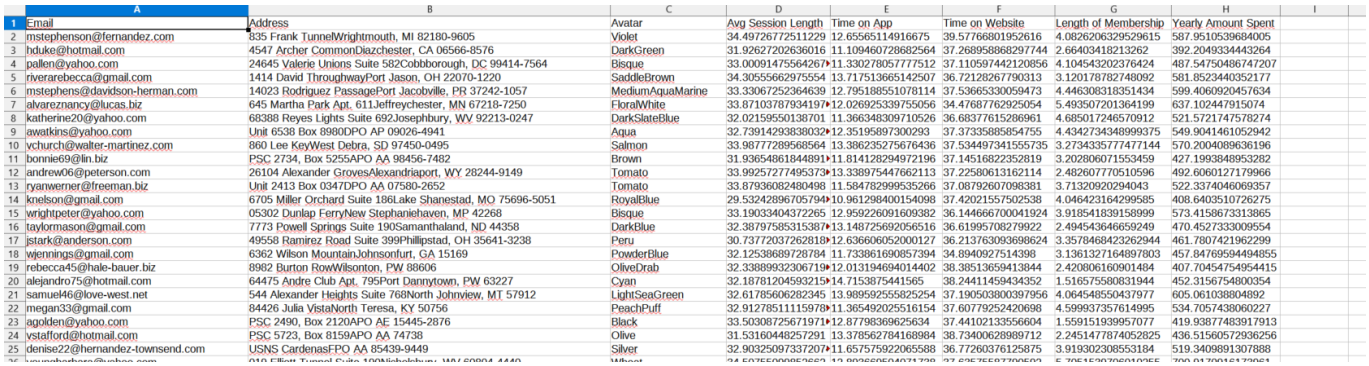
GOBIERNO DE ESPAÑA
MINISTERIO DE EDUCACIÓN Y FORMACIÓN PROFESIONAL
GENERALITAT VALENCIANA
Conselleria d'Educació, Investigació, Cultura i Esport
cefire
Fons Social Europeu
L'FSE inverteix en el teu futur
UNIÓ EUROPEA

# EJERCICIO MLLIB

# Sumario

# MLLIB

Ejercicio de Machine Learning con Spark Mllib.
Tenemos un excel (customers.csv) en el cual tenemos información sobre clientes de una web:



| | Email | Address | Avatar | Avg Session Length | Time on App | Time on Website | Length of Membership | Yearly Amount Spent |
|---|---|---|---|---|---|---|---|---|
| 2 | mstephenson@fernandez.com | 835 Frank TunnelWrightmouth, MI 82180-9605 | Violet | 34.49726772511229 | 12.65565114916675 | 39.57766801952616 | 4.0826206329529615 | 587.9510539684005 |
| 3 | hduke@hotmail.com | 4547 Archer CommonDiazchester, CA 06566-8576 | DarkGreen | 31.92627202636016 | 11.109460728682564 | 37.268958868297744 | 2.66403418213262 | 392.2049334443264 |
| 4 | pallen@yahoo.com | 24645 Valerie Unions Suite 582Cobbborough, DC 99414-7564 | Bisque | 33.000914755642677 | 11.330278057777512 | 37.110597442120856 | 4.104543202376424 | 487.54750486747207 |
| 5 | riverarebecca@gmail.com | 1414 David ThroughwayPort Jason, OH 22070-1220 | SaddleBrown | 34.305555662975554 | 13.717513665142507 | 36.72128267790313 | 3.120178782748092 | 581.8523440352177 |
| 6 | mstephens@davidson-herman.com | 14023 Rodriguez PassagePort Jacobville, PR 37242-1057 | MediumAquaMarine | 33.33067252364639 | 12.795188551078114 | 37.53665330059473 | 4.4463083183514 34 | 599.4060920457634 |
| 7 | alvareznancy@lucas.biz | 645 Martha Park Apt. 611Jeffreychester, MN 67218-7250 | FloralWhite | 33.871037879341976 | 12.026925339755056 | 34.47687762925054 | 5.493507201364199 | 637.102447915074 |
| 8 | katherine20@yahoo.com | 68388 Reyes Lights Suite 692Josephbury, WV 92213-0247 | DarkSlateBlue | 32.02159550138701 | 11.366348309710526 | 36.68377615286961 | 4.685017246570912 | 521.5721747578274 |
| 9 | awatkins@yahoo.com | Unit 6538 Box 8980DPO AP 09026-4941 | Aqua | 32.739142938380326 | 12.35195897300293 | 37.37335885854755 | 4.4342734348999375 | 549.9041461052942 |
| 10 | vchurch@walter-martinez.com | 860 Lee KeyWest Debra, SD 97450-0495 | Salmon | 33.98777289568564 | 13.386235275676436 | 37.534497341555735 | 3.2734335777477144 | 570.2004089636196 |
| 11 | bonnie69@lin.biz | PSC 2734, Box 5255APO AA 98456-7482 | Brown | 31.936548618448914 | 11.814128294972196 | 37.14516822352819 | 3.202806071553459 | 427.1993848953282 |
| 12 | andrew06@peterson.com | 26104 Alexander GrovesAlexandriaport, WY 28244-9149 | Tomato | 33.99257277495373 | 13.338975447662113 | 37.22580613162114 | 2.482607770510596 | 492.6060127179966 |
| 13 | ryanwerner@freeman.biz | Unit 2413 Box 0347DPO AA 07580-2652 | Tomato | 33.87936082480498 | 11.584782999535266 | 37.08792607098381 | 3.71320920294043 | 522.3374046069357 |
| 14 | knelson@gmail.com | 6705 Miller Orchard Suite 186Lake Shanestad, MO 75696-5051 | RoyalBlue | 29.532428967057947 | 10.961298400154098 | 37.42021557502538 | 4.046423164299585 | 408.6403510726275 |
| 15 | wrightpeter@yahoo.com | 05302 Dunlap FerryNew Stephaniehaven, MP 42268 | Bisque | 33.19033404372265 | 12.959226091609382 | 36.144666700041924 | 3.918541839158999 | 573.4158673313865 |
| 16 | taylormason@gmail.com | 7773 Powell Springs Suite 190Samanthaland, ND 44358 | DarkBlue | 32.387975853153877 | 13.148725692056516 | 36.61995708279922 | 2.494543646659249 | 470.4527333009554 |
| 17 | jstark@anderson.com | 49558 Ramirez Road Suite 399Phillipstad, OH 35641-3238 | Peru | 30.737720372628187 | 12.636606052000127 | 36.213763093698624 | 3.3578468423262944 | 461.7807421962299 |
| 18 | wjennings@gmail.com | 6362 Wilson MountainJohnsonfurt, GA 15169 | PowderBlue | 32.125386897287847 | 11.733861690857394 | 34.8940927514398 | 3.1361327164897803 | 457.84769594494855 |
| 19 | rebecca45@hale-bauer.biz | 8982 Burton RowWilsonton, PW 88606 | OliveDrab | 32.338899323067197 | 12.013194694014402 | 38.385136594138444 | 2.4208061609014 84 | 407.70454754954415 |
| 20 | alejandro75@hotmail.com | 64475 Andre Club Apt. 795Port Dannytown, PW 63227 | Cyan | 32.187812045932157 | 14.71538754415 65 | 38.24411459434352 | 1.5165755808319 44 | 452.3156754800354 |
| 21 | samuel46@love-west.net | 544 Alexander Heights Suite 768North Johnview, MT 57912 | LightSeaGreen | 32.617856062823457 | 13.989592555825254 | 37.190503800397956 | 4.064548550437977 | 605.061038804892 |
| 22 | megan33@gmail.com | 84426 Julia VistaNorth Teresa, KY 50756 | PeachPuff | 32.912785111159787 | 11.365492025516154 | 37.60779252420698 | 4.599937357614995 | 534.7057438060227 |
| 23 | agolden@yahoo.com | PSC 2490, Box 2120APO AE 15445-2876 | Black | 33.503087256719717 | 12.877983696256 34 | 37.441021335566004 | 1.5591519399570 77 | 419.9387748391791 3 |
| 24 | vstafford@hotmail.com | PSC 5723, Box 8159APO AA 74738 | Olive | 31.531604482572917 | 13.378562784168984 | 38.73400628989712 | 2.2451477874052825 | 436.5156057293625 6 |
| 25 | denise22@hernandez-townsend.com | USNS CardenasFPO AA 85439-9449 | Silver | 32.903250973372077 | 11.657575922065588 | 36.77260376125875 | 3.9193023085531 84 | 519.3409891307888 |

La idea es que a partir del tiempo de sesion, tiempo en la app, tiempo en la web… podamos saber cuanto dinero se ha gastado el usuario en un año.

**1.- Importamos nuestros datos de clientes, en este caso vamos a importar un csv con la información usamos inferschema para que nos coja el tipo de dato y header true para que nos cree las cabeceras**

```
datos = spark.read.csv("customers.csv",inferSchema = True,header = True)
```

**2.- Debemos saber que para trabajar, el modelo va a necesitar un formato concreto que es de tipo vector que necesita spark para trabajar, esto lo vamos a conseguir usando lo que se llama Assembler**

**Vamos a crear un Assembler el cual nos permite crear el formato que necesita el modelo para trabajar con spark(formato de vector) para ello primero necesitamos importar:**

```
from pyspark.ml.linalg import Vectors
from pyspark.ml.feature import VectorAssembler
```

**3.- Vamos a crear nuestro Assembler crearemos con la funcion que acabamos de importar, añadimos en el inputCols las columnas que queremos que estén en nuestro vector y el segundo elemento que es outputCols que se va a llamar features**

assembler = VectorAssembler(inputCols = ['Avg Session Length', 'Time on App', 'Time on Website','Length of Membership'], outputCol='features')

**4.- Vamos a crear un output utilizando el assembles que acabamos de crear y transform usando los datos,para tener los datos como necesitamos**

output = assembler.transform(datos)

**5.- Ahora vamos a coger los datos finales, que son el vector que hemos creado que será lo que nos ayude a predecir, y lo que queremos predecir que es el Yearly Amount Spent**

datos_finales = output.select('features','Yearly Amount Spent')

**6.- A partir de estos datos hacemos un train y un test usando randomSplit, le vamos a dar 0,7 y 0,3, lo que estamos haciendo aquí es coger el 70% de los datos para entrenar el modelo y un 30 para testear el modelo**

train, test = datos_finales.randomSplit([0.7,0.3]

**7.- Consultamos los datos y vemos que tenemos 354 registros para entrenar:**

train.describe().show()

```
+-------+------------------+
|summary|Yearly Amount Spent|
+-------+------------------+
|  count|               354|
|   mean| 496.25968353126103|
| stddev|  78.68652306131044|
|    min| 256.67058229005585|
|    max|  765.5184619388373|
+-------+------------------+
```

**Comprobamos los datos que tenemos para testear y vemos que en test tenemos 146 registros**

test.describe().show()

```
+-------+------------------+
|summary|Yearly Amount Spent|
+-------+------------------+
|  count|               146|
|   mean| 506.71980246047167|
| stddev|  80.60943345005087|
|    min| 327.37795258965207|
|    max|  725.5848140556806|
+-------+------------------+
```

**8. Vamos a entrenar el modelo con train y evaluar el modelo con test, entonces vamos a crear una regresión lineal pasándole features que son los datos que vamos a usar para entrenar y el dato que queremos calcular en este caso el yearly amount Spent, y por ultimo añadiremos una columna de predicción que será el resultado que vamos a obtener**

lr = LinearRegression(featuresCol = 'features',labelCol = 'Yearly Amount Spent',predictionCol = 'prediction')

**Si por defecto tienen los nombres features y label no hace falta indicarle el nombre, pero como aquí se llama Yearly... se lo pasamos**

**9.- Creamos el modelo con lr.fit() y le pasamos los datos de entrenamiento**

lr_modelo = lr.fit(train)

**10.Evaluamos los datos test**

resultado = lr_modelo.evaluate(test)

**11. Con residuals podemos ver en lo que se ha equivocado por predicción**

resultado.residuals.show()

```
+------------------+
|         residuals|
+------------------+
|  8.178019986211666|
|-11.662579206420503|
|0.28375387925063933|
|  23.88033922548516|
|  5.328554360332532|
| 1.5247878290938388|
| -4.585938823420406|
|  5.016282615349667|
|   17.7858835593359|
| -1.585812084553197|
|-3.2538229452102883|
|-16.324447019822003|
| 1.3243068169010712|
|  8.979596065303951|
| -16.35742247597409|
| -3.271236363257799|
|   4.72471901011221|
|  5.612109941364338|
| 7.1134779211603245|
|  23.82819269785142|
+------------------+
only showing top 20 rows
```

**Si hacemos:**

resultado.rootMeanSquaredError

```
In [24]: resultado.rootMeanSquaredError
Out[24]: 10.522935932789585
```

**Podemos ver en lo que se ha equivocado, por lo que si los precios son de 400 500 dólares, se esta equivocando en unos 9 dólares, podemos considerar que es una muy buena predicción Si el valor que tenemos que calcular es 9 y el error nos da 9, pues obviamente es muy mal resultado pero en nuestro caso que son valores altos, 9 es muy poco**

**12. Por ultimo vamos a hacer un sumary y ver las predicciones:**

summary = lr_modelo.summary

summary.predictions.show()

```
+------------------+------------------+-----------------+
|          features|Yearly Amount Spent|       prediction|
+------------------+------------------+-----------------+
|[29.5324289670579...|  408.6403510726275| 396.2129552626909|
|[30.3931845423455...|  319.9288698031936| 330.7251777036697|
|[30.4925366965402...|  282.4712457199145| 287.1825725547285|
|[30.5743636841713...| 442.06441375806565| 440.3762661797134|
|[30.7377203726281...|  461.7807421962299|449.70867929914334|
|[30.8162006488763...|   266.086340948469| 282.6664667274133|
|[30.8364326747734...|  467.5019004269896|470.24814039939565|
|[30.8794843441274...|  490.2065999848547|492.21998777593853|
|[31.0472221394875...|  392.4973991890214|386.77926727686736|
|[31.0613251567161...|  487.5554580579016|  492.245354616555|
|[31.1239743499119...|  486.9470538397658|507.06124817285877|
|[31.1280900496166...|  557.2526867470547| 562.4946330032003|
|[31.1695067987115...|  427.3565308022928|416.07943078351013|
|[31.2681042107507...|  423.4705331738239|426.60481489817926|
|[31.3091926408918...|  432.7207178399336|428.90727593277074|
|[31.3123495994443...|  463.5914180279406|443.30478342340825|
|[31.3662121671876...|  430.5888825564849| 425.9061480940511|
|[31.4252268808548...|  530.7667186547619| 533.0027416017085|
|[31.4459724827577...| 484.87696493512857| 480.5407962121369|
|[31.4474464941278...|   418.602742095224|  425.332599466542|
+------------------+------------------+-----------------+
only showing top 20 rows
```

**Como podemos observar nuestro modelo funciona bastante bien ya que las predicciones son muy buenas**