# linear-ecom-customers

March 19, 2024

[1]:

```
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

[2]:

```
df = pd.read_csv("ecomm-customers.csv")
df
```

[2]:

```
                             Email  \
0        mstephenson@fernandez.com
1                hduke@hotmail.com
2                 pallen@yahoo.com
3          riverarebecca@gmail.com
4    mstephens@davidson-herman.com
..                             …
495   lewisjessica@craig-evans.com
496            katrina56@gmail.com
497             dale88@hotmail.com
498            cwilson@hotmail.com
499      hannahwilson@davidson.com

                                               Address            Avatar  \
0         835 Frank Tunnel\nWrightmouth, MI 82180-9605            Violet
1       4547 Archer Common\nDiazchester, CA 06566-8576         DarkGreen
2    24645 Valerie Unions Suite 582\nCobbborough, D…            Bisque
3     1414 David Throughway\nPort Jason, OH 22070-1220       SaddleBrown
4    14023 Rodriguez Passage\nPort Jacobville, PR 3…  MediumAquaMarine
..                                                 …                …
495  4483 Jones Motorway Suite 872\nLake Jamiefurt,…               Tan
496  172 Owen Divide Suite 497\nWest Richard, CA 19320     PaleVioletRed
497  0787 Andrews Ranch Apt. 633\nSouth Chadburgh, …          Cornsilk
498  680 Jennifer Lodge Apt. 808\nBrendachester, TX…              Teal
499  49791 Rachel Heights Apt. 898\nEast Drewboroug…       DarkMagenta

     Avg. Session Length  Time on App  Time on Website  Length of Membership  \
0              34.497268    12.655651        39.577668              4.082621
1              31.926272    11.109461        37.268959              2.664034
2              33.000915    11.330278        37.110597              4.104543
3              34.305557    13.717514        36.721283              3.120179
```

```
4              33.330673    12.795189        37.536653              4.446308
..                     …            …             …                    …
495            33.237660    13.566160        36.417985              3.746573
496            34.702529    11.695736        37.190268              3.576526
497            32.646777    11.499409        38.332576              4.958264
498            33.322501    12.391423        36.840086              2.336485
499            33.715981    12.418808        35.771016              2.735160

     Yearly Amount Spent
0             587.951054
1             392.204933
2             487.547505
3             581.852344
4             599.406092
..                     …
495           573.847438
496           529.049004
497           551.620145
498           456.469510
499           497.778642

[500 rows x 8 columns]
```

[3]:
```
df.shape
```

[3]:
```
(500, 8)
```

[4]:
```
df.isnull().sum()
```

[4]:
```
Email                   0
Address                 0
Avatar                  0
Avg. Session Length     0
Time on App             0
Time on Website         0
Length of Membership    0
Yearly Amount Spent     0
dtype: int64
```

[5]:
```
x = df.drop(columns=["Email", "Address", "Avatar", "Yearly Amount␣
 ↪Spent"],axis=1)
x
```

[5]:
```
     Avg. Session Length  Time on App  Time on Website  Length of Membership
0              34.497268    12.655651        39.577668              4.082621
1              31.926272    11.109461        37.268959              2.664034
2              33.000915    11.330278        37.110597              4.104543
```

```
3            34.305557    13.717514    36.721283              3.120179
4            33.330673    12.795189    37.536653              4.446308
..                 …            …            …                   …
495          33.237660    13.566160    36.417985              3.746573
496          34.702529    11.695736    37.190268              3.576526
497          32.646777    11.499409    38.332576              4.958264
498          33.322501    12.391423    36.840086              2.336485
499          33.715981    12.418808    35.771016              2.735160

[500 rows x 4 columns]
```

[6]:
```
y = df["Yearly Amount Spent"]
y
```

[6]:
```
0      587.951054
1      392.204933
2      487.547505
3      581.852344
4      599.406092
          …
495    573.847438
496    529.049004
497    551.620145
498    456.469510
499    497.778642
Name: Yearly Amount Spent, Length: 500, dtype: float64
```

[7]:
```
from sklearn.model_selection import train_test_split
```

[8]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,␣
 ↪random_state=42)
```

[9]:
```
from sklearn.linear_model import LinearRegression
```

[10]:
```
model = LinearRegression().fit(x_train, y_train)
model
```

[10]:
```
LinearRegression()
```

[11]:
```
y_pred = model.predict(x_test)
y_pred
```

[11]:
```
array([402.86230051, 542.53325708, 426.62011918, 501.91386363,
       409.6666551 , 569.92155038, 531.50423529, 505.94309188,
       408.10378607, 473.45942928, 441.18668812, 424.52463471,
       424.83341694, 527.12061508, 430.87985533, 423.47062047,
       575.8751518 , 484.6563331 , 457.77896975, 481.58742311,
       501.56110993, 513.12815188, 507.49166899, 646.63377343,
       449.70050586, 496.26290484, 556.18523776, 554.78684161,
       399.1582784 , 325.16921284, 532.62732659, 477.73025415,
       500.76491535, 305.09971374, 505.46811902, 483.52069444,
       519.09464122, 437.75549737, 456.25005245, 470.63517876,
       494.11207805, 444.65549239, 508.57079732, 500.88197484,
       488.35128728, 535.34025218, 594.58301773, 513.59474408,
       279.69877702, 432.71590835, 421.06976164, 480.94327496,
       584.59481888, 608.61734059, 564.42312991, 494.47224504,
       393.95593318, 456.11321352, 572.92228417, 499.27385693,
       512.42973545, 391.56170305, 479.60705887, 481.05023229,
       474.71926117, 546.37716047, 430.11675694, 601.91418143,
       422.26508516, 493.11622454, 528.10614863, 581.06630842,
       620.60774498, 512.47838603, 411.2147464 , 498.07095351,
       461.44587681, 445.63453258, 447.63898998, 534.81030495,
       598.85091016, 619.46554961, 494.43362232, 672.2442837 ,
       532.15516513, 438.41740681, 514.80907179, 546.73893548,
       331.73069072, 510.33949236, 536.21660556, 499.50696031,
       375.86919792, 573.61952185, 479.18212334, 588.32862943,
       485.18137257, 455.93070091, 398.67820721, 451.70869105])
```

[12]:

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

[13]:

```
r2_sc = r2_score(y_test, y_pred)
```

[14]:

```
print(f"R2 Score = ", r2_sc)
```

```
R2 Score =  0.9778130629184127
```

OPTIMIZATION

[15]:

```
from sklearn.model_selection import GridSearchCV
```

[16]:

```
model = LinearRegression()
model
```

[16]:

```
LinearRegression()
```

[17]:

```
param_grid = {
     'fit_intercept':[True,False],
    'copy_X':[True, False],
    'n_jobs':[-1, None],
    'positive':[False, True]
}
```

[18]:

```
grid_search = GridSearchCV(model, param_grid, cv=5, n_jobs=-1)
grid_search.fit(x_train, y_train)
```

```
[18]: GridSearchCV(cv=5, estimator=LinearRegression(), n_jobs=-1,
                   param_grid={'copy_X': [True, False],
                               'fit_intercept': [True, False], 'n_jobs': [-1, None],
                               'positive': [False, True]})
```

```
[19]: best_params = grid_search.best_params_
      print("Best Parameters :", best_params)
```

```
Best Parameters : {'copy_X': True, 'fit_intercept': True, 'n_jobs': -1,
'positive': False}
```

```
[20]: best_model = LinearRegression(**best_params)
      best_model.fit(x_train, y_train)
      best_model
```

```
[20]: LinearRegression(n_jobs=-1)
```

```
[21]: y_pred = best_model.predict(x_test)
      y_pred
```

```
[21]: array([402.86230051, 542.53325708, 426.62011918, 501.91386363,
             409.6666551 , 569.92155038, 531.50423529, 505.94309188,
             408.10378607, 473.45942928, 441.18668812, 424.52463471,
             424.83341694, 527.12061508, 430.87985533, 423.47062047,
             575.8751518 , 484.6563331 , 457.77896975, 481.58742311,
             501.56110993, 513.12815188, 507.49166899, 646.63377343,
             449.70050586, 496.26290484, 556.18523776, 554.78684161,
             399.1582784 , 325.16921284, 532.62732659, 477.73025415,
             500.76491535, 305.09971374, 505.46811902, 483.52069444,
             519.09464122, 437.75549737, 456.25005245, 470.63517876,
             494.11207805, 444.65549239, 508.57079732, 500.88197484,
             488.35128728, 535.34025218, 594.58301773, 513.59474408,
             279.69877702, 432.71590835, 421.06976164, 480.94327496,
             584.59481888, 608.61734059, 564.42312991, 494.47224504,
             393.95593318, 456.11321352, 572.92228417, 499.27385693,
             512.42973545, 391.56170305, 479.60705887, 481.05023229,
             474.71926117, 546.37716047, 430.11675694, 601.91418143,
             422.26508516, 493.11622454, 528.10614863, 581.06630842,
             620.60774498, 512.47838603, 411.2147464 , 498.07095351,
             461.44587681, 445.63453258, 447.63898998, 534.81030495,
             598.85091016, 619.46554961, 494.43362232, 672.2442837 ,
             532.15516513, 438.41740681, 514.80907179, 546.73893548,
             331.73069072, 510.33949236, 536.21660556, 499.50696031,
             375.86919792, 573.61952185, 479.18212334, 588.32862943,
             485.18137257, 455.93070091, 398.67820721, 451.70869105])
```

```
[22]: r2_sc = r2_score(y_test, y_pred)
```

[23]:

```
print(f"R2 Score = ", r2_sc)
print("Best Parameters :", best_params)
```

```
R2 Score =  0.9778130629184127
Best Parameters : {'copy_X': True, 'fit_intercept': True, 'n_jobs': -1,
'positive': False}
```

[ ]:

```
```