Estudio predictivo de contrataciones vía Modelos Econométricos de Elección Binaria y Machine Learning

por

Jonathan Llamas Crespillo

Tesis presentada en conformidad con los requisitos del Máster en Economía, Finanzas y Computación.

Universidad de Huelva & Universidad Internacional de Andalucía

uhu.es

Universidad Internacional de Andalucía

Julio de 2023

Estudio predictivo de contrataciones vía Modelos Econométricos de Elección Binaria y Machine Learning

Jonathan Llamas Crespillo
Universidad de Málaga y Universidad Internacional de Andalucía

Supervisado por:

Nicola Rubino
Universidad de Huelva y Universidad de Barcelona

2023

Abstract

Information technology and big data processing are tools that can be applied to almost any field or industry. In this master's thesis, machine learning techniques, econometric models, and big data processing techniques will be used in the field of labour studies. The tool which I will be using to execute these techniques and models is Python. The goal will be firstly, to process the dataset and to validate and exploit it through statistical techniques and econometric models. Secondly, we are going to employ some classification techniques through Machine Learning. Finally, I will show my conclusions on the exercise, which are oriented to have utility for professional and research goals. The aim of this study is ultimately to analyse job seekers features to compare them to each other and determine if they are ideally employable, or not, using Machine Learning and Econometrics.

JEL classification: C25, C53, J24.

Keywords: Algorithm, Prediction, Big Data, Job, Hiring, Machine Learning, Econometrics, Collective Intelligence.

Resumen

Las tecnologías de la información y el procesamiento de datos son herramientas que pueden aplicarse a casi cualquier campo o sector. En este Trabajo de Fin de Máster se van a utilizar en el campo del estudio laboral, técnicas de aprendizaje automático y modelos econométricos. Con ello, la herramienta que se ha empleado para ejecutar dichas técnicas, modelos y tratamientos es Python. Así pues, el objetivo en primer lugar será tratar el conjunto de datos y estudiarlos estadística y econométricamente. En segundo lugar, se validará y explicarán las predicciones vía Machine Learning. Finalmente se realizarán una serie de conclusiones del trabajo realizado, las cuales pretenden tener una utilidad en el ámbito profesional e investigador. En definitiva, con este estudio se analizarán las características de personas en búsqueda de empleo para relacionarlas entre sí y decidir si son potencialmente empleables combinando Machine Learning y Econometría.

Clasificación JEL: C25, C53, J24.

Palabras clave: Algoritmo, Predicción, Big Data, Empleo, Contratación, Aprendizaje Automático, Econometría, Inteligencia Colectiva.

Índice General

Índice de Tablas

Índice de Figuras

1. Introducción

Para comenzar el presente Trabajo de Fin de Máster (en adelante, TFM) debemos tener asimilados ciertos conceptos. Comprender estos conceptos y acepciones servirá para entender mejor el estudio y, por ende, será más sencillo extraer información y conocimiento útil.

Asimismo, en este primer capítulo del TFM se exponen además de los conceptos básicos, el objetivo del estudio, las metodologías y las herramientas utilizadas. En resumen, lo que se pretende con esta primera toma de contacto es tener una imagen general y organizada del TFM.

1.1 Conceptos básicos

En este primer apartado se van a definir conceptos con los que se ha construido el TFM. Se explicarán primero los de carácter laboral, luego los relacionados con matemáticas y economía y, por último, los que pertenecen a una rama más tecnológica.

- Mercado laboral. Espacio donde intervienen la oferta y la demanda de trabajo. La oferta de trabajo está formada por el conjunto de individuos en búsqueda de empleo y la demanda de trabajo por el conjunto de empresas o empleadores que están buscando contratar trabajadores.

- Estadística. Disciplina científica que se ocupa de recopilar, ordenar y analizar un conjunto de datos con el fin de conseguir información y predicciones sobre fenómenos observados.

- Econometría. Rama de la economía que, a través del uso de métodos matemáticos, estadísticos y de programación lineal, entre otros, tiene por objeto la interpretación y la elaboración de predicciones sobre los diferentes sistemas económicos o para estimar las relaciones económicas. Es decir, la finalidad de la econometría es explicar una o más variables en función de otras.

- Big Data. Datos cuya escala, diversidad y complejidad requieren de nuevas arquitecturas, técnicas, algoritmos y modelos de análisis para extraer valor y conocimiento oculto. En esta disciplina la ciencia de datos es clave en el estudio de cómo manipular y tratar los propios datos, para conseguir en primer lugar información, y luego, conocimiento.

- Inteligencia Colectiva. Forma de inteligencia que surge a partir de la colaboración de individuos con relación a una temática específica. En otras palabras, consiste en una inteligencia que recoge una serie de decisiones individuales para después, recopilarlas y

extraer una decisión conjunta. Esta decisión conjunta podrá ser de forma teórica, igual o superior que la mejor decisión individual que haya dentro del propio conjunto.

- Machine Learning. Es una forma de la Inteligencia Artificial que se basa en un aprendizaje automático e iterativo. Esto permite a un sistema aprender de los datos en lugar de aprender mediante la programación explícita.

- Inteligencia Artificial. La IA tiene como principal meta imitar la realización de tareas que pueden ser realizadas por seres humanos, mediante la experiencia, la adaptación y el aprendizaje.

Con esta breve recopilación de las áreas en los que se ha basado el estudio, se han repasado los conceptos más importantes a tener en cuenta. No obstante, a lo largo del estudio se podrán dar términos derivados de estos, los cuales, en unas ocasiones se explicarán con detenimiento y en otras, se darán por conocidos. La diferencia estará si el término es complejo o básico.

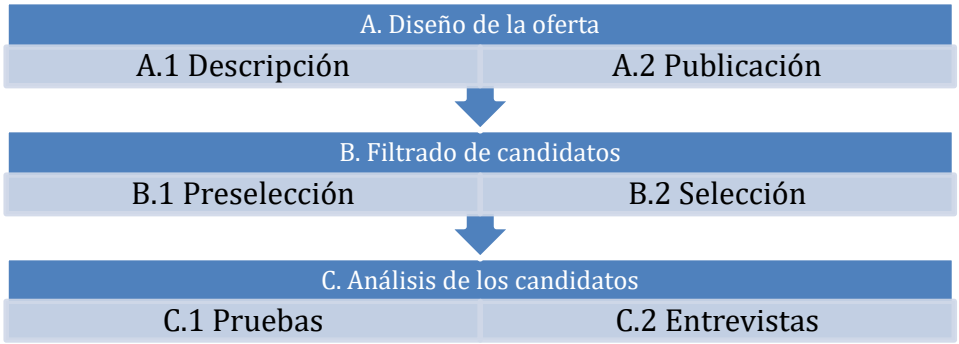
1.2 Contexto y visión empresarial

A continuación, se va a realizar un ejercicio de contexto para argumentar la elección del sector laboral como objeto de estudio.

Con el estudio de este sector como eje principal de datos, se pone sobre la mesa un precedente más para extrapolar un estudio de estas características al sector profesional. En este escenario, en relación con la aplicación de estas tecnologías y procesos caben destacar las iniciativas y análisis de empresas como Oracle, CEF, Business Insider o Adecco.

En primer lugar, para ponernos en contexto, un proceso de selección por parte del personal o departamento de RRHH tiene las siguientes etapas:

Figura 1.1. Proceso de reclutamiento en el sector de RRHH



Fuente: Adecco y CEF

En estas etapas, donde se va a centrar el estudio es en la de preselección. Esta fase conlleva a los departamentos de RRHH una ingente cantidad de tiempo para procesar todas las solicitudes, pero, además, también es una cantidad de tiempo importante la que el candidato invierte. Por ello, la meta es conseguir cierta eficiencia para ambas partes.

En este enfoque, uno de los ejemplos de digitalización y automatización de procesos selectivos es el ATS (del inglés, 'Applicant Tracking System') que posee Oracle. Este sistema lo que hace básicamente es realizar un seguimiento de los candidatos durante todo el proceso de selección y contratación. Además, incorpora una IA que filtra candidatos según sea la oferta de trabajo. Luego, esta herramienta es un complemento más que los reclutadores tienen para ser más eficientes y eficaces. Ahí es donde entra el presente estudio, en la eficacia a la hora de contratar candidatos no sólo usando machine learning, sino también técnicas de validación econométricas e inteligencia colectiva.

Sin embargo, según ManpowerGroup, citando su empresa textualmente vía Business Insider España: "La utilización de la tecnología tendrá que ser en colaboración con la persona, porque tiene que haber una persona que interprete ese dato, que sepa de dónde sale, que entienda porque…". Bajo esta premisa, en el esquema anterior la IA sólo podría intervenir de forma prácticamente completa en las fases A.1, A.2, B.1 y C.1. Las otras fases, que son las de selección y entrevistas, es aún incierto si la IA y sus tecnologías adyacentes van a progresar lo suficiente para sustituir la labor del ser humano.

En consecuencia, donde va a hacer hincapié el estudio, es en la fase B.1, la de preselección.

1.3 Objetivo del estudio

Ya expuestas las principales disciplinas y el contexto, seguidamente se va a tratar de coger las piezas del puzle y encajarlas.

Por un lado, el objetico inicial de este estudio es aplicar a un conjunto de datos de forma combinada, técnicas de gestión de datos a gran escala (Big Data), análisis estadístico, modelos econométricos y aprendizaje automático (o Machine Learning).

Por otro lado, el objetivo final será predecir las probabilidades de contratación dados una serie de perfiles en un conjunto de datos. Estas predicciones deberán determinar si un candidato es el ideal, o no, para un puesto de trabajo. En nuestro caso estas predicciones se considerarán fruto de la Inteligencia Colectiva, ya que los perfiles que se manejan son de tan solo 8 tipos y las instancias superan las 800. Por tanto, se da por hecho que las decisiones de contratación de

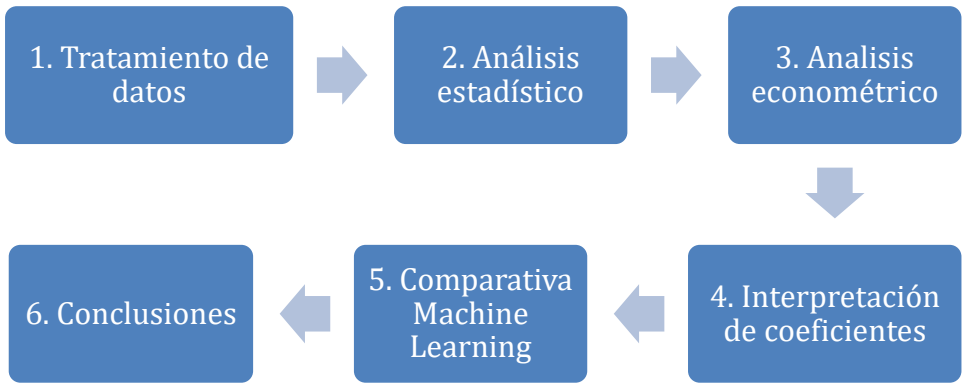
perfiles parecidos por distintas personas son un caso de Inteligencia Colectiva indirecta. Con el término 'indirecta' nos referimos a que las decisiones individuales no han sido realizadas inicialmente para un fin colectivo, sino que son decisiones independientes entre sí.

En definitiva, lo que se pretende realizar con el estudio es sentar un precedente sólido en el que se analice de forma combinada un conjunto de datos con técnicas de Machine Learning y Econometría, demostrando que esta combinación, más que interesante, es necesaria.

1.4 Metodología y herramientas

En este apartado se va a comentar la metodología del estudio y los pasos que se van a llevar cabo. Además, también se van a explicar qué herramientas se han utilizado para lograr los resultados del estudio.

Figura 1.2. Esquema del proceso en el tratamiento y análisis del TFM



Fuente: Elaboración propia

En esta hoja de ruta, en primer lugar, el tratamiento de los datos se realiza vía proceso KDD (del inglés, Knowledge Discovery in Databases). Este proceso se caracterizará por simplificar y dejar como resultado, un conjunto de datos de los que se pueda extraer información. En este proceso las herramientas utilizadas son Excel y Python. Excel se ha utilizado para realizar un tratamiento inicial de prueba de forma previa, para luego, realizar toda la gestión y manipulación de datos con Python y la librería Pandas.

En segundo lugar, el análisis estadístico se llevará a cabo también vía Python. En este caso, además de la librería Pandas, se usarán también las librerías NumPy y MatPlotLib. Con esto, se analizarán la distribución y frecuencia de los datos a nivel numérico y gráfico.

En tercer lugar, además del análisis de heterocedasticidad vía test de White, el modelo econométrico seleccionado para el análisis de las variables será el modelo de regresión logística, más conocido como 'logit'. Este modelo es ideal para el estudio de nuestras variables, ya que el conjunto de datos estará en formato binario. En lo que respecta a las herramientas utilizadas, de forma principal se ha empleado Python, aunque para contrastar ciertos resultados, se ha utilizado también STATA.

En cuarto lugar, el cálculo y la interpretación de los coeficientes nos confirmará si el modelo tal y como se va a plantear en la fase de Machine Learning, valdrá la pena o no someterlo al análisis de los algoritmos. Los coeficientes que se van a emplear son AUC-ROC, Kappa de Cohen, Correlación de Pearson, VIF Invertido, Recall y F1, todo con la herramienta Python vía librería Scikit-Learn (también conocido como 'sklearn').

En quinto lugar, se compararán los resultados obtenidos del conjunto de datos empleando los algoritmos kNN (del inglés, k-Nearest Neighbors), además de árboles de decisión. Estos algoritmos se han elegido para contrastar resultados entre una técnica simple catalogada comúnmente en el grupo de las técnicas 'perezosas', con otra más compleja que se basa en hojas, ramas y nodos. Luego, habiendo estudiado la naturaleza de los datos y su carácter dicotómico, estos algoritmos eran de los más interesantes entre las opciones barajadas. En este caso, de nuevo, la herramienta elegida ha sido Python y la ya mencionada librería Scikit-Learn.

Finalmente, las conclusiones consistirán en una serie de conocimientos extraídos del estudio. Por un lado, se comentarán de forma individual los resultados de cada fase, y por otro, los resultados obtenidos conjuntamente en el TFM.

2. Tratamiento de datos

2.1 Características y formato

Los datos que se han utilizado en el estudio se caracterizan inicialmente por ser de carácter laboral, y por tener un formato CSV. Estos datos no provienen de un solo conjunto de datos (en adelante, dataset), sino que son 2 datasets distintos que hemos fusionado durante el proceso.

En esta fusión, los datasets se han depurado para que, a final de cuentas, las variables coincidieran y pudieran ser primero concatenados, y luego distribuidos aleatoriamente para evitar sesgos por datasets.

El origen de los datasets es la plataforma Kaggle, la cual se dedica a poner a disposición de los usuarios una serie de problemas para solucionar con temáticas como la ciencia de datos, el análisis predictivo y el machine learning.

Por un lado, aunque el primer dataset se denomina originalmente 'Job Placement Dataset', lo renombraremos a 'fichero1', ya que así es como nos vamos a referir a él en el resto del TFM. Este dataset se caracteriza por poseer una serie de variables y datos que teóricamente nos va a ayudar a predecir si van a ser contratados los perfiles, o no. Asimismo, este dataset tiene 13 columnas o variables y 215 instancias.

Por otro lado, el segundo dataset se denomina originalmente 'Recruitment data', y lo renombraremos como 'fichero2'. Nuevamente, este dataset nos va a ayudar a predecir si van a ser contratados los perfiles, o no. Además, en este caso el dataset tiene 11 columnas o variables y 614 instancias.

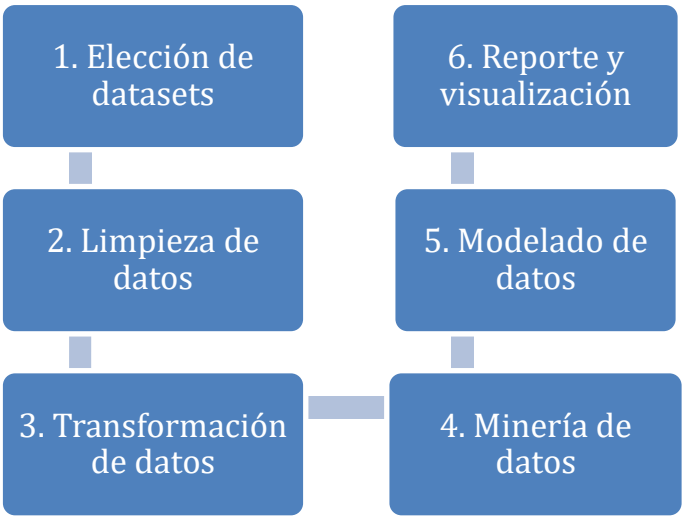
Como se puede observar, el número de variables (exógenas) no coincide, por lo que se además de estudiar cuales podremos o no poner en común, debemos para concatenarlos tener en cuenta el número y orden de las columnas. Para esto, se opta por quedarnos con las variables que, siendo comunes entre ambos datasets, puedan transformarse de forma binaria, ya que el resultado (variable endógena) también lo es (contratado, o no contratado).

En definitiva, lo que se busca es ampliar el número de instancias, eliminar el sesgo que pudieran tener individualmente, y transformar ambos datasets en uno sólo que posea los datos transformados de forma binaria.

2.2 Proceso KDD

El proceso KDD consiste en una serie de pasos en los que conjuntos de datos son sometidos a muestreos, limpiezas, transformaciones y modelados.

Figura 2.1. Esquema del proceso KDD



Fuente: UAEH, Escuela Superior Huejutla

Este proceso se caracteriza por ser iterativo e interactivo. Es decir, se realizan pasos que son necesarios entre sí. Estos son el muestreo y selección de los datos, su preprocesamiento, exploración y transformación, reconocimiento de patrones y finalmente, su evaluación e interpretación. Por tanto, cada fase del KDD es necesaria para poder llegar a un resultado que nos pueda ser útil. Sin embargo, cabe destacar que según la naturaleza de los datos y lo que se pretenda hacer con ellos, el proceso puede verse modificado, por lo que no es un proceso rígido, sino que debe adaptarse a nuestras necesidades.

2.2.1 Fuentes de datos

Durante la fase de búsqueda de datos se han explorado varias fuentes donde poder consultarlos y compararlos entre sí. Algunas de esas fuentes son generalistas, como por ejemplo el 'Google Dataset Search', el cual es un buscador indexado especializado de datasets que se encuentran alojados en internet. Otros que funcionan como repositorios, también han sido consultados. Ejemplos son la 'Machine Learning Repository' de la UCI (del inglés, University of California, Irvine), además del repositorio abierto de Kaggle.

Las características que se buscan en el conjunto de datos objetivo es que sean de carácter laboral, binarios y que al menos tengan un mínimo de 4 variables y 500 instancias. El desempeño que tengamos con esta colección de datos tendrá como meta final, su implantación a nivel profesional o investigador de una manera más amplia y masiva.

En este buceo de datos y búsquedas, utilizando las palabras clave 'hiring', 'jobs' o 'labor market', la mayoría de datasets y problemáticas estaban orientadas al caso particular de un proyecto o problema concreto. Además, los datasets binarios o que pudieran ser transformados de forma binaria son escasos.

Por otra parte, en lo que respecta a una búsqueda de datasets de trabajos o proyectos en los que se encuentren los términos 'machine learning' junto con 'econometrics', son prácticamente inexistentes. Sólo en el buscador del Google Dataset Search se muestran 13 resultados y si añades a estos términos anteriores 'hiring' o 'job', saldrá 1 solo resultado, el cual data del 13 de enero de 2023.

Ante esto, con los pocos datos disponibles, se decide trabajar con 2 datasets, ambos provenientes de Kaggle. Se escogen 2 y no 1, porque el número de instancias es bajo y lo que se pretende es evitar el sesgo individual que pueda tener un dataset. De esta forma, el conjunto de datos será más heterogéneo.

Luego, esta fase va a dar como resultado, un dataset totalmente nuevo fruto de los 2 que se van a emplear y con ello, se va a contribuir a ser uno de los escasos estudios y datasets en la lista del Google Data Search con las palabras clave: 'machine learning', 'econometrics', 'hiring' y 'job'.

En definitiva, este TFM tiene el ambicioso objetivo de ofrecer una forma de estudio y análisis poco convencional combinando aprendizaje automático y econometría.

2.2.2 Preprocesamiento

Antes de iniciar la fase de preprocesamiento, se van a mostrar los datasets elegidos. En primer lugar, tenemos el dataset 'Job Placement Dataset', renombrado como 'fichero1'.

Tabla 2.1. Job Placement Dataset – fichero1

	gender	ssc_percenta	ssc_board	hsc_percenta	hsc_board	hsc_subject	degree_perc	undergrad_d	work_experi	emp_test_p	specialisatio	mba_percen	status
2	M	67	Others	91	Others	Commerce	58	Sci&Tech	No	55	Mkt&HR	58.8	Placed
3	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech	Yes	86.5	Mkt&Fin	66.28	Placed
4	M	65	Central	68	Central	Arts	64	Comm&Mgn	No	75	Mkt&Fin	57.8	Placed
5	M	56	Central	52	Central	Science	52	Sci&Tech	No	66	Mkt&HR	59.43	Not Placed
6	M	85.8	Central	73.6	Central	Commerce	73.3	Comm&Mgn	No	96.8	Mkt&Fin	55.5	Placed
7	M	55	Others	49.8	Others	Science	67.25	Sci&Tech	Yes	55	Mkt&Fin	51.58	Not Placed
8	F	46	Others	49.2	Others	Commerce	79	Comm&Mgn	No	74.28	Mkt&Fin	53.29	Not Placed
9	M	82	Central	64	Central	Science	66	Sci&Tech	Yes	67	Mkt&Fin	62.14	Placed
10	M	73	Central	79	Central	Commerce	72	Comm&Mgn	No	91.34	Mkt&Fin	61.29	Placed
11	M	58	Central	70	Central	Commerce	61	Comm&Mgn	No	54	Mkt&Fin	52.21	Not Placed
12	M	58	Central	61	Central	Commerce	60	Comm&Mgn	Yes	62	Mkt&HR	60.85	Placed

Fuente: Kaggle

En este dataset se define de izquierda a derecha, las siguientes variables: género; nota de un primer examen preuniversitario; lugar de dicho examen; nota de un segundo examen preuniversitario; de nuevo lugar del examen; rama del curso preuniversitario; nota media obtenida en la universidad; rama de la carrera universitaria; si poseen experiencia laboral o no; nota de una prueba de aptitud; especialización del MBA; nota del MBA y finalmente, si han sido contratados, o no.

En segundo lugar, se muestra el dataset 'Recruitment data', renombrado como 'fichero2'.

Tabla 2.2. Recruitment data – fichero2

	Serial_no	Gender	Python_exp	Experience_	Education	Internship	Score	Salary * 10E4	Offer_Histor	Location	Recruitment_Status
2	1	Male	Yes	0	Graduate	No	5139	0	1	Urban	Y
3	2	Male	No	1	Graduate	No	4583	128	1	Rural	N
4	3	Male	No	0	Graduate	Yes	3000	66	1	Urban	Y
5	4	Male	No	0	Not Graduat	No	2583	120	1	Urban	Y
6	5	Male	Yes	0	Graduate	No	6000	141	1	Urban	Y
7	6	Male	No	2	Graduate	Yes	5417	267	1	Urban	Y
8	7	Male	No	0	Not Graduat	No	2333	95	1	Urban	Y
9	8	Male	No	3	Graduate	No	3036	158	0	Semiurban	N
10	9	Male	No	2	Graduate	No	4006	168	1	Urban	Y
11	10	Male	No	1	Graduate	No	12841	349	1	Semiurban	N
12	11	Male	No	2	Graduate	No	3200	70	1	Urban	Y

Fuente: Kaggle

En este segundo dataset, se define de izquierda a derecha, las siguientes variables: cardinalidad de las instancias, género; si tienen experiencia con Python; si tienen experiencia laboral; si tienen título universitario; si han realizado prácticas; una nota general del candidato; sueldo; ofertas de trabajo recibidas; tipo de localización demográfica y finalmente, si han sido contratados, o no.

Como se puede observar, ambos datasets poseen variables en común. En este sentido, antes de armonizar ambos datasets comprobamos con la librería Pandas de Python si estos poseen valores nulos (más conocidos como NaN, del inglés, 'Not a Number'), y si sus valores son consistentes, es decir, que no haya errores tipográficos ni de variables con entradas erróneas.

Después de subsanar los pocos NaN y errores vía reemplazo de valores medios, los cuales apenas sumaban entre ambos datasets menos de un 2.50% sobre el total de instancias, ya tenemos los datos limpios.

2.2.3 Exploración y transformación

En esta fase de tratamiento de datos nos vamos a centrar en la normalización de los datasets. Esto es necesario para poder concatenarlos.

En primer lugar, eliminamos las columnas o variables que no puedan armonizarse entre los datasets, luego renombramos las columnas o variables comunes para tengan ambos conjuntos de datos un mismo estándar y después, las colocamos en el mismo orden.

Tabla 2.3. Ficheros 1 y 2 normalizados y concatenados – fichero3

	Gender	Education	Work Exp	Programming Exp	Status
0	M	Graduate	No	1	Y
1	M	Graduate	Yes	1	Y
2	M	Graduate	No	0	Y
3	M	Graduate	No	1	N
4	M	Graduate	No	0	Y
...	...	...	...	...	...
609	F	Graduate	No	1	Y
610	M	Graduate	Yes	0	Y
611	M	Graduate	Yes	0	Y
612	M	Graduate	Yes	0	Y
613	F	Graduate	No	1	N

829 rows × 5 columns

Fuente: Elaboración propia vía Python

En segundo lugar, ya habiendo normalizado y concatenado los datos, se puede observar las variables que han sobrevivido en esta transformación de los datos, los cuales son de izquierda a derecha los siguientes: género; si posee educación universitaria; si tiene experiencia laboral; si tiene habilidades de programación y, por último, si han conseguido empleo, o no.

En total hay 5 variables de las cuales, la variable objetivo es 'Status'. Además, se han reunido entre ambos datasets más de 800 instancias. Esto quiere decir que cumplimos con el objetivo mínimo de variables, y superamos satisfactoriamente el número de instancias preestablecido.

Por otra parte, cabe destacar, que el nuevo archivo en formato CSV resultante de esta transformación se denomina 'fichero3', tal y como se puede ver en la tabla 2.3.

2.2.4 Reconocimiento de patrones e interpolación

En este apartado procedemos a analizar el dataset resultante, de forma que nuestro objetivo será entender de una forma preliminar qué relación poseen las 4 variables exógenas con la endógena (o variable objetivo).

En este sentido, lo primero que hacemos es estudiar las correlaciones que existen entre las variables. Estas correlaciones nos darán pistas de cómo crear una variable que imite el comportamiento de la endógena 'Status'. El método consistirá en dotar de un peso mayor dentro de una puntuación sintética a aquellas variables que poseen menos correlación con Status, aunque manteniendo al margen la variable género considerándola un factor constante.

Tabla 2.4. Resultado de añadir las columnas Score y Matching

	Gender	Education	Work Exp	Programming Exp	Score	Matching	Status
0	F	Graduate	No	0	0.7	NOPE	N
1	M	Graduate	No	1	0.8	OK	Y
2	M	Graduate	Yes	0	0.9	OK	Y
3	M	Graduate	No	0	0.7	NOPE	N
4	M	Not Graduate	Yes	0	0.6	NOPE	Y
...	...	...	...	...	...	...	...
824	M	Graduate	No	0	0.7	OK	Y
825	M	Graduate	Yes	0	0.9	OK	Y
826	M	Graduate	No	1	0.8	NOPE	N
827	M	Graduate	No	0	0.7	OK	Y
828	M	Graduate	Yes	0	0.9	OK	Y

Fuente: Elaboración propia vía Python

Como resultado obtenemos una puntuación que denominaremos 'Score' y una variable auxiliar que determinará de forma binaria si la puntuación determina contratación o no: 'Matching'. La forma matemática de la fórmula de Score sería de la siguiente forma:

$$Score = Grad * 0.5 + NGrad * 0.2 + WE * 0.3 + NWE * 0.1 + PE * 0.2 + NPE * 0.1$$

Por lo que, según la instancia, se deberá otorgar a cada par de parámetros un 1 y un 0 dependiendo de si son universitarios o no; poseen experiencia o no, y si tienen habilidades en programación, o no. Todo ello según el orden establecido en la ecuación.

Además, la puntuación se interpretará tal que así:

- Poseerá un intervalo entre 0.40 y 1. A más cerca de 1, más probabilidades que tiene el candidato de ser contratado.

- Probando con varios pesos en las variables, establecemos el límite de ser considerado potencialmente empleable o no, en 0.70.

- En el caso de que el Score y Status coincidan, en la columna Matching se mostrará un 'OK', mientras que, si no lo hacen, se mostrará 'NOPE'. Es decir, todos los candidatos con una puntuación que sean igual o superior a 0.70 y en Status tenga un 'Y' se considerarán 'OK' y los que sean de inferior puntaje a 0.70, y un 'N', 'NOPE'.

- Por otra parte, el 'Score' dará una importancia mayor primero a la educación, luego a la experiencia laboral y finalmente, a las habilidades en programación, ya que es el orden de correlaciones que se han obtenido. Este apartado se explica de forma más profunda en el capítulo 5.

Asimismo, para favorecer el entendimiento de la columna 'Score', procedemos a normalizarla y crear un nuevo intervalo que parta de cero y tenga como límite el uno. Es decir, el puntaje funciona igual sólo que el intervalo va desde 0 a 1 en lugar de entre 0.40 y 1. Así, la frontera de decisión en lugar de ser 0.70, será 0.50. Por tanto, los resultados que sean igual o mayores a 0.50 se considerarán potencialmente contratables y los que no, estarán por debajo de 0.50.

Para llevar un control de las versiones de ficheros y sus cambios, renombramos la columna Score por 'Assurance' (lo que en castellano equivale a 'confianza'). Con este cambio, nace el archivo CSV que denominaremos 'fichero4'.

Tabla 2.5. Resultado de transformar la columna Score a Assurance – fichero4

	Gender	Education	Work Exp	Programming Exp	Assurance	Matching	Status
0	F	Graduate	No	0	0.50	NOPE	N
1	M	Graduate	No	1	0.67	OK	Y
2	M	Graduate	Yes	0	0.83	OK	Y
3	M	Graduate	No	0	0.50	NOPE	N
4	M	Not Graduate	Yes	0	0.33	NOPE	Y
...	...	...	...	...	...	...	...
824	M	Graduate	No	0	0.50	OK	Y
825	M	Graduate	Yes	0	0.83	OK	Y
826	M	Graduate	No	1	0.67	NOPE	N
827	M	Graduate	No	0	0.50	OK	Y
828	M	Graduate	Yes	0	0.83	OK	Y

829 rows × 7 columns

Fuente: Elaboración propia vía Python

Tal y como se puede observar, por un lado, siempre que en la columna auxiliar Matching aparezca un OK, significará que Assurance y Status han coincidido en sus resultados, es decir, éxito en la predicción de Assurance. Por otro lado, la salida NOPE significará que Assurance y Status no coinciden en sus resultados, por lo que será en este caso, un fracaso en la predicción.

2.2.5 Resultados

Si nos fijamos las variables que han quedado para el objeto de estudio, son en esencia, binarias. Por ello, lo primero que vamos a hacer con los resultados del dataframe anterior, es sustituir los valores que se consideren perjudiciales con ceros, y los que se consideren como ventajosos con unos.

La única excepción en este método es la variable género, la cual se tratará como una variable puramente neutra, aunque también la normalicemos ceros y unos.

Tabla 2.6. Resultado de transformar las variables en salidas binarias – fichero5

	Gender	Education	Work Exp	Programming Exp	Assurance	Matching	Status
0	0	1	0	0	0.50	0	0
1	1	1	0	1	0.67	1	1
2	1	1	1	0	0.83	1	1
3	1	1	0	0	0.50	0	0
4	1	0	1	0	0.33	0	1
...	...	...	...	...	...	...	...
824	1	1	0	0	0.50	1	1
825	1	1	1	0	0.83	1	1
826	1	1	0	1	0.67	0	0
827	1	1	0	0	0.50	1	1
828	1	1	1	0	0.83	1	1

829 rows × 7 columns

Fuente: Elaboración propia vía Python

Con esta transformación, obtenemos lo que será el fichero base de estudio. Si bien es cierto que a lo largo del TFM el fichero va a recibir hasta 2 modificaciones más, estas serán mínimas y por supuesto, fruto de la necesidad de adaptar el dataset.

En definitiva, en esta nueva tabla vemos como todo lo que antes eran valores nominales, ahora son valores numéricos binarios, excepto la columna Assurance, la cual no se ha eliminado

en esta fase del tratamiento de datos dado que nos será útil su estudio estadístico en el siguiente capítulo.

Además, ya sea contando con Assurance o con Matching (no podemos contarlas por separado porque representan casi lo mismo), tenemos como resultado un dataset con una variable más, lo que nos deja con un total de 6 variables.

3. Análisis Estadístico

Para comprender mejor el dataset que ha resultado durante el proceso KDD, en este capítulo se van a estudiar las características de las variables, sus estadísticos descriptivos, y, además, las frecuencias y distribuciones de las variables. En resumidas cuentas, obtendremos una síntesis preliminar de los datos con los que se va a trabajar. Todo ello vía Python.

3.1 Análisis básico

Durante este análisis se van a exponer las características de los datos y sus tablas de frecuencia y distribución.

Tabla 3.1. Información básica del fichero5

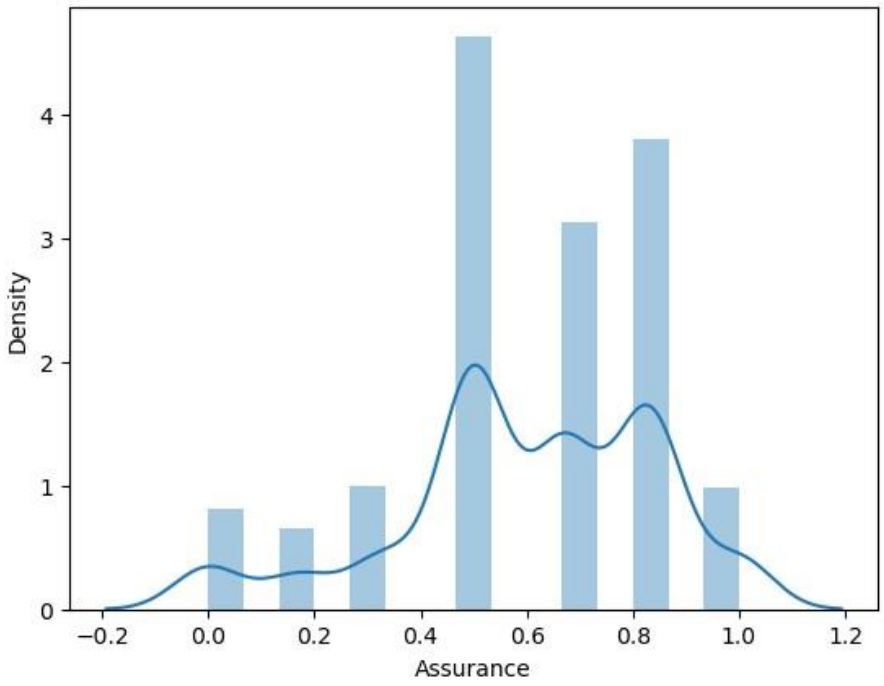
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 829 entries, 0 to 828
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Gender           829 non-null    int64
 1   Education        829 non-null    int64
 2   Work Exp         829 non-null    int64
 3   Programming Exp  829 non-null    int64
 4   Assurance        829 non-null    float64
 5   Matching         829 non-null    int64
 6   Status           829 non-null    int64
dtypes: float64(1), int64(6)
memory usage: 45.5 KB
```

Fuente: Elaboración propia vía Python

Con el comando 'fichero5.info()' Python muestra las características básicas del dataset, reafirmando lo expuesto en los anteriores dataframe: 829 instancias, 7 columnas, y, además, te dice si los datos son enteros o poseen decimales.

Entonces, si en alguna columna se da alguna instancia que no siga el estándar, con este comando lo vas a saber ya que, si no son todos números enteros, no te va a mostrar esa columna como 'int64'. Por otra parte, también informa que en ninguna columna se dan valores nulos. En resumen, los datos están completamente limpios y con este comando se puede comprobar directamente.

Figura 3.1. Densidad y distribución de Assurance – fichero5



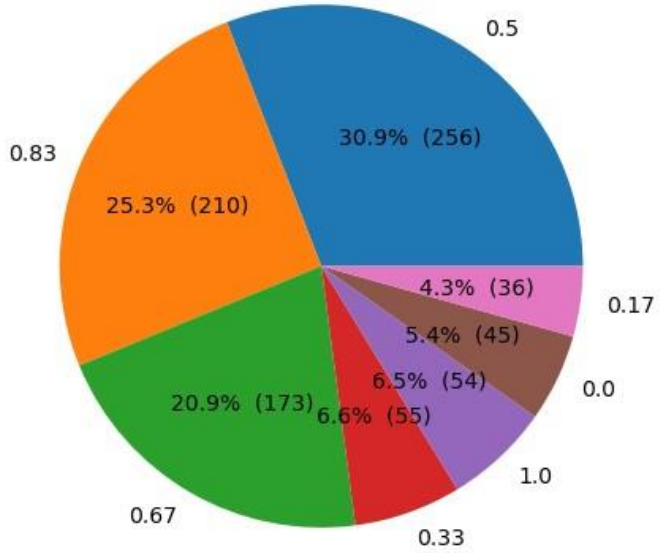
Fuente: Elaboración propia vía Python

Con esta tabla de frecuencias y distribuciones, se puede analizar de forma gráfica la variable Assurance. En este aspecto, vemos como entre los candidatos hay una clara tendencia a ser potencialmente contratados.

Además, aunque sea de una forma algo irregular, la forma de la función de densidad parece que se aproxima una función normal. Luego, si tomamos como verdadera la afirmación anterior, el conjunto de datos estará cerca de ser simétrico y poseerá una cierta tendencia central (0.50), y tendrán un comportamiento positivo respecto a la desviación estándar (en este caso 0.24).

Por otra parte, para visualizar de otra forma esta variable, y poder ver la frecuencia de los datos de una manera más clara, se muestra el siguiente gráfico:

Figura 3.2. Frecuencia y distribución de Assurance – fichero5

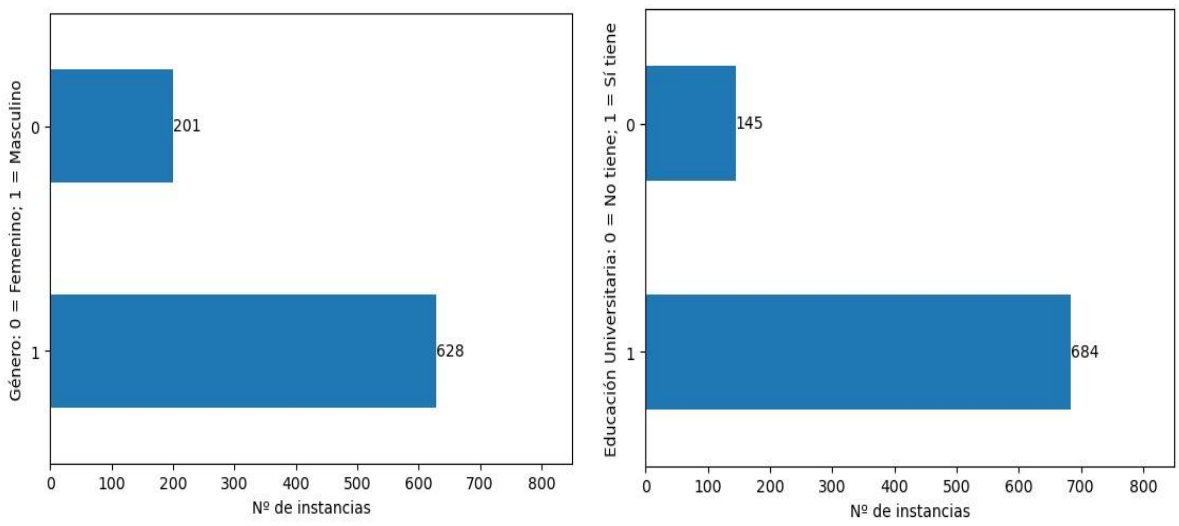


Fuente: Elaboración propia vía Python

Luego, con este gráfico se puede observar de forma sencilla qué puntuación es la moda (0.50), y cuál es la que menos se da en las salidas de la variable (0.17).

A continuación, se van a exponer las distribuciones y frecuencias del resto de variables:

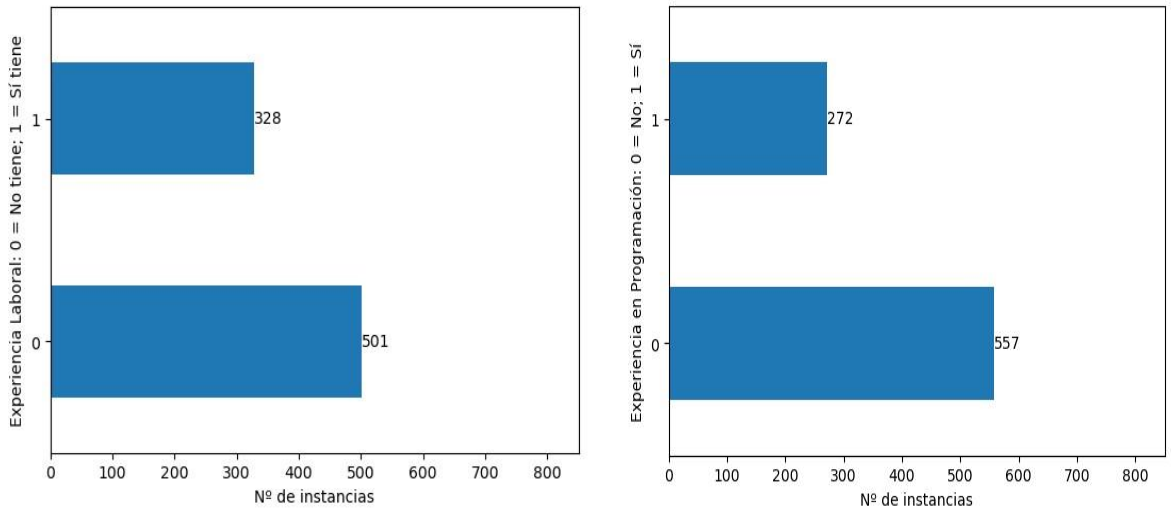
Figura 3.3. Frecuencia y distribución de Género y Educación – fichero5



Fuente: Elaboración propia vía Python

En este caso vemos que el género predominante en el dataset es el masculino y que poseer educación universitaria es tendencia entre los candidatos.

Figura 3.4. Frecuencia y distribución de Experiencia laboral y Experiencia en programación – fichero5

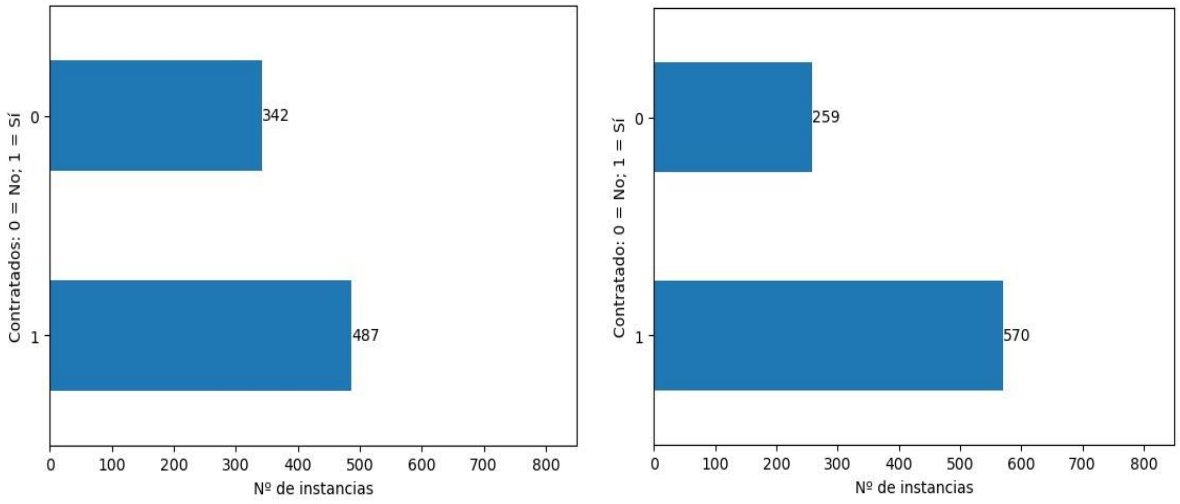


Fuente: Elaboración propia vía Python

En ambas variables no tener experiencia es mayoría. Sin embargo, la distribución está algo más equilibrada, ya que una variable no duplica ni triplica la frecuencia de una salida con la otra.

Importante: se debe observar bien en el eje vertical qué se está estudiando, ya que el orden sólo en este caso es diferente al resto de gráficas.

Figura 3.5. Frecuencia y distribución de Matching y Status – fichero5



Fuente: Elaboración propia vía Python

Finalmente, comparamos la variable auxiliar Matching, y la endógena, Status.

El objeto de Matching mostrar las predicciones correctas de Assurance respecto a Status. Por ello, la frecuencia y la distribución de ambas variables son muy parecidas. En este caso se puede llegar a la conclusión de que se ha alcanzado dicho objetivo.

3.2 Análisis descriptivo

En este apartado nos centraremos en el análisis descriptivo. Este análisis nuevamente se ha realizado íntegramente con la herramienta Python.

Tabla 3.2. Estadísticos descriptivos del fichero5

	Gender	Education	Work Exp	Programming Exp	Assurance	Matching	Status
count	829.000000	829.000000	829.000000	829.000000	829.00000	829.000000	829.000000
mean	0.757539	0.825090	0.395657	0.328106	0.59889	0.587455	0.687575
std	0.428830	0.380119	0.489287	0.469807	0.24579	0.492589	0.463762
min	0.000000	0.000000	0.000000	0.000000	0.00000	0.000000	0.000000
25%	1.000000	1.000000	0.000000	0.000000	0.50000	0.000000	0.000000
50%	1.000000	1.000000	0.000000	0.000000	0.67000	1.000000	1.000000
75%	1.000000	1.000000	1.000000	1.000000	0.83000	1.000000	1.000000
max	1.000000	1.000000	1.000000	1.000000	1.00000	1.000000	1.000000

Fuente: Elaboración propia vía Python

En orden descendente, los estadísticos descriptivos son: número de instancias, media, desviación típica, valor mínimo, primer cuartil, mediana, tercer cuartil y valor máximo.

En primer lugar, en lo que respecta al género la media al ser mayor que 0.50 vuelve a revelar, que hay más hombres que mujeres, ya que se le asignó el dígito 1 al sexo masculino y 0 al sexo femenino.

En segundo lugar, la media es mayor que 0.50, lo que indica en la misma línea una gran mayoría en posesión de título universitario. No obstante, en lo que se refiere a experiencia laboral y de programación, las medias rondan los valores 0.30 y 0.40. Esto se debe a que los candidatos poseen una tendencia a no tener experiencia ni laboral ni de programar en general.

En tercer lugar, se puede observar como la variable Assurance al tener una media superior a 0.50, nos permite volver a afirmar que una mayoría de los candidatos son potencialmente empleables.

En cuarto y último lugar, tenemos las columnas Matching y Status. Como se puede ver, ambas se comportan de una forma muy parecida, por lo que los resultados son casi idénticos.

3.3 Conclusiones

Después de haber analizado las variables a nivel estadístico, se pueden llegar a los siguientes resultados y conclusiones:

a) La variable género está distribuida en un 75% hombres y 25% mujeres.

b) Un 82.50% de los candidatos poseen título universitario.

c) Los perfiles que no poseen experiencia laboral suponen un 60%.

d) En lo que se refiere a las habilidades de programación, un 67% no las posee.

e) Si nos fijamos en Assurance, los perfiles potencialmente empleables son en total un 83.60%, lo que difiere del resultado de Matching, ya que sólo aproximadamente 2/3 partes de este resultado serían finalmente un perfil contratado según Matching.

f) Respecto a la variable Matching obtenemos un resultado coincidente entre Assurance y Status del 58.74%. Esto supone sólo una diferencia de 10 puntos básicos entre Matching y Status, ya que este último tiene una ratio de contratación de los candidatos del 68.75%.

En adición, los tipos de perfiles que se pueden dar en los candidatos calculando sus diferentes combinaciones con las variables educación, experiencias laborales y de programación, son en total 8. Si incluimos la variable auxiliar Matching nos encontraríamos con 16 tipos de candidatos, un número que, sigue siendo apto para seguir con la tesis respecto a la inteligencia colectiva, ya que el abanico de opciones sigue siendo limitado.

4. Análisis Econométrico

Con este análisis obtendremos un punto de vista distinto al meramente computacional o matemático, ya que la intención de este es estudiar y comprender el comportamiento de las variables de forma individual, y en conjunto.

Con ello, el modelo escogido para el estudio del dataset es el de regresión logística (logit). La regresión logística (o logit) es una técnica estadística utilizada para predecir la probabilidad de

que un evento binario ocurra, basándose en variables predictoras. Su fórmula matemática genérica es la siguiente:

$$Logit(p) = \ln \frac{p}{1-p}$$

Donde 'p' representa la probabilidad de que ocurra un evento o resultado binario y 'ln' es la función de logaritmo neperiano.

Por tanto, como este modelo está orientado a analizar conjuntos de datos binarios, por lo que en nuestra problemática será ideal.

De nuevo, durante todo el proceso se ha utilizado Python y su librería Scikit-Learn. Sin embargo, para ciertas comprobaciones se ha utilizado puntualmente STATA, lo cual se irá comentando en el propio capítulo.

4.1 Análisis de heterocedasticidad

Antes de comenzar con el análisis logit, en primer lugar, debemos evaluar si el modelo tiene problemas de heterocedasticidad o no. En consecuencia, para este análisis se va a utilizar el test o prueba de White.

Esta prueba estadística es utilizada en econometría para verificar la presencia de heterocedasticidad en los errores de un modelo de regresión. En este orden, la heterocedasticidad se refiere a la situación en la que la varianza de los errores no es constante a lo largo de los valores de las variables independientes en un modelo de regresión. Su forma de cálculo sería tal que así:

$$Test\ de\ White = \frac{nR2}{2} \sim \chi\text{^}2_{m-p}$$

Donde 'n' es el número de observaciones, 'R2' es el coeficiente de determinación del modelo de regresión, 'm' es el número de variables explicativas y 'p' es el número de parámetros estimados en el modelo.

En nuestro caso, según lo calculado en Python con la librería 'Statsmodels', el p-value (0.0882), al no ser el resultado inferior al 5% establecido, no se rechaza la hipótesis nula (H0), por lo que no se puede afirmar que exista heterocedasticidad (H1 o hipótesis alternativa).

Además, a nivel computacional sólo se han necesitado 5 iteraciones, por lo que es un signo de que el algoritmo es eficiente.

4.2 Modelo de elección binaria

Después de analizar el modelo original, toca prescindir de alguna de las columnas Assurance o Matching, ya que se explican mutuamente y no es lógico mantenerlas de forma simultánea en testeos para los análisis de econometría y machine learning. En este caso concreto prescindiremos de Assurance, ya que no es una variable binaria.

Tabla 4.1. Resultado de eliminar la columna Assurance – fichero6

	Gender	Education	Work Exp	Programming Exp	Matching	Status
0	0	1	0	0	0	0
1	1	1	0	1	1	1
2	1	1	1	0	1	1
3	1	1	0	0	0	0
4	1	0	1	0	0	1
...	...	...	...	...	...	...
824	1	1	0	0	1	1
825	1	1	1	0	1	1
826	1	1	0	1	0	0
827	1	1	0	0	1	1
828	1	1	1	0	1	1

829 rows × 6 columns

Fuente: Elaboración propia vía Python

Después de esta modificación, a continuación, se va a aplicar el modelo logit en el dataset.

En primer lugar, lo primero que se va a realizar es una regresión logística teniendo en cuenta todas las variables del dataset. Sin embargo, la variable Matching es ignorada por el modelo test, por lo que se opta por testear el modelo original, es decir, sólo tenemos en cuenta género, educación, experiencia laboral y de programación, y Status. Como resultado, a este fichero lo denominaremos 'fichero6*', ya que ha sido modificado y es importante tener en cuenta el hecho de que la variable auxilixar Matching no se está teniendo en cuenta. Es decir, se está evaluando el modelo original.

Con este pequeño cambio, ahora sí procedemos a testear el conjunto de datos vía logit:

*Tabla 4.2. Logit aplicado sin Assurance ni Matching – fichero6**

```
Optimization terminated successfully.
         Current function value: 0.612652
         Iterations 5
                           Logit Regression Results
==============================================================================
Dep. Variable:                 Status   No. Observations:                  829
Model:                          Logit   Df Residuals:                      824
Method:                           MLE   Df Model:                            4
Date:                Sat, 15 Apr 2023   Pseudo R-squ.:                 0.01349
Time:                        21:32:49   Log-Likelihood:                -507.89
converged:                       True   LL-Null:                       -514.83
Covariance Type:            nonrobust   LLR p-value:                  0.007668
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.2261      0.245      0.924      0.356      -0.254       0.706
Gender         0.1592      0.178      0.897      0.370      -0.189       0.507
Education      0.5065      0.191      2.653      0.008       0.132       0.881
Work Exp       0.2750      0.162      1.698      0.089      -0.042       0.592
Programming Exp -0.2159    0.165     -1.306      0.192      -0.540       0.108
==============================================================================
```

Fuente: Elaboración propia vía Python

De esta tabla de resultados, nos vamos a fijar en los siguientes parámetros: valor actual de la función, pseudo-R2, 'LLR p-value', 'p-value' (o lo que es lo mismo, P>|z|), y los coeficientes. En total son 6 indicadores.

En primer lugar, el valor actual de la función posee un valor de 0.61. Este valor se refiere al valor numérico que se obtiene al evaluar la función objetivo en el punto actual de la optimización en un análisis logit. La función objetivo es una medida de la calidad del ajuste del modelo y se utiliza para encontrar los valores óptimos de los parámetros del modelo a través de técnicas de optimización.

En general, en un análisis logit, se busca encontrar los valores de los coeficientes de regresión que maximizan la función objetivo, por lo que los coeficientes mostrados en la tabla son los que teóricamente según este análisis, mejor pueden explicar la variable endógena.

En segundo lugar, se debe tener en cuenta que el pseudo-R2 como es un valor relativo, no podemos interpretarlo directamente, ya que no funciona de la misma forma que el R2.

En tercer lugar, vamos a interpretar el LLR p-value y los p-value individuales de cada variable.

En lo que respecta a los p-value, únicamente es considerada como significativa la variable educación, aunque si pusiéramos un límite más flexible, la experiencia laboral podría considerarse

como significativa. Para ello deberíamos fijar una frontera en lugar de una frontera de 0.05, una de 0.1, lo cual no es muy común.

Por otro lado, en lo que se refiere al LLR p-value, aunque tan sólo 1 de las 4 variables es considerada por el modelo logit como significativa, parece ser que el conjunto de las variables es muy significativo, ya que el valor de este p-value conjunto es de 0.0076, lo cual lo sitúa incluso a un mejor nivel de significatividad que educación (0.008). Esto quiere decir que el desempeño de las variables en conjunto es satisfactorio.

A continuación, vamos a interpretar los coeficientes:

- Género. Al ser el coeficiente más cercano a cero, se puede llegar a la conclusión de que es la variable que menos impacto tiene en la variable endógena.

- Educación. Es con diferencia, la variable más determinante. Su signo es positivo por lo que, si este coeficiente aumenta, habrá más probabilidades de que la variable endógena nos dé el resultado que consideramos como positivo.

  A modo de aclaración, un resultado positivo sería que Status tuviera en la instancia un valor de 1 por lo que el candidato estaría contratado.

- Experiencia laboral. Posee un coeficiente con un valor de la mitad del de educación, por lo que se podría decir que su impacto es menor.

- Experiencia/Habilidades en programación. En este caso el coeficiente es negativo, por lo que a medida que aumente el valor de este coeficiente, menos cerca estaremos de que la variable endógena nos dé un resultado o evento que consideremos como positivo. Sin embargo, a pesar de que tomemos esta variable como 'marginalmente' significativa, estadísticamente no es significativa, ya que su desviación típica es bastante alta en comparación con el tamaño del coeficiente. Es decir, su resultado en términos estadísticos no sería distinto de cero.

Por último, nos quedaría la constante. Esta posee un valor de 0.22, y al ser positivo, estaríamos más cerca de que ocurra el evento o resultado que consideramos como positivo.

4.3 Conclusiones

Habiendo calculado y analizado los resultados del modelo de elección binaria logit, se exponen las siguientes conclusiones:

a) Que el modelo original no posee problemas de heterocedasticidad según test de White.

b) Que las variables de forma individual únicamente es significativa educación, pero en conjunto, su significatividad es aún mayor.

c) Que las predicciones realizadas con Python y STATA son idénticas, en términos de coeficientes y p-value.

d) Que el resultado de la regresión logística como valor óptimo en Python obtenemos un valor de entre 0.61 y 0.73, y en STATA un valor medio de 0.68. Esto quiere decir que los resultados son hasta cierto punto, similares.

e) Que al predecir vía modelo logit en STATA, obtenemos los mismos resultados que utilizando la columna Assurance en el momento de realizar un 'cutoff' en 0.50 unidades. Esto quiere decir que la variable Assurance, posee un desempeño satisfactorio.

En definitiva, el modelo logit nos ha dado una imagen de las variables y los resultados han sido buenos. Sin embargo, el conjunto de variables es mejorable y explorar la exclusión o inclusión de nuevas variables, podría ser una buena forma de mejorar la significatividad de estas.

5. Análisis de coeficientes

En este capítulo se van a calcular vía herramienta Python varios coeficientes en función del fichero6. Este fichero posee todas las variables en formato binario e incluye la variable auxiliar Matching.

Por tanto, este análisis se puede considerar una especie de extensión de los análisis estadístico y econométrico. Es decir, si en ambos análisis ya se analizaron las variables y sus características, en este apartado se va a estudiar el dataset como unidad.

Luego, estos coeficientes valorarán de una forma u otra, su desempeño y comportamiento como conjunto de datos, en el que la única excepción será el coeficiente de correlación de Pearson, el cual, también analizará las variables a nivel individual.

5.1 Coeficiente AUC-ROC

El AUC-ROC (del inglés, 'Area Under the Curve of the Receiver Operating Characteristic') es una medida utilizada en problemas de clasificación. Representa la capacidad de un modelo para distinguir entre clases positivas y negativas. Su fórmula matemática sería la siguiente:

$$AUC - ROC = \int [0,1] Sensibilidad(TPR) * Especificidad(FPR)\, d\theta$$

Donde TPR (Tasa de Verdaderos Positivos) es la proporción de verdaderos positivos respecto a todos los positivos, y FPR (Tasa de Falsos Positivos) es la proporción de falsos positivos respecto a todos los negativos. Por otro lado, 'θ' representa el umbral de clasificación utilizado por el modelo.

Por tanto, valor de AUC-ROC próximo a 1 indica un modelo con un buen desempeño; un valor cercano a 0.5 indica aleatoriedad; y un valor cercano a 0 indica que no tiene un buen comportamiento predictivo.

En este caso, el resultado ha sido de 0.92, lo que indica un muy buen desempeño en la distinción entre clases positivas y negativas. Esto se traduce en una buena calidad de predicción.

5.2 Coeficiente Kappa de Cohen

El Kappa de Cohen es una medida de concordancia entre clasificadores. Es utilizado para evaluar la consistencia en la clasificación de datos en categorías o clases. A continuación, se expone su fórmula matemática:

$$Kappa = \frac{\mathrm{Po} - \mathrm{Pe}}{1 - \mathrm{Pe}}$$

Donde 'Po' es la proporción de acuerdo observado y 'Pe' es la proporción de acuerdo esperado bajo el azar.

Entonces, un valor de Kappa cercano a 1 indica una alta concordancia, mientras que un valor que esté por debajo de 0.50-0.60 podría indicar que los datos no son todo lo consistentes que deberían de ser.

El resultado que se ha obtenido de este coeficiente es de 0.78, lo que lo coloca por encima del límite establecido de 0.50. Esto significa que no abundan los eventos aleatorios en el modelo.

5.3 Coeficiente de correlación de Pearson

La correlación de Pearson es una medida de la relación lineal entre dos variables. Indica la fuerza y dirección de la asociación entre las variables, donde un valor de 1 indica una correlación positiva perfecta, -1 indica una correlación negativa perfecta, y 0 indica ausencia de correlación. Su fórmula matemática sería tal que así:

$$C.Pearson = \frac{\sum[(\mathrm{xi} - \bar{\mathrm{x}}) * (\mathrm{yi} - \bar{\mathrm{y}})]}{(\mathrm{n} * \mathrm{sx} * \mathrm{sy})}$$

Donde, en primer lugar, 'xi' y 'yi' son los valores de las dos variables, '$\bar{x}$' e '$\bar{y}$' son las medias de las variables, y, en segundo lugar, 'sx' y 'sy' son las desviaciones estándar de las variables, y 'n' es el número de pares de observaciones.

En lo que respecta este coeficiente, a continuación, se expone la siguiente tabla:

Tabla 5.1. Correlación de Pearson – fichero6

	Gender	Education	Work Exp	Programming Exp	Matching	Status
Gender	1.000000	-0.060435	0.129666	-0.204122	0.023318	0.043740
Education	-0.060435	1.000000	-0.043051	0.017417	0.523623	0.086997
Work Exp	0.129666	-0.043051	1.000000	-0.234425	0.041665	0.071721
Programming Exp	-0.204122	0.017417	-0.234425	1.000000	-0.045861	-0.066631
Matching	0.023318	0.523623	0.041665	-0.045861	1.000000	0.804385
Status	0.043740	0.086997	0.071721	-0.066631	0.804385	1.000000

Fuente: Elaboración propia vía Python

Como se puede observar, el coeficiente revela que las variables explicativas no están correlacionadas. Es más, rondan entre -0.2 y 0.2. Mismo caso entre las variables exógenas y las endógenas (exceptuando Matching), las cuales no poseen prácticamente correlación entre sí.

Cabe destacar que como sucedía en el análisis econométrico, por un parte, la única variable con un comportamiento negativo es la experiencia de programación, por otra, el orden de variables de más significativas a menos es prácticamente el mismo respecto a la variable que posee más correlación y la que menos conforme a la variable endógena (Status).

Además, como era de esperar, Matching es la que más correlación posee con Status. Es importante tener en cuenta que un valor cercano a cero no necesariamente significa que no haya relación entre las dos variables, ya que puede haber una relación no lineal entre ellas que no se pueda capturar con el coeficiente de correlación de Pearson.

5.4 Distancia del coseno

La distancia del coseno es una medida de similitud entre dos vectores, en el cual, si el valor de la distancia posee un valor de 1 significa que son idénticos; si posee un valor -1 significa que son inversos; y, por último, si posee un valor de 0 significaría que no poseen ninguna similitud. Su fórmula matemática es la siguiente:

$$Distancia\ del\ coseno = \cos(\theta) = \frac{\vec{a} \cdot \vec{b}}{\| \vec{a} \| \| \vec{b} \|}$$

Donde 'a' y 'b' son los vectores que se comparan y 'θ' es el ángulo entre ellos.

En nuestro caso, al ser una distancia de 0.5, los vectores coinciden en un 50%, lo que significa que hay un nivel intermedio de similitud entre los vectores seleccionados. En otras palabras, se trata de un resultado que se puede considerar razonablemente bueno, ya que lo ideal es que esté lo más cercano a 0.

5.5 Coeficiente VIF Inverso

El VIF (del inglés, 'Variance Inflation Factor') es una medida que se utiliza en estadística para evaluar la multicolinealidad en un modelo de regresión. La multicolinealidad es una situación en la que dos o más variables predictoras en un modelo de regresión están altamente correlacionadas entre sí, lo que puede causar problemas en la interpretación y precisión del modelo.

En este caso, al ser el conjunto de datos de carácter binario, lo ideal es utilizar el coeficiente VIF inverso, más conocido como 'tolerancia'. Su fórmula es:

$$Tolerancia = \frac{1}{1 - \mathrm{VIF}}$$

Por consiguiente, el resultado que nos ha dado el análisis de tolerancia en el modelo original ha sido muy positivo, ya que todos los valores han rondado cerca del valor 1, lo que sugiere la no

existencia de multicolinealidad. En caso contrario, si los valores hubieran sido muy alejados de 0 o 1 o cercanos a 0, indicaría una presencia de multicolinealidad entre las variables.

5.6 Coeficiente Recall

El Recall, también conocido como Tasa de Verdaderos Positivos (TVP), es un coeficiente que mide la capacidad de un modelo para identificar correctamente los casos positivos. La fórmula matemática de este coeficiente de cobertura sería la siguiente:

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

Donde TP (del inglés, 'True Positives') es la cantidad de casos positivos correctamente clasificados y FN (del inglés, 'False Negatives') es la cantidad de casos positivos incorrectamente clasificados.

Por tanto, valor de Recall cercano a 1 indica un alto nivel de detección de casos positivos, mientras que un valor cercano a 0 indica un bajo nivel de detección.

En este coeficiente el resultado ha sido de 0.85, lo cual nos revela que se predice en torno a un 85% de las salidas de Status.

5.7 Coeficiente F1

La medida F1 es una métrica utilizada en estadísticas y aprendizaje automático para evaluar la precisión y exhaustividad de un modelo de clasificación binaria. Este coeficiente tiene un rango de valores entre 0 y 1, donde 1 indica una clasificación perfecta y 0 indica una clasificación muy pobre. Por otro lado, su fórmula matemática es:

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

Donde, 'Precision' es la proporción de verdaderos positivos y 'Recall' (o exhaustividad), como se ha mencionado anteriormente en este capítulo, es la proporción de verdaderos positivos respecto al total de instancias positivas.

En nuestro caso, los resultados obtenidos tras calcular la F1 en el modelo son de entre 0.79 y 0.93, sin incluir e incluyendo la variable auxiliar Matching respectivamente. Por lo que, se puede afirmar que la medida muestra un resultado muy positivo.

5.8 Conclusiones

Si bien es cierto que la mayoría de los coeficientes han dado resultados positivos respecto al modelo, cabe recordar que el dataset es simple, posee variables limitadas y el número de instancias no suponen un gran desafío a nivel computacional. Además, existen más coeficientes que pueden aplicarse al dataset.

Por tanto, el motivo por el cual se han elegido los presentes coeficientes en este capítulo es por su uso asiduo en este tipo de análisis.

Tabla 5.2. Resultados de los coeficientes – fichero6

Coeficientes	**Resultados**
AUC-ROC	0.92
Kappa de Cohen	0.78
C. C. de Pearson	< |0.2|
D. Coseno	0.5
VIF Inverso	<1.24
Recall	0.85
F1	0.79 - 0.93

Fuente: Elaboración propia

Sin embargo, los resultados han sido bastante prometedores y puede servir para implantar esta forma de estudio en un problema con un mayor número de variables e instancias. Recordemos que el valor añadido de este TFM es la combinación de técnicas econométricas y de machine learning en un mismo estudio, por lo que, si además le añadimos el factor disciplina laboral, no se dan apenas precedentes ni en datasets ni en análisis similares previos. Con todo ello, los resultados y análisis a nivel estadístico, econométrico y de coeficientes han sido muy satisfactorios en el intento de conocer el dataset, evaluarlo y llegar a la conclusión de que su desempeño, aunque sea a pequeña escala, es bueno.

6. Aplicación de algoritmos vía Machine Learning

En el campo del aprendizaje automático se dan multitud de algoritmos con los que se pueden testar y analizar diferentes conjuntos de datos. Algunos de estos algoritmos son, de menos a más complejo: kNN, Naïve Bayes, SVM, árboles de decisión y las redes neuronales.

En nuestro caso particular, se han elegido los algoritmos kNN y árboles de decisión. Esta elección viene dada principalmente por tres factores: no disponemos de muchas instancias, el problema que se plantea no es especialmente complejo, y los datos son de corte binario, por lo que después de probar algunos de los principales algoritmos de aprendizaje supervisado, estos, eran los más interesantes.

En este capítulo se va a utilizar el fichero7 en los modelos 'A', y fichero6 en los modelos 'B'. Esto se hace para comparar el desempeño del dataset sin la columna Matching (modelos A) y con columna Matching (modelos B), vía algoritmos anteriormente citados.

Por último, cabe destacar que en ambos algoritmos el tamaño del conjunto destinado al test es de un 25%, y el de entrenamiento de un 75%. Con ello, al realizarse varias pruebas en el caso del kNN según la distancia, en concreto 10 por distancia al ser diferentes 'k' (30 pruebas en total), podría decirse que al construirse de forma aleatoria (lo cual está comprobado vía Python) este tipo de subconjuntos, pueden ser considerados una validación cruzada.

6.1 Aplicación kNN

El algoritmo kNN (k-Nearest Neighbors) trata de encontrar los 'k' vecinos más cercanos a un nuevo punto de datos utilizando una medida de distancia, luego imita una votación mayoritaria o promedio para clasificar el nuevo punto, y, por último, evalúa la precisión del modelo para determinar su rendimiento. En términos de regresión su formulación matemática sería la siguiente:

$$X = \{(x1, y1), (x2, y2), \ldots, (xn, yn)\}$$

Donde 'xi' es un vector de características del punto de datos 'i', e 'yi', es su valor a predecir.

En este sentido, las distancias utilizadas son: Euclídea, Jaccard y Hamming. Estas distancias a excepción de la Euclídea, que suele ser la predeterminada, están mejor diseñadas que el resto para modelos de corte binario.

En lo que respecta a la distancia Euclídea, es una medida de distancia entre dos puntos en un espacio euclidiano, como un plano o un espacio tridimensional. Su formulación matemática es la siguiente:

$$D.Euclídea = \sqrt{((x2 - x1)\text{^}2 + (y2 - y1)\text{^}2 + \ldots + (xn - xn)\text{^}2)}$$

Donde (x1, y1, ..., xn) y (x2, y2, ..., xn) son las coordenadas de los dos puntos en el espacio euclidiano.

Por otro lado, en lo que concierne a la distancia Jaccard, es una medida de similitud entre dos conjuntos. Esta se utiliza para calcular la similitud entre dos conjuntos en función de la cantidad de elementos compartidos en relación con la cantidad total de elementos en los conjuntos. Su fórmula matemática es:

$$D.Jaccard = 1 - \frac{|A \cap B|}{|A \cup B|}$$

Donde '|A ∩ B|' es la cantidad de elementos compartidos entre los conjuntos 'A' y 'B', y '|A ∪ B|' es la cantidad total de elementos en la unión de los conjuntos 'A' y 'B'.

En tercer y último lugar, la distancia Hamming es una medida de distancia entre dos cadenas de bits de igual longitud. Se utiliza para calcular la diferencia entre dos cadenas de bits en términos de la cantidad de bits diferentes. Luego, su fórmula matemática sería la siguiente:

$$D.Hamming = \sum(\text{xi} \neq \text{yi})$$

Donde 'xi' y 'yi' son los bits en la posición 'i' de las dos cadenas de bits, y el símbolo '≠' representa la diferencia entre los bits, es decir, si los bits son diferentes, se suma 1, y si son iguales, se suma 0.

6.1.1 Modelo A

En este primer apartado del kNN se va a utilizar el fichero7, el cual no posee la columna Matching. Es decir, se va a someter el modelo original a los distintos kNN según sean las distancias utilizadas.

Antes de comenzar a interpretar la tabla, debemos saber que: 'm' es el promedio más bajo de todos los 'k' utilizados; 'M' es el promedio más alto; y 'O', es el óptimo en el que se encuentra un mejor equilibrio entre el menor valor de 'k' y el mayor promedio.

Tabla 6.1. Algoritmo kNN en el modelo A – fichero7

Valores de k	D. Euclídea	D. Jaccard	D. Hamming
2	52.40% (m)	47.59% (m)	45.67% (m)
4	61.53%	49.51%	49.51%
6	61.53%	51.44%	48.07%
8	55.76%	62.02%	56.73%
10	**69.23% (O)**	65.86%	**68.75% (O)**
20	67.30%	**70.67% (O)**	68.26%
40	70.19% (M)	74.51% (M)	69.23% (M)
80	64.90%	66.34%	66.34%
150	63.94%	68.26%	67.78%
300	68.75%	68.26%	68.26%

Fuente: Elaboración propia

Respecto a los resultados, se puede comprobar que el desempeño de las tres distancias es similar, aunque si entramos en detalles, la distancia Euclídea y la Jaccard son de forma mínima las que peores y mejores resultados poseen.

Cabe destacar también, que las tres distancias tienen como parámetro óptimo 'k' un valor comprendido entre 10 y 20, mientras que los mínimos y máximos coinciden en ambos en valores de 2 y 40 respectivamente.

Además, las matrices de confusión de las tres distancias en el 'k' óptimo dan cero errores tanto en falsos positivos como en falsos negativos.

6.1.2 Modelo B

En este segundo apartado del kNN se va a utilizar el fichero6, el cual sí posee la columna Matching en el modelo.

Tabla 6.2. Algoritmo kNN en el modelo B – fichero6

Valores de k	D. Euclídea	D. Jaccard	D. Hamming
2	89.90%	91.82%	92.30%
4	90.86%	93.26% (M)	92.78% (M)
6	94.71% (M)	**93.26% (O)**	**92.78% (O)**
8	**94.71% (O)**	91.82%	88.94%
10	93.75%	88.46%	91.82%
20	93.75%	93.26%	92.30%
40	92.30%	92.78%	92.78%
80	88.46%	78.84%	92.78%
150	85.09%	78.84%	81.25%
300	79.32% (m)	71.15% (m)	72.11% (m)

Fuente: Elaboración propia

Respecto a los resultados, se puede comprobar, de nuevo, que el desempeño de las tres distancias es muy similar.

En cambio, en este modelo las tres distancias tienen como parámetro óptimo 'k' un valor comprendido entre 6 y 8, mientras que los mínimos y máximos coinciden en ambos en valores de 300 y entre 4 y 6 respectivamente.

De nuevo, tal y como sucede también en el modelo A, las matrices de confusión de las tres distancias en el 'k' óptimo dan cero errores tanto en falsos positivos como en falsos negativos.

6.2 Aplicación de árboles de decisión

El algoritmo de árbol de decisiones consiste en seleccionar las características óptimas para dividir los datos en subconjuntos de los cuales construye un árbol de decisiones que se puede utilizar para clasificar datos. La precisión del modelo se evalúa y se pueden realizar ajustes

para mejorar su rendimiento en base a las métricas de evaluación, lo cual en términos de regresión está estrechamente relacionado con el SRS (del inglés, 'Sum of Residual Squares').

En otras palabras, un árbol de decisión, aunque depende de las hojas, ramas y talas (o podas) para definirse, lo clave en este algoritmo será antes de avanzar en la iteración minimizar el SRS. Esto quiere decir que el árbol, sólo avanzará por la una región sea la que menor SRS ofrezca. Así pues, dado que el algoritmo en sí no posee una fórmula matemática maestra, si debemos de tener en cuenta alguna, será la de SRS:

$$SRS = \sum (yi - \hat{y}i)\text{^}2$$

Donde 'yi' es el valor observado en los datos, y 'ŷi' es el valor predicho por el modelo de regresión.

En lo que respecta a los árboles de decisión en Python, en concreto en la librería Scikit-Learn, es un árbol de decisiones binario con criterio de división predeterminado basado en el índice de Gini. Por otro lado, utiliza la estrategia de seleccionar la mejor característica en cada división; maneja valores faltantes; no tiene profundidad máxima predeterminada; requiere un mínimo de 2 muestras para dividir y 1 muestra en una hoja por defecto.

Respecto al uso predeterminado de Gini, utiliza este índice para medir la impureza de los subconjuntos de datos en cada división.

Por último, en los que se refiere a los modelos A y B, se seguirá el mismo patrón que en el kNN: en el modelo A no habrá variable Matching incorporada y en el B, sí. Además, en este algoritmo se incluirán los resultados de la regresión, el error cuadrático medio (ECM), la raíz del ECM, y el rango de la variable.

6.2.1 Modelo A

En este primer apartado del árbol de decisión se va a utilizar otra vez como es lógico, el fichero7. A continuación se muestran los valores y datos resultantes:

Tabla 6.3. Algoritmo de árbol de decisión en el modelo A – fichero7

Valor de clasificación	0.68
ECM	0.23

Raíz del ECM	0.48
Rango de la variable	1

Fuente: Elaboración propia

La precisión del modelo según el test ha sido de un 68%, un resultado en la línea del kNN. Además, el ECM es de 0.23. Por sí mismo no posee valor o interpretación absoluta, pero lo que sí sabemos es que cuanto más cerca del valor cero, mejor.

Por otro lado, el rango de valores de la variable endógena es 1, lo que significa que la diferencia entre el valor máximo y mínimo de la variable dependiente es 1 unidad. En este caso, la raíz del ECM es aproximadamente un 48% del rango de valores. Esto sugiere que las predicciones del modelo son según el contexto, imprecisas.

La matriz de confusión posee un alto número de falsos positivos, ya que su distribución ha sido de 6 verdaderos positivos (TP), 60 falsos positivos (FP), 5 falsos negativos (FN), y 137 verdaderos negativos (TN). Esto quiere decir que el modelo mide bien las salidas negativas, pero posee un alto índice de error en la predicción de salidas positivas.

6.2.2 Modelo B

En este segundo apartado se va a utilizar de nuevo, el fichero6. A continuación se muestran los valores y datos resultantes:

Tabla 6.4. Algoritmo de árbol de decisión en el modelo B – fichero6

Valor de clasificación	0.94
ECM	0.03
Raíz del ECM	0.19
Rango de la variable	1

Fuente: Elaboración propia

La precisión del modelo según la prueba ha sido de un 94.23%, un resultado similar a los kNN y mejor que el ofrecido por el modelo A. Por otro lado, el ECM ha sido de 0.038. Además, a esto se le suma que la raíz del ECM sí es un valor comparable por lo que en el modelo B

nuevamente se ha conseguido un mejor valor de desempeño que en el A. Esto sugiere que las predicciones del modelo B son bastante precisas en comparación con el A.

En otro orden, el rango de valores de la variable endógena es 1, lo que significa tal y como se ha mencionado también en el modelo A, que la diferencia entre el valor máximo y mínimo de la variable dependiente es 1 unidad.

La matriz de confusión esta vez ha tenido una mejor distribución de los errores, siendo mitigado en su mayoría. La distribución es la siguiente: 55 TP, 5 FP, 7 FN, y 141 TN. Luego, esto significa que el modelo B se comporta mejor que su predecesor, el modelo A.

6.3 Comparación entre los modelos A y B

En general, los datos obtenidos entre los modelos A y B, se ve una clara tendencia de mejores resultados en B, que en A. Esto quiere decir, que la inclusión de la variable auxiliar Matching impacta de forma muy positiva en el modelo:

Tabla 6.5. Comparativa entre los modelos A y B – fichero7 y fichero6

Algoritmos	Resultados fichero7	Resultados fichero6
kNN (D. Euclídea)	0.69	0.94
kNN (D. Jaccard)	0.70	0.93
kNN (C. Hamming)	0.68	0.92
Árbol de Decisión	0.68	0.94
Error Cuadrático Medio	0.23	0.03
Raíz del ECM	0.48	0.19
Rango de la variable	1	1

Fuente: Elaboración propia

Los datos que se ven en la tabla son los resultados óptimos de cada algoritmo y en el caso del kNN, de cada distancia. En esta línea, cabe destacar, por un lado, que los resultados independientemente de las distancias son sólidos, y por otro, que la selección de instancias en los

conjuntos de testeo se realiza de forma aleatoria y estratificada. Esto es debido a que la librería Scikit-Learn lo realiza de forma predeterminada.

Luego, tal y como se puede apreciar, los resultados provenientes del fichero6 arrojan una clara mejoría frente a los del fichero7. Sin embargo, los datos que se manejan en el fichero7 sin aplicación alguna salvo el proceso KDD a inicios del estudio, son cuanto menos, positivos.

En otras palabras, el potencial de mejora del modelo si se decidiera prescindir de la variable auxiliar Matching es amplio, ya que con sólo 4 variables el modelo es capaz en conjunto de resolver el 70% de las salidas de Status. Este porcentaje sube hasta una media del 93% cuando se añade la variable auxiliar.

En este contexto, además, las matrices de confusión como es lógico distribuyen mejor las clases en el modelo B.

Sin embargo, hay una gran diferencia entre ambos modelos que se debe analizar: la tendencia de los 'k' y la situación de los máximos y mínimos. Como se puede ver en las tablas 6.1 y 6.2, el valor de los 'k' en el que se alojan los mínimos y máximos se invierte. Lo único que se mantiene constante en este aspecto es que los óptimos siempre están cerca de los valores máximo, sea en el modelo A o B. En esta perspectiva, como las diferencias entre los modelos A y B respecto a los árboles ya se ha tratado anteriormente, nos centraremos en el caso del kNN.

Por tanto, por un lado, si el valor de 'k' es bajo el modelo puede volverse demasiado sensible a pequeñas variaciones (o ruido) en los datos de entrenamiento, y, por ende, en un sobreajuste. Por otro lado, si el valor de 'k' es alto, puede llegar a suavizar la predicción, ya que se están tomando en cuenta más puntos en el proceso de clasificación. Esto puede ser útil cuando los datos tienen características importantes que se pueden capturar mejor con un enfoque más amplio.

Por consiguiente, seleccionar los valores óptimos de 'k' en kNN implica encontrar un equilibrio entre la sensibilidad al ruido y la capacidad de capturar patrones relevantes en los datos. En nuestro caso, los valores de 'k' necesarios en el modelo A duplican a los necesarios en el modelo B para encontrar el óptimo. Esto sugiere, que los datos en el modelo A poseen quizá menos debilidad respecto a cambios en el modelo y en el B, son algo más sensibles al ruido y los cambios. También, se puede decir que en el modelo B se capturan mejor los cambios locales y en el B, al tener un mayor número de 'k', está más enfocado a cambios generales.

Lo ideal, en resumidas cuentas, sería un valor de 'k' que se alejara todo lo posible de valores muy bajos o muy altos, y en este intervalo, que el valor de 'k' fuese medio-alto para datasets con datos

generalistas y pocas variables, y un numero de 'k' medio-bajo, para aquellos datasets con datos muy específicos con un enfoque al detalle. De nuevo, en nuestro caso, los datos son generalistas, disponemos de un número de variables limitado y en adición, son de tipo binario, por lo que un 'k' medio-alto sería lo más conveniente en nuestro estudio. Todo esto teniendo en cuenta que los valores predictivos que nos dé el valor 'k', serán los que determinen qué rango de 'k' escogeremos para su uso, tal y como se ha hecho en este estudio en la elección de los óptimos de los modelos A y B.

7. Conclusiones

A continuación, se van a exponer varias conclusiones del estudio, en el que se van a abordar las siguientes cuestiones y resultados:

a) Creación del nuevo dataset. La confección del nuevo conjunto de datos es un éxito de este estudio, ya que como se ha comentado durante el TFM, son escasos los datasets binarios orientados a la contratación laboral. En este sentido, el dataset posee una serie de características que lo hacen aún más interesante, ya que posee de serie varianzas constantes (homocedasticidad), clases balanceadas, y no da señales de correlación ni multicolinealidad.

b) Incorporación de una variable auxiliar. En muchos estudios la escasez de variables y la falta de capacidad predictiva son un problema. Este, se puede resolver incorporando lo que en este TFM se ha denominado como 'variable auxiliar'. Esta variable es también una contribución del presente estudio, ya que ha mejorado la capacidad predictiva del modelo en la mayoría de los análisis a los que ha sido sometido.

   Sin embargo, aunque hace mejorar al modelo de forma notable lo cual es su objetivo principal, al estar construida en base al resto de variables originales del dataset, los niveles de correlación y multicolinealidad que provoca no son positivos.

c) Construcción de la variable auxiliar. En primer lugar, para la construcción de la variable se han tenido en cuenta inicialmente por un lado los datos mostrados de la correlación de Pearson, con el que nos hemos apoyado para asignar pesos a cada variable original, y después, del Recall, del que nos hemos inspirado para calcular la nueva variable.

   En segundo lugar, hemos iterado varias veces hasta conseguir el formato de variable que más interesaba, ya que hemos pasado de tener una columna denominada 'Score', la cual hemos normalizado y renombrado a 'Assurance' para finalmente, convertirla a formato binario y renombrarla de nuevo a 'Matching'.

En tercer lugar, aunque el proceso ha sido un éxito, la forma de asignar inicialmente pesos a las variables para la construcción de la variable es mejorable, ya que, en lugar de únicamente utilizar la correlación de Pearson, sería recomendable, añadir al análisis previo algún test de multicolinealidad, de significatividad y en caso de que los datos sean de carácter binario, la distancia del coseno. De esta forma, la construcción de la variable se apoyaría en cuatro indicadores y no sólo en uno. En nuestro caso, aunque no se han realizado todos los test mencionados previamente a su construcción, los resultados han sido mayoritariamente positivos.

d) Análisis completo y multidisciplinar. En este contexto, la metodología y las comprobaciones que se han realizado durante el estudio ha sido amplia. Se ha sometido al dataset a un análisis estadístico, econométrico, de coeficientes y de aprendizaje automático. Todo ello siendo de nuevo, una contribución notable del TFM el combinar las disciplinas de econometría y aprendizaje automático en un estudio predictivo.

   Por otro lado, durante la aplicación de algoritmos de machine learning, la validación cruzada ha sido el eje principal por el que se han analizado los datos. Por ende, se puede afirmar que el estudio en sí posee una cierta robustez.

   Además, cabe destacar, que las salidas de la variable endógena (Status) no se deben exclusivamente a un evento que se pueda calcular matemáticamente, sino que las decisiones humanas son clave en decantar si un candidato es contratado, o no. Este dataset se basa entonces, en un conjunto de decisiones humanas que pueden ser catalogadas como fruto de la inteligencia colectiva.

e) Resultados. En lo que se refiere a los resultados obtenidos, de forma general, han sido positivos, destacando 2 de ellos: los relacionados con el análisis de características de las variables originales del dataset (fichero3), y los relacionados con la capacidad predictiva incluyendo la variable auxiliar Matching (fichero6).

En definitiva, se ha conseguido mejorar el desempeño del conjunto de datos original (fichero3) al modificado (fichero6) de forma satisfactoria. Esto se ha llevado a cabo vía pruebas y análisis que dotan a los resultados y conclusiones del estudio, de una gran solidez.

Bibliografía

Adecco España (Desconocido): Cómo es el proceso de selección de personal: fases y resultados, https://www.adecco.es/insights/como-es-el-proceso-seleccion-personal-fases-y-resultados (Último acceso desconocido)

Anaconda Inc. (2016): Start coding immediately with Anaconda's brand new cloud notebook, https://www.anaconda.com/products/distribution/start-coding-immediately (Último acceso: 2023)

Aranda Corral, Gonzalo A. Universidad de Huelva (UHU), (2023): Programación: Librería Pandas, pp. 4-35.

Aranda Corral, Gonzalo A. Universidad Internacional de Andalucía (UNIA), (2023): Sistemas de Recomendación: Inteligencia Colectiva (Parte 1), pp. 4-20, 40 y 41.

Aranda Corral, Gonzalo A. Universidad Internacional de Andalucía (UNIA), (2023): Sistemas de Recomendación: Inteligencia Colectiva (Parte 2), pp. 5-33.

Armero Ramón. Business Insider (2022): Por qué una inteligencia artificial no debería ser la encargada de contratarte, según un directivo de la firma de contratación ManpowerGroup, https://www.businessinsider.es/inteligencia-artificial-no-deberia-ser-encargada-contratarte-1173980 (Último acceso desconocido)

BBC, (2021): 3 consejos para "burlar" los algoritmos que te seleccionan cuando buscas trabajo, https://www.bbc.com/mundo/noticias-59017785 (Último acceso: 2023)

Calavia Rogel Miriam. Cinco Días, El País. (2021): Inteligencia Artificial al servicio de los procesos de selección, https://cincodias.elpais.com/cincodias/2021/11/18/companias/1637245533_878212.html (Último acceso: 2023)

Carmona García, Mónica. Universidad Internacional de Andalucía (2023): Sistemas de Recomendación: Introducción, pp. 17-71.

Fernández de la Cigoña Fraga, José Ramón. Centro de Estudios Financieros (CEF). (2018): Inteligencia Artificial al servicio de los procesos de selección, https://www.laboral-social.com/7-fases-que-puede-tener-un-proceso-de-seleccion.html (Último acceso: 2023)

Fernández De Viana González, Iñaki. Universidad de Huelva (UHU), (2023): Programación: Estructura de datos, pp. 6-49.

Gorelli Marco. GitHub (Desconocido): Pandas: Flexible and powerful data analysis / manipulation library for Python, providing labeled data structures similar to R dataframe objects, statistical functions, and much more, https://github.com/pandas-dev/pandas (Último acceso: 6/4/23)

IBM México, (Desconocido): ¿Qué es el Machine Learning?, https://www.ibm.com/mx-es/analytics/machine-learning (Último acceso: 2023)

Jupyter Labs, (Desconocido): Installing Jupyter, https://jupyter.org/install (Último acceso: 2023)

Kiziryan Mariam. Economipedia (2018): Mercado laboral, https://economipedia.com/definiciones/mercado-laboral.html (Último acceso: 1/03/21)

Marín Santos, Diego. Universidad de Huelva (UHU), (2023): Técnicas de Aprendizaje Automático: Análisis discriminante, pp. 45-51.

Marín Santos, Diego. Universidad de Huelva (UHU), (2023): Técnicas de Aprendizaje Automático: Introducción, pp. 37-39.

Marín Santos, Diego. Universidad de Huelva (UHU), (2023): Técnicas de Aprendizaje Automático: K-vecinos más cercanos, pp. 7-31, 33-35.

Marín Santos, Diego. Universidad de Huelva (UHU), (2023): Técnicas de Aprendizaje Automático: Modelos basados en árboles, pp. 35-45.

Marín Santos, Diego. Universidad de Huelva (UHU), (2023): Técnicas de Aprendizaje Automático: Teoría de la decisión, pp. 9, 19, 23-26.

Microsoft Azure, (Desconocido): Algoritmos de aprendizaje automático, https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-are-machine-learning-algorithms (Último acceso: 2023)

Oracle (2019): ¿Qué es un sistema de seguimiento de solicitantes?, https://www.oracle.com/es/human-capital-management/recruiting/what-is-applicant-tracking-system/ (Último acceso: 2023)

Pandas, PyData (NumFOCUS) (Desconocido): 'DataFrame.corr', https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.corr.html (Último acceso: 2023)

Peregrín Rubio, Antonio. Universidad de Huelva (UHU), (2023): Sistemas de Procesamiento Masivo de Datos: Big Data (Ciencia de datos), pp. 12-14, 20, 33-43.

Peregrín Rubio, Antonio. Universidad de Huelva (UHU), (2023): Sistemas de Procesamiento Masivo de Datos: Big Data (Minería de datos), pp. 6-75.

Python Software Foundation (2001): Python 3.11.3 Documentation, https://docs.python.org/3/ (Último acceso: 2023)

Raza Ahsan. Kaggle Inc. (2023): Job Placement Dataset, https://www.kaggle.com/datasets/ahsan81/job-placement-dataset?select=Job_Placement_Data.csv (Último acceso: 01/23)

Rodella Francesco. Xataka (2018): Inteligencia colectiva, https://www.xataka.com/empresas-y-economia/hay-algoritmos-ayudando-a-seleccionar-personal-busqueda-trabajo-era-algoritmo (Último acceso desconocido)

Rodó Paula. Economipedia (2019): Distribución t de Student, https://economipedia.com/definiciones/distribucion-t-de-student.html (Último acceso desconocido)

Rodó Paula. Economipedia (2019): Multicolinealidad, https://economipedia.com/definiciones/multicolinealidad.html (Último acceso desconocido)

Rohan Lal Kshetry. Kaggle Inc. (2021): Recruitment data, https://www.kaggle.com/datasets/rafunlearnhub/recruitment-data (Último acceso desconocido)

Roldán N. Paula. Economipedia (2020): Estadística, https://economipedia.com/definiciones/estadistica.html (Último acceso desconocido)

Román Díaz, Concepción. Universidad Internacional de Andalucía (UNIA), (2023): Econometría: Modelos de elección binaria, pp. 2-9, 25-28.

Rubino, Nicola. Universidad Internacional de Andalucía (UNIA), (2023): Modelos Predictivos: Introducción al análisis aleatorio, pp. 22-27, 38-43.

SAS (Desconocido): Inteligencia Artificial: Qué es IA y Por Qué Importa, https://www.sas.com/es_cl/insights/analytics/what-is-artificial-intelligence.html (Último acceso: 2023)

Scikit-Learn Consortium (2007): Supervised Learning, https://scikit-learn.org/stable/supervised_learning.html#supervised-learning (Último acceso: 2023)

Scikit-Learn (2008): Cohen Kappa Score, https://scikit-learn.org/stable/modules/generated/sklearn.metrics.cohen_kappa_score.html (Último acceso: 2023)

Scikit-Learn (Desconocido): Confusion Matrix, https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html (Último acceso: 2023)

Scikit-Learn (Desconocido): Cosine Distance, https://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.cosine_distances.html (Último acceso: 2023)

Scikit-Learn (2009): Decision Tree, https://scikit-learn.org/stable/modules/tree.html (Último acceso: 2023)

Scikit-Learn (Desconocido): Decision Tree Regressor, https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html (Último acceso: 2023)

Scikit-Learn (Desconocido): F1 Score, https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html (Último acceso desconocido)

Scikit-Learn (Desconocido): K Neighbors Classifier, https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html (Último acceso: 2023)

Scikit-Learn (Desconocido): Logistic Regression, https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html (Último acceso: 2023)

Scikit-Learn (Desconocido): Recall Score, https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html (Último acceso: 2023)

Scikit-Learn (Desconocido): Receiver Operating Characteristic (ROC), https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html (Último acceso: 2023)

StackOverFlow (2017): Variance Inflation Factor in Python, https://stackoverflow.com/questions/42658379/variance-inflation-factor-in-python (Último acceso: 1/12/21)

Statsmodels, Sphinx (Desconocido): White's Lagrange Multiplier Test for Heteroscedasticity, https://www.statsmodels.org/dev/generated/statsmodels.stats.diagnostic.het_white.html (Último acceso: 24/4/23)

Software DELSOL (Desconocido): Econometría, https://www.sdelsol.com/glosario/econometria/ (Último acceso: 2023)

Tallón Ballesteros, Antonio Javier. Universidad de Huelva (UHU), (2023): Técnicas de Aprendizaje Automático: Clases y lecciones prácticas del Máster Universitario en Economía, Finanzas y Computación (MECOFIN).

Universidad de Valladolid (UVA), (Desconocido): Distribución de una variable aleatoria, https://www5.uva.es/estadmed/probvar/d_univar/d_univar2.htm (Último acceso desconocido)

Waskom, Michael. L. PyData (NumFOCUS) (2021): Seaborn: Statistical Data Visualization, https://seaborn.pydata.org/installing.html (Último acceso desconocido)

Westreicher, Guillermo. Economipedia (2020): Experiencia laboral, https://economipedia.com/definiciones/experiencia-laboral.html (Último acceso desconocido)

Wikipedia (2019): Lenguaje de programación, https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n (Último acceso: 30/3/23)

Wikipedia (2017): Inteligencia colectiva, https://es.wikipedia.org/wiki/Inteligencia_colectiva (Último acceso: 15/2/23)

Anexo

A continuación, se muestran los links de consulta para los ficheros CSV, STATA y Python utilizados durante el TFM.

1. Acceso directo a los ficheros CSV:
https://drive.google.com/file/d/1SIoQ9rbW0YB5N9RKmVDh2rIiKqubsDWw/view?usp=share_link

2. Acceso directo a los ficheros STATA:
https://drive.google.com/file/d/1__NRmfX6ua0hG9UYaol_4o5ZimBbTPne/view?usp=share_link

3. Acceso directo a los ficheros Python:

https://drive.google.com/file/d/1CxokJXOt-m6XTIXZI7jlTnu5U4t1A-sl/view?usp=sharing