PROYECTO FINAL

Inteligencia Empresarial

Gonzalo Tamariz-Martel Sánchez
201810031@alu.comillas.edu

Contents

1. Introducción 2
2. Análisis e inspección de los datos 3
Visualización de los datos 4
Conclusiones extraídas tras la inspección de los datos 7
Módulo 1: Predicción de contratación de depósito a plazo fijo 8
Tratamiento de los datos 8
Modelo 1: KNN 9
Modelo 2: Random Forest 11
Modelo 3: GradientBoosting 14
Modelo 4: MLP 16
Comparativa de los modelos 18
Módulo 2: Segmentación de clientes de la Entidad 20
Modelo 1: K-Means 20
Modelo 2: HC 21
Conclusiones y reflexiones finales 23

1. Introducción

En esta sección se van a definir los objetivos y necesidades que cubriría el proyecto. Además, se van a plantear las distintas partes del proyecto indicando los algoritmos que se emplearán en estas.

El conjunto de datos que se va a emplear está titulado como "*Bank Marketing Data Se*t" y se puede encontrar en el repositorio de *Machine Learning* de la UCI. Este conjunto de datos se puede encontrar en: https://archive.ics.uci.edu/ml/datasets/bank+marketing.

Los datos de este conjunto están relacionados con campañas directas de marketing realizadas mediante llamadas telefónicas por una institución bancaria portuguesa. El *dataset* cuenta con una variable *target* indicando si un cliente contrató el producto, un depósito bancario a plazo fijo, o no.

Bank Marketing Data Set cuenta con varias versiones. Para este proyecto se va a emplear la versión "bank-additional.csv". Esta versión cuenta con 4119 ejemplos seleccionados aleatoriamente de la versión completa del *dataset*. Cuenta con 20 *features* o *inputs* y una variable *target*.

En este proyecto final se quiere realizar un estudio de los datos para dar respuesta a las siguientes dos preguntas:

- En función de los parámetros recopilados de un cliente, ¿contratará un depósito bancario a plazo fijo?
- En función de los datos disponibles de clientes, ¿a quién debería dirigirse las campañas de marketing?

Para contestar a estas preguntas, se va a hacer uso de algoritmos y modelos de Machine Learning vistos en la asignatura.

1. Uso y comparativa de distintos algoritmos de clasificación con el objetivo de predecir si un cliente contrató o no un depósito bancario a plazo fijo. Los algoritmos empleados serán los siguientes:
    - KNN
    - Red neuronal Perceptrón Multicapa (MLP)
    - Random Forest
    - Gradient Boosting
2. Realizar una segmentación de los clientes del banco mediante algoritmos de clustering para conocer los perfiles de cliente que tiene el banco pudiendo extraer información valiosa para las campañas de marketing. Los algoritmos de clustering serán los siguientes:
    - K-Means
    - Hierarchical Clustering (HC)

De este modo, se podrán emplear los datos para contestar a las preguntas planteadas, extrayendo información que ayude a la toma de decisiones estratégicas, permitiendo que una institución bancaria pueda centrar sus esfuerzos de marketing en aquellos clientes que puedan llevar a mayores ganancias para la Entidad. El primer objetivo, puede ayudar a la Entidad a determinar si un cliente es probable que contrate un depósito bancario a plazo fijo, por lo que podrá realizar acciones comerciales sobre este. El segundo objetivo, permitirá segmentar los

clientes, agrupándolos con otros clientes de características similares y pudiendo de este modo realizar campañas de marketing específicas para cada grupo.

2. Análisis e inspección de los datos

Durante esta sección, se pretende profundizar en el dataset que se va a tratar durante el proyecto. Tal y como se ha explicado antes, este ha sido obtenido del repositorio oficial de la UCI. El *dataset* empleado *Bank Marketing Data Set*, concretamente la versión bank-addtional cuenta con 4119 ejemplos. Esta versión contiene una fracción de los datos del conjunto original.

El *dataset* cuenta con 21 variables de entrada o *features* y una variable *target* indicando si un cliente contrató un depósito bancario a plazo fijo ("Yes"), o si, por el contrario, no lo hico ("No"). Está formado por datos de una institución bancaria de Portugal. Los datos están relacionados con campañas de marketing directo que se llevaron a cabo por la entidad mediante llamadas telefónicas. Además, se indica que en ocasiones se necesitó más de un contacto con el mismo cliente para saber si obtenía el depósito bancario a plazo fijo.

A continuación, se va a explicar de forma resumida las variables disponibles en el conjunto de datos indicando si se trata de una variable categórica o numérica.

Datos del cliente:

- **Age:** Edad del cliente → Numérica
- **Job:** Tipo de trabajo → Categórica
- **Marital:** Estado civil → Categórica
- **Educación:** Nivel de estudios → Categórica
- **Default:** ¿Tiene créditos en estado de morosidad? Yes/No → Categórica
- **Housing:** ¿Tiene préstamo para vivienda? Yes/No → Categórica
- **Loan:** ¿Tiene préstamo personal? Yes/No → Categórica

Datos del cliente relacionados con el último contacto de la campaña actual:

- **Contact[1]:** Tipo de comunicación de contacto → Categórica
- **Month[1]:** Último mes de contacto del año → Categórica
- **Day_of_week[1]:** Último día de contacto de la semana → Categórica
- **Duration[1]:** Duración del último contacto, en segundos → Numérica

Otros atributos:

- **Campaign:** Número de contactos realizados durante esta campaña y para este cliente → Categórica
- **Pdays:** Número de días transcurridos desde la última vez que se contactó con el cliente en una campaña anterior - Categórica
- **Previous:** Número de contactos realizados antes de esta campaña y para este cliente → Numérica
- **Poutcome:** Resultado de la última campaña → Categórica

[1]Estas variables se van a eliminar para el módulo de clasificación. Se indica en el repositorio de la UCI que la variable duration tiene mucha relación con la variable de salida y no debe ser empleada en modelos de predicción que busquen ser realistas. Por otra parte, el resto de las variables de esta categoría se ha creído que no están relacionadas con el problema a tratar y se van a eliminar

Atributos del contexto social y económico

- **Emp.var.rate:** Tasa de variación del empleo - indicador trimestral. → Numérica
- **Cons.price.idx:** Índice de precios al consumo - indicador mensual. → Numérica
- **Cons.conf.idx:** Índice de confianza del consumidor - indicador mensual. → Numérica
- **Euribor3m:** Tipo euribor 3 meses - indicador diario → Numérica
- **Nr.employed:** Número de empleados - indicador trimestral. → Numérica

Target

- **y:** ¿Ha suscrito el cliente un depósito a plazo fijo? Yes/No → Categórica

Visualización de los datos

En primer lugar, se va a representar la matriz de correlación de las variables numéricas. De este modo, se podrá ver que variables tienen una mayor correlación:

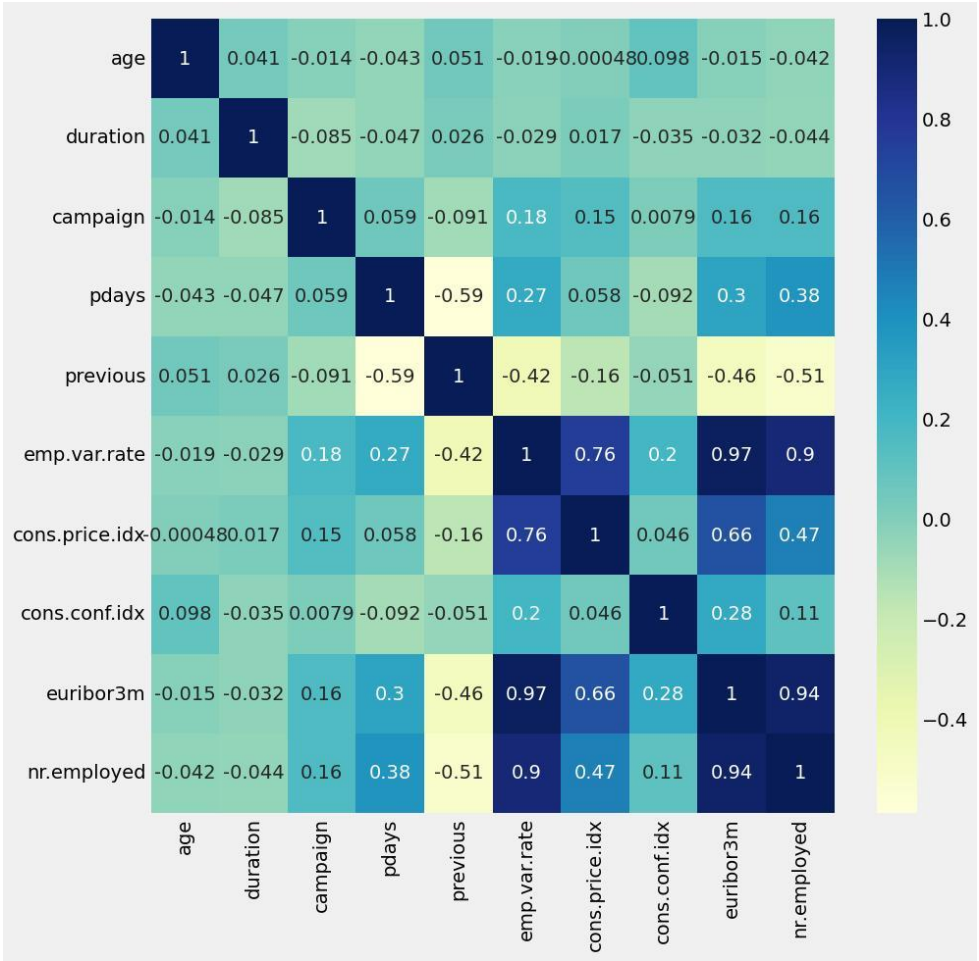


Ilustración 1 - Correlación de las variables numéricas

Se observa que existe una alta correlación entre los siguientes pares de variables:

- euribor3m y emp.var.rate → 0.97
- euribor3m y nr.employed → 0.94
- nr.employed y emp.var.rate → 0.9

Todas estas variables son relativas a atributos del contexto social y económico. Se puede eliminar emp.var.rate y nr.employed del modelo.

Por otra parte, se ha analizado la variable de salida y. A continuación, se muestra un *pie chart:*

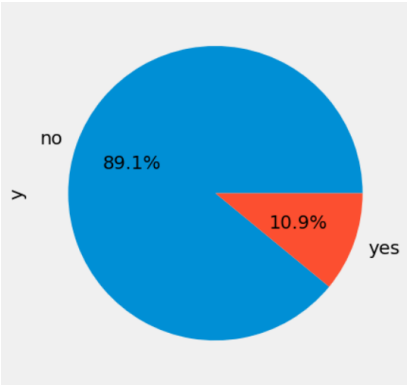


Ilustración 2 - Distribución de la variable y

Se observa claramente que el *dataset* no está balanceado ya que hay 3668 (89.1%) ejemplos en los que la variable *target* es no, es decir, que no compraron el producto, y tan solo 451 (10.9%) casos en los que el cliente contrato el deposito bancario a plazo fijo. Para el módulo de clasificación será conveniente realizar sobremuestreo con SMOTE para el entrenamiento de los distintos modelos.

También se han representado las variables numéricas por pares:

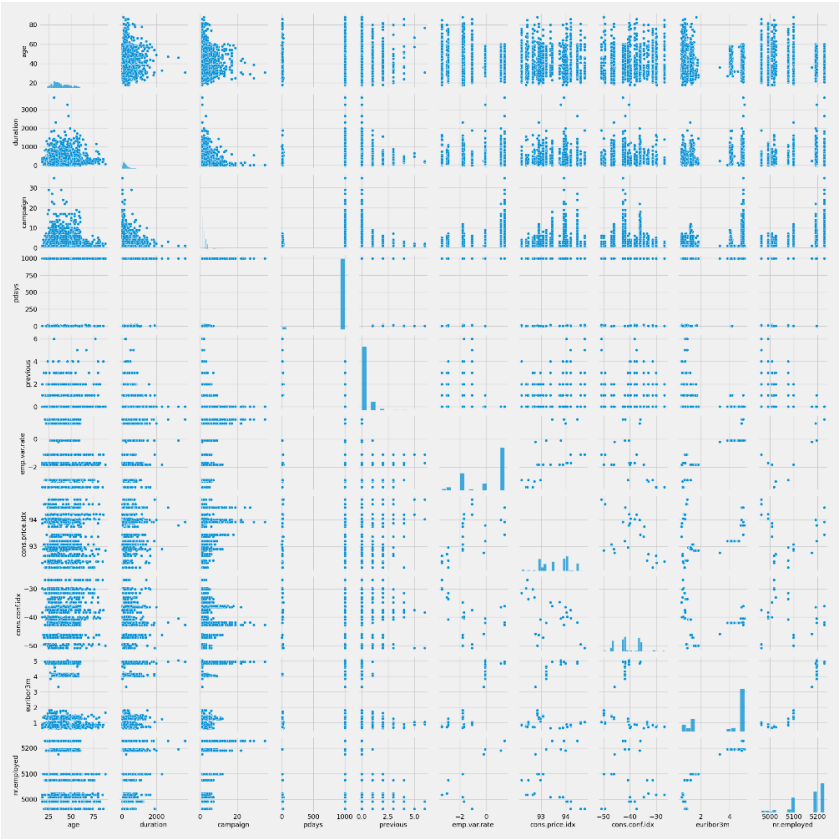


Ilustración 3 - Representación de las variables numéricas por pares

Por otra parte, se ha representado el histograma de algunas variables categóricas:

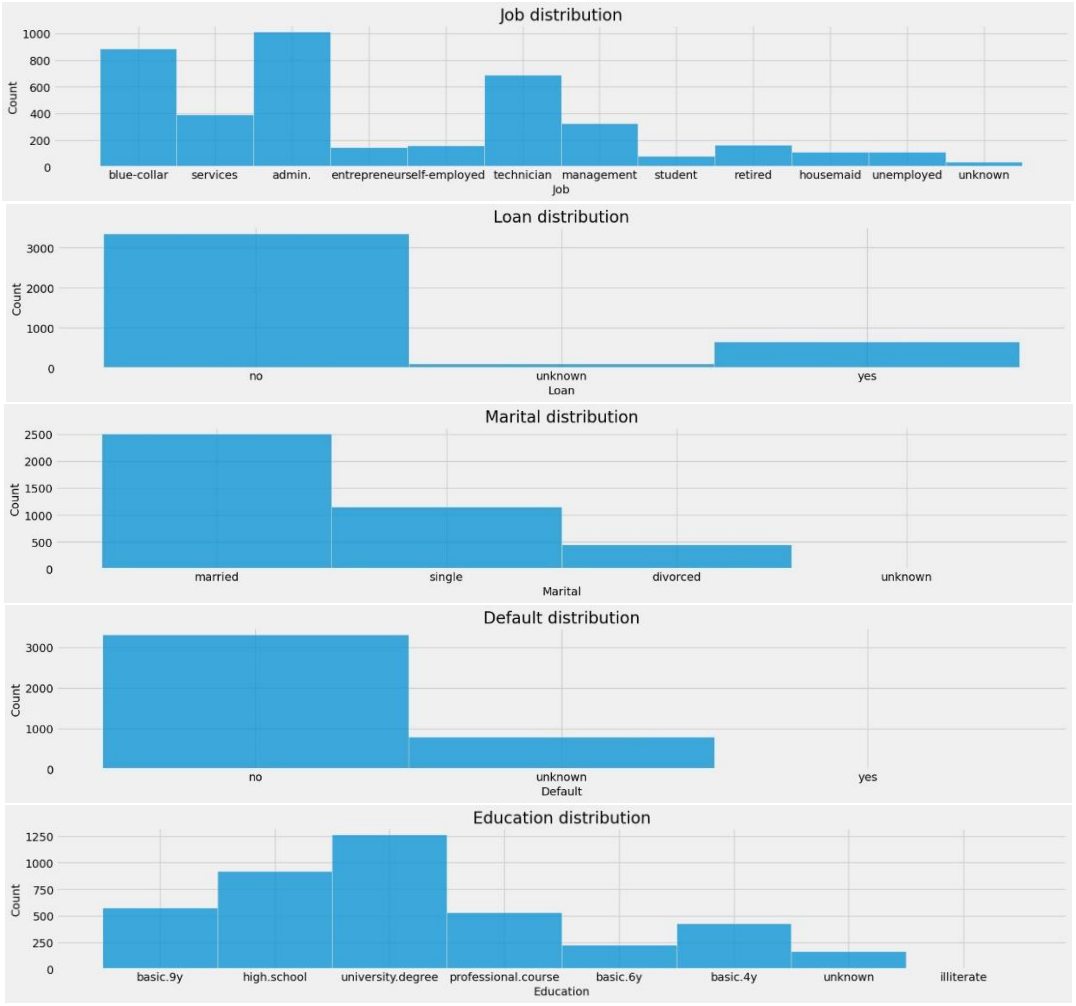


Ilustración 4 - Histogramas de la variable de salida

Visualizando los histogramas de las variables de salida, se pude ver que:

- Aproximadamente la mitad de los clientes tienen un préstamo para la vivienda, mientras que la otra mitad no.
- Los trabajos más realizados por los clientes son: administrador, manufactura (*blue-collar*) y técnico.
- La mayoría de los clientes no tienen un préstamo personal.
- Existe un mayor número de clientes con estado civil casado. Siendo el siguiente soltero
- La mayoría de los clientes no tienen créditos en estado de morosidad.
- En cuanto al nivel de educación, se observa que la categoría mayoritaria es estudios universitarios (*University Degree*), seguido por Educación secundaria (*High School*).

Conclusiones extraídas tras la inspección de los datos

Módulo 1: Clasificación

Se ha decidido eliminar las variables relativas a datos del cliente relacionados con el último contacto de la campaña actual. Tal y como indican en el repositorio, la variable "duration" afecta mucho a la variable target y, sin embargo, la duración de la llamada no se conoce antes de realizar una llamado, sino después. Por lo que como se busca realizar modelos de predicción realistas, se va a eliminar dicha variable.

Por otra parte, el resto de las variables de esta categoría también se ha creído que no están directamente relacionados con el problema a tratar, por lo que se van a eliminar para evitar que afecten a los modelos generados.

Además, como se ha visto una alta correlación entre las variables euribor3m, emp_var_rate y nr_employed, se ha decidido eliminar del modelo la variables emp_var_rate y nr_employed.

Por último, dado que el dataset está muy desbalanceado y cuenta con muy pocos ejemplos con la variable target Y igual a "yes", se deberá emplear SMOTE para sobremuestrear de cara al entrenamiento de los distintos modelos de este módulo.

En resumen:

- Eliminar variables: duration, day_of_week, month, emp_var_rate y contact
- Tratamiento de las variables categóricas.
- Normalización de los datos.
- Balanceo del dataset mediante sobremuestreo con SMOTE.

Módulo 2: Clustering

Para la sección de clustering, únicamente se emplearán aquellas variables relativas a los datos del cliente debido que son las que realmente interesan para determinar los grupos o sectores de clientes que tiene la entidad bancaria

Las variables para emplear en este módulo serán:

- Age
- Job
- Marital
- Educación
- Default
- Housing
- Loan

Módulo 1: Predicción de contratación de depósito a plazo fijo

Durante esta sección, se van a exponer los distintos algoritmos de clasificación empleados para con el objetivo de predecir si un cliente contrató o no un depósito bancario a plazo fijo. Los algoritmos empleados serán los siguientes:

- KNN
- Red neuronal Perceptrón Multicapa (MLP)
- Random Forest
- Gradient Boosting

Por otra parte, se va a realizar una comparativa entre los distintos modelos generados para determinar cual de ellos debería emplearse para realizar predicciones. Las métricas que se van a emplear en la comparativa serán *Accuracy*, F1-Score, *Recall* y *Precision*. A continuación, se describen las métricas empleadas:

- *Accuracy*: Calcula la proporción de casos predichos correctamente sobre el número total de casos. Proporciona una medida general de la corrección de un modelo
$$Accuracy = \frac{TP + TN}{TP + TN + F`P + FN}^{*}$$
- *Recall*: Mide la proporción de casos positivos predichos correctamente de todos los casos positivos reales. Se centra en la capacidad de identificar casos positivos.
$$Recall = \frac{TP}{TP + F`N}^{*}$$
- *Precision*: mide la proporción de casos positivos predichos correctamente de todos los casos predichos como positivos. Se centra en la precisión de las predicciones positivas.
$$Precision = \frac{TP}{TP + F`P}^{*}$$
- F1-*Score*: Es la media armónica de precision y recall. Proporciona una medida equilibrada del rendimiento de un modelo al tener en cuenta tanto la precisión como la recuperación. Resulta útil cuando los datos están desequilibrados.

*Donde TP es True Positive, TN es True Negative, FP es False Positive y FN es False Negative.

Tratamiento de los datos

Se ha realizado un tratamiento de los datos. En primer lugar, tal y como se ha explicado en la sección anterior, se han eliminado las variables relacionadas con datos del cliente relacionados con el último contacto de la campaña actual.

Estas variables son: Contact, Month, Day_of_week y Duration. Tal y como se ha explicado, Duration se trata de una variable que solo se conoce posteriormente a que el cliente haya contratado el producto y está altamente relacionada con la variable *target.* Por lo que no debe ser empleada en un modelo de predicción que busque ser realista. Por otra parte, las variables Month, Day_of_week y Duration se han eliminado porque se ha considerado que no están realmente relacionadas con el problema a tratar y no se quiere que afecten a los distintos modelos.

Además, debido a la alta correlación de la variable emp.var.rate t nr.employed con otras, esta también se va a eliminar.

Por otra parte, se ha convertido las variables categóricas mediante la función get_dummys de la librería pandas.

En cuanto a las variables numéricas, se han normalizado empleando MinMaxScaler() perteneciente a la librería sklearn.

Una vez se han tratado las variables categóricas y las variables numéricas, se han separado las variables de entrada de la variable target. También se ha realizado una división del conjunto de datos dividiéndolo en dos, un conjunto de entrenamiento (con el 70% de los datos) y un conjunto de test (con el 20% de los datos).

Finalmente, debido a que el conjunto de datos estaba desbalanceado se ha empleado una técnica de oversampling o sobremuestreo mediante SMOTE (de la librería imblearn) con los siguientes parámetros:

- **sampling_strategy:** 0.5
- **random_state:** 2
- **k_neighbors:** 5

De esta forma, el conjunto de entrenamiento ha quedado de la siguiente forma:

```
Before Over Sampling, count of the label '1': 320
Before Over Sampling, count of the label '0': 2563

After Over Sampling, the shape of the train_X: (3844, 36)
After Over Sampling, the shape of the train_y: (3844,)

After Over Sampling, count of the label '1': 1281
After Over Sampling, count of the label '0': 2563
```

Ilustración 5 - Antes y después del oversampling

Aunque el conjunto sigue estando algo desbalanceado, se ha conseguido mediante la generación de muestras sintéticas que no haya tanta desproporción entre el número de datos perteneciente a cada clase.

Una vez se han tratado los datos, se ha procedido a generar los distintos modelos.

Modelo 1: KNN

El primer modelo que se trata de KNN. K-nearest neighbors (KNN) es un algoritmo de aprendizaje supervisado que clasifica o predice nuevos puntos de datos basándose en la clase mayoritaria de sus k vecinos más cercanos en un espacio de características.

Para el modelo de KNN que se va a entrenar, se va a emplear la distancia euclídea.

En primer lugar, se ha buscado el número óptimo de vecinos k. Para ello, se ha comprobado para el rango de k=1 a k=40 que modelo obtenía mejores resultados de *accuracy* en el conjunto de test.

En la siguiente gráfica se pueden observar los resultados:

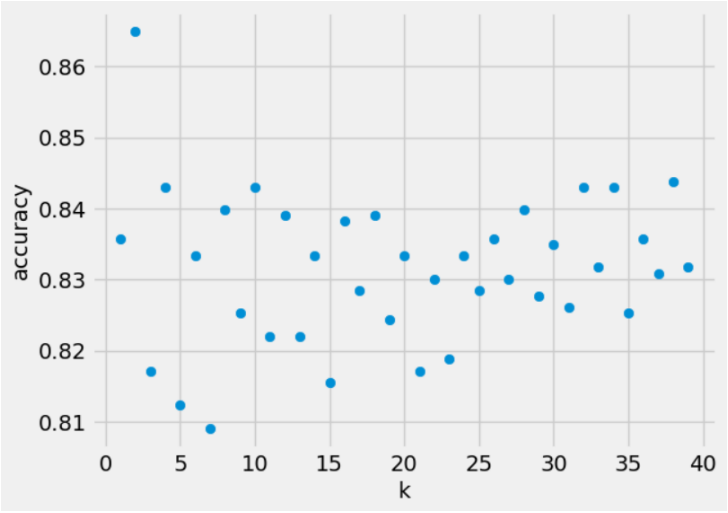


Ilustración 6 - Selección de la mejor K en KNN

Como se puede observar, los mejores resultados se obtienen para K=2.

Generando el modelo para k =2 se han obtenido los siguientes resultados en el conjunto de entrenamiento y test:

- **Conjunto de entrenamiento**

```
TRAINING
Accuracy of K-NN classifier on training set: 0.97
[[2563    0]
 [ 102 1179]]
```

Ilustración 7 - Accuracy y matriz de confusión training set - KNN

- **Conjunto de test**

```
TEST
Accuracy of K-NN classifier on test set: 0.86
[[1040   65]
 [ 102   29]]
```

Ilustración 8 - Accuracy y matriz de confusión test set - KNN

Además, se ha calculado el Recall, Precision y F1-Score para el conjunto de test. Se han obtenido los siguientes resultados:

	precision	recall	f1-score	support
0.0	0.91	0.94	0.93	1105
1.0	0.31	0.22	0.26	131

Ilustración 9 - Precision, Recall y F1-Score sobre el conjunto de test – KNN

Se observa que el modelo tiene una mayor accuracy en el conjunto de training del 97% en comparativa con la obtenida con el conjunto de test (86%). El modelo presenta una alta varianza. Aun así, el porcentaje de aciertos en test es bastante elevado.

Por otra parte, observando la precisión, recall y f1-score se observa que el modelo es capaz de predecir mejor cuando un cliente NO va a contratar un depósito a plazo fijo, que cuando un cliente SÍ lo va a contratar.

Modelo 2: Random Forest

El siguiente modelo empleado se trata de Random Forest. Random Forest es un algoritmo de aprendizaje supervisado que combina varios árboles de decisión para realizar predicciones mediante la agregación de sus resultados individuales, lo que mejora la precisión y la solidez en el manejo de conjuntos de datos complejos.

Además, una de las características de Random Forest es que ayuda a reducir la varianza utilizando el concepto de aprendizaje conjunto, en el que se construyen múltiples árboles de decisión utilizando diferentes subconjuntos de datos y características. El promedio o votación de varios árboles ayuda a suavizar la tendencia del árbol de decisión individual a sobreajustarse a los datos de entrenamiento, reduciendo así la varianza y mejorando el rendimiento de la generalización.

Para seleccionar los hiperparámetros del modelo de Random Forest se han empleado dos técnicas distintas. La primera de ellas, realizando una búsqueda de los mejores hiperparámetros empleando el out-of-bag error. La segunda, empleando Cross-Validation o validación cruzada.

Ambos enfoques tienen sus ventajas y pueden utilizarse en función de los requisitos específicos del problema. El error OOB proporciona una estimación cómoda del rendimiento del modelo sin necesidad de dividir los datos adicionalmente, mientras que el CV ofrece una evaluación más exhaustiva, pero puede requerir más recursos a nivel computacional.

Por ello, se van a emplear ambas técnicas y desarrollar un modelo final con los hiperparámetros que mejores resultados hayan tenido.

En ambos casos, los parámetros sobre los que se han realizado la búsqueda han sido los siguientes:

- **n_estimators:** 50, 100, 150, 250, 500
- **max_features:** 5, 10, 15, 20, 25, 35
- **max_depth:** None, 3, 10, 20
- **criterion:** “Gini”, “Entropy”

A continuación, se muestran los resultados empleando out-of-bag error:

	oob_accuracy	criterion	max_depth	max_features	n_estimators
124	0.926899	entropy	NaN	5	500
4	0.926119	gini	NaN	5	500
3	0.925338	gini	NaN	5	250
2	0.924298	gini	NaN	5	150
94	0.924298	gini	20.0	5	500
123	0.924037	entropy	NaN	5	250
128	0.923777	entropy	NaN	10	250
127	0.923517	entropy	NaN	10	150
122	0.922737	entropy	NaN	5	150
92	0.922737	gini	20.0	5	150

Ilustración 10 - Resultado de mejores hiperparámetros usando Out-of-bag error

```
--------------------------------------------------
Better hyperparameters found (oob-accuracy)
--------------------------------------------------
oob_accuracy    0.926899
criterion        entropy
max_depth            NaN
max_features           5
n_estimators         500
Name: 124, dtype: object : 0.9268990634755463 accuracy
```

Ilustración 11 - Mejores hiperparámetros con Out-of-bag error

Los resultados empleando CV han sido los siguientes:

	param_criterion	param_max_depth	param_max_features	param_n_estimators	mean_test_score	std_test_score	mean_train_score	std_train_score
124	entropy	None	5	500	0.919008	0.007054	0.999783	0.000153
4	gini	None	5	500	0.918488	0.006133	0.999783	0.000153
3	gini	None	5	250	0.917881	0.005753	0.999783	0.000153
2	gini	None	5	150	0.917794	0.006767	0.999783	0.000153

Ilustración 12 - Resultados de mejores hiperparámetros usando CV

```
--------------------------------------
Better hyperparameters found (cv)
--------------------------------------
{'criterion': 'entropy', 'max_depth': None, 'max_features': 5, 'n_estimators': 500} : 0.9190077075205896 accuracy
```

Ilustración 13 - Mejores hiperparámetros usando CV

Como se puede observar, se ha obtenido un poco más de accuracy empleando OOB error frente a la obtenida mediante CV. Aun así, los hiperparámetros encontrados coinciden a excepción de n_estimators.

Por ello, para el modelo final se van a emplear los obtenidos mediante OOB error:

- **criterion:** “Entropy”
- **max_depth:** None
- **max_features:** 5
- **n_estimators:** 500

Los resultados para este modelo han sido los siguientes:

- **Conjunto de entrenamiento:**

```
Accuracy in train 1s: 99.97 %
```

```
Confussion Matrix -- TRAIN
------------------

array([[2563,    0],
       [   1, 1280]], dtype=int64)
```

Ilustración 14 - Accuracy y matriz de confusión training set - RF

- **Conjunto de test:**

```
Accuracy in test 1s: 89.16 %
```

```
Confussion Matrix -- TEST
------------------

array([[1068,   37],
       [  97,   34]], dtype=int64)
```

Ilustración 15 - Accuracy y matriz de confusión test set - RF

Además, se ha calculado el Recall, Precision y F1-Score para el conjunto de test. Se han obtenido los siguientes resultados:

	precision	recall	f1-score	support
0.0	0.92	0.97	0.94	1105
1.0	0.48	0.26	0.34	131

Ilustración 16 - Precision, Recall y F1-Score sobre el conjunto de test – RF

De nuevo se observa que el modelo presenta una alta varianza ya que se ha obtenido un 99% de accuracy en training y un 89% en test. Además, el modelo también es capaz de predecir mejor cuando un cliente NO va a comprar un depósito a plazo fijo, que cuando SÍ va a hacerlo.

Por último, se ha comprobado que variables tienen más importancia. Se muestran las diez primeras:

	predictor	importance
6	euribor3m	0.184281
5	cons.conf.idx	0.134309
4	cons.price.idx	0.102434
0	age	0.097600
1	campaign	0.090821
31	housing_yes	0.033654
3	previous	0.029236
2	pdays	0.028130
28	default_unknown	0.025576
35	poutcome_success	0.022406

Ilustración 17- Importancia de las variables RF

Como se puede ver, la variable con mayor importancia es euribor3m, seguido de cons.conf.idx y cons.price.idx.

Modelo 3: GradientBoosting

El tercer modelo empleado ha sido Gradient Boosting, que es una técnica de aprendizaje automático por conjuntos que combina varios modelos de predicción débiles, normalmente árboles de decisión, para crear un modelo de predicción más potente mediante el ajuste iterativo de nuevos modelos a los residuos de los modelos anteriores, haciendo más hincapié en los ejemplos que se han predicho erróneamente en iteraciones pasadas.

Para encontrar los mejores hiperparámetros, se va a realizar una búsqueda mediante GridSearchCV.

A continuación, se muestran los parámetros sobre los que se va a realizar la búsqueda:

- **n_estimators:** 50, 100, 500, 1000
- **max_features:** “auto”, “sqrt”, “log2”
- **max_depth:** None, 1, 3, 5, 10, 20
- **subsample:** 0.5, 1
- **learning_rate:** 0.001, 0.01, 0.1

Los resultados obtenidos mediante GridSearchCV han sido:

	param_learning_rate	param_max_depth	param_max_features	param_n_estimators	param_subsample	mean_test_score	std_test_score	mean_train_score	std_train_score
395	0.1	10	sqrt	100	1	0.923517	0.003187	0.99987	0.000184
407	0.1	10	log2	1000	1	0.922737	0.002277	0.99987	0.000184
255	0.01	10	sqrt	1000	1	0.922477	0.001443	0.99987	0.000184
429	0.1	20	log2	500	1	0.922217	0.000952	0.99987	0.000184
431	0.1	20	log2	1000	1	0.921436	0.001601	0.99987	0.000184
397	0.1	10	sqrt	500	1	0.921436	0.000354	0.99987	0.000184
254	0.01	10	sqrt	1000	0.5	0.921177	0.004435	0.99935	0.000184
310	0.1	None	log2	1000	0.5	0.920918	0.005993	0.99987	0.000184
405	0.1	10	log2	500	1	0.920916	0.002395	0.99987	0.000184
399	0.1	10	sqrt	1000	1	0.920915	0.002059	0.99987	0.000184

Ilustración 18 - Resultados de mejores hiperparámetros usando GridSearchCV

Por lo que los mejores hiperparámetros son:

```
--------------------------------------------------
Better hyperparameters found (oob-accuracy)
--------------------------------------------------
{'learning_rate': 0.1, 'max_depth': 10, 'max_features': 'sqrt', 'n_estimators': 100, 'subsample': 1} : 0.9235171592656056 accuracy
```

Ilustración 19 - - Mejores hiperparámetros usando GridSearchCV

Por lo que se ha generado un modelo con esos hiperparámetros. Los resultados obtenidos para este modelo son:

- **Conjunto de entrenamiento:**

```
Accuracy in train 1s: 99.97 %
```

```
Confussion Matrix -- TRAIN
--------------------

array([[2563,    0],
       [   1, 1280]], dtype=int64)
```

Ilustración 20 Accuracy y matriz de confusión training set - Gradient Boosting

- **Conjunto de test:**

```
Accuracy in test 1s: 89.00 %
```

```
Confussion Matrix -- TEST
-------------------

array([[1067,   38],
       [  98,   33]], dtype=int64)
```

Ilustración 21 - Ilustración 16 Accuracy y matriz de confusión test set - Gradient Boosting

Además, se ha calculado el Recall, Precision y F1-Score para el conjunto de test. Se han obtenido los siguientes resultados:

	precision	recall	f1-score	support
0.0	0.92	0.97	0.94	1105
1.0	0.46	0.25	0.33	131

Ilustración 22 - Precision, Recall y F1-Score sobre el conjunto de test – Gradient Boosting

Este modelo, como los anteriores, también presenta una alta varianza. La accuracy en entrenamiento es del 99.97%, frente a un 89.00% en test. Por otra parte, acierta de forma casi total cuando un cliente NO va a contratar un depósito a plazo fijo, pero tiene un menor ratio de acierto cuando SÍ va a suscribirse al producto.

Por último, se ha comprobado que variables tienen más importancia. Se muestran las diez primeras:

	predictor	importance
6	euribor3m	0.240130
5	cons.conf.idx	0.124997
4	cons.price.idx	0.115610
1	campaign	0.093394
0	age	0.072718
2	pdays	0.038588
34	poutcome_nonexistent	0.036682
28	default_unknown	0.030386
31	housing_yes	0.027743
35	poutcome_success	0.021286

Ilustración 23 - Importancia de las variables Gradient Boosting

Como se puede ver, la variable con mayor importancia es euribor3m, seguido de cons.conf.idx y cons.price.idx.

Modelo 4: MLP

El último modelo que se va a emplear se trata de una red neuronal. Concretamente de una red neuronal Perceptrón Multicapa o *Multilayer Perceptron* (MLP) caracterizada por tener múltiples capas de neuronas interconectados y ser capaz de aprender patrones y relaciones complejas de los datos. En esta ocasión, se va a utilizar para una tarea de clasificación.

Se va a emplear MLPClassifier de la librería sklearn.

Para ajustar los hiperparámetros se ha optado por emplear RandomizedSearchCV. El espacio de búsqueda ha sido el siguiente:

- **hidden_layer_sizes:** (10), (10,10), (20,20) y (38, 10,2)
- **alpha:** np.logspace(-3, 3, 7)
- **learning_rate_init:** 0.001, 0.01, 0.1

Los resultados obtenidos han sido los siguientes:

	param_learning_rate_init	param_hidden_layer_sizes	param_alpha	mean_test_score	std_test_score	mean_train_score	std_train_score
42	0.01	(38, 10, 2)	0.001	0.846520	0.022056	0.937306	0.014442
17	0.01	(20, 20)	0.01	0.834815	0.024975	0.939128	0.012747
41	0.01	(38, 10, 2)	0.01	0.832734	0.021978	0.927161	0.010068
39	0.001	(20, 20)	0.01	0.829872	0.019840	0.932755	0.019162
8	0.01	(20, 20)	0.1	0.829612	0.020801	0.920398	0.016911
34	0.001	(20, 20)	0.001	0.828308	0.013368	0.931975	0.020367
11	0.001	(38, 10, 2)	1.0	0.818685	0.018666	0.874480	0.008631
32	0.01	(10, 10)	0.01	0.809585	0.031739	0.882934	0.002543
47	0.001	(20, 20)	1.0	0.804375	0.012254	0.836890	0.008457
24	0.01	10	0.001	0.802552	0.007651	0.863947	0.017846

Ilustración 24 - Resultados de mejores hiperparámetros usando RandomizedSearchCV

Tomando la mejor combinación de hiperparámetros, se ha generado el siguiente modelo:

```
MLPClassifier(alpha=0.001, hidden_layer_sizes=(38, 10, 2),
              learning_rate_init=0.01, max_iter=2000)
```

Ilustración 25 - Modelo MLP con mejores hiperparámetros encontrados

Por lo que se ha generado un modelo con esos hiperparámetros. Los resultados obtenidos para este modelo son:

- **Conjunto de entrenamiento:**

```
Confussion Matrix -- TRAIN
------------------

array([[2563,    0],
       [   1, 1280]], dtype=int64)
```

```
Accuracy in train 1s: 99.97 %
```

Ilustración 26 - Accuracy y matriz de confusión training set - MLP

- **Conjunto de test:**

```
Confussion Matrix -- TEST
------------------

array([[1105,    0],
       [ 131,    0]], dtype=int64)
```

```
Accuracy in test 1s: 89.40 %
```

Ilustración 27 Accuracy y matriz de confusión test set – MLP

Además, se ha calculado el Recall, Precision y F1-Score para el conjunto de test. Se han obtenido los siguientes resultados:

	precision	recall	f1-score	support
0.0	0.89	1.00	0.94	1105
1.0	0.00	0.00	0.00	131

Ilustración 28 - Precision, Recall y F1-Score sobre el conjunto de test – MLP

Por último, se ha representado la curva de pérdidas donde se puede ver como la función de coste va decreciendo en función de las iteraciones:

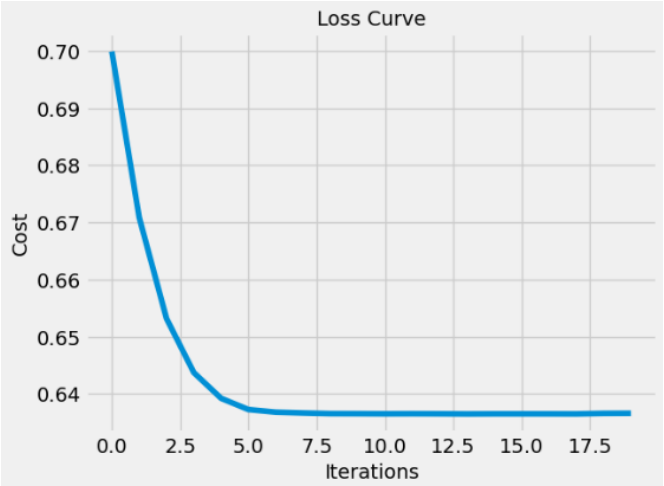


Ilustración 29 - Curva de pérdidas

Comparativa de los modelos

Finalmente, se va a realizar una comparativa de los distintos modelos.

A continuación, se muestra una tabla resumen de la Accuracy obtenida en entrenamiento y test para los distintos modelos generados:

Modelo	*Accuracy Training*	*Accuracy Test*
KNN (k=2)	0.97	0.86
RF	0.99	0.89
Gradient Boosting	0.99	0.89
MLP	0.99	0.89

Tabla 1 - Resumen Accuracy para los distintos modelos

Se observa que los modelos tienen un rendimiento similar en cuanto a *Accuracy*.

A continuación, se muestra una tabla resumen de la *Precision*, *Recall* y F1-Score obtenidas en cada uno de los modelos:

	Y = 0			Y = 1		
	Precision	*Recall*	F1- Score	*Precision*	*Recall*	F1- Score
KNN (k=2)	0.91	0.94	0.93	0.31	0.22	0.26
RF	0.92	0.97	0.94	0.48	0.26	0.38
Gradient Boosting	0.92	0.97	0.94	0.46	0.25	0.33
MLP	0.89	1	0.94	0	0	0

Tabla 2 - Resumen Precision, Recall y F1-Score para los distintos modelos

Observando la Precision y Recall se ve que todos los modelos fallan al predecir cuando un cliente SÍ va a contratar un depósito a plazo fijo, pero aciertan cuando un cliente NO va a contratar un depósito a plazo fijo. De hecho, el modelo MLP falla en test todos los ejemplos de clientes que No van a contratar un depósito a plazo fijo.

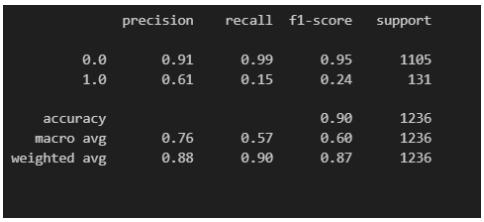
Con los resultados obtenidos, se cree que los modelos pueden ser empleados para reducir el esfuerzo que realiza el banco mediante acciones de marketing a aquellos clientes que NO van a contratar un crédito. De esta forma, aunque no pueden predecir que clientes SÍ lo van a contratar de forma fiable, el modelo se puede usar para descartar acciones comerciales sobre una gran cantidad de clientes que de forma casi segura NO van a contratar el producto. Estos modelos pueden permitir un ahorro en costes a la Entidad.

Se cree que con un dataset con más datos de clientes que sí fuesen a contratar un crédito bancario a plazo fijo se podrían mejorar los resultados.

Por último, se ha realizado una prueba con el conjunto de entrenamiento combinando todos los modelos. Esto sería similar a generar un modelo de votación en el cual cada modelo vota el output que cree correcto.

De esta forma, se ha conseguido mejorar la accuracy en test a 90%. La mejora es mínima frente a usar cualquier modelo de forma individual. Esto implica que la mayoría de los modelos para un

mismo ejemplo están prediciendo lo mismo. A continuación, se muestran los resultados obtenidos:



	precision	recall	f1-score	support
0.0	0.91	0.99	0.95	1105
1.0	0.61	0.15	0.24	131
accuracy			0.90	1236
macro avg	0.76	0.57	0.60	1236
weighted avg	0.88	0.90	0.87	1236

Ilustración 30 - Resultados combinando todos los modelos

Módulo 2: Segmentación de clientes de la Entidad

Tal y como se ha indicado en el apartado 2. Análisis e inspección de los datos para esta sección únicamente se van a emplear aquellas variables relativas a los datos de los clientes. De este modo, se podrá obtener una segmentación de los clientes mediante algoritmos de clustering que permita identificar aquellos grupos similares. De este modo, la Entidad podrá poner el foco en estos grupos mediante diferentes campañas de marketing dirigidas hacia ellos. Estas variables son:

Datos del cliente:

- **Age:** Edad del cliente → Numérica
- **Job:** Tipo de trabajo → Categórica
- **Marital:** Estado civil → Categórica
- **Educación:** Nivel de estudios → Categórica
- **Default:** ¿Tiene créditos en estado de morosidad? Yes/No → Categórica
- **Housing:** ¿Tiene préstamo para vivienda? Yes/No → Categórica
- **Loan:** ¿Tiene préstamo personal? Yes/No → Categórica

Los algoritmos de clustering que se van a emplear durante esta sección son:

- K-Means
- Hierarchical Clustering (HC)

Dado que ambos son algoritmos de aprendizaje no supervisado, no se va a realizar una comparativa directa de ellos.

Para esta sección, únicamente se han tratado las variables categóricas mediante OneHotEncoding y se ha normalizado las variables numéricas. Al emplear técnicas de aprendizaje no superivisado, no será necesario dividir el conjunto de datos.

Modelo 1: K-Means

El primer modelo empleado es K-Means, que se trata de un algoritmo de aprendizaje no supervisado que busca separar el dataset en K clusters o grupos asignando los datos de forma iterativa al centroide más cercano hasta la convergencia.

Para determinar la K óptima se va a emplear el método del codo. Este método permite determinar un número óptimo de clusters K. A continuación, se muestra la gráfica obtenida en la Ilustración 31.

Como no existe un punto claro para determinar la posición del codo. Se recomienda tomar una K entre 5 y 7. Para el análisis a realizar se ha decidido tomar k = 5, ya que se quiere priorizar el menor número posible de grupos. De esta forma, la entidad tendrá que trabajar en un menor número de campañas de marketing que abarquen más clientes.

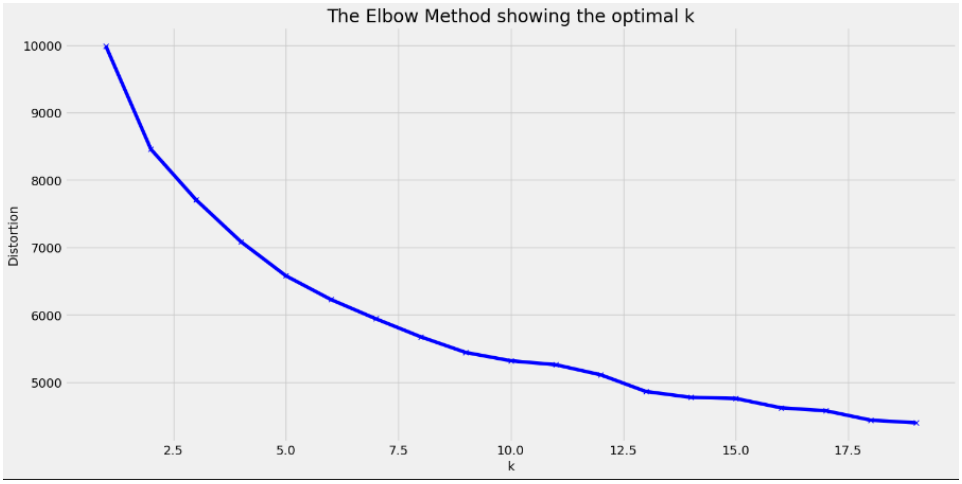


Ilustración 31 - Elbow Method K-Means

De este modo, la Entidad podrá realizar 5 campañas de marketing diferentes dirigidas a un amplio número de usuarios con características similares.

Modelo 2: HC

El segundo modelo empleado ha sido clustering jerárquico (HC). Este algoritmo también es no supervisado y busca crear una jerarquía de clusterings aglomerando o dividiendo iterativamente. Para ello, se basa en la similitud (o diferencia) existente entre los puntos.

El resultado es un dendrograma que representa la estructura de clusters de los datos. Además, no requiere un número predeterminado de clusters, cosa que en K-Means si es necesario.

A continuación, se muestra el dendrograma mostrando la distancia entre los distintos ejemplos pertenecientes al conjunto de datos:

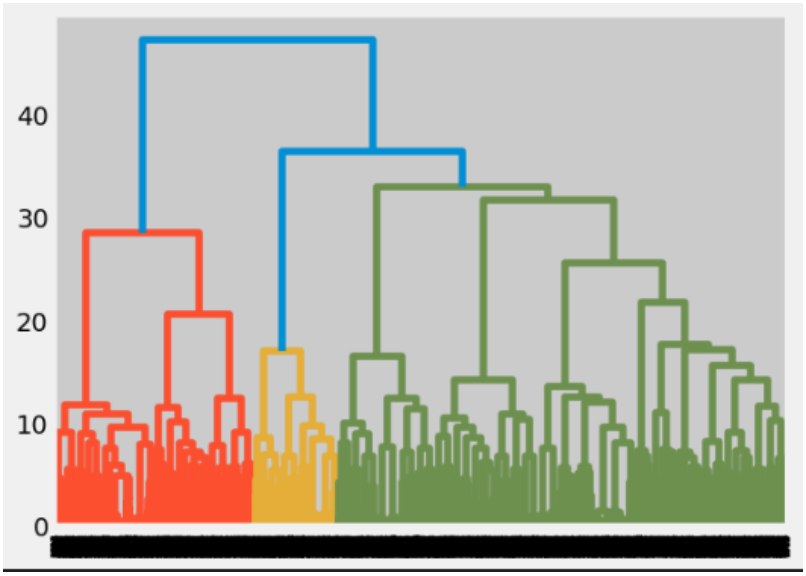


Ilustración 32 - Dendrograma de los datos

El punto óptimo de corte sería aproximadamente en 35, existiendo 2 clusters que agrupan la materia de los datos.

Además, se ha realizado el dendrograma de la construcción inversa. En este se muestra la disimilitud entre las variables.

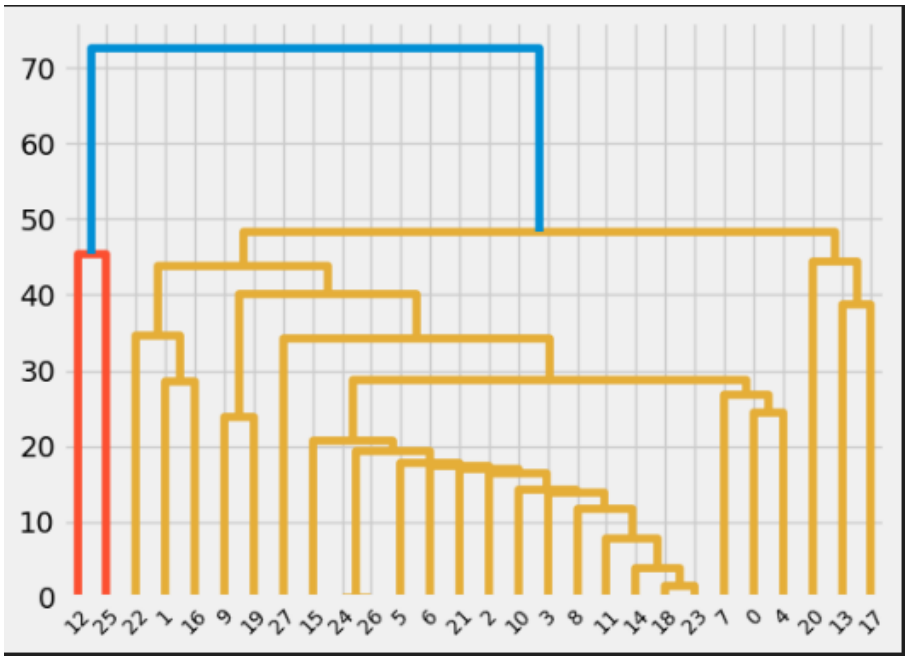


Ilustración 33 - Dendograma de la construcción inversa

De nuevo, se obtienen dos clusters mayoritarios que agrupan la mayoría de las variables.

Se recomienda tomar los resultados de HC, ya que esto permitiría a la Entidad realizar únicamente dos campañas de marketing diferentes agrupando a los clientes en dos grupos. De esta forma, podría llegar a todos los clientes de forma sencilla y ahorrar en costes.

Para ello, solo necesita emplear el modelo generado para saber a que grupo pertenece el cliente y enviarle la campaña de marketing de ese grupo.

Conclusiones y reflexiones finales

Como se ha visto en el primer módulo, que trataba de generar modelos para predecir si un cliente contrataría o no un depósito a plazo fijo, los resultados para los distintos modelos empleados han sido muy similares.

Se ha observado que todos los modelos predicen muy bien cuando un cliente NO va a contratar un depósito a plazo fijo. Sin embargo, los modelos no aciertan con tanta frecuencia para predecir cuando un cliente SÍ va a contratar un depósito a plazo fijo.

La recomendación sería emplear los modelos para reducir el esfuerzo de realizar acciones comerciales sobre aquellos clientes que el modelo predice que NO van a contratar un depósito a plazo fijo. De esta manera, como estos se trata de una mayoría, pueden descartar una gran cantidad de clientes y ahorrar en costes.

Por otra parte, se ha visto que el número de grupos por los que se puede segmentar a los clientes en la entidad no está claro.

Con el modelo de clustering jerárquico (HC) se han obtenido dos tipos de clientes. Mientras que con el modelo de K-Means se podrían haber diferenciado entre 5 y 7 grupos de clientes.

La recomendación sería tomar los dos grupos hallados mediante clustering jerárquico y realizar campañas similares de marketing a aquellos clientes que se encuentren en un mismo grupo. Se ha elegido esta opción porque al agrupar únicamente por dos perfiles a los clientes, se podrá diseñar dos campañas de marketing distintas enfocadas en cada uno de los grupos, mientras que si se toman más grupos el esfuerzo requerido de marketing por la Entidad será mucho mayor.