

Desafío - Implementar modelos desde AWS EMR

- Para realizar este desafío debes haber estudiado previamente todo el material disponibilizado correspondiente a la unidad.
- Una vez terminado el desafío, comprime la carpeta que contiene el desarrollo de los requerimientos solicitados y sube el `.zip` en el LMS.
- Desarrollo desafío:
 - El desafío se debe desarrollar de manera Individual/Grupal.
 - Adjunta todos los screenshots solicitados, así como los scripts y la dirección de su bucket s3, la cual debe hacerla pública.

Ejercicio 1 - Simulación de datos

- En esta actividad trabajaremos con datos simulados.
- El primer ejercicio tiene que ver con generar dos archivos csv con las siguientes columnas:

Variable	Código
deliverer_id	<code>np.random.choice(range(100), 1)[0]</code>
delivery_zone	<code>np.random.choice(['I', 'II', 'III', 'IV', 'V', 'VI', 'VII', 'VIII'])</code>
monthly_app_usage	<code>np.random.poisson(15)</code>
subscription_type	<code>np.random.choice(['Free', 'Prepaid', 'Monthly', 'Trimestral', 'Semestral', 'Yearly'], 1, [.30, .20, 10, .15, .20, .05])[0]</code>
paid_price	<code>np.random.normal(25.45, 10)</code>
customer_size	<code>np.random.poisson(2) + 1</code>
menu	<code>np.random.choice(['Asian', 'Indian', 'Italian', 'Japanese', 'French', 'Mexican'], 1)[0]</code>
delay_time	<code>np.random.normal(10, 3.2)</code>

Utilizando una instancia AWS EMR:

- (En local) Escriba un script llamado `simulate_data.py` que permita simular un número finito de registros con las columnas.
- Suba el script a su instancia de trabajo. (Adjunte screenshot de esta etapa)
- Desde su instancia de trabajo, ejecute el script para que simule 1000 datos, con una semilla pseudoaleatoria=11238 (puede ocupar `np.random.seed` para esto) . Guarde esos datos con el nombre `train_delivery_data.csv`. (Adjunte screenshot de esta etapa)
- Desde su instancia de trabajo, ejecute el script para que simule 10000 datos, con una semilla pseudoaleatoria=42. Guarde estos datos con el nombre `test_delivery_data.csv`. (Adjunte screenshot de esta etapa)
- Desde su instancia de trabajo, copie ambos `csv` a un bucket S3 que cree usted mismo.

```
0,VI,Semestral,17,4,16.94099079811256,French,11.332126139995461
98,VII,Free,18,5,39.06417712024473,Asian,10.429259959053653
14,IV,Trimestral,16,2,32.82654382121793,Italian,7.9008918807783
72,V,Free,17,3,7.556762164991376,Mexican,14.689806469978173
11,VIII,Semestral,12,1,34.20068369067669,Indian,12.51384063783114
81,VII,Free,27,3,21.51587429598476,French,5.141044125803192
85,IV,Semestral,21,5,29.970787225017745,French,7.534726624956967
21,VII,Prepaid,10,5,26.25805639563678,French,8.989187910382848
68,II,Yearly,14,2,41.04552212183895,Indian,11.1300756818118031
```

Ejercicio 2 - Entrenamiento de modelos

- **Utilizando los datos de entrenamiento en su instancia EMR.** Genere un script `train_models.py` que lea los datos de entrenamiento e implemente los siguientes puntos:
 - Recodifique el vector objetivo `delay_time` entre 1 y 0, identificando como 1 aquellos casos donde hubo una demora superior al promedio.
 - Para aquellos atributos string, recodifíquelos en K-1 columnas, manteniendo la primera categoría como referencia. Elimine el atributo original. Puede utilizar `pd.get_dummies` para eso.
 - Genere conjuntos de entrenamiento y validación con los atributos y el vector objetivo, preservando un .33 para test y un random state de 11238.
 - Entrene los siguientes modelos `LogisticRegression`, `DecisionTreeClassifier`, `RandomForestClassifier`, `GradientBoostingClassifier` y `BernoulliNB` **sin modificar hiperparámetros.**
 - Genere un print para cada modelo donde presente el `classification_report`.
 - Suba el script a su instancia de trabajo. (Adjunte screenshot de esta etapa)
 - Ejecute el script desde la instancia de trabajo y preserve el output con el nombre `candidate_models.txt`. (Adjunte screenshot de esta etapa)
 - Preserve el mejor modelo en un archivo serializado `.pkl`.

Ejercicio 3 - Evaluación y predicción

- **Utilizando su mejor modelo y sus datos de validación generados.** Genere un script llamado `predict_model.py` que lea los datos de validación e implemente los siguientes puntos:
 - Implemente el preprocesamiento de los datos que realizó en el conjunto de entrenamiento.
 - (Desde su instancia de trabajo) Con el mejor modelo entrenado, genere las predicciones de los datos y genere las siguientes evaluaciones:
 - Evalúe cuál es la probabilidad que un pedido se retrase por sobre la media, para cada una de las zonas de envío. (adjunte screenshot de esta etapa)
 - Evalúe cuál es la probabilidad que un pedido se retrase por sobre la media, para cada uno de los repartidores. (adjunte screenshot de esta etapa)
 - Evalúe cuál es la probabilidad que un pedido se retrase por sobre la media, para cada uno de los menús. (adjunte screenshot de esta etapa)
 - Evalúe cuál es la probabilidad que un pedido se retrase por sobre la media, para cada una de las suscripciones. (adjunte screenshot de esta etapa).
 - Genere un archivo con nombre `eval_pr.txt` con todas las probabilidades solicitadas.

Ejercicio 4

- (Desde su instancia de trabajo) Suba los resultados de sus `.txt` y sus archivos `.py` al bucket S3 creado en el punto 1. Haga público su bucket y envíe la ruta de este.
- Termine la instancia de trabajo. (Adjunte screenshot de esta etapa)
- En un `.zip`, adjunte los siguientes archivos:
 - Los scripts `simulate_data.py`, `train_models.py` y `predict_model.py`.
 - Los datos generados `train_delivery_data.py` y `test_delivery_data.py`.
 - Los resultados `candidate_models.txt` y `eval_pr.txt`.
 - Los screenshots de la actividad 1, 2 y 3.
 - El screenshot de la terminación de la instancia.