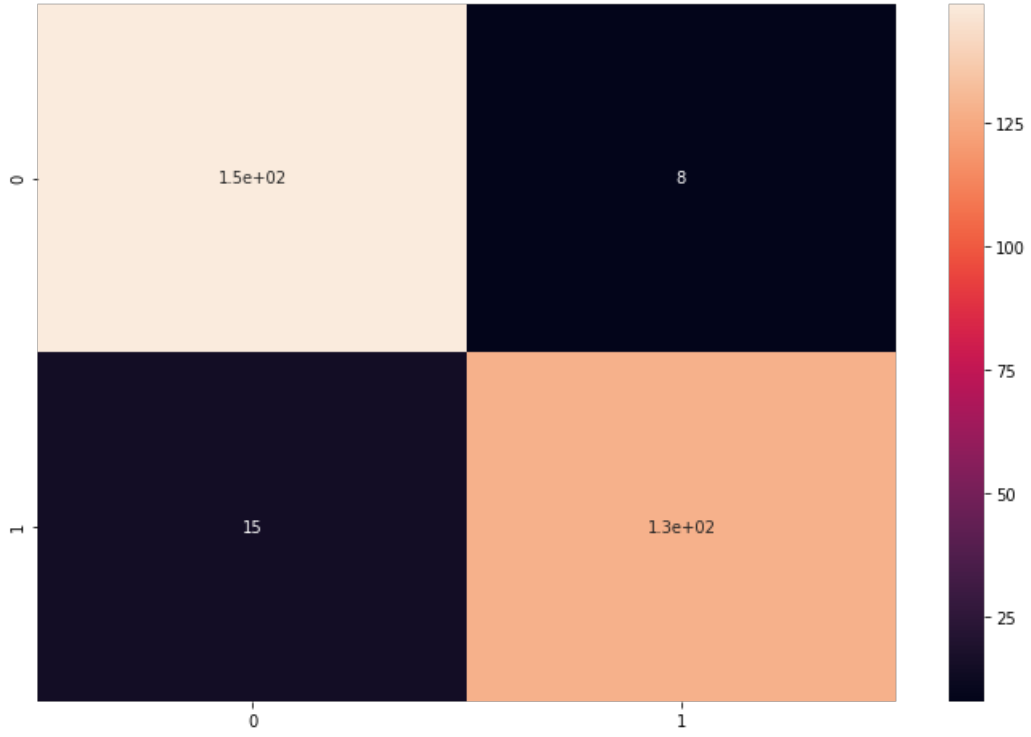
```
'              precision    recall  f1-score   support\n\n           0       0.91      0.95      0.93
157\n           1       0.94      0.90      0.92       143\n\navg / total       0.92      0.92      0.92
300\n'
```

In [27]:

```
cm = confusion_matrix(y_test, predictions)
```

In [29]:

```
plt.figure(figsize=(12, 8))
sns.heatmap(cm, annot=True);
```



In [ ]:

```

```