Aplicación de técnicas de inteligencia artificial para la extracción de información a partir de facturas de consumo eléctrico

Trabajo de Fin de Título

Grado en Ingeniería Informática

David Evaristo Araña Guedes

TUTORIZADO POR:
Javier Sánchez Pérez

Fecha [Julio/2024]

Agradecimientos

Quiero agradecer a todos los docentes y maestros que me han enseñado durante la carrera, sobre todo por la paciencia que tuvieron a la hora de responder mis preguntas y atenderme en las tutorías y sobre todo en las correcciones de exámenes donde me explicaban mis errores y como no volver a cometerlos. Así como agradezco la vocación que emanaban en las asignaturas que más les apasionaban por trasmitir esa pasión a pesar de la aparente dificultad en el aprendizaje.

Igualmente quiero agradecer a mi tutor Javier Sánches Pérez por tutorizarme y guiarme además de enseñarme en un campo en el que tenía muy poca experiencia, además agradezco a mi tutor y a José Daniel Hernández Sosa por su paciencia y la guía dada hacia la presentación del Trabajo Fin de Grado.

No puedo olvidarme de mi familia a quien agradezco su apoyo y alegría durante el trascurso de la carrera a pesar de los sacrificios que hayan podido hacer para ayudarme en las formas que pudieran, así como mis abuelos maternos y mi abuela paterna por su interés y animo continuo.

Agradezco a mis amigos y compañeros en la carrera por tantas horas estudiando juntos apoyándonos y resolviendo dudas unos a otros formando un grupo de estudio muy eficaz además del apoyo emocional y anímico.

Gracias a todos

David Evaristo Araña Guedes

Índice general

Índice de figuras

Índice de cuadros

Resumen

Se quiere desarrollar una aplicación que lea facturas de empresas eléctricas ya sea fotos o pdfs y se extraiga su información más relevante en forma de objeto JSON. Los datos que se busca extraer se repiten en las distintas facturas de las distintas empresas, pero todas tienen un formato distinto y no se puede aplicar algoritmos tradicionales por lo que se optará por utilizar mecanismos de aprendizaje automático que aprenda de cada tipo de factura cual es la información relevante creando así un sistema mucho más flexible.

Abstract

I want to develop an application that reads electricity company bills either photos or pdfs and extracts its most relevant information in a JSON object. The data that is sought to be extracted is repeated in the different invoices of the different companies, but all have a different format and traditional algorithms cannot be applied, so it will be chosen to use machine learning mechanisms that learn from each type of invoice what it is relevant information thus creating a much more flexible system.

Prólogo

En este trabajo de fin de grado (TFG) queremos analizar los resultados entre diferentes métodos de aprendizaje automático, como son por ejemplo Naive Bayes y Stochastic Gradient Descent, con el objetivo de encontrar cual es el mejor para crear un algoritmo con el cual extraer información de facturas del suministro eléctrico y almacenarlas digitalmente en formato Json. De esta forma podremos organizar y clasificar mejor la información extraída al estar digitalizada y poder trabajar mejor con estos datos. Por ejemplo, de esta forma se podrán crear más herramientas y algoritmos para tratar la información extraída pudiendo clasificarla para realizar análisis de costos, de consumo, comparar datos de empresas suministradoras o realizar un seguimiento de consumo en el servicio contratado.

Vivimos en un tiempo donde cada vez somos más dependientes de la electricidad pues casi todas las herramientas que usamos en nuestro día a día necesitan de la electricidad la cual también se quiere emplear más ampliamente para combatir y sustituir el uso de combustibles fósiles y la energía nuclear debido a la contaminación que producen, lo cual añaden más presión y demanda al actual mercado eléctrico y aumenta su consumo por parte de empresas y particulares. A todo esto hay que sumarle el precio de la electricidad que ha tenido un notable ascenso especialmente debido a la invasión de Ucrania por parte de Rusia que ha provocado el uso de la energía con fines geopolíticos como el de cambiar las voluntades políticas de los países más dependientes de las generación de electricidad con el gas que le compraban a Rusia, además por la subsecuente inestabilidad geopolítica propia de este tipo de conflictos complicando o bloqueando las vías anteriores de los suministros con la que se producía electricidad. Cada vez es más evidente el sector energético se esta volviendo de una importancia estratégica vital para las naciones pues sin la electricidad muchas empresas dejarían de producir y las naciones modernas se paralizarían, lo cual crea un gran caos en la economía y las finanzas en un mundo muy globalizado y competitivo.

Con todo lo anterior, este trabajo pretende ofrecer una herramienta que aunque es muy insignificante en comparación con las grandes investigaciones y proyectos que buscan promover y mejorar los sistemas eléctricos, espero que pueda generar algo de interés en la gestión y análisis en lo referente a consumo y costes energéticos, y tal vez, ayudar a promover aún más el interés por un sistema energético más robusto, limpio y barato.

David Evaristo Araña Guedes

Las Palmas de Gran Canaria, abril de 2024

Capítulo 1
Introducción

La energía eléctrica tiene un rol estratégico y fundamental en la civilización moderna. Siendo algo casi comparable a como es la sangre para un organismo humano, es la electricidad para la sociedad. Con el paso del tiempo la energía eléctrica se ha estado encareciendo al ser tan ampliamente demandada en los distintos sectores de la sociedad como el industrial y el civil, y cada vez más, debido a que la elevada demanda que iguala o supera a la oferta.

En otras industrias más tradicionales al producirse un mismo producto masivamente con el tiempo se estandariza su producción y su coste baja, y con él su precio, a no ser que la materia prima que se utilice para su producción escasee y exista una alta demanda de la misma. De forma similar ocurre con la generación de la electricidad utilizando materias primas como los recursos fósiles, derivados del petroleo, el gas, o el uranio en centrales nucleares, al también necesitar de recursos externos para conseguir el resultado deseado, en este caso la generación de electricidad.

No obstante hay una diferencia sustancial entre la electricidad y la fabricación de productos, la electricidad puede ser generada por medios diversos incluso sin necesidad de materias primas o recursos fósiles como por medio del movimiento cinético, la luz solar y demás fuerzas de la naturaleza como el viento, las mareas, o la energía geotérmica, a diferencia de la fabricación de productos que siempre consistirá en la transformación y procesamiento de recursos.

La tendencia en el aumento del coste de la electricidad contrasta enormemente con la capacidad de producirla de formas tan variadas e incluso de forma limpia algo que cada vez más está cobrando importancia en un mundo post-industrial con tanta contaminación, no obstante dependiendo del coste de aplicar estas tecnologías experimentales para obtener energía de forma limpia y renovable, este puede llegar a superar a corto y medio plazo que el de medios de generación más tradicionales y dependientes de materias primas y sus respectivas lineas de suministros.

Una vez estandarizados las tecnología y herramientas para generarla mediante tecnologías renovables, esta brecha entre los medios tradicionales y los renovables se reducirá, pues es debido a la naturaleza experimental de esta tecnología y herramientas que aún se sigue utilizando recursos fósiles, que debido a un uso estandarizado durante décadas los hace más económicos y fiables además de ser una tecnología más desarrollada por los mismos motivos.

Esto hace la generación de electricidad dependiente de cadenas de suministro y precios globales de dichas materias primas y por tanto haciéndolas susceptible a cambios drásticos de estas como ha sucedido con el conflicto Ruso-Ucraniano donde se ha usado el suministro de gas como arma por parte de Rusia, un recurso que es ampliamente usado en Europa central

y este.

Esto puede ser paliado desarrollando y estandarizando las energías renovables aprovechando la ventaja de la energía eléctrica de no depender intrínsecamente de procesos de fabricación, es decir, transformación y elaboración de materias primas y evitar el necesitar de líneas de suministro de combustible de las materias primas para generar esta electricidad. No obstante las energías renovables como son la eólica y solar entre otros, requieren de una inversión inicial mayor que los sistemas para generar energía mediantes recursos fósiles, lo cual encarece en un primer momento la factura de la luz, a pesar de esto a largo plazo el mantenimiento y los costos operativos suelen ser más bajos, abaratando así el precio de la luz. Por lo tanto la gestión del costo de la electricidad tanto a nivel cotidiano como industrial es algo bastante más relevante con un simple pago al mes.

Por lo tanto sería de mayor provecho el desarrollo de las tecnologías que permitan un abaratamiento de la electricidad tal como lo es el agua en nuestra sociedad, ya que a diferencia del agua, podemos generar electricidad con tecnologías limpias. Estoy es vital en el estado actual de nuestra civilización ya que es imposible desarrollarse sin energía y podemos hablar de pobreza energética cuando los costes de la misma la hace inviable para algunos sectores de la población.

En con este contexto podemos deducir la importancia de la factura de la luz habiendo explicado ya la importancia de la electricidad y los diversos motivos que pueden afectar a su precio por ello lo que se quiere lograr con este proyecto es crear un sistema automático que sea capaz de digitalizar facturas, en concreto, facturas de empresas de la luz extrayendo de ellas el contenido más relevante como cobros, energía contratada y demás datos, generando un objeto JSON el cual podamos fácilmente almacenar y gestionar dicha información.

Con la información más organizada podemos crear perfiles de consumo y dará más control a los clientes debido que estarán mejor informados de sus necesidades y su consumo, pudiendo optimizar sus requerimientos en caso de desearlo.

Será necesaria una gran base de datos de facturas con sus campos etiquetados (para emplear aprendizaje supervisado) y se usará algoritmos de aprendizaje automático como los Support Vector Machines (SVM) para lograr la correcta extracción de la información de los documentos tratados. Los algoritmos de aprendizaje automático supervisado, son algoritmos entrenados con datos etiquetados y sin etiquetar, los etiquetados son usados para entrenar el algoritmo en que datos debe extraer, las muestras ya vienen marcadas y luego este entrenamiento es probado en las muestras sin etiquetar donde el algoritmo tiene que decidir qué datos pertenecen a qué campos en base al entrenamiento previo.

Una opción para solucionar esta cuestión habría sido usando expresiones regulares, pero para ello hace falta un experto [14] en la realización de expresiones regulares para poder desarrollar nuevas o modificar las existentes cada vez que se actualice o exista una nueva factura donde, por razones de diseño, mueven los campos de información relevante en el documento con respecto a versiones anteriores o de otros distribuidores. Es decir, existiría la necesidad de contar con alguien con un gran conocimiento en manipulación de cadenas de texto alfanuméricas y habría necesariamente un algoritmo para cada tipo de factura y empresa, ya que los datos que se busca extraer son comunes para las facturas y se encuentran

en todas ellas, pero dichas facturas se diferencian significativamente según el año o la empresa de la que dicha factura proceda, pues las propias compañías pueden cambiar dichas facturas a lo largo del tiempo, ya sea para mejorar su legibilidad u ofrecer una presentación más moderna y refinada.

También pueden surgir nuevas distribuidoras, lo que hace las facturas en cuestión muy susceptibles al cambio, de manera que la opción de usar expresiones regulares personalizadas para cada factura de cada tipo, año y empresa eléctrica se vuelve inviable.

La opción propuesta es la de usar técnicas de aprendizaje automático de manera que solo sea necesario entrenar dicho algoritmo para que pueda extraer la información relevante de las distintas facturas dividiendo la facturas en campos etiquetados. De esta forma al producirse cambios en la forma de las facturas solo será necesario volver a entrenar el algoritmo para la factura nueva, pues estará preparado para encontrar dichos campos y extraerá automáticamente los datos pertinentes.

Las facturas serán usadas como base de datos para entrenar los algoritmos de aprendizaje automático. Para esto, etiquetaremos los campos relevantes de la factura, así como los clasificadores que usemos, es decir, los algoritmos de aprendizaje automático se entrenarán en detectar qué datos se encuentran en esos campos. Con el suficiente número de muestras serán capaces de extraer los datos de una factura real sin etiquetar.

En este proyecto, el autor utiliza y ha ayudado a refinar una amplia cantidad de muestras generadas con ayuda del tutor para entrenar 3 algoritmos de aprendizaje automático, comparándolos para encontrar el más adecuado. También se desarrolló un método para crear facturas reales con datos aleatorios pero realistas usando plantillas de facturas reales.

El presente Trabajo de Fin de Grado está organizado por capítulos donde podemos encontrar en el primero y actual la introducción, motivación, organización del documento y los objetivos del proyecto. En el capítulo 2 analizamos el estado del arte y realizamos una explicación sobre algoritmos de aprendizaje automático que se analizaron para usarlos en el proyecto y temas relacionados. Además, realizaremos una revisión bibliográfica para el proyecto y una extensa explicación sobre un artículo en el apartado de revisión bibliográfica que fue especialmente inspirador a la hora de entender conceptos básicos del proyecto que se estaba planteando.

En el capítulo 3 describimos la planificación del proyecto y enumeramos las herramientas que se emplearon en el mismo. En los capítulos siguientes entramos en la materia del propio proyecto donde, en el capítulo 4, desglosamos todo el proceso de anotación de las facturas y su generación a la hora de tener muestras suficientes, que usaremos en el capítulo 5 además de explicar como se usan para el entrenamiento de los clasificadores, cuáles son y qué características tienen, aparte de una revisión del código más relevante.

En los capítulos 6 y 7 veremos los resultados finales de los clasificadores donde los compararemos y veremos las conclusiones del proyecto, los objetivos alcanzados, los problemas encontrados y qué más trabajo se puede hacer. En el anexo A veremos las competencias especificas de este proyecto basadas en las que se adquieren durante la carrera.

1.1. Motivación

Me sentí interesado en aprender acerca de las técnicas de aprendizaje automático y de poder aplicarlo en un extractor de datos de facturas de empresas distribuidoras de energía. Además me gusta las capacidades de generación de energía y siempre me extrañó el precio elevado que alcanzaba en algunos casos la luz cuando hay tantas formas de generar energía, incluso sin el uso de materias primas como recursos fósiles.

Cuando comienzo a descubrir las diversas formas en las que es posible generar energía, solo puedo pensar en lo inverosímil de situaciones como la pobreza energética, ya que se puede obtener de tantas formas la electricidad, que el hecho de que no se abarate por exceso de oferta resulta decepcionante y se vuelve evidente una falta de inversión en este ámbito por parte de las naciones desarrolladas.

Además en tiempos recientes se ha usado la energía como arma en el conflicto ruso-ucraniano por parte de Rusia como una estrategia para evitar que países de Europa, especialmente del centro y este que dependen mucho del gas exportado de Rusia, se opusieran a la invasión a gran escala de Rusia aunque pudieron evitar esa situación cambiando rápidamente los proveedores de gas a otras naciones.

Esto es debido precisamente a la falta de diversificación de las fuentes de generación de energía en estas naciones, haciéndolas vulnerables a estos cambios geopolíticos que por otro lado eran altamente predecibles. Ya desde 2014, cuando Rusia invadió a menor escala Ucrania se podía prever una situación como esta, lo que debería haber servido como aviso a los países europeos que la dependencia energética con el país invasor era un error. Finalmente no se hicieron esfuerzos en diversificar la generación de energía eléctrica para evitar el chantaje energético de Rusia.

Este tipo de sucesos generan la duda de si realmente hay un interés de las naciones en el desarrollo de energía limpias y sobre todo renovables que no dependan de materias primas y por tanto de cadenas de importación y extracción. Aunque en comparación con todo lo expuesto, este TFT es insignificante en la aportación que me gustaría hacer para ayudar (lo que me fuera posible) en universalizar el acceso a la energía eléctrica.

1.2. Organización del documento

Los capítulos siguientes del documento constarán del siguiente contenido:

- Capítulo 1: En este capítulo introducimos el contexto y la visión general del proyecto, las motivaciones para desarrollarlo y sus objetivos a cumplir.
- Capítulo 2: En este capítulo se menciona el estado del arte y, por lo tanto, veremos distintos algoritmos de aprendizaje automático y cuál es su funcionamiento. También veremos la revisión bibliográfica y dentro de esta una amplia explicación sobre al articulo mencionado en la introducción[14] que sirvió de gran inspiración y ayuda para introducirse en el mundo del aprendizaje automático.
- Capítulo 3: En este capítulo se detallan las herramientas utilizadas y la planificación del proyecto.
- Capítulo 4: En este capítulo se explican las clases que tendrá que extraer el clasificador y el proceso para anotar estas clases en las facturas, así como los dos planteamientos, uno mediante facturas en formato SpaCy y otro usando plantillas de facturas reales, para generar facturas automáticamente con contenido realista y coherente para así imitar facturas reales, generando así una gran base de muestras con las que entrenar los algoritmos de extracción de la información.
- Capítulo 5: En este capítulo se explica el proceso con el que se extrae la información de las facturas, las herramientas utilizadas explicadas en más detalle, cuáles son los métodos de aprendizaje automático utilizados para los clasificadores explicados siguiendo la documentación de Scikit Learn y cual es su funcionamiento a nivel de código.
- Capítulo 6: En este capítulo se comparan los resultados de los distintos clasificadores, cuáles son los tipos de errores que se tienen en cuenta y se muestran tablas de errores de los clasificadores entrenados con distintas cantidades de muestras y los mapas de calor que dan una ayuda adicional a la hora de llegar a conclusiones.
- Capítulo 7: Se mencionan los objetivos alcanzados, los problemas encontrados y el trabajo futuro. En este capítulo vemos qué se ha conseguido desarrollar exitosamente en el proyecto, es decir, los objetivos alcanzados, en Trabajo Futuro7.3 proponemos mejoras al proyecto actual y luego resaltamos los problemas que se han encontrado en la realización del proyecto.
- Anexo A: En este anexo se detallan las competencias especificas relacionadas con las competencias que se adquieren en el grado de Ingeniería Informática y sus especializaciones.

1.3. Objetivos del proyecto

En este proyecto de Trabajo de Fin de Grado se han marcado una serie de objetivos para lograr el fin de extraer información relevante de facturas eléctricas y presentar los resultados en formato JSON para un uso mas sencillo y manejable de los datos.

El objetivo con respecto a las facturas consiste en anotar qué clases o campos dentro de una factura eléctrica son relevantes para el usuario final y generar una base de datos de muestras con las que entrenar a los clasificadores. Esta información deber ser realista y razonable para garantizar el máximo rendimiento de los mismos.

Los objetivos con respecto a los clasificadores constan de preparar una serie de métodos de aprendizaje automático los cuales son las Máquinas de Soporte de Vectores, Bayes "Inocente" y Descenso de gradiente Estocástico o por sus nombres y siglas en inglés *Support Vector Machine(SVM), Naive Bayes(NB)* y *Stochastic Gradient Descent(SGD)*, para ser entrenados y ajustados con el fin de extraer una serie de datos de facturas eléctricas que serán previamente anotadas y formarán las clases en las que se dividirá la información extraída. Se elegirá el mejor clasificador de entre los 3 mediante la comparación de resultados finales.

El clasificador elegido por sus resultados deberá estimar a qué clase pertenece la información que se le presente en forma de factura eléctrica para presentarla en formato JSON al usuario final.

Capítulo 2
Estado del Arte

En la actualidad existen varios proyectos de extracción de información de textos usando tanto redes neuronales como algoritmos de aprendizaje automático, entre otros. El uso más frecuente suele ser las traducciones de textos, a varios idiomas donde se necesita obtener un contexto para la información extraída, pues un mismo dato no significa lo mismo en un contexto que en otro, por ejemplo, en el caso que nos ocupa en la factura de la luz existen campos como la potencia contratada y el importe de los impuestos, en ambos casos se representan con una cifra pero son de campos completamente distintos y una cifra se mide en kWh y la otra en euros.

El objetivo es crear un software para extraer la información de los campos relevantes de una factura eléctrica como pueden ser potencia consumida, potencia contratada, etc. y generar un objeto JSON con los pares de los campos de la factura con los datos que ha extraído el software. Para este proyecto usaremos algoritmos de aprendizaje automático supervisados, algunos de los más comunes son las Máquinas de Vectores de Soporte o en inglés *Support Vector Machine (SVM)*, aunque también podemos encontrar en este ámbito los Árboles de Decisión o *Decision Tree(DS)* y la Regresión Lineal o *Linear Regression*.

En el caso del presente proyecto se necesita que el algoritmo "entienda" el contexto de la información de los campos claves de la factura donde debe extraer los datos para clasificarlos adecuadamente, como decíamos en el ejemplo anterior, una cifra numérica puede indicar tanto la potencia como el costo en euros en la factura y el contexto puede ser extraído en este caso gracias a la unidades de la cifra numérica, por ejemplo del campo "Potencia Consumida" debería extraer algo como 289 kWh. Una vez el algoritmo toma la decisión de etiquetar los datos extraídos en el campo estimado creará un objeto JSON con los pares campo-dato, por ejemplo Distribuidora: [Iberdrola].

2.1. Algoritmos y técnicas de aprendizaje automático

La extracción de datos de texto mediante técnicas como SVM, Regresión Lineal o Naive Bayes es recurrente en el ámbito de la inteligencia artificial, más concretamente en el del aprendizaje automático y el procesamiento del lenguaje natural, como es el caso que nos ocupa al tratar el proyecto de la extracción de información de facturas de distribuidores de energía eléctrica. La extracción de información de facturas de luz es un proceso útil en la gestión eficiente de la energía y la monitorización del consumo eléctrico. Existen trabajos donde la utilización de técnicas avanzadas de aprendizaje automático, como las Máquinas de

Vectores de Soporte (SVM) o Descenso de Gradiente Estocástico (SGD), ha ayudado a la manera en que se aborda este problema.

Existen diferentes formas de abarcar problemas con estas técnicas. Se han propuesto diferentes formas y métodos para mejorar la extracción de datos de texto por ejemplo:

2.1.1. Sequential Minimal Optimization (SMO)

Es importante abordar la teoría básica de optimización y las Support Vector Machine (SVM), junto con el algoritmo Sequential Minimal Optimization (SMO). El algoritmo SMO es ampliamente utilizado para entrenar el SVM y encontrar el hiperplano óptimo de separación. Es un algoritmo utilizado para entrenar Máquinas de Vectores de Soporte (SVM) de manera eficiente. Este algoritmo fue propuesto por John Platt en 1998[17] y se ha convertido en una técnica ampliamente utilizada para optimizar el proceso de entrenamiento de SVM. El SMO se caracteriza por descomponer el problema de optimización de SVM en subproblemas más pequeños y más fáciles de resolver. En lugar de abordar el problema de entrenamiento de SVM en su totalidad, el SMO se centra en optimizar pares de coeficientes de Lagrange de manera secuencial.

2.1.2. Clasificación de textos mediante algoritmos de Machine Learning: Support Vector Machine Lineal

El SVM es un algoritmo de aprendizaje automático supervisado que se utiliza para la clasificación y regresión de datos. Su objetivo principal es encontrar un hiperplano óptimo que separe los datos en diferentes clases[8], en nuestro caso son los distintos campos clave que podemos encontrar en una factura de la luz. En el contexto de la extracción de datos de texto, el SVM se utiliza para clasificar y extraer información relevante de documentos no estructurados, no obstante, en nuestro caso los documentos sí están estructurados pero como se ha explicado anteriormente son susceptibles al cambio. El uso de SVM en este contexto permite la creación de modelos precisos y eficientes para identificar y extraer información relevante de las facturas de la luz, como el consumo de energía, los costos asociados, los períodos de facturación y otros datos importantes para análisis y toma de decisiones.

Uno de los enfoques más comunes en la extracción de datos de texto mediante SVM es el uso de kernels polinomiales. Estos kernels permiten mapear los datos de texto a un espacio de características (los campos relevantes de las facturas) de mayor dimensión, lo que facilita la separación de las diferentes clases. Además, los kernels polinomiales son especialmente útiles cuando se trabaja con vectores de datos dispersos, como es el caso de la categorización de texto.

En este caso exploraremos el kernel Lineal que se diferencia principalmente en su implementación. Este kernel es más flexible al configurar los parámetros de función de pérdida ("*loss*") y de penalización ("*penalty*"). La función de pérdida permite adaptar el algoritmo a diferentes tipos de conjuntos de datos y problemas de clasificación, mientras que la función

de penalización ayuda a controlar el sobreajuste (*overfitting*) al penalizar los coeficientes del modelo. Esto es crucial para evitar que el modelo se ajuste demasiado a los datos de entrenamiento y pierda capacidad predictiva en datos nuevos. La elección entre normas L1 y L2 (o diferentes tipos de penalización) puede influir en cómo se manejan las características.

2.1.3. Clasificación de textos mediante algoritmos de Machine Learning: Naive Bayes

Es relevante mencionar otros algoritmos de clasificación de textos, como el clasificador Naive Bayes. El algoritmo Naive Bayes es un método de aprendizaje automático supervisado basado en el teorema de Bayes, que se utiliza comúnmente en problemas de clasificación y análisis predictivo. Es conocido por su simplicidad y eficiencia.

La premisa fundamental detrás del algoritmo Naive Bayes es la suposición de independencia condicional entre las características, es decir, que cada característica contribuye de manera independiente a la probabilidad de pertenencia a una determinada clase. Aunque esta suposición puede no ser realista en todos los casos, en la práctica suele funcionar bien y permite un cálculo más sencillo de las probabilidades.

El algoritmo se basa en el teorema de Bayes, que establece cómo se puede calcular la probabilidad de que ocurra un evento A dado que ha ocurrido un evento B. En el contexto de la clasificación, se busca determinar la probabilidad de que un nuevo dato pertenezca a una clase específica dadas sus características observadas.

El proceso de entrenamiento del modelo Naive Bayes implica el cálculo de las probabilidades de cada clase, así como las probabilidades condicionales de las características dadas las clases. Estas probabilidades se utilizan luego para predecir la clase más probable de un nuevo dato en función de sus características. Estas características pueden ser, por ejemplo, que el dato extraído tenga formato de fecha por lo tanto, el algoritmo tendrá más fácil saber que pertenece una clase fecha, pero debe predecir si sería de inicio de contrato, o del fin, solo de la factura, etc.

A pesar de su simplicidad y suposiciones simplificadas, el algoritmo Naive Bayes ha demostrado ser efectivo en una amplia gama de aplicaciones, incluyendo la clasificación de texto, detección de spam, diagnóstico médico y análisis de sentimientos. Su rapidez de entrenamiento y predicción lo hacen especialmente útil en entornos donde se requiere una alta eficiencia computacional.

El algoritmo Naive Bayes se destaca por su capacidad para manejar conjuntos de datos grandes y complejos, su facilidad de implementación y su buen desempeño en situaciones donde la suposición de independencia condicional es razonable. Su versatilidad y robustez lo convierten en una herramienta valiosa en el campo del aprendizaje automático y la minería de datos.[20]

2.1.4. Clasificación de textos mediante algoritmos de Machine Learning: Stochastic Gradient Descent

El algoritmo de Descenso de Gradiente Estocástico o *Stochastic Gradient Descent (SGD)* es una variante del Descenso de Gradiente que se utiliza comúnmente en el entrenamiento de modelos de aprendizaje automático.

A diferencia del descenso de gradiente no estocástico, que calcula el gradiente de la función de pérdida utilizando el conjunto de datos completo en cada iteración, el SGD calcula el gradiente de forma estocástica, es decir, utilizando una sola muestra o un pequeño lote (mini-batch) de datos en cada paso.

El descenso de gradiente estocástico (SGD) es un enfoque simple pero muy eficiente para ajustar clasificadores y regresores lineales bajo funciones de pérdida convexas, como Support Vector Machines (lineales) y regresión logística. Aunque SGD existe en la comunidad de aprendizaje automático desde hace mucho tiempo, recientemente ha recibido una cantidad considerable de atención en el contexto del aprendizaje a gran escala. SGD se ha aplicado con éxito a problemas de aprendizaje automático dispersos y a gran escala que a menudo se encuentran en la clasificación de textos y el procesamiento del lenguaje natural.[9]

El SGD es especialmente útil cuando se trabaja con conjuntos de datos grandes, ya que permite actualizar los parámetros del modelo de manera más eficiente al procesar solo una fracción de los datos en cada iteración. Esto puede acelerar significativamente el proceso de entrenamiento y hacer que el algoritmo sea más escalable para conjuntos de datos masivos.

En el contexto de SVM, el SGD se puede aplicar para optimizar los hiperparámetros[9] del modelo, como los pesos asociados a las características y los términos de regularización. Al utilizar el SGD, el algoritmo busca encontrar los parámetros que minimizan la función de costo, lo que se traduce en la búsqueda del hiperplano óptimo de separación entre las clases.

El SGD es particularmente útil en problemas de clasificación con grandes volúmenes de datos, donde el tiempo de entrenamiento es una consideración significativa. Al actualizar los parámetros de forma estocástica, el algoritmo puede converger más rápidamente a una solución satisfactoria, aunque puede ser más susceptible a variaciones en la convergencia debido a la estocasticidad en el cálculo del gradiente.

2.1.5. Clasificación de textos mediante algoritmos de Machine Learning: Linear Model

En este caso, nos enfocaremos en el clasificador SVM Linear Model el cual podemos encontrar en Scikit-Learn [3], una biblioteca de aprendizaje automático en Python ampliamente utilizada.

Es un tipo de modelo de aprendizaje automático que busca establecer una relación lineal entre las características de entrada y la variable objetivo. En el caso específico del clasificador

SVM, se trata de un modelo lineal que busca encontrar el hiperplano óptimo que mejor separe las diferentes clases de datos en un espacio de características multidimensional.

En un clasificador, se busca encontrar el hiperplano de separación que maximice el margen entre las clases de datos. Este hiperplano es la línea que mejor divide los datos en dos clases distintas. Aunque el SVM es capaz de realizar clasificaciones no lineales a través de trucos de kernel, en su forma más básica, es un clasificador lineal que separa las clases mediante un hiperplano en un espacio de características. En un Clasificador lineal, la función de decisión es lineal, lo que significa que la clasificación de un nuevo punto se realiza calculando en qué lado del hiperplano se encuentra.

En resumen, un clasificador como modelo lineal utiliza un enfoque lineal para separar y clasificar datos en un espacio multidimensional, con el objetivo de encontrar el hiperplano óptimo que maximice el margen de separación entre las clases.

Como explicábamos, un clasificador linear model es un tipo de modelo que busca establecer una relación lineal entre las características de entrada y la variable objetivo. Estos modelos son implementados a través de diferentes algoritmos que permiten realizar tareas de clasificación. En la documentación de scikit klearn comentan que si $\hat{y}$ es el valor a predecir tenemos la siguiente notación matemática.

$$\hat{y}(w, x) = w_0 + w_1 x_1 + ... + w_p x_p \tag{2.1}$$

A continuación: scikit learn presenta variable como coef_ $w = (w_1, ..., w_p)$ e intercept_ como w_0

Scikit-Learn ofrece una variedad de algoritmos de clasificación lineal que pueden adaptarse a diferentes tipos de conjuntos de datos y problemas.

Mínimos cuadrados ordinarios: LinearRegression se ajusta un modelo lineal con coeficientes $w = (w_1, ..., w_p)$ a minimizar la suma residual de cuadrados entre los objetivos observados en el conjunto de datos y los objetivos predichos por la aproximación lineal. Matemáticamente resuelve un problema de la forma:

$$\min_{w} ||Xw - y||_2^2 \tag{2.2}$$

La documentación contigua diciendo que LinearRegression usará en su metodo fit las matrices X e Y, y almacenará los coeficientes W del modelo lineal en coef_:

```
from sklearn import linear_model
reg = linear_model.LinearRegression()
reg.fit([[0, 0], [1, 1], [2, 2]], [0, 1, 2])
reg.coef_
```

Las estimaciones de coeficientes para los mínimos cuadrados ordinarios se basan en la independencia de las características. Cuando las características están correlacionadas y las columnas de la matriz diseñada X tienen una dependencia aproximadamente lineal, la matriz diseñada se vuelve casi singular y, como resultado, la estimación de mínimos cuadrados se vuelve altamente sensible a errores aleatorios en el objetivo observado, lo que produce una

gran variabilidad. Esta situación de multicolinealidad puede surgir, por ejemplo, cuando se recopilan datos sin un diseño experimental terminan por comentar en scikit learn.

Mínimos cuadrados no negativos

Es posible restringir todos los coeficientes para que no sean negativos, lo que puede ser útil cuando representan algunas cantidades físicas o naturalmente no negativas (por ejemplo, recuentos de frecuencia o precios de bienes). LinearRegression acepta un parámetro booleano positivo cuando se establece True y entonces aplican mínimos cuadrados no negativos.

Complejidad de mínimos cuadrados ordinarios:

Sobre los mínimos cuadrados aportan la siguiente solución: La solución de mínimos cuadrados se calcula utilizando la descomposición en valores singulares de X. Si X es una matriz de forma, este método tiene un costo de (n_samples, n_features) $O(n_{\text{samples}} n_{\text{features}}^2)$ asumiendo que $n_{\text{samples}} \geq n_{\text{features}}$

Para utilizar un clasificador linear model en Scikit-Learn, es necesario seguir una serie de pasos estándar que incluyen la importación del modelo, el ajuste a los datos de entrenamiento y la evaluación del rendimiento del modelo. Scikit-Learn proporciona una documentación detallada y ejemplos prácticos para guiar a los usuarios en este proceso como hemos mostrado.

2.1.6. Decision Trees

Los árboles de decisión (DT) son un método de aprendizaje supervisado no paramétrico que se utiliza para la clasificación y la regresión . El objetivo es crear un modelo que prediga el valor de una variable objetivo mediante el aprendizaje de reglas de decisión simples inferidas a partir de las características de los datos. Un árbol puede considerarse como una aproximación constante por partes.

Por ejemplo, en el ejemplo siguiente 2.1, los árboles de decisión aprenden de los datos para aproximarse a una curva sinusoidal con un conjunto de reglas de decisión if-then-else. Cuanto más profundo sea el árbol, más complejas serán las reglas de decisión y más adecuado será el modelo.[4]

En la documentación de Scikit Learn podemos encontrar algunas ventajas de los árboles de decisión que son las siguientes:

- Fácil de entender e interpretar. Los árboles se pueden visualizar.
- Requiere poca preparación de datos. Otras técnicas a menudo requieren la normalización de datos, es necesario crear variables ficticias y eliminar valores en blanco. Algunas combinaciones de árboles y algoritmos admiten valores faltantes.
- El costo de usar el árbol (es decir, predecir datos) es logarítmico en el número de puntos de datos utilizados para entrenar el árbol.
- Puede manejar datos tanto numéricos como categóricos. Sin embargo, la implementación de scikit learn no admite variables categóricas por ahora. Otras técnicas suelen

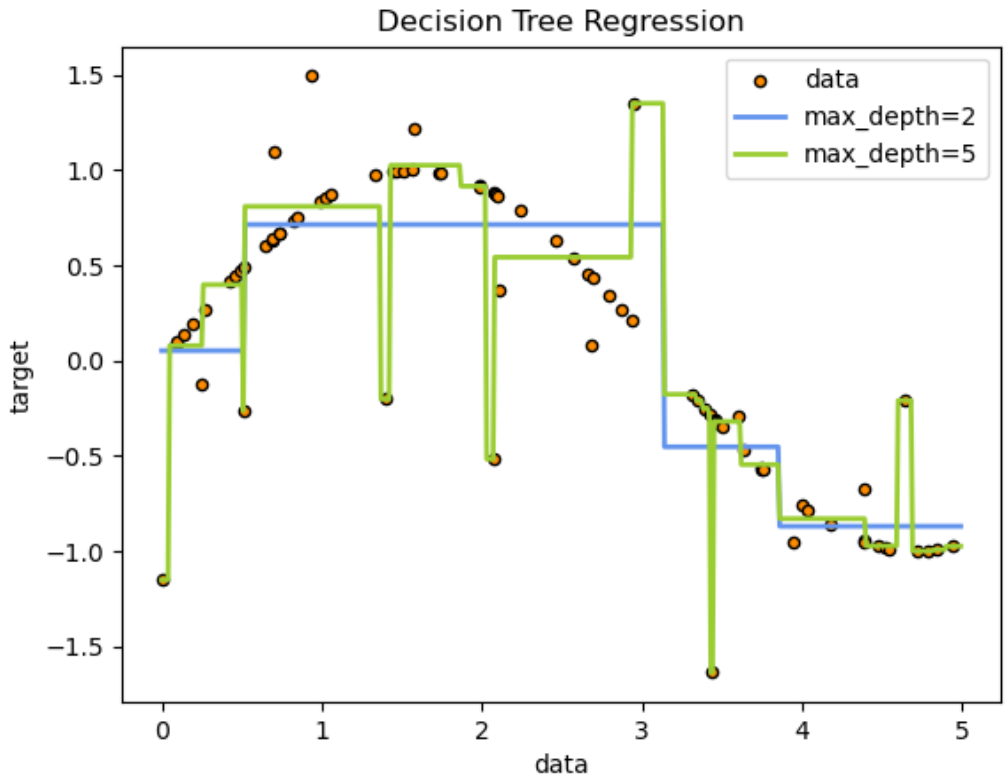

Ilustración 2.1: Gráfica de la documentación de árbol de decisión de sckit learn

estar especializadas en el análisis de conjuntos de datos que tienen solo un tipo de variable.

- Capaz de manejar problemas de múltiples salidas.
- Utiliza un modelo de caja blanca. Si una situación dada es observable en un modelo, la explicación de la condición se explica fácilmente mediante lógica booleana. Por el contrario, en un modelo de caja negra (por ejemplo, en una red neuronal artificial), los resultados pueden ser más difíciles de interpretar.
- Es posible validar un modelo mediante pruebas estadísticas, lo que permite dar cuenta de la confiabilidad del modelo.
- Funciona bien incluso si sus supuestos son violados en cierta medida por el modelo real a partir del cual se generaron los datos.

También podemos encontrar varias desventajas de los árboles de decisión que incluyen:

- Los aprendices de árboles de decisión pueden crear árboles demasiado complejos que no generalizan bien los datos. Esto se denomina sobreajuste (overfitting). Para evitar este problema, son necesarios mecanismos como la poda, la configuración del número mínimo de muestras necesarias en un nodo de hoja o la configuración de la profundidad máxima del árbol.

- Los árboles de decisión pueden ser inestables porque pequeñas variaciones en los datos pueden generar un árbol completamente diferente. Este problema se mitiga utilizando árboles de decisión dentro de un conjunto.
- Las predicciones de los árboles de decisión no son uniformes ni continuas, sino aproximaciones constantes por partes, como se ve en la figura anterior.2.1 Por lo tanto, no son buenos para la extrapolación.
- Se sabe que el problema con el aprendizaje de un árbol de decisiones óptimo es NP-complete en varios aspectos óptimos e incluso para conceptos simples. En consecuencia, los algoritmos prácticos de entrenamiento de árboles de decisiones se basan en algoritmos heurísticos como el algoritmo greedy, en el que se toman decisiones óptimas a nivel local en cada nodo. Dichos algoritmos no pueden garantizar que se obtenga el árbol de decisiones globalmente óptimo. Esto se puede mitigar entrenando varios árboles en un conjunto de entrenamiento, en el que las características y las muestras se muestrean aleatoriamente con reemplazo.
- Hay conceptos que son difíciles de aprender porque los árboles de decisión no los expresan fácilmente, como los problemas de XOR, paridad o multiplexores.
- Los entrenamientos de árboles de decisión crean árboles sesgados si algunas clases predominan. Por lo tanto, se recomienda equilibrar el conjunto de datos antes de ajustarlo al árbol de decisión.

Por lo tanto los arboles de decisión (DTs) son algoritmos flexibles y se pueden configurar de forma sencilla donde se deja al algoritmo encontrar la mejor solución sin grandes y complejas configuraciones. No obstante, en casos complejos se vuelven casi inviables debido a su forma de ajustarse y requerir de métodos de poda con los que se puede evitar por accidente las soluciones óptimas.

2.1.7. Utilización de Grandes Conjuntos de Datos

En la investigación actual, se ha observado un aumento en el uso de grandes conjuntos de datos para entrenar y evaluar los modelos de extracción de datos de texto mediante algoritmos de aprendizaje automático.

Por ejemplo anteriormente se evitaba el uso de SVM, para grandes conjuntos de datos debido al costo computacional y por tanto al tiempo de entrenamiento que se tenia que emplear. No obstante, en la actualidad se han mejorado estos factores volviendo los SVM (y otros modelos) viables para conjuntos significativamente grandes de datos.[11][18]

Estos conjuntos de datos contienen una gran cantidad de documentos no estructurados y permiten mejorar la precisión y el rendimiento de los modelos. Los grandes conjuntos de datos pueden conducir a un mejor rendimiento del modelo en términos de precisión, recall (muestras clasificadas positivas correctamente) y otras métricas de evaluación. Al tener más datos de entrenamiento disponibles, el modelo puede aprender con mayor precisión los patrones relevantes para la extracción de datos de texto aunque también existe el riesgo de

overfitting, es decir, de sobreentrenar el clasificador o modelos de forma que en las pruebas el resultado es óptimo pero en cuanto se ponen a prueba usando muestras que no estén en el conjunto de pruebas empieza a fallar.

2.1.8. Aplicaciones y Casos de Uso

En la extracción de datos de texto mediante algoritmos de aprendizaje automático, es importante mencionar las diversas aplicaciones y casos de uso de esta técnica. Algunos ejemplos incluyen la extracción de información de documentos legales, la clasificación de opiniones en redes sociales y la detección de spam en correos electrónicos. También se incluyen otras aplicaciones como el geo-sondeo, evaluación del potencial de licuefacción sísmica, reconocimiento de voz, detección facial y clasificación de expresiones.[10]

2.2. Revisión Bibliográfica

Primeramente se optó por estudiar otros artículos, donde uno en concreto me sirvió mucho para comprender los conceptos básicos de lo que se pretendía hacer, ya que era similar en problema al proyecto que nos ocupa. [14].

Se uso la documentación de la pagina Scikit Learn [16] para la comprensión de conceptos teóricos del funcionamiento de la librerías, así como para consultar las herramientas de las que disponía y como configurarlas como las distintas clases y métodos que se podían utilizar además de ejemplos explicados.[9][8]

También se usaron algunas webs especializadas y otros artículos para obtener información básica de algunos conceptos.[17][20][18]

Como trabajo previo en la extracción de datos mediante aprendizaje automático tenemos un artículo[14] cuyos autores se enfrentan a un problema similar al del actual proyecto, que no es otra que extraer información de un texto en cuestión para clasificarlos en campos o categorías específicos.

Este artículo ha resultado de gran utilidad a la hora de comprender el modo de operar para lograr el objetivo del problema se quiere abarcar. En el trabajo de este artículo que mencionamos se pretendía extraer metadatos de las cabeceras de artículos de investigación, donde afirman que con SVM obtienen mejor el resultado que con otros métodos de aprendizaje automático en esta misma tarea. En el propio resumen comentan cómo su método empieza con la clasificación de cada línea de la cabecera en una clase o más de un total de 15 clases.

El problema que tratan de solucionar es que los buscadores de estos artículos de investigación suelen usar pocos o ningún tipo de metadatos para realizar la búsqueda.

Las formas de solucionar este problema según sus propuestas son tres, que además consideran las más populares, las cuales incluyen los ya mencionados métodos de aprendizaje automático, las expresiones regulares y los métodos de análisis basados en reglas.

Según la información que hemos podido encontrar "los modelos basados en reglas constituyen uno de los paradigmas más populares para la modelización del comportamiento y el control de sistemas complejos, donde es necesaria una representación de conocimiento simbólica, cualitativa, basada en relaciones. Básicamente, cualquier sistema basado en reglas está compuesto por un conjunto de reglas que definen acciones o conclusiones si se cumplen determinadas condiciones, y un mecanismo de control de inferencia que determina la manera y orden de ejecución de las reglas. Los sistemas basados en reglas son probablemente la herramienta más general y más popular de la Inteligencia Artificial. En el caso de sistemas de supervisión pueden aplicarse a la monitorización y evaluación de situaciones, detección y diagnóstico de fallos y comportamientos no deseados, control jerarquizado, razonamiento sobre el estado y las características del sistema, apoyo a decisiones, etc."[1]

Por otro lado, proponen el uso de expresiones regulares que como "un modelo o una forma de comparar con una cadena de caracteres. Esta comparación es conocida con el nombre de pattern matching o reconocimiento de patrones, permite identificar las ocurrencias del modelo en los datos tratados. La utilización principal de las expresiones regulares en los lenguajes de programación consiste en la identificación de cadenas de caracteres para la búsqueda, modificación y extracción de palabras clave. De esta manera se pueden dividir las expresiones regulares en varios tipos que son: expresiones regulares de sustitución, expresiones regulares de comparación y expresiones regulares de traducción".[2]

Es precisamente en este artículo donde se menciona que las expresiones regulares no son óptimas para extraer la información en estos casos a pesar de poderse aplicar inmediatamente al no necesitar entrenamiento del algoritmo como sí ocurre con los métodos de aprendizaje automático, ya que las expresiones regulares dependen del dominio al que se les va a aplicar y también de un experto en las expresiones regulares, teniendo que desarrollar unos patrones de expresiones regulares cada vez que cambien el dominio o estructura del documento en cuestión o aparezca uno nuevo, en el caso del presente TFG los diferentes formatos de facturas de la luz como ya se ha explicado previamente, mientras que en la introducción consideran los métodos de aprendizaje automático como más robustos y adaptables. En el algoritmo usado especifican que se usa información contextual para estimar la clase a la que pertenece la información extraída en este caso, revisando palabra por palabra su antecesora y predecesora.

Comentan distintos tipos de técnivas de aprendizaje automático como SVM, aprendizaje simbólico, modelo oculto de Markov *(Hidden Markov Models o HMMs)*, inducción gramatica *(grammar induction)* e inductive logic programming, donde evalúan que al clasificar las clases directamente indexales los SVM son mejores que los HMMs. Las clases directamente indéxales se pueden entender como nombres, afiliaciones, direcciones y el título del artículo, que aparecen en mayor medida en las cabeceras de los artículos (desde el principio del artículo hasta la primera sección, generalmente la introducción).

Los chunks (que viene del inglés trozos o fragmentos) los definen como varias palabras consecutivas que pertenecen a una misma clase, por ejemplo los nombres y apellidos de los autores. El problema a tratar es el límite óptimo de estos chunks, como habíamos explicado los chunks son palabras consecutivas que pertenecen a la misma clase, pero en el caso de los nombre de los autores si tenemos la línea "Jack Johnson John Jackson" podríamos tener un autor o dos autores "Jack Johnson" y "John Jackson" u otras variaciones.

En este artículo dividen el problema en dos pasos, la clasificación de cada línea y la identificación de chunks multiclase y multiautor, donde la precisión en la clasificación de la línea es crítica, pues de ella depende el rendimiento de la identificación de los chunks. Para su método de extracción, en el clasificador eligieron el núcleo gaussiano (Gaussian Kernel) y usaron como base el software SVM_light, configuraron el parámetro gamma(-g) a 0.1 dejando el resto de parámetros por defecto.

Al ser un problema multiclase la estrategia que siguen para el clasificador es de una clase contra las demás, donde se trata una clase como positiva (la que se esta analizando) y las demás como negativas. Para la clasificación de a qué clases pertenece las líneas usan las extracción de características para líneas de texto y palabras. La extracción de características en la función por la cual el clasificador SVM obtiene el contexto para poder realizar su estimación, es el conjunto de características más significativas de la palabra o de la línea que tiene que clasificar como de una clase u otra y puede ser usado para identificar patrones en el análisis del texto ayudando a la clasificación del mismo.

Al mismo tiempo, en este artículo continúan explicando que diseñaron un método de agrupamiento de palabras basado en reglas y dependiente del contexto para la generación de características especificas de palabras y cuyas reglas son extraídas de varios dominios de bases de datos y reglas basadas en propiedades ortográficas como por ejemplo las mayúsculas. Utilizan bases de datos para tener un conocimiento inicial del dominio, se encuentran por ejemplo nombres de ciudades estadounidenses, apellidos chinos o el diccionario estándar en línea de Linux.

Apoyándose en lo anterior hacen uso de métodos de agrupaciones de palabras (cluster words) agrupándolas por palabras similares que a su vez, es usado como característica también. Estos métodos de agrupación de palabras muestran una reducción de la dimensionalidad y un aumento del rendimiento.

Para las clases affiliation, address, degree, pubnum, note, abstract, keyword, introduction, y phone crearon listas de palabras con las palabras y par de palabras más frecuentes en las líneas de cada clase para entrenar el clasificador. Las palabras y pares de palabras de una misma agrupación (cluster) son representadas por una misma característica que llamaron "word-specifc feature", en cuanto a esto ultimo definen su peso como en numero de veces que aparece una misma "word-specifc feature" en la muestra.

Además definen también características para las líneas "line-specifc feature", por ejemplo establecen CsenLen como el número de palabras que contiene y ClinePos como la posición de la línea o número de la línea. Añadieron una característica que representa el porcentaje de una clase especifica de palabra en una línea por ejemplo usan "CdateNumPer" como porcentaje de palabras de fecha. También afirman que los SVM no manejan bien grandes rangos de diferencia entre valores de las características donde características de mayor escala dominaban a las de menor escala y deben normalizar el peso de las características para mejorar el rendimiento.

Siguen dos pasos para la clasificación de las líneas, donde en el primer paso es las clasificación de la línea en una o más clases, por lo que lo han llamado un clasificador de líneas independiente, seguidamente el segundo paso consta de una clasificación contextual de la

línea:

En el primer paso genera vectores de características en base a los criterios explicados anteriormente, este vector es una representación de las características extraídas como el numero de repetición de palabras claves, si es una única letra mayúscula, etc.

Se etiqueta el vector como una determinada clase si contiene palabras de la clase en cuestión, se genera un conjunto de vectores de características de entrenamiento para la clase en cuestión con los vectores recopilados de dicha clase como muestras positivas y el resto como negativas y este procedimiento se aplica a todas las demás clases. Destacan que los vectores multiclase estarán en los distintos conjuntos de vectores de las clases a las que estén etiquetados.

En el siguiente paso usan un clasificador contextual para las líneas haciendo uso de la información secuencial entre las mismas cuyo objetivo es mejorar la clasificación de cada línea. Codifican las etiquetas de N líneas antes y después de la línea actual L como características binarias y las concatenan al vector de características de la línea L formado en el paso anterior.

Continúan comentando que entrenan a cada clasificador contextual por cada meta-etiqueta en base a las etiquetas del vector de características con información contextual adicional.

Las etiquetas de clases de las líneas adyacentes son clasificadas por el clasificador de línea independiente. Las líneas del conjunto de pruebas son reclasificadas por el clasificador contextual en una o más clases, la clasificación contextual de las líneas es repetida de modo que, en cada iteración el vector de características de cada línea es expandido al agregarle la predicción de la información de la etiqueta de clase de las líneas vecinas predicha por el clasificador en la iteración anterior.

El método converge cuando el porcentaje de la líneas con nuevas etiquetas de clase es inferior a un umbral que establecieron como 0.7 % en este caso. Después de clasificar cada línea en una o más clases se extrae los meta-datos de cada multi-clase basada en la predicción de la etiqueta de clase de esa línea.

Enfatizan que la tarea de extracción de metadatos desde líneas multi-clases se vuelve un trabajo de identificación de "chunks", el cual cada uno pertenecerá a una clase. Consideran que los signos de puntuación y los espacios entre palabras son posibles "chunk boundaries" si hay más de dos en una línea.

Buscan el límite óptimo para los "chunks boundaries" que produzca la mayor diferencia entre dos chunks, donde los "chunks boundaries" son lo que conecta dos chunks. Solo consideran candidatos los signos de puntuación si se utilizan dos o más signos de puntuación en la línea; de lo contrario, apuestan por probar con cada signo de puntuación y espacio. Suponiendo que cada clase tiene sólo un fragmento en la línea, la identificación de chunks de dos clases es encontrar los chunk boundary óptimos.

Para ejemplificar más la situación utilizan el siguiente ejemplo que considero bastante revelador: "The Ohio State University, Columbus, OH 43210-1277" es un ejemplo de una línea de dos clases, la afiliación y la dirección. Cada coma es un candidato a chunk boundary la elección aquí es crucial para identificar cada chunk adecuadamente.

Explican el procedimiento de la siguiente forma: Llamaron el clasificador de la afiliación como clasificador 1 y el clasificador de la dirección como clasificador 2, donde cada uno es un SVM que ha sido entrenado por líneas individuales de una única clase. Consideran cada chunk como una pequeña línea independiente.

Establecen que:

P1 es la puntuación de clasificación del chunk P por el clasificador 1

P2 es la puntuación de clasificación del chunk P por el clasificador 2

N1 es la puntuación de clasificación del chunk N por el clasificador 1

N2 es la puntuación de clasificación del chunk N por el clasificador 2

P12 = P1 - P2; N21 = N2 - N1

PN1 = P1 - N1; PN2 = P2 - N2

Y eligen como el chunk boundary óptimo el signo de puntuación o el espacio que produce el máximo P12 * N21. El chunk P es clasificado en la clase 1 si $PN1 > 0$ y $PN1 * PN2 < 0$ o $PN1 * PN2 > 0$ y $||PN1|| = max(||PN1||, ||PN2||)$, de otra forma será de la clase 2. Continúan afirmando que este método les dio un 75.5 % de precisión siendo esta misma la cantidad de líneas cuyos chunks fueron correctamente clasificados entre el total de líneas multi-clases de 2 clases.

Luego consideran el problema de los chunks en cuanto a los nombres de autor, donde una línea cuya clase es autor con menos de cuatro palabras como línea de un único autor y con cuatro o más como línea multi-autor. Definen las líneas donde los autores están separados solo por espacios como "space-separated multi-author line" y las que están solo separadas por signos de puntuación como "punctuation-separated multi-author line".

A la hora de tratar las líneas multi autor separadas por signos de puntuación vuelven a enfatizar que el objetivo es encontrar correctamente el chunk boundary para que cada chunk este claramente separado y se pueda diferenciar cada nombre correctamente. En este artículo los autores continúan diciendo que el problema reside en clasificar cada espacio o signo de puntuación como chunk boundary en caso de que lo sea y solo consideran como candidato a chunk boundary los signos de puntuación si hay dos o más en la línea.

Además consideran la preposición "y"("and" en inglés) como un signo de puntuación, pero ignoran los espacios siguientes a un signo de puntuación. Diseñaron vectores de características para los signos de puntuación y señalan que convierten cada palabra analizada en una tupla de 5 elementos $< FN, LN, L, FC, D >$ que son definidos como:

FN: vale 1 si la palabra está en la lista de nombres, de lo contrario vale 0.

LN: vale 1 si la palabra está en la lista de apellidos, de lo contrario vale 0.

L: vale 1, 2 o 0 si es una letra, dos letras o más de dos letras respectivamente.

FC: vale 1 si la palabra tiene su primera letra mayúscula en mayúscula, de lo contrario vale 0.

D: vale 1 si la palabra está en el diccionario de palabras, de lo contrario vale 0.

La intención de esto, explican en el artículo, es que si la palabra más cercana al signo de puntuación aparece solo en la lista de nombres o solo en la de apellidos ayudará en la clasificación de que el signo de puntuación es el chunk boundary correcto. Ponen el siguiente ejemplo: suponiendo que "Leonidas Fegaras, David Maier" satisface el siguiente patrón "[10010(nombre)] [01011(apellido)], [10011(nombre)] [00010(apellido)]", será razonable clasificar la coma como el chunk boundary, que recuerdo es la unión de dos chunks. No obstante encuentran que la característica dominante en la clasificación de los chunk boundaries era el signo de puntuación en cuestión el cual era una característica en su vector de características, de esta forma implementaron sencillas reglas heurísticas usando los signos de puntuación para extraer el nombre de las líneas con signos de puntuación multi-autor.

En cuanto a las líneas multi-autor separadas por espacios, explican que no tienen ninguna información explícita para el reconocimiento de los chunks boundaries al contrario que en el caso anterior. El patrón válido para los nombres de los autores son la fuente de la información en este caso.

El algoritmo que plantean tiene 4 pasos:

- El paso 1 es generar todos los nombres potenciales de la línea que sean razonables.
- El paso 2 es diseñar un vector de características para cada sentencia de nombres de autor potencial.
- El paso 3 es entrenar un clasificador SVM de nombres usando las muestras de entrenamiento etiquetadas.
- El paso 4 consta que si la línea de prueba tiene solo un sentencia con el nombre potencial esa será la sentencia extraída. De otra forma clasificará cada potencial nombre.

Por ejemplo, la línea "Alan Fekete David Gupta Victor Luchangco Nancy Lynch Alex Shvartsman" tiene tres potenciales nombres completos."[Alan Fekete] [David Gupta] [Victor Luchangco] [Nancy Lynch] [Alex Shvartsman]"; "[Alan Fekete] [David Gupta] [Victor Luchangco Nancy] [Lynch Alex Shvartsman]"; "[Alan Fekete] [David Gupta Victor] [Luchangco Nancy] [Lynch Alex Shvartsman]". La sentencia con los nombres que generan la máxima puntuación será la que extraiga el clasificador.

Afirman que generaron todas las sentencias de nombres potenciales de las 99 líneas de nombres multi-autor separadas por espacios del conjunto de muestras de pruebas. Afirmaron conseguir una precisión de 90,9 % gracias a la ten-fold validation. Desde que eligieron la secuencia potencial más alta para cada secuencia desconocida la precisión aumentó hasta el 99 % el cual es la división entre las sentencias correctamente extraídas y el total de sentencias.

Por último, explican los parámetros de para validar la tasa de éxito de los resultados, algo importante en todo desarrollo de aprendizaje automático donde estos datos y medidas nos ayudan a medir la eficacia de las soluciones propuestas, destacando en este caso precision, recall, F-measure y accuaracy como las medidas para evaluar los resultados.

A continuación en este artículo se explican estas medidas antes mencionadas y se describen

los procesos de evaluación que utilizaron. La evaluación "Overall evaluation" o evaluación general que es la certeza (accuracy) de las palabras que ha clasificado correctamente.

La siguiente evaluación se trata como "Class-specific evaluation" o evaluación especifica de clase, donde definen A como el número de verdaderos positivos clasificados como positivos, B como el numero de verdaderos positivos clasificados como negativos, C como el numero de verdaderos negativos clasificados como positivos y D como el número de de verdaderos negativos clasificados correctamente como negativos. Comparten las formulas de precision, accuracy, f-measure y recall con el A, B, C, D establecido anteriormente para clarificar el proceso.

Precision: (Precision = $\frac{\mathrm{A}}{\mathrm{A+C}}$)

Recall: (Recuperación = $\frac{\mathrm{A}}{\mathrm{A+B}}$)

Accuracy: (Accuracy = $\frac{\mathrm{A+D}}{\mathrm{A+B+C+D}}$)

F-measure: (F-measure = $2 \times \frac{\mathrm{Precision \times Recall}}{\mathrm{Precision+Recall}}$)

Afirman que aplicaron el método de extracción discutido anteriormente con parámetros que eligieron por la validación "ten-fold" (validación cruzada) en 500 muestras usadas para entrenamiento y 435 para validación. En cuanto a la evaluación general (Overall evaluation) consiguieron una certeza (*accuracy*) de 92,9 %. Resaltan su bien rendimiento en las líneas de una única clase y en la extracción de los autores en las líneas multi-autor.

Destacan una línea en particular en la cual hay dos emails y esta etiquetada originalmente como note y el SVM lo etiqueta como email en el cual ellos cuentan como falsa predicción pero puede debatirse según comentan. También comentan otros errores como al clasificar una línea de 5 clases que se trata como de 4 clases causando un reconocimiento erróneo de los chunks, además de clasificar como de clase única una línea de 5 clases porque sobrevalora la información contextual del texto "nota" de la línea cercana.

Es debido a este extenso artículo cuyo problema es similar al del actual proyecto que ha resultado de gran ayuda para comprender los problemas relacionados con la extracción de información utilizando algoritmos de aprendizaje automático, el concepto de "chunk" como fragmentos de información (grupos de palabras juntos p.e.: Potencia Contratada) y también preparar al método de aprendizaje de forma que sea capaz de extraer información de una palabra (feature o características)para ser entrenado en conocer su valor en las muestras en la fase de entrenamiento y diferenciarla de palabras que no aportan nada en la muestra, ayudándole a extraer la información requerida.

Capítulo 3
Planificación y Herramientas

3.1. Planificación del Proyecto

Fases	**Duración Estimada (horas)**	**Tareas (nombre y descripción)**
Estudio previo / Análisis	50	Tarea 1.1: Revisión Bibliográfica Tarea 1.2: Anotación de facturas
Diseño / Desarrollo / Implementación	140	Tarea 2.1: Generación automática de datos a partir de las facturas Tarea 2.2: Métodos basados en técnicas de aprendizaje automático Tarea 2.3: Desarrollo de método combinado Tarea 2.4: Desarrollo de servicio web API REST
Evaluación / Validación / Prueba	80	Tarea 3.1: Evaluación de los métodos implementados Tarea 3.2: Configuración del servidor
Documentación / Presentación	30	Tarea 4.1: Revisión de documentación producida Tarea 4.2: Redacción de la memoria y preparación para la presentación

Cuadro 3.1: Planificación del proyecto con sus fases y tareas

Tarea 1.1: Revisión Bibliográfica:

Se realiza un estudio de otros proyectos similares en el contexto de la extracción de información de textos para ver como se enfrentaron a los desafíos del proyecto y estudiando sus aportaciones y sus conclusiones quedándonos con los trabajos más relevantes, en este caso el paper "Automatic document metadata extraction using support vector machines"[14] fue especialmente relevante. También se procedió al estudio de los conceptos básicos necesarios para llevar acabo el proyecto como los métodos más usados de aprendizaje automático y las herramientas que se iban a usar, especialmente sklearn[16] y su documentación.

Tarea 1.2: Anotación de facturas

Se selecciona un conjunto de facturas y se anotan los campos que se considera información relevante a extraer del texto.

Se extraerán las facturas de cada compañía que sean más representativas y se creará una plantilla de cada una con el fin de generar facturas aleatorias.

En cada tipo de factura se sustituyen los valores de la factura de los campos anotados previamente sustituyéndolos por etiquetas que permitirá al generador de facturas sustituirlas por valores aleatorios pero realistas y así generar una gran cantidad de muestras para entrenar la inteligencia artificial.

Tarea 2.1: Generación automática de datos a partir de las facturas

Se desarrollará un script que permitirá generar facturas de forma automática. Se utilizarán ficheros de nombres y apellidos reales, así como un listado de empresas obtenidas de la CNMC, con el fin de que los datos que figuren en las facturas sean lo más realistas posibles, además se contará con listados de pueblos, localidades, provincias, códigos postales y todo el material que sea necesario para poder sustituir los campos anotados y etiquetados por valores aleatorios de esas listas.

Lo que se persigue con esta tarea es generar un amplio número de facturas aleatorias, a partir de las cuales se puedan obtener bastantes datos de entrenamiento.

Tarea 2.2: Métodos basados en técnicas de aprendizaje automático

Se implementarán varios métodos basados en técnicas de aprendizaje automático como pueden ser Support Vector Machines (SVM), con el fin de mejorar la capacidad de predicción.

Esta tarea es la que va a llevar más tiempo, ya que requiere implementar cada uno de los métodos, haciendo uso de las librerías disponibles en Python, entrenarlos con los datos del módulo anterior y probar si el resultado es satisfactorio.

Habrá que refinar los parámetros de cada método, probar con distintas configuraciones y extraer características que faciliten el entrenamiento y la predicción. El objetivo es conseguir estimadores que detecten el mayor número de campos con una alta precisión. De dichos métodos se usará el mecanismo de la librería scikit learn (GridSearchCV) para probar y seleccionar las mejores configuraciones de las distintas técnicas de aprendizaje.

Tarea 2.3: Desarrollo de método combinado

Una vez que se hayan desarrollado varios métodos en la tarea anterior, se estudiará la posibilidad de combinarlos con el fin de mejorar los resultados.

Se implementarán varias estrategias para seleccionar los datos a partir del método que sea más fiable en cada caso. Se buscará, por un lado, que el tiempo de ejecución sea lo más reducido posible y que se mantenga una alta precisión en las predicciones.

Tarea 3.1: Evaluación de los métodos implementados

Se desarrollarán varios algoritmos para evaluar los métodos implementados. A partir de las facturas de prueba generadas automáticamente, se calcularán varias métricas estándares como la precisión, las tasas de aciertos y fallos, así como el F-score. Esto nos permitirá conocer cuál es la capacidad de predicción de cada método.

También se usará la librería seaborn de python para generar diversas gráficas como los heatmap para visualizar y mostrar fácilmente las medidas obtenidas por los distintos métodos, además de guardar la configuración de los métodos implementados para que en caso de ser aceptables y exitosos se puedan cargar a la aplicación y ser usados para la predicción de las muestras entrantes

Tarea 4.1: Revisión de documentación producida

Se mostrarán todos los avances conseguidos y se usarán tablas y gráficas para mostrar los resultados de los métodos usados con diferentes tamaños de muestras.

Tarea 4.2: Redacción de la memoria y preparación para la presentación

Se prepara la memoria del proyecto y la presentación del mismo de cara a la defensa.

Las tareas 2.4 y 3.2 fueron removidas durante la realización del presente proyecto y han sido propuestas como trabajo futuro en el apartado en cuestión.7.3

3.2. Herramientas utilizadas

Para la realización del proyecto se uso un ordenador portátil privado contando con los siguientes componentes principales: CPU i7-6700HQ, 16GB RAM, Gráfica: Nvidia 980M 4GB VRAM. Se usaron herramientas como el editor de código Pycharm con python 3.8 y librerías de sklearn[13] para la implementación de los métodos de aprendizaje automático así como la librería Pandas para manejo de dataframes y la librería python-docx para editar docx desde python, para el uso de mapas de calor se usó la librería seaborn para crear gráficas en python.

Capítulo 4
Conjunto de Datos de Facturas

Ahora pasaremos a explicar el proceso de creación de las muestras para entrenar a los clasificadores.

Lo que queremos es extraer la información relevante de una factura eléctrica, como por ejemplo la potencia contratada, el importe total y los distintos importes que la suman, NIF, nombre y dirección del titular, etc. Lo que hacemos es hacer una lectura y análisis de las diferentes facturas disponibles en el mercado para definir las clases que usaremos para el aprendizaje automático, es decir, las etiquetas que guiarán al método en su aprendizaje para que luego de forma autónoma sepa predecir a que clase pertenece una parte del texto a la hora de procesarlo y extraer la información pertinente.

Realizamos una anotación de las facturas donde identificamos un conjunto de facturas de distintas empresas y seleccionamos las palabras claves que se usarán como etiquetas que son los campos de los que se quiere extraer la información de las facturas. Las etiquetas son claves a la hora de entrenar el algoritmo ya que tendrá que analizar cada palabra y tener en cuenta sus características (features) para identificar su importancia dentro del texto, ayudando a clasificar cada extracto de información de la manera correcta para ser capaz de extraerla en formato JSON como se ha explicado anteriormente.

Como explicábamos elegimos una serie de etiquetas extraídas en base a la información relevante de una factura eléctrica las cuales usaremos para marcar las partes del texto donde queremos extraer la información, es decir, donde se encuentran las palabras claves que queremos que el clasificador extraiga y ponemos una etiqueta (label) para que en la fase de entrenamiento se ajuste de la mejor manera posible para que en los casos de validación (pruebas para comprobar la efectividad del método una vez halla finalizado el entrenamiento) y en los reales conozca qué información extraer a través de la información contextual que el clasificador ha sido capaz de extraer de cada palabra asociada a una etiqueta concreta.

Podemos definir los campos de una factura eléctrica de la siguiente manera: En el contexto de una factura eléctrica, existen diversos campos de información relevante que son los que queremos extraer, es decir, los datos que representan, y es de utilidad conocerlos para comprender y analizar los detalles asociados a los servicios de suministro energético.

Entre los elementos relevantes que suelen encontrarse en una factura eléctrica se encuentra la Comercializadora, la cual es la empresa encargada de la venta de la energía eléctrica al consumidor. Esta entidad es responsable de la facturación y el suministro de electricidad al cliente. La comercializadora de electricidad es la empresa con la que el cliente finaliza un contrato para la compra y venta de energía eléctrica. Es la encargada de facturar el consumo de electricidad, establecer las tarifas, ofrecer diferentes planes de precios, y brindar atención

al cliente.

Otro campo en una factura eléctrica es el consumo de energía, que indica la cantidad de energía eléctrica consumida por el usuario en un período específico. Este dato se expresa generalmente en kilovatios hora (kWh) y es fundamental para determinar el importe a pagar por el consumo realizado.

El CUPS, o Código Universal del Punto de Suministro, es un identificador único asignado a cada punto de suministro eléctrico en España. Puedes ubicarlo en el apartado de Datos del contrato de la factura. Este código facilita la identificación precisa de la instalación eléctrica y es utilizado por las empresas distribuidoras y comercializadoras para gestionar el suministro de electricidad.

La Distribuidora es la empresa encargada de la distribución de la energía eléctrica en la zona donde se encuentra el punto de suministro contratado por el cliente. Es la empresa encargada de la red de distribución eléctrica que lleva la electricidad desde las subestaciones hasta los puntos de suministro de los consumidores.

Esta entidad gestiona la infraestructura eléctrica y garantiza la correcta entrega de la electricidad a los usuarios finales, es responsable de mantener y gestionar la red eléctrica, realizar las conexiones, gestionar las averías y garantizar la calidad del suministro eléctrico.

Otros datos relevantes que suelen incluirse en una factura eléctrica son la Fecha de Emisión y la Fecha de Cargo, que indican el momento en que se emitió la factura y la fecha donde se realiza el pago de la factura, respectivamente.

La fecha de emisión nos da información sobre el momento de generación de la factura eléctrica la cual es importante junto al número de identificación de la factura para identificarla y poder realizar seguimiento de la misma. La fecha de cargo es importante pues es cuando se realiza el pago del periodo de facturación de todo el consumo registrado. Asimismo, las fechas de inicio y fin del periodo de facturación son fundamentales para establecer el intervalo de tiempo al que corresponde el consumo facturado, es decir, es el periodo donde se realiza el registro de la energía consumida que se reflejará en la factura como importe.

El importe de alquiler en una factura eléctrica se refiere al costo asociado al alquiler de equipos o dispositivos que son propiedad de la compañía eléctrica. Este importe suele aparecer desglosado en la factura como un cargo adicional. Representa el coste asociado al alquiler del contador eléctrico, que es el dispositivo que mide la cantidad de electricidad consumida en una vivienda o local. El importe de alquiler puede estar relacionado con el uso y mantenimiento de este contador y cualquier equipo suministrado. Su propósito es cubrir los costos de mantenimiento, instalación y gestión de los equipos de medición que la compañía proporciona para garantizar un adecuado registro del consumo de electricidad.

El Importe de la energía corresponde al coste de la electricidad consumida. El importe de la energía se refiere al costo asociado a la cantidad de electricidad consumida durante el período de facturación. Este importe se calcula multiplicando la cantidad de energía consumida (expresada en kilovatios-hora, kWh) por el precio por kWh establecido por la compañía eléctrica. Este concepto refleja el coste de la electricidad en sí misma, es decir, el precio por la energía consumida.

Por otro lado, el importe de la potencia se refiere al coste asociado a la potencia contratada por el usuario. La potencia contratada es la capacidad máxima de consumo eléctrico que el cliente tiene disponible en su instalación. Este importe es un cargo fijo que se basa en la potencia contratada y se paga independientemente del consumo de energía. Representa el coste de tener disponible esa potencia en todo momento, aunque no se consuma energía en determinados momentos. Los importes de los impuestos de la factura son asociados a los impuestos por consumir la energía que depende del país y gobierno en cuestión. Todos estos importes sumados forman el importe total que suele ser el campo más relevante para el cliente pues no deja de ser la suma total a pagar por parte del cliente.

Además, la factura eléctrica suele incluir información personal del titular del contrato, como el NIF, "Número de Identificación Fiscal", que es un código único asignado a cada persona física o jurídica con residencia en España para su identificación fiscal. Este número es utilizado para identificar a los contribuyentes ante la Agencia Tributaria.

El Nombre y apellidos del titular son otro campo de gran importancia para identificar claramente al titular del contrato. Así como el Número de contrato y el Número de factura que son utilizados para la identificación de la relación entre cliente y comercializadora. El número de contrato es un identificador único asignado por la compañía eléctrica a cada contrato de suministro eléctrico. Este número se utiliza para identificar de manera específica el contrato que vincula al cliente con la empresa suministradora de electricidad.

Es importante tener este número a mano para cualquier gestión relacionada con el contrato, como consultas, modificaciones o reclamaciones. Por otro lado, el número de factura es un código único asignado a cada factura emitida por la compañía eléctrica. Este número identifica de manera individual cada factura, permitiendo su seguimiento y gestión de forma ordenada. Es de utilidad tener el número de factura a mano para consultas, pagos y cualquier trámite relacionado con la factura en cuestión.

El Peaje de Acceso y la Potencia Contratada son elementos importantes que determinan los costes fijos asociados al suministro eléctrico. El peaje de acceso tiene como objetivo cubrir los costos de transporte y distribución de la electricidad, así como otros conceptos relacionados con la garantía de suministro y el fomento de energías renovables. Forma parte de los costos asociados al suministro de electricidad y este peaje está regulado por el Gobierno y se aplica a todos los consumidores de energía eléctrica. Mientras que la potencia contratada es la cantidad máxima de energía que se puede consumir que ha sido contratada previamente por el titular de la factura con la comercializadora.

El Código Postal identifica la ubicación geográfica del punto de suministro eléctrico. Este dato es relevante pues ayuda a garantizar que la información sobre la ubicación del suministro sea correcta y se pueda realizar el seguimiento adecuado de los servicios prestados en esa área específica. Es parte de la información detallada que se incluye en la factura para identificar claramente la ubicación del suministro de energía eléctrica. La Población y la Provincia son datos que identifican la ubicación del consumidor y también son utilizados para determinar aspectos como la tarifa eléctrica aplicable en función de la zona geográfica. Esta información facilita la gestión administrativa y logística de la empresa comercializadora de energía.

Todos estos elementos conforman los campos más relevantes que hemos analizado de una

factura eléctrica y permiten al consumidor comprender detalladamente los aspectos de su suministro de energía eléctrica, los importes asociados, etc.

Con todo esto necesitamos una forma donde tener gran cantidad de muestras con las que entrenar el algoritmo. Teniendo las facturas en formato SpaCy y generándolas masivamente con información simulada podríamos generar una gran cantidad de muestras. El plan consistiría en unir todas las palabras en líneas de 11 palabras donde la palabra central, es decir, la sexta será en la que anotamos la etiqueta de la clase a la que pertenece, dicha clase es la etiqueta del campo del que se quiere extraer la información, es decir, si tenemos una línea como "Fecha emisión: 24/03/1996 Fecha vencimiento: 27/03/1996 Potencia 235 kWh x 0,01423" donde la sexta cadena de caracteres(string) es "27/03/1996", la etiquetamos como fecha de cargo que es la clase a la cual pertenece esa string. Mientras que la siguiente línea a analizar sería "emisión: 24/03/1996 Fecha vencimiento: 27/03/1996 Potencia 235 kWh x 0,01423 €/kWh" donde la sexta string es "Potencia" la etiquetamos como other pues aquí no es una información que queramos extraer.

En el primer caso estamos extrayendo la fecha de cargo, es decir, "27/03/1996" que es la fecha que aparece en el campo fecha de cargo de la factura y es lo que el algoritmo debe extraer y presentar en formato JSON, no obstante en el segundo caso la palabra (string) analizada es "Potencia" que no es información relevante aun siendo una palabra que pudiera indicar que le sigue información relevante por eso es etiquetada como other ya que no debe ser extraída por el algoritmo.

Este proceso es muy laborioso y se pueden cometer multitud de errores debido al gran número de datos y factores que se deben de tener en cuenta a la hora de generar esta información simulada, de hecho puede llegar a ser muy difícil de detectar un error en estos casos, pues en el script para generar las facturas en formato SpaCy nos dimos cuenta que no respetaba la separación del texto en sentencias de 11 palabras y se tardó mucho tiempo en que se detectará dicho fallo, el cual provocaba que el entrenamiento de un método de aprendizaje automático(SVM con kernel rbf) tardara más de 5 días en entrenarse con el conjunto de datos más grande, ya que era un método muy costoso en términos de procesamiento y daba unos resultados coherentes a la hora de verificar la validación de los resultados lo cual no despertaba sospechas y en una revisión del código con el tutor se detectó el error y una vez solventado el proceso redujo su duración a 1 día.

Para validar el resultado final de los clasificadores se usan las facturas originales para poder crear plantillas con las que generar facturas realistas ya que los campos con información relevantes están etiquetados y con un script previamente creado se remplazan las etiquetas por información realista para generar facturas automáticamente como veremos en el apartado 4.2.

Etiqueta(label)	**D**escripción
Comercializadora	Nombre de la comercializadora, empresa con la que el cliente final tiene el contrato
ConsumoEnergia	La energía consumida registrada en la factura
CP	Código postal del lugar donde se está consumiendo la energía que se ha contratado
CUPS	Código Universal de Punto de Suministro, es un código único e invariable formado por letras y números que sirve para identificar la instalación o punto de suministro de cada casa
Direccion	Dirección que corresponde al lugar donde se está consumiendo la energía que se ha contratado
Distribuidora	La distribuidora es la encargada de la red de distribución eléctrica que lleva la electricidad hasta los puntos de suministro de los consumidores
FechaEmision	Fecha de generación de la factura
FechaCargo	Se refiere al momento en el que se registra el consumo de electricidad que se está facturando
FechaFin	Es la fecha donde se termina de contabilizar el consumo de energía, es decir la fecha donde termina del periodo de facturación
FechaInicio	Es la fecha donde comienza a contabilizar el consumo de energía, es decir la fecha donde inicia del periodo de facturación
FinContrato	Fecha de la finalización del contrato
ImporteAlquiler	Importe de alquiler sobre el equipo que se usa en la instalación eléctrica
ImporteEnergia	Importe por la energía consumida durante el periodo de facturación. Este importe es la cantidad de energía consumida (en kWh) por el precio por kWh establecido por la compañía eléctrica
ImporteImpuestos	Importe asociado a los impuestos sobre la energía consumida
ImportePotencia	Importe sobre el coste asociado a la potencia contratada por el titular del contrato
ImporteTotal	Coste total de la factura de la luz
NIF	Numero de identificación fiscal del titular del contrato.
Nombre	Nombre y apellidos del titular del contrato.
NumeroContrato	Número único dado por la compañía eléctrica a cada contrato de suministro eléctrico, este número se utiliza para identificar el contrato que vincula al cliente con la empresa
NumeroFactura	El número de factura es un código único asignado a cada factura emitida por la compañía eléctrica. Este número identifica de manera individual cada factura
PeajeAcceso	Costos asociados al suministro de electricidad, el peaje de acceso tiene como objetivo cubrir los costos de transporte y distribución de la electricidad
PotenciaContratada	Potencia contratada por el titular del contrato
Poblacion	Población en la que se encuentra ubicado el punto de suministro de electricidad
Provincia	Provincia a la que pertenece la población en la que se encuentra ubicado el punto de suministro de electricidad
Other	Etiqueta que se asocia a información que no es relevante ni se busca extraer

4.1. Generación automática de datos a partir de las facturas

A la hora de desarrollar el método SVM necesitamos una gran cantidad de muestras para desarrollar el entrenamiento de los algoritmos de los clasificadores que como hemos dicho previamente eran Naive bayes, SGD (*Stochastic Gradient Descent*), y SVM con el kernel "rbf" (*Radial Base Function*).

Habíamos explicado que el plan era la generación de facturas en formato spacy donde cada línea contaría con 11 palabras cada línea donde la sexta palabra o palabra del medio exacto de la línea estará etiquetada con la clase a la que pertenece para poder realizar el aprendizaje automático. Estas muestras serán simulaciones de las facturas de distintas comercializadoras eléctricas, siguiendo su patrón y estructura de los distintos campos para que en la validación final el algoritmo tenga un rendimiento con la mayor eficacia.

4.1.1. SpaCy

También cabría explicar que es Spacy:

SpaCy es una librería de software de código abierto diseñada para facilitar tareas de procesamiento avanzado del lenguaje natural. SpaCy está escrita en Python y Cython (extensiones del lenguaje C para Python que permiten realizar una programación de bajo nivel muy eficiente). SpaCy tiene como objetivo facilitar la puesta en producción (hacer una aplicación lista para su uso por el consumidor final) de las aplicaciones de software en el ámbito del lenguaje natural.[12]

Diseñada específicamente para la producción, SpaCy es rápida y eficiente, y está construida sobre las últimas investigaciones en el campo del NLP. A diferencia de otras bibliotecas de NLP, como NLTK, que están diseñadas principalmente para la enseñanza y la investigación, SpaCy se enfoca en tareas industriales. SpaCy es una herramienta potente y versátil para el procesamiento del lenguaje natural que ha ganado popularidad en la industria y la academia por su robustez y facilidad de uso.[15]

Algunas caracteristicas de SpaCy son las siguientes:

- Tokenización: Divide el texto en palabras, signos de puntuación, etc.
- Reconocimiento de entidades nombradas (NER): Identifica entidades como nombres de personas, organizaciones, lugares, fechas y más.
- Análisis de dependencia: Identifica las relaciones gramaticales entre las palabras en una oración.
- Lematización: Reduce las palabras a su forma base o raíz.
- Etiquetado POS (part-of-speech): Etiqueta las palabras como sustantivos, verbos, adjetivos, etc.

- Integración con modelos de lenguaje profundo: spaCy se puede integrar con bibliotecas como TensorFlow y PyTorch, permitiendo la incorporación de modelos de lenguaje más avanzados, como BERT y transformer.
- Soporte multilingüe: Ofrece modelos preentrenados para varios idiomas.
- Extensibilidad: Puedes añadir tus propios componentes y extensiones.
- Integración con word vectors y embeddings: Permite la incorporación de vectores de palabras para la similitud semántica y otras tareas.
- Rendimiento: Es conocida por su velocidad y eficiencia, lo que la hace adecuada para aplicaciones en tiempo real.

En SpaCy se presentan aspectos adicionales que complementan su utilidad y versatilidad:

Modelos Preentrenados:

SpaCy proporciona modelos preentrenados para varios idiomas, lo que facilita el análisis de texto en diferentes contextos lingüísticos. Estos modelos incluyen información sobre algunos elementos antes mencionados como tokenización, etiquetado POS, lematización, entre otros, permitiendo a los usuarios trabajar con datos en varios idiomas sin la necesidad de entrenar modelos desde cero.

Pipeline de Procesamiento Personalizable:

Una de las ventajas de SpaCy es su capacidad para personalizar el pipeline de procesamiento según las necesidades específicas de cada proyecto. Los usuarios pueden configurar qué componentes se aplican al texto y en qué orden, lo que brinda flexibilidad para adaptar el flujo de trabajo a diferentes escenarios de análisis de texto.

Visualización de Resultados:

SpaCy ofrece herramientas integradas para visualizar los resultados de las tareas de procesamiento de texto, como la visualización de árboles de dependencia, la identificación de entidades nombradas resaltadas en el texto y la representación gráfica de las relaciones gramaticales entre las palabras. Estas capacidades visuales facilitan la interpretación de los resultados y el análisis de los datos procesados.

Comunidad Activa y Recursos Educativos:

Además de su documentación detallada y extensa, SpaCy cuenta con una comunidad activa de desarrolladores y usuarios que comparten conocimientos, recursos y colaboran en el desarrollo de nuevas funcionalidades. Esto permite a los usuarios acceder a tutoriales, ejemplos de código y discusiones en línea que enriquecen su experiencia con la herramienta y fomentan el aprendizaje continuo.

Esta última característica es vital en la mayoría de herramientas de código abierto especialmente relacionado con el desarrollo de aprendizaje automático e inteligencia artificial donde al compartir progresos y conocimientos entre la comunidad de usuarios e incluso investigadores se llega a contar con mucha ayuda para desarrollar el proyecto que se desea, y

de ejemplos para inspirarse o directamente aprender, también puede provocar cierta estandarización de la herramienta la cual causa que esta característica resalte aún más al aumentar la afluencia de usuarios.

Estos aspectos destacan la versatilidad y potencial de SpaCy como una herramienta fundamental para el análisis de texto y aplicaciones del procesamiento del lenguaje natural. Su combinación de rendimiento, facilidad de uso y funcionalidades avanzadas y una comunidad que apoya su desarrollo y uso aportan un gran valor tanto para profesionales de la industria como para investigadores en el campo del NLP, así como para el resto de usuarios y desarrolladores de esta herramienta.

4.1.2. Generación de facturas en formato SpaCy

Ahora procedemos a explicar la forma en la que generamos una gran cantidad de muestras realistas para entrenar a los clasificadores la cual fue proporcionada por ayuda del propio tutor del actual TFG y que se explica a continuación:

Este diseño es un diseño anterior del publicado por mi propio tutor y sus compañeros en el un artículo que publicaron en el 2022.[19]

En este paso ya hemos seleccionado las facturas de empresas de energía eléctrica y hemos escogido qué campos vamos a extraer, que son los campos de información relevante como nombres, fechas, pagos, potencias, etc, que hemos explicado previamente 4.1, luego se convierten a texto plano y, además, en los campos en los que estamos interesados se borra la información relevante a extraer, sustituyéndola por etiquetas de los campos que queremos extraer de las siguiente manera "<etiqueta>".

Con un script que es capaz de generar información realista de nombres, calles, empresas, fechas, etc rellenamos los huecos de información sustituyendo las etiquetas por estos datos. Esta información se extrae aleatoriamente de ficheros a modo de diccionarios con los nombres reales de las clases anteriormente mencionadas y en el caso de fechas o cifras como el NIF se crean aleatoriamente siguiendo patrones realistas y razonables, y como decíamos, sustituimos las etiquetas por estos datos para generar facturas realistas en formato spacy como en la siguiente imagen. 4.5.

Es decir, gracias a tener la facturas en texto plano podemos cambiar cada uno de los campos con estos ficheros generados que se usan para entrenar el método de aprendizaje automático o lo que es lo mismo el clasificador. Para esta generación de facturas realistas necesitamos a su vez un gran número de datos sobre tales cosas como calles, distribuidoras, comercializadoras, nombres y apellidos en español para los cuales haremos uso de grandes diccionarios que son en la práctica ficheros .txt, es decir, ficheros de texto plano, con la información necesaria, en este caso como dijimos, calles, distribuidoras, comercializadoras, poblaciones, etc, las cuales se usarán a la hora de generar estas facturas simuladas realistas.

Esta información será seleccionada aleatoriamente de los diccionarios con el fin de tener gran variedad de facturas realistas diferentes cuya información contenida no sea descabellada para entrenar correctamente a los clasificadores y que en casos reales, no tengan un

rendimiento nefasto aunque en las pruebas y validaciones tengan un buen rendimiento. Esto último podría ocurrir si las facturas no están correctamente generadas, es decir, de forma realista y razonable y se alejaran demasiado de las reales volviendo inútil al clasificador por mucho que tenga buen rendimiento con las facturas generadas automáticamente. Mostramos parte de estos "generadores de datos" el la siguiente imagen.4.1

```
nombres = open(data_directory+"spanish_names.txt").readlines()
apellidos = open(data_directory+"spanish_surnames.txt").readlines()
comercializadoras = [line.replace('\n', '') for line in
                open(data_directory+"spanish_marketers.txt").readlines()]
distribuidoras = open(data_directory+"spanish_distributors.txt").readlines()
poblaciones = open(data_directory+"spanish_villages.txt").readlines()
calles = open(data_directory+"spanish_streets.txt").readlines()
```

Ilustración 4.1: Generación aleatoria de datos mediante diccionarios

Por otro lado hay ciertos datos que podemos generar y simular con el uso de algoritmos, es decir, conociendo como es la estructura del campo a simular podemos crear datos realistas para estos campos como son el NIF, fechas de cargo, emisión, inicio y fin de periodo de facturación, CUPS, Número de contrato, etc. Para estos casos desarrollamos los métodos necesarios para simular dichos datos de manera creíble, por ejemplo, el NIF es en esencia una ristra de 8 números y una letra, con un simple algoritmo que reúna estas condiciones podemos generar los NIF que necesitemos para cada factura generada. En este caso basta con generar 8 cifras aleatoriamente y agregar una letra mayúscula para tener un resultado razonable de NIF 4.2. Un ejemplo de lo expuesto lo podemos ver en las siguientes imágenes: 4.3

```
result['NIF'] = str(random.randint(10**7, 10**8-1)) + \
                random.choice(string.ascii_uppercase)
```

Ilustración 4.2: Generación aleatoria de NIF

```
result['CUPS'] = 'ES' + str(random.randint(10**15, 10**16-1))
letters = string.ascii_uppercase
result['CUPS'] += ''.join(random.choice(letters) for i in range(4))
result['NumeroContrato'] = str(random.randint(10**12, 10**13-1))
result['PotenciaContratada'] = random.choice([2.00, 2.50, 3.00, 3.50, 4.00, 4.50, 5.00, 5.50, 6.00, 6.50])
result['PeajeAcceso'] = random.choice(['2.0A', '20DHA'])
```

Ilustración 4.3: Generación aleatoria de datos

Como se había indicado anteriormente una vez estén los resultados ya generados se procede a su inclusión en la factura sustituyendo las etiquetas por los resultados que las representan. 4.4 Esto se hace al sustituir la marca <etiqueta> donde la etiqueta estará nombrada en el txt por la clase que le corresponda por ejemplo, usando ejemplos anteriores podríamos tener algo como NIF: <NIF> que sería sustituido de la siguiente manera NIF: 58311579G.

```
for label in result:
    bill = bill.replace("<" + label + ">", str(result[label]))
```

Ilustración 4.4: Se rellena la estructura de la factura

El resultado final de las factura en formato SpaCy se visualiza en la siguiente imagen 4.5, la etiqueta se sitúa después de "Entities" justo después de valores reservados para expandir el proyecto como por ejemplo indicar si es multietiqueta la propia línea.

```
 {'entities': [(0, 0, 'NumeroFactura')]}),
('E) #eol NO Factura: FI8248659842 Doc Impresión: #eol Periodo de '
 'facturación:',
 {'entities': [(0, 0, 'Other')]}),
('Factura: FI8248659842 Doc Impresión: #eol Periodo de facturación: '
 '22/01/1996 - 22/03/1996',
 {'entities': [(0, 0, 'Other')]}),
('Impresión: #eol Periodo de facturación: 22/01/1996 - 22/03/1996 #eol Fecha '
 'emisión:',
 {'entities': [(0, 0, 'FechaInicio')]}),
('#eol Periodo de facturación: 22/01/1996 - 22/03/1996 #eol Fecha emisión: '
 '24/03/1996',
 {'entities': [(0, 0, 'Other')]}),
('Periodo de facturación: 22/01/1996 - 22/03/1996 #eol Fecha emisión: '
 '24/03/1996 Fecha',
 {'entities': [(0, 0, 'FechaFin')]}),
('22/01/1996 - 22/03/1996 #eol Fecha emisión: 24/03/1996 Fecha vencimiento: '
 '27/03/1996 Potencia',
 {'entities': [(0, 0, 'Other')]}),
('- 22/03/1996 #eol Fecha emisión: 24/03/1996 Fecha vencimiento: 27/03/1996 '
 'Potencia 235',
 {'entities': [(0, 0, 'FechaEmision')]}),
('Fecha emisión: 24/03/1996 Fecha vencimiento: 27/03/1996 Potencia 235 kWh x '
 '0,01423',
 {'entities': [(0, 0, 'FechaCargo')]}),
('emisión: 24/03/1996 Fecha vencimiento: 27/03/1996 Potencia 235 kWh x '
 '0,01423 €/kWh',
 {'entities': [(0, 0, 'Other')]}),
('vencimiento: 27/03/1996 Potencia 235 kWh x 0,01423 €/kWh día x 59',
 {'entities': [(0, 0, 'Other')]}),
```

Ilustración 4.5: líneas de la factura en formato spacy

4.2. Generación de facturas realistas

Otra vía de experimentación a la hora de generar grandes cantidades de datos de muestras para entrenar los clasificadores fue la idea de generar las facturas directamente usando el documento en word, es decir, formatos como .doc o .docx entre otros, y se usó el ya mostrado algoritmo de generación de datos realistas siguiendo la misma idea de sustituir los campos etiquetados en la factura pero en lugar de hacerlo leyendo el documento como texto plano y reescribiéndolo en formato SpaCy, modificar directamente el word. De las facturas seleccionadas se crean plantillas como hemos explicado anteriormente sustituyendo en el mismo documento word la etiqueta del campo por la información generada aleatoriamente como en el script anterior pero ahora la intención es crear facturas muy parecidas a la realidad con la disposición e imágenes que tendría normalmente la factura.4.6 La idea es acercarse al máximo a la factura real y que sea el clasificador el que tenga que entrenarse en estos casos, buscando que a la hora de que procese una factura autentica el rendimiento sea el mejor posible.

En este punto el proyecto se ha encontrado con problemas, pues a la hora de generar las facturas en formato docx convertidas de pdf se han generado con textbox el cual es un elemento de los documentos docx que aunque es posible editarlo en la propia aplicación, es decir, etiquetar los campos donde está la información relevante en ellos usando la aplicación Word, en los campos etiquetados al intentar remplazarlos con el script desde python no se puede escribir ni manipular en esos textbox para remplazar las etiquetas por los datos generados aleatoriamente ya que se usa la librería Python-Docx para editar en estos documentos docx y dicha librería no es capaz de escribir ni manipular los textbox. Eliminar los textbox no es una solución ya que modifican demasiado las facturas y serían muy distintas de las originales incumpliendo el requisito esencial del objetivo impuesto de crear facturas lo más cercanas a la realidad.

Había otra opción que era escribir directamente en el documento modificándolo como un xml lo cual resultaba demasiado complicado y se gasto demasiado tiempo investigando esta posible solución la cual requería realizar patrones de expresiones regulares opción que ya había sido rechazada para el problema de la extracción de información de una factura de la luz como se explico en la introducción 1 y eso esta basado en el artículo explicado en el apartado de trabajos previos 2.2[14].

A continuación podemos ver algunas capturas de unas facturas plantilla donde sus campos con información relevante han sido remplazados por etiquetas, 4.6 esta factura es del tipo que se usaría tanto en el generador de facturas en formato SpaCy como en con este generador de facturas realistas. Seguidamente podemos ver una plantilla con las etiquetas ya remplazadas por la información generada por el generador de facturas realistas.4.84.9 En sus dos páginas, en comparación con una plantilla con las etiquetas remplazadas por datos generados por el script de facturas realistas, donde debido a sus textbox no es posible editar todas sus etiquetas.4.7, es decir, donde aparecen resaltados los importes, es un objeto de .docx llamado textbox y en el no puede escribir la librería antes mencionada que usamos para escribir en documentos docx. Sin embargo se puede ver como otros datos los remplaza correctamente como el nombre o el importe total de la factura.

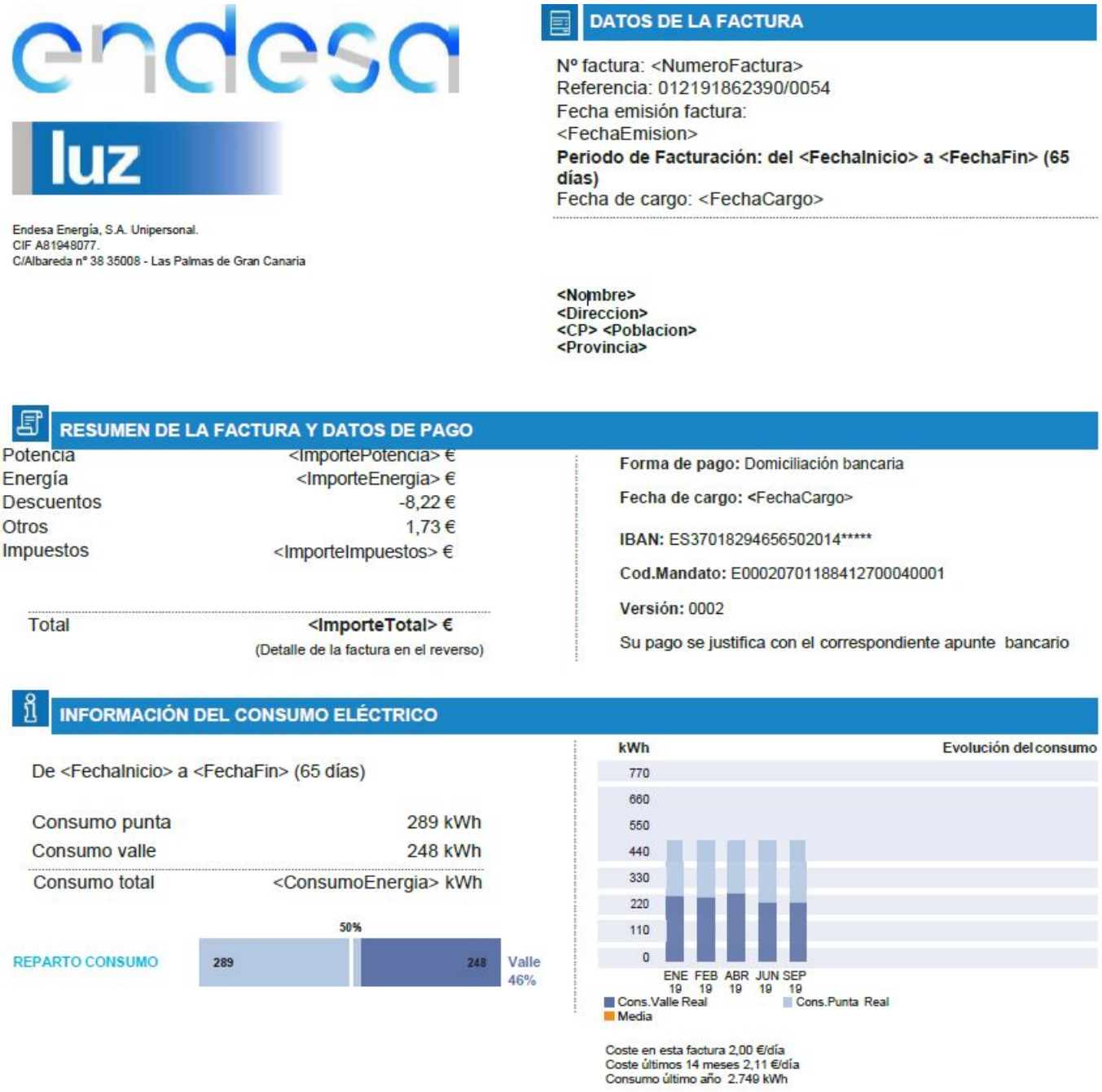

Ilustración 4.6: Captura de plantilla de factura realista

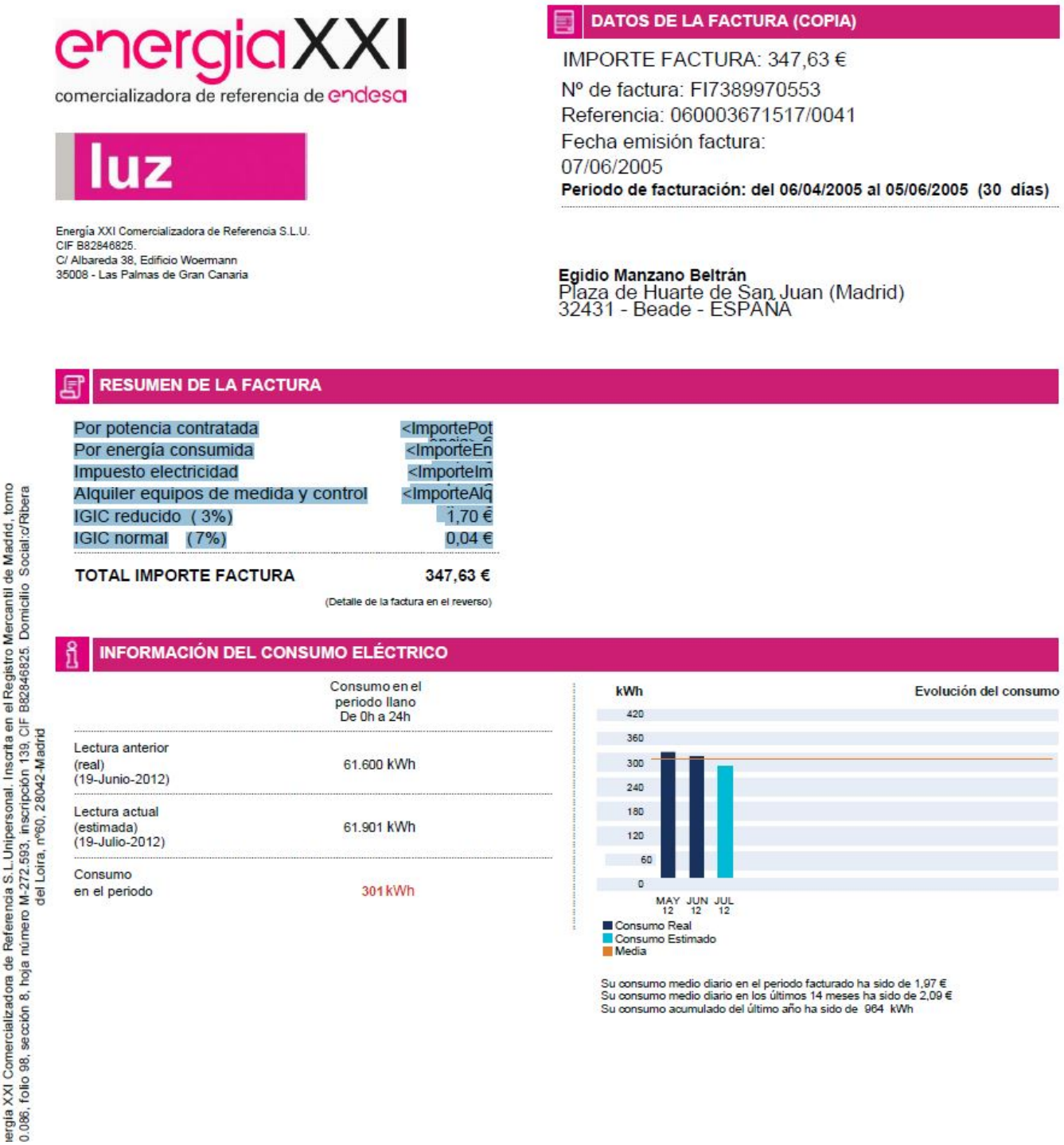

Ilustración 4.7: Captura de plantilla de factura realista con textbox

Ilustración 4.8: Captura de plantilla de factura realista 1 con campos rellenados

DATOS DEL CONTRATO

Titular del contrato: Albertina Gutiérrez Campos
NIF: 31393958C
Dirección de suministro: Calle de Rafael Alberti (Camarma de Esteruelas) Vinebre, Tarragona
Producto contratado: Tempo Siempre Ganas
Potencia contratada: 2.5 kW
CUPS: ES4067871052227227MDNC

Número de contador: 7470933258541
Referencia del contrato: 012191862390
Su comercializadora: BSG ENERGIA S.L.
Su distribuidora: PEDRO SANCHEZ IBAÑEZ S.L.
Referencia del contrato de acceso: 500002056849 Peaje de acceso: 2.0A
Fin de contrato de suministro: 07/03/1999 (renovación anual automática)

LUZ

Importe por potencia contratada:

5 kW x 0,133229 Eur/kW x 2 días	1,33 €
5 kW x 0,133229 Eur/kW x 63 días	41,97 €
	39,67 €

Importe por energía consumida:

17 kWh x 0,161225 Eur/kWh	2,74 €
520 kWh x 0,161151 Eur/kWh	83,80 €
	217,07 €

SUBTOTAL	**129,84 €**
OTROS CONCEPTOS	
Descuento por e-factura 86,54 Eur x -2 % DTO	-1,73 €
Descuento -5 % x 129,84 Eur	-6,49 €
Impuesto electricidad (121,62 X 5,11269632 %)	5,24 €
Alquiler equipos de medida y control (65 días x 0,026615 Eur/día)	2,11 €
SUBTOTAL	**-0,27 €**
Importe total	**129,57 €**
IGIC reducido (0%) 0% s/ 127,84	0,00 €
IGIC normal (6,5%) 6,5% s/ 1,73	0,11 €
TOTAL IMPORTE FACTURA	**266,80 €**

Incluido en el importe facturado está el coste del peaje de acceso que ha sido de 52,34 € (33,87 € potencia y 18,47 € por energía activa). Precios del peaje de acceso publicados en la Orden TEC/1366/2018 (BOE 22-12-2018). Precio del alquiler de los equipos de medida y control en Orden IET 1491/2013 de 3 de agosto

El destino del importe de su factura, 129,68 euros, es el siguiente:

- Incentivos a las energías renovables, cogeneración y residuos **20,33 €**
- Coste de redes de transporte y distribución **20,79 €**
- Otros costes regulados (incluida la anualidad del déficit) **10,42 €**

A los importes indicados en el diagrama debe añadirse, en su caso, el importe del alquiler de los equipos de medida y control así como los conceptos no energéticos.

INFORMACIÓN DEL CONSUMO ELÉCTRICO
Efectos facturación de los peajes de acceso

	28/06/2019 L.Ant real	01/09/2019 Lectura real	Multipl.	Ajuste	Consumo
Punta	1.408	1.697	1	0	289
Valle	1.070	1.318	1	0	248

INFORMACIÓN DE SU PRODUCTO

Los precios de la energía de esta tarifa se han actualizado el 01/07/2019 trasladando las variaciones reguladas en la Orden IET/2013/2013 de 31 de octubre, en la Orden ETU/1133/2017 de 21 de noviembre, en la Orden ETU/362/2018 de 6 de abril y en la Resolución de 24 de mayo de 2019.

ATENCIÓN AL CLIENTE: CONSULTAS, GESTIONES Y RECLAMACIONES 24 HORAS

800760909 (tlf. gratuito)
www.endesaclientes.com
atencionalcliente@endesaonline.com

Reclamaciones
C/ Ribera del Loira 60
28042 Madrid

Urgencias
900 85 58 85
(tlf. gratuito)

Para reclamaciones sobre el contrato de suministro o facturaciones podrá dirigirse a: Consejería de Empleo, Industria y Comercio de la Comunidad Autónoma de Canarias en el teléfono: 928 899 400 o a través de su página web. http://www.gobcan.es/ceic/energia

Endesa Energía está adherida al Sistema Arbitral de Consumo. Para ampliar la información sobre las reclamaciones que pueden ser tratadas a través del arbitraje consultar www.endesaclientes.com.

Ilustración 4.9: Captura de plantilla de factura realista 1 con campos rellenados 2ª pagina

Capítulo 5
Extracción de información a partir de facturas

5.1. Pandas y Dataframes

A la hora de tratar con las muestras y las etiquetas correspondientes se necesitan herramientas para gestionar de forma más sencilla semejante cantidad de datos y muestras, aquí es donde entra la librería Pandas. De Pandas podemos decir lo siguiente según la información que podemos encontrar en línea: [5]

Pandas es una biblioteca de lenguaje de programación Python, dedicada por completo a la Data Science. La biblioteca de software de código abierto Pandas está diseñada específicamente para la manipulación y el análisis de datos en el lenguaje Python. Es potente, flexible y fácil de usar. Gracias a Pandas, por fin se puede utilizar el lenguaje Python para cargar, alinear, manipular o incluso fusionar datos. El rendimiento es realmente impresionante cuando el código fuente del back-end está escrito en C o Python.

El nombre «Pandas» es en realidad una contracción del término «Panel Data» para series de datos que incluyen observaciones a lo largo de varios periodos de tiempo. La biblioteca se creó como herramienta de alto nivel para el análisis en Python.

Los creadores de Pandas pretenden que esta biblioteca evolucione hasta convertirse en la herramienta de análisis y manipulación de datos de código abierto más potente y flexible en cualquier lenguaje de programación. Además del análisis de datos, Pandas se utiliza mucho para la Data Wrangling. Este término engloba los métodos de transformación de datos no estructurados para hacerlos procesables. Por lo general, Pandas también destaca en el procesamiento de datos estructurados en forma de tablas, matrices o series temporales. También es compatible con otras bibliotecas de Python.

Pandas trabaja sobre DataFrames: tablas de datos bidimensionales, donde cada columna contiene los valores de una variable y cada fila contiene un conjunto de valores de cada columna. Los datos almacenados en un DataFrame pueden ser números o caracteres. Los Data Scientists y los programadores familiarizados con el lenguaje de programación R para cálculo estadístico utilizan DataFrames para almacenar datos en una cuadrícula muy sencilla de revisar. Por eso Pandas se utiliza mucho para Machine Learning.

Esta herramienta permite importar y exportar datos en distintos formatos, como CSV o JSON. Además, Pandas también ofrece la funcionalidad de Data Cleaning. Esta biblioteca es muy útil para trabajar con datos estadísticos, datos tabulares como tablas SQL o Excel,

con datos de series temporales y con datos matriciales arbitrarios con etiquetas de filas y columnas.

Para los Data Scientists y desarrolladores, Pandas aporta varias ventajas. Esta biblioteca permite compensar fácilmente los datos que faltan. Es una herramienta flexible, ya que las columnas pueden insertarse o eliminarse fácilmente dentro de los DataFrames. La alineación de los datos con las etiquetas puede automatizarse. Otra gran ventaja es una potente herramienta de agrupación de datos que permite realizar operaciones de split-apply-combine sobre las series de datos para agregarlos o transformarlos. Es muy fácil convertir datos indexados de forma diferente en otras estructuras de Python y NumPy en objetos DataFrame. Del mismo modo, los datos pueden indexarse u ordenarse mediante un sistema inteligente basado en etiquetas.

Los conjuntos de datos pueden fusionarse de forma intuitiva y reestructurarse con flexibilidad. Las herramientas de E/S simplifican la carga de datos desde archivos CSV, Excel o bases de datos, o la carga de datos en formato HDF5. La funcionalidad de series temporales completa el cuadro, principalmente con la generación de intervalos de fechas, la conversión de frecuencias o el desplazamiento de ventanas estadísticas. Todos estos puntos fuertes hacen de Pandas una biblioteca imprescindible para la Data Science en Python. Se trata de una herramienta muy útil para los Data Scientists.

Algunos lenguajes de programación se utilizan tradicionalmente en entornos científicos o en equipos de investigación y desarrollo de empresas. Sin embargo, estos lenguajes suelen plantear problemas a los Data Scientists. Sin embargo, Python permite superar la mayoría de estas limitaciones. Es un lenguaje ideal para las distintas etapas de la ciencia de datos: limpieza, transformación, análisis, modelización, visualización y elaboración de informes. Su interfaz es agradable, la documentación es completa y el uso es relativamente intuitivo. La popularidad de Pandas también está ligada a su antigüedad. Fue la primera biblioteca de este tipo en crearse, o al menos una de las primeras. Además, es una herramienta de código abierto y muchas personas han contribuido al proyecto. Esto es lo que le ha dado tanto éxito.

Además de Pandas, existen otras bibliotecas de software de Python dedicadas a la Data Science. NumPy es una biblioteca matemática que permite implementar álgebra lineal y cálculos estándar de forma muy eficiente. Pandas está basado en NumPy. Muchas estructuras de datos y características de Pandas provienen de NumPy. Estas dos bibliotecas están estrechamente interrelacionadas y a menudo se utilizan juntas.

Por su parte, Scikit-learn es la referencia para la mayoría de las aplicaciones de Machine Learning en Python. Para crear un modelo predictivo, se suele usar Pandas y NumPy para cargar, analizar y dar formato a los datos que se van a utilizar. A continuación, estos datos se utilizan para alimentar el modelo de Scikit-learn. Este modelo se utiliza después para hacer predicciones. Por lo tanto, Pandas, NumPy y Scikit-learn son tres herramientas de uso común en Data Science.

5.2. Dataframes de entrenamiento y validación

Una vez tenemos un extenso dataset, es decir, una gran cantidad de muestras con la que entrenar el clasificador, podemos comenzar con la preparación de los clasificadores.

Para la extracción de la información de las facturas, las muestras de las facturas en formato SpaCy y cargadas en memoria, se separan las líneas de texto de las etiquetas las cuales especifican qué información extraíble contiene el texto, en el código 5.1 podemos observar que las variables ts y ls almacenan las líneas y etiquetas respectivamente, hay que recordar que en la línea cada muestra en SpaCy consta de una línea de 11 palabras donde se analiza la sexta palabra y de una lista de 3 elementos donde en el tercero esta etiquetada la clase a la que pertenece la línea.

A continuación creamos un dataframe usando la librería de Pandas donde se introduce la palabra central del texto y la palabra inmediatamente siguiente además del texto y las etiquetas de las clases como se observa en la imagen. 5.1 Buscamos analizar la palabra central "world" y para agregar contexto la siguiente "next world".

```
ts = [t for t,v in data if len(v['entities']) !=0]
ls = [l for t,v in data if len(v['entities']) !=0 for _,_,l in v['entities']]


F = pd.DataFrame()
F["Word"] = [t.split()[int(len(t.split())/2)] for t in ts]
F["NextWord"] = [t.split()[int(len(t.split())/2)+1] for t in ts]
F["Text"] = ts
F["Label"] = ls
F.index.names = ['Id']
```

Ilustración 5.1: De factura en SpaCy a dataframe de Pandas

Para poder usar las etiquetas primero tenemos codificarlas, donde gracias a esto cada etiqueta de cada clase estará asociada a un número que la representa, que utilizará el clasificador para trabajar con las etiquetas sin tener que trabajar con cadenas de caracteres. 5.1

Las etiquetas teniendo ya un código asignado en números son remplazados por sus propios códigos en el dataframe y que es formado a partir del dataframe X extrayendo solo las etiquetas, este proceso se realiza gracias al método applymap que reemplaza cada elemento del dataframe por el valor en el diccionario label_codes que es el conjunto clave valor que relaciona las etiquetas con sus códigos como vemos en la siguiente tabla.5.1

A su vez son separadas del dataframe principal que es el que contiene todos los datos quedando dos dataframes X e Y que son los datos y las etiquetas ya representadas por su código. A su vez estos dos dataframes son separados en los pares de entrenamiento y validación

formando 4 grupos, las líneas de texto para entrenamiento, las etiquetas correspondientes a esas líneas para entrenamiento, líneas de texto para validación y sus respectivas etiquetas para validación. Las muestras de cada uno de los grupos es barajada para evitar overfitting. 5.2

```
X, labels = spacy_to_multiclass_dataframe(X)
y = pd.DataFrame(X["Label"], columns = ["Label"])
y = y.applymap(lambda x: label_codes[x])

X.drop("Label", axis=1, inplace=True)
X_train, X_valid, y_train, y_valid = train_test_split(
    X, y, train_size=0.8, test_size=0.2, random_state=0,
    shuffle=True
)
```

Ilustración 5.2: Dataframes de datos(X) y Etiquetas(y) para test y validación

Etiqueta	**Código**
Comercializadora	3
ConsumoEnergia	12
CP	71
CUPS	14
Direccion	18
Distribuidora	19
FechaEmision	26
FechaCargo	27
FechaFin	28
FechaInicio	29
FinContrato	30
ImporteAlquiler	38
ImporteEnergia	39
ImporteImpuestos	43
ImportePotencia	46
ImporteTotal	47
NIF	54
Nombre	55
NumeroContrato	77
NumeroFactura	57
PeajeAcceso	58
PotenciaContratada	59
Poblacion	60
Provincia	85
Other	100

Cuadro 5.1: Codificación de las etiquetas(labels)

5.3. Métodos basados en técnicas de aprendizaje automático

Con los 2 pares de muestras (dataframes de entrenamiento) solo queda entrenar los clasificadores. En el entrenamiento usamos 3 métodos de aprendizaje automático: Naive Bayes Multinomial, "Support Vector Machine" con el kernel "Radial Basis Function" (SVM-RBF) y Stochastic Gradient Descent cada uno para un clasificador y con los cuales una vez tengamos los resultados podremos compararlos, lo que ayudará a ver en qué clases se desempeña mejor cada uno y decidirse por el mejor en cuestiones de rendimiento a la hora de cometer menos fallos en la clasificación. Estos algoritmos estarán brevemente explicados en los siguientes apartados con la información en línea y en especial, la que proporciona scikit learn [13][6].

5.3.1. Naive Bayes Multinomial

Hemos comentado en anteriores apartados (2.1.3) sobre Naive Bayes que como algoritmos de clasificación son un conjunto de métodos de algoritmos de aprendizaje automático supervisado basado en aplicar el teorema de Bayes con la suposición "inocente" o "naive" en inglés, de independencia condicional entre cada par de características dado el valor de la variable de clase.

El teorema de Bayes establece la siguiente relación según la documentación ofrecida en Scikit Learn[6][7], dada la variable de clase y y el vector de características dependiente x_1 a traves de x_n:

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots, x_n \mid y)}{P(x_1, \ldots, x_n)} \tag{5.1}$$

Utilizando el supuesto "ingenuo" de independencia condicional:

$$P(x_i|y, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(x_i|y), \tag{5.2}$$

para todos i, esta relación se simplifica a:

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \ldots, x_n)} \tag{5.3}$$

$P(x_1, \ldots, x_n)$ es constante dando la entrada (*input*), podemos utilizar la siguiente regla de clasificación:

$$\begin{gathered} P(y \mid x_1, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y) \\ \Downarrow \\ \hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y), \end{gathered} \tag{5.4}$$

Continúan documentando en Scikit Learn que con todo lo anterior podemos utilizar la estimación Máximo A Posteriori (MAP) para estimar $P(y)$ y $P(x_i \mid y)$; el primero es la frecuencia relativa de la clase y en el conjunto de entrenamiento.

Los diferentes clasificadores Bayesianos "ingenuos"(*Naives Bayes*) difieren principalmente por las suposiciones que hacen con respecto a la distribución de $P(x_i \mid y)$.

A pesar de sus supuestos aparentemente demasiado simplificados, los clasificadores bayesianos ingenuos han funcionado bastante bien en muchas situaciones del mundo real, como la clasificación de documentos y el filtrado de spam. Requieren una pequeña cantidad de datos de entrenamiento para estimar los parámetros necesarios.

Los clasificadores y aprendices de Naive Bayes pueden ser extremadamente rápidos en comparación con métodos más sofisticados. La disociación de las distribuciones de características condicionales de clase significa que cada distribución puede estimarse independientemente como una distribución unidimensional. Esto, a su vez, ayuda a aliviar los problemas derivados de la dimensionalidad.

Por otro lado, aunque es sabido que el método Naive Bayes es un clasificador decente, es conocido como un mal estimador, por lo que los resultados de probabilidad desde predict_proba no deben tomarse demasiado en serio.

La documentación sobre el método Naive Bayes Multinomial de Scikit Learn afirma que el método MultinomialNB implementa el algoritmo Naive Bayes para datos distribuidos multinomialmente, y es una de las variantes clásicas de Naive Bayes que se utilizan en la clasificación de texto donde los datos se representan normalmente como recuentos de arrays de palabras es decir en los Term Frequency (TF), aunque también los arrays tf-idf (Term Frequency - Inverse document frequency), funcionan bien en la práctica. La distribuciones parametrizadas por vectores $\theta_y = (\theta_{y1}, \ldots, \theta_{yn})$ para cada clase y, dónde n es el número de características (en la clasificación de texto, el tamaño del vocabulario) y θ_{yi} es la probabilidad $P(x_i \mid y)$ de característica i que aparece en una muestra perteneciente a la clase y.

Los parámetros θ_y se estima mediante el recuento de la frecuencia relativa:

$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n} \tag{5.5}$$

dónde $N_{yi} = \sum_{x \in T} x_i$ es el número de veces que aparece la característica i en una muestra de clase y en el conjunto de entrenamiento T, y $N_y = \sum_{i=1}^{n} N_{yi}$ es el recuento total de todas las características de la clase y.

Los priores $\alpha \geq 0$ tiene en cuenta características que no están presentes en las muestras de aprendizaje y evita probabilidad 0 en cálculos posteriores. Establecer $\alpha = 1$ se le llama suavizado de Laplace, mientras que $\alpha < 1$ es llamado suavizado Lidstone.

5.3.2. Support Vector Machine (Kernel RBF)

Los support vector machines (SVMs) según podemos ver en la propia documentación de Scikit Learn,[8] son un conjunto de métodos usados para la clasificación, regresión y la detección de valores atípicos. Si bien explicamos en apartados anteriores2.1.2 los aspectos generales de los SVM más concretamente acerca del kernel linearSVC ahora trataremos de profundizar un poco más.

Los SVM son muy eficaces en espacios de gran dimensionalidad, son efectivos donde el número de dimensiones es mayor al de muestras, es eficiente a nivel de memoria y versátil ya que podemos configurar diversos kernels para la función de decisión, sin embargo se destaca que se debe elegir el núcleo y la regularización con cuidado en caso de que sea muy superior en número de características que el de muestras y otra desventaja es que las estimaciones las realiza de forma muy costosa realizando validación cruzada.

En SkLearn se usan las clases SVC, NuSVC y linearSVC para definir los clasificadores SVM, donde SVC y NuSVC son similares pero aceptan parámetros ligeramente diferentes y tienen formulaciones matemáticas distintas y en el caso de linearSVC es otra implementación de SVC donde usa un kernel lineal, además no usa el atributo kernel pues ya esta establecido con un kernel lineal y carece de algunos atributos en comparación con las otras 2 clases mencionadas.2.1.2

SVC, NuSVCy linearSVC son clases capaces de realizar clasificación binaria y multiclase en un conjunto de datos.5.3

Para problemas de regresión también existen clases equivalentes a las anteriores las cuales son SVR, NuSVR y linearSVR que tiene entre ellas las mismas características antes explicadas pero para problemas de regresión.

Al entrenar una SVM con el kernel de función de base radial (RBF), se deben considerar dos parámetros: C y gamma. El parámetro C, es común a todos los núcleos de SVM y compensa la clasificación errónea de los ejemplos de entrenamiento frente a la simplicidad de la superficie de decisión. Un C bajo suaviza la superficie de decisión, mientras que un C alto apunta a clasificar correctamente todos los ejemplos de entrenamiento. Gamma define cuánta influencia tiene un solo ejemplo de entrenamiento. Cuanto más grande gamma es, más cerca deben estar otros ejemplos para verse afectados.

La elección adecuada de C y gamma es fundamental para el rendimiento de la SVM. Se recomienda usar GridSearchCV con C y gamma para elegir buenos valores.

Entrando más en lo que es el núcleo de función de base radial (*Radial basis function kernel*) también conocido como núcleo exponencial al cuadrado (*squared-exponential kernel*) podemos consultar la documentación de Scikit Learn donde podemos ver lo siguiente: El núcleo RBF es un núcleo estacionario. También se lo conoce como núcleo "exponencial al cuadrado". Está parametrizado por un parámetro de escala de longitud $l > 0$, que puede ser un escalar (variante isotrópica del núcleo) o un vector con el mismo número de dimensiones

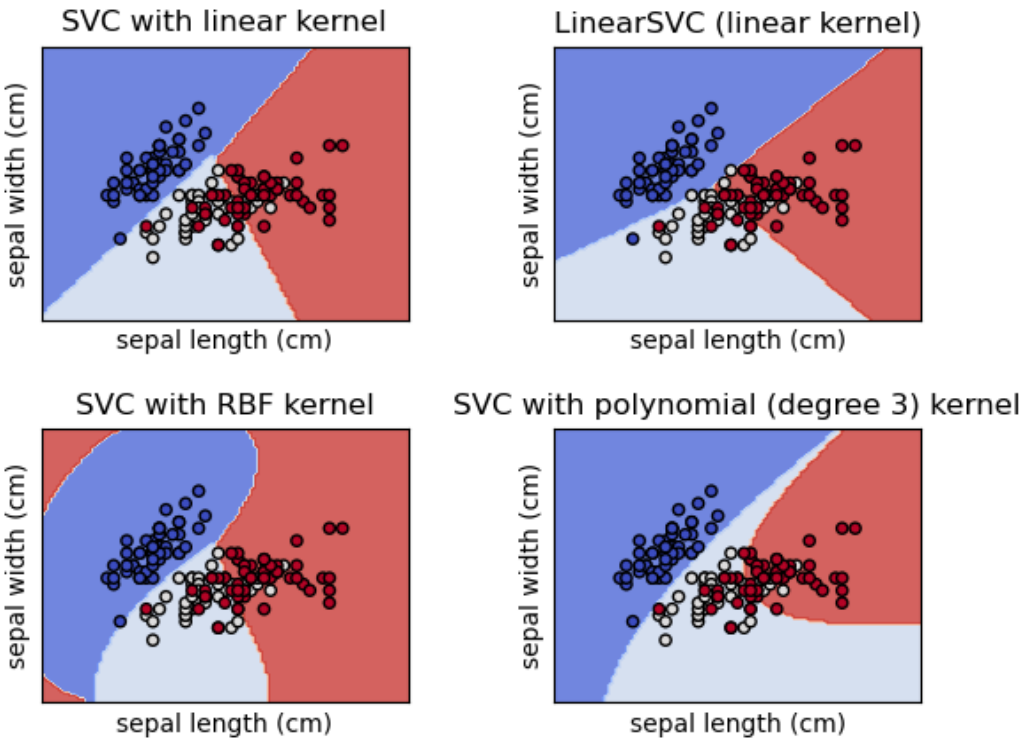

Ilustración 5.3: Imagen propiedad de SKlearn que compara 4 clasificadores SVC https://scikit-learn.org/stable/modules/svm.html

que las entradas de X (variante anisotrópica del núcleo). El núcleo está definido por:

$$k(x_i, x_j) = \exp\left(-\frac{d(x_i, x_j)^2}{2l^2}\right) \tag{5.6}$$

dónde l es la escala de longitud del núcleo y $d(\cdot, \cdot)$ es la distancia euclídea. Este núcleo es infinitamente diferenciable, lo que implica que los procesos gausianos (*Gaussian Processes o GP*) con este núcleo como función de covarianza tienen derivadas cuadráticas medias de todos los órdenes y, por lo tanto, son muy suaves.

5.3.3. Stochastic Gradient Descent

Stochastic Gradient Descent también ha sido explorado con anterioridad en el presente TFG 2.1.4. De Stochastic Gradient Descent podemos decir gracias a la documentación aportada por Scikit Learn[9] que es una forma eficiente y sencilla de ajustar clasificadores lineales.

Se considera más un optimizador debido a que implementa modelos líneales regularizados con aprendizaje de descenso de gradiente estocástico (SGD). Sus ventajas consisten en ser eficiente y en su facilidad de implementación mientras que las desventajas son el gran número de parámetros que necesita y cuando las características aumentan mucho su número empieza a ser menos efectivo.

Para contextualizar, en la propia documentación indica que es recientemente cuando se le ha prestado atención debido a su capacidad para ser usado en aprendizaje a gran escala y ha dado buenos resultados en problemas del tipo sparse, problemas de aprendizaje automático comunes en clasificación de texto y procesamiento de lenguaje natural. Dado que los datos son dispersos, los clasificadores de este módulo se adaptan fácilmente a problemas con más de 10^5 ejemplos de entrenamiento y más de 10^5 características.

En sklearn la clase SGDClassifier se puede configurar con diferentes funciones de pérdidas y de penalización ajustándose esto en sus parámetros. Al igual que otros clasificadores, SGD debe estar equipado con dos matrices: una matriz X de forma (n_muestras, n_características) que contiene las muestras de entrenamiento, y una matriz y de forma (n_muestras), que contiene los valores objetivo (etiquetas de clase) para las muestras de entrenamiento:

```
from sklearn.linear_model import SGDClassifier
X = [[0., 0.], [1., 1.]]
y = [0, 1]
clf = SGDClassifier(loss="hinge", penalty="l2", max_iter=5)
clf.fit(X, y)
```

5.4. Creación y entrenamiento de los clasificadores

En la creación de los clasificadores se usa la clase Pipeline como una forma de poder configurar distintos atributos que el clasificador tendrá que usar como el tf-idf, las stop-words, el wordvectorizer y el modelo concreto de clasificador que se va a usar, de esta forma se declara todo de forma sencilla y fácil de modificar. 5.4

```
clf1[label] = Pipeline([
    ('features', FeatureUnion([
        ('tfidf', TfidfVectorizer(stop_words=stop_words,
                  ngram_range=(1, 2), use_idf=True)),
        ('word_features', WordVectorizer(all=True))
    ])),
    ('clf', create_model('nbayes'))
])
```

Ilustración 5.4: Declaración de clasificador con Pipeline

Como información adicional sobre la clase pipeline y los pipelines en general podemos decir lo siguiente, que los pipelines son una herramienta (métodos de sklearn en este caso) en el desarrollo del aprendizaje automático, especialmente al trabajar con algoritmos como las Support Vector Machines (SVMs).

En el aprendizaje automático, un pipeline es una secuencia de pasos que se utilizan para procesar y transformar los datos antes de aplicar un modelo de machine learning, es decir, los clasificadores que serán entrenados usando las muestras. Estos pasos pueden incluir desde la limpieza y preprocesamiento de datos hasta la selección de características y la creación del modelo en sí, como en la imagen anterior. 5.4

Las ventajas de utilizar Pipelines son por ejemplo, que se evita la fuga de estadísticas. Los pipelines ayudan a evitar que las estadísticas de los datos de prueba se filtren al modelo entrenado durante la validación cruzada. También facilitan la reproducibilidad, al encapsular todos los pasos en un solo objeto, se facilita la reproducción del flujo de trabajo y se simplifica. Además podemos decir que mejora la organización y mantenimiento del código ya que permite una organización clara de los pasos de preprocesamiento y modelado, lo que facilita el mantenimiento y la depuración del código.

Al utilizar la clase Pipeline para entrenar a los clasificadores, se pueden combinar múltiples pasos en un solo objeto que actúa como un clasificador. Esto incluye la normalización de datos, selección de características y entrenamiento del modelo, todo en un solo flujo de trabajo.

Podemos encontrar las siguientes recomendaciones para el uso de Pipelines:

Hay que asegurar el incluir todas las etapas necesarias de preprocesamiento de datos en el pipeline, como la estandarización de características y la selección de características relevantes. Optimización de hiperparámetros utilizando técnicas para encontrar los mejores hiperparámetros para el modelo dentro del pipeline y la validación cruzada que emplea la

validación para evaluar el rendimiento del modelo entrenado a través del pipeline y garantizar su generalización.

Los pipelines son una herramienta común en el aprendizaje automático que permite organizar y automatizar el flujo de trabajo, especialmente al entrenar modelos como las Support Vector Machines (SVMs) u otros modelos. Al utilizar la clase Pipeline de manera efectiva, se puede mejorar la eficiencia y la reproducibilidad al codificar los modelos a usar y tenerlos organizados.

En cuanto a las stop words son palabras que se usan frecuentemente en un idioma pero que no aportan información relevante en la extracción de datos ni análisis del texto como "de","a", "el", etc. 5.5 y dificultan la labor del tf-idf.

Para que el clasificador las ignoren apuntamos en un fichero txt las palabras en cuestión que consideremos stop words y al leer los datos la cargamos en una variable que luego al introducirla en el pipeline hará que el clasificador no las tenga en cuenta a la hora de utilizar el tf-idf. Esto es así ya que las stop-words pueden afectar negativamente la precisión del modelo al tener un alto índice de frecuencia en los documentos.

```
['_', 'a', 'aquella', 'aquellas', 'aquel', 'aquello', 'aquellos', 'ante',
'bajo', 'con', 'contra', 'de', 'del', 'desde', 'durante', 'en', 'este',
'esta', 'estos', 'estas', 'entre', 'el', 'hacia', 'hasta', 'mediante', 'la',
'las', 'los', 'para', 'por', 'según', 'sin', 'sobre', 'tras', 'un', 'una',
'unos', 'unas', 'versus', 'vía']
```

Ilustración 5.5: Stops words en un fichero de texto

Podemos pasar a explicar terminos como TF y TF-IDF. En el procesamiento de lenguaje natural y la extracción de información, dos conceptos fundamentales son el TF (Frecuencia de Término) y el TF-IDF (Frecuencia de Término-Inversa de Frecuencia de Documento). Estas métricas desempeñan un papel crucial en la representación y el conteo de términos en un texto.

La Frecuencia de Término (TF) es una métrica que se utiliza para medir la importancia de una palabra en un documento específico dentro de un texto. Básicamente, el TF de una palabra en un documento es la cantidad de veces que esa palabra aparece en ese documento. Se calcula dividiendo el número de veces que aparece un término en un documento por el total de términos en ese documento.

Por otro lado, el TF-IDF combina la Frecuencia de Término (TF) con la Inversa de la Frecuencia de Documento (IDF). Mientras que el TF se centra en la importancia de una palabra en un documento específico, el TF-IDF va un paso más allá al considerar la relevancia de esa palabra en el conjunto de documentos. El objetivo del TF-IDF es dar mayor peso a las palabras que son importantes en un documento pero relativamente poco comunes en todo el texto.

Podemos diferéncialos de la siguiente manera:

TF (Frecuencia de Término):
Se calcula como la frecuencia de una palabra en un documento específico. No tiene en cuenta la importancia relativa de la palabra en el texto completo. Puede llevar a una sobreponderación de palabras comunes que no son necesariamente significativas.

TF-IDF (Frecuencia de Término-Inversa de Frecuencia de Documento):
Considera tanto la frecuencia de una palabra en un documento como su rareza en el texto. Penaliza las palabras demasiado comunes y otorga mayor peso a las palabras más distintivas. Ayuda a identificar términos clave que son relevantes para un documento en particular.

Por lo tanto podemos decir que TF y TF-IDF permiten representar documentos de texto de manera numérica para su procesamiento por algoritmos de aprendizaje automático. Ayudan a asignar pesos a los términos en función de su importancia relativa en un documento. Al penalizar las palabras comunes y destacar las palabras distintivas, TF-IDF contribuye a reducir el ruido en el análisis de texto.

Estas métricas son esenciales para tareas como la clasificación de texto, la recuperación de información y análisis de texto en general.

La variable "use_idf" especifica si al calcular el term frequency (tf) se le aplicaría o no el "inverse document frequency" es decir usar el tf-idf, mientras que la variable ngram_range indica el rango de los n-gramas que se analizarán, donde (1,1) son unigramas mientras que con el parámetro (1,2) serían bigramas.

Por ultimo la clase Wordvectorized es la encargada de extraer las características del dataframe. Gracias a la asistencia del tutor en la realización de este código se ha sido capaz de analizar características relevantes a ser examinadas y el como realizar el proceso de extracción de características. Por defecto extrae las características de la palabra central del texto guardado en el dataframe y también de la palabra anterior y siguiente 5.6, en caso de usar el parámetro "all" extraerá las características de todas las palabras del texto del datagrama.

La clase almacena el trío de palabras antes mencionada(o todas si se ha usado el parámetro "all") la siguiente información sobre la palabra y para esto usará métodos previamente definidos los cuales usarán tanto expresiones regulares como mapas y otros métodos de control de cadenas de caracteres:

El método word_type:5.7 si es alfabetico, alfanumerico, número con comas o solo numérico.

El método word_pretype: Si es una cifra de dinero, un email, un enlace web, un dni, un código postal o una fecha.

El método word_mesure: Que tipo de medida se encuentra. Por ejemplo: €, €/día, €/kw, €/kwh, eur, eur/día, eur/kw, eur/kwh, euros, euros/día, euros/kw, euros/kwh, kw, kw/mes, kwh, %

El método word_capital: Analiza si la palabra tiene la primera letra mayúscula, si no tiene letra mayúscula, si todas las letras son mayúsculas y si contiene alguna letra mayúscula.

El método word_colons: Analiza si la palabra va seguida de un punto y coma o no.

```
columns = ["Type", "Pretype", "Measure", "Capital", "Colons",
           "NType", "NPretype", "NMeasure", "NCapital", "NColons",
           "PType", "PPretype", "PMeasure", "PCapital", "PColons"]

# Extract middle word and the following word
F = pd.DataFrame(index = X.index)
F["Word"]     = [t.split()[int(len(t.split())/2)] for t in X]
F["NextWord"] = [t.split()[int(len(t.split())/2)+1] for t in X]
F["PrevWord"] = [t.split()[int(len(t.split())/2)-1] for t in X]

# Extract cutom word features
W[columns[0]]  = self.word_type(F['Word'])
W[columns[1]]  = self.word_pretype(F['Word'])
W[columns[2]]  = self.word_mesure(F['Word'])
W[columns[3]]  = self.word_capital(F['Word'])
W[columns[4]]  = self.word_colons(F['Word'])
W[columns[5]]  = self.word_type(F['NextWord'])
W[columns[6]]  = self.word_pretype(F['NextWord'])
W[columns[7]]  = self.word_mesure(F['NextWord'])
W[columns[8]]  = self.word_capital(F['NextWord'])
W[columns[9]]  = self.word_colons(F['NextWord'])
W[columns[10]] = self.word_type(F['PrevWord'])
W[columns[11]] = self.word_pretype(F['PrevWord'])
W[columns[12]] = self.word_mesure(F['PrevWord'])
W[columns[13]] = self.word_capital(F['PrevWord'])
W[columns[14]] = self.word_colons(F['PrevWord'])
```

Ilustración 5.6: Wordvectorized por defecto

Como podemos ver a la hora de analizar una palabra la clase wordvectorized debe extraer información de los tipos indicados anteriormente en relación a la palabra y sus "vecinos" o todas las de la línea si se usa el parámetro all como se especifico previamente. Esto ayudará en gran medida a la hora de obtener el contexto de las palabras y pudiendo ajustar el clasificador lo mejor posible para que sea capaz de extraer la información relevante del texto tanto en la validación como en caso de funcionamiento normal del predictor.

```
money = '^\d+[,.]\d\d$'
email = "^[a-zA-Z0-9.!#$%&'*+/=?^_`{|}~-]+@[a-zA-Z0-9](?:[a-zA-Z0-9-]{0,61}"\
        "[a-zA-Z0-9])?(?:\.[a-zA-Z0-9](?:[a-zA-Z0-9-]{0,61}[a-zA-Z0-9])?)*$"
web   = "^(https?:\/\/)?(www\.)?([a-zA-Z0-9]+(-?[a-zA-Z0-9])*\.)+[\w]{2,}"\
        "(\/\S*)?$"
dni1  = '^[klmxyzKLMXYZ][-]?\d{7}[-]?[a-zA-Z]$'
dni2  = '^\d{8}[-]?[a-zA-Z]$'
pcode = "^[0-5]\d{4}$"
date  = "^\d\d[/\-.]\d\d[/\-.]\d{4}$"
```

Ilustración 5.7: Expresiones regulares para detectar características

Una vez creado el estimador se usa el método fit con los conjuntos de entrenamiento tanto con los textos como las etiquetas, el cual aplicará todo el entrenamiento del clasificador, para verificar los resultados con los conjuntos de validación se usa el método evaluate al cual se le pasa como parámetros tanto dichos conjuntos como el clasificador, donde se usará el método predict con el conjunto de validación. En el método evaluate también se hará uso de la librería seaborn para mostrar mapas de calor de los fallos y aciertos cometidos por el clasificador además de la clase metrics de sklearn para mostrar esa misma información como una matriz o como una tabla 5.8.

Con la clase seaborn podemos crear diferentes gráficas entre ellas los mapas de calor:

Los mapas de calor son una forma visualmente efectiva de representar datos de manera gráfica. En Python, la librería seaborn ofrece herramientas para crear mapas de calor y visualizar resultados de algoritmos como las Support Vector Machines (SVMs).

Un mapa de calor esencialmente es una representación gráfica de datos donde los valores se representan con colores para indicar su magnitud. En el caso de seaborn, los mapas de calor son útiles para visualizar matrices de datos de manera clara y comprensible, lo que facilita la identificación de patrones y tendencias, en este caso donde el clasificador ha etiquetado cada palabra, es decir, en qué clase, con respecto a la clase a la que verdaderamente pertenece, esto gracias a las etiquetas que usamos en el conjunto de validación, nos servirán para ver más fácilmente los errores que cometió el estimador.

Seaborn es una librería de visualización de datos en Python que ofrece unos métodos y clases simples y potentes para crear gráficos atractivos. Para generar un mapa de calor con seaborn, se pueden utilizar funciones como heatmap que permiten visualizar matrices de datos de forma matricial con colores que representan los valores.

Al utilizar las matrices resultado de un algoritmo por ejemplo de Support Vector Machines (SVMs), los mapas de calor pueden ser una herramienta útil para visualizar la distribución de las predicciones del modelo como explicamos anteriormente. Por ejemplo, al representar una matriz de confusión generado por un SVM, un mapa de calor puede resaltar las áreas con mayores aciertos y errores de clasificación de forma intuitiva y rápida.

Una ventaja de utilizar mapas de calor es su visualización clara ya que los mapas de calor facilitan la identificación de patrones en las matrices resultado del SVM u otro modelo de manera visual y rápida.

Además ofrecen una interpretación sencilla mediante colores y graduaciones, los mapas de calor permiten interpretar los datos de una forma más intuitiva que simplemente revisando números. Al combinar los resultados del clasificador con mapas de calor, se puede obtener una perspectiva adicional que puede ayudar en la toma de decisiones.

Los mapas de calor de seaborn son útiles para visualizar los resultados de clasificadores. Su uso con matrices resultado de una estimación de un clasificador puede proporcionar una representación gráfica clara y efectiva como podremos ver en el apartado de conclusiones.6.11

- MAE: 1.1704662876813314

	precision	recall	f1-score	support
Comercializadora	0.92	0.88	0.90	7279
ConsumoEnergia	0.96	1.00	0.98	1042
CUPS	1.00	0.89	0.94	989
Direccion	0.91	0.98	0.94	10974
Distribuidora	0.97	0.96	0.97	1901
FechaEmision	1.00	1.00	1.00	865
FechaCargo	0.93	1.00	0.96	841
FechaFin	0.99	0.92	0.95	1206
FechaInicio	1.00	1.00	1.00	1174
FinContrato	0.90	1.00	0.95	911
ImporteAlquiler	1.00	0.89	0.94	1208
ImporteEnergia	0.95	0.96	0.95	802
ImporteImpuestos	0.97	0.93	0.95	1169
ImportePotencia	1.00	0.76	0.86	898
ImporteTotal	0.99	0.91	0.95	2413
NIF	1.00	0.99	0.99	1182
Nombre	0.96	0.88	0.92	5729
NumeroFactura	0.97	0.92	0.94	1100
PeajeAcceso	0.97	0.84	0.90	969
PotenciaContratada	1.00	0.98	0.99	1280
Poblacion	0.86	0.73	0.79	3659
CP	0.98	1.00	0.99	1166
NumeroContrato	0.81	1.00	0.89	1134
Provincia	0.88	0.62	0.73	1463
Other	0.99	0.99	0.99	272845
accuracy			0.98	324199
macro avg	0.96	0.92	0.94	324199
weighted avg	0.98	0.98	0.98	324199

Ilustración 5.8: Tabla de resultados de estimador SGD

Capítulo 6
Resultados

6.1. Comparación de Resultados

En el entrenamiento usamos 3 conjuntos de muestras con los mismos tipos de facturas de los cuales solo varían el número de muestras teniendo el conjunto short, medium y large como los tres conjuntos de muestras cada uno con 32485, 97294 y 324199 muestras respectivamente. El tiempo de entrenamiento para los clasificadores NB y SGD es muy bajo en el caso más exigente que es con el mayor conjunto de muestras (large). Este tiempo es inferior a los 10 minutos no siendo así para el SVM que llega a las 24h aproximadamente con el mayor número de muestras.

Para comparar los resultados compararemos los errores de los 3 clasificadores en cada tamaño de muestras, estos errores son *precision* o precisión, que es la capacidad de no etiquetar una muestra a un campo al que no pertenece, el *recall* o exhaustividad, que es la capacidad de etiquetar correctamente todas las muestras que pertenecen a un campo a ese campo sin falsos negativos, el f-score, que es la ponderación de *precision* y *recall*, y Meant Absolute Error(MAE) o error absoluto medio que es la suma de los errores cometidos por el clasificador. La columna support indicara las muestras totales.

Los datos mostrados son los obtenidos de la clase metrics usando el método classification_report y luego pasados a un documento word como texto donde cambiando todos los espacios en blanco por una única coma entre palabras luego se pasa todo el texto a tablas y se vuelven a pasar a excel para asegurarse de que tienen la misma disposición entre las tablas de los 3 clasificadores de un mismo tamaño de muestra.

Naive Bayes Short Samples

	precision	recall	f-score	support
Comercializadora	0.72	0.99	0.84	742
ConsumoEnergia	0.66	0.92	0.77	98
CUPS	0.58	0.9	0.71	98
Direccion	0.85	0.98	0.91	1157
Distribuidora	0.7	0.99	0.82	168
FechaEmision	0.89	1	0.94	111
FechaCargo	0.69	1	0.82	79
FechaFin	0.92	0.91	0.91	119
FechaInicio	0.93	0.89	0.91	111
FinContrato	0.87	1	0.93	93
ImporteAlquiler	0.87	1	0.93	134
ImporteEnergia	0.39	0.84	0.54	83
ImporteImpuestos	0.5	0.91	0.64	134
ImportePotencia	0.47	1	0.64	89
ImporteTotal	0.75	1	0.86	248
NIF	0.9	0.99	0.95	122
Nombre	0.66	0.99	0.79	619
NumeroFactura	0.43	0.93	0.59	98
PeajeAcceso	0.53	1	0.69	108
PotenciaContratada	0.9	1	0.95	122
Poblacion	0.65	0.69	0.67	389
CP	0.73	1	0.84	122
NumeroContrato	0.85	1	0.92	111
Provincia	0.48	0.77	0.59	152
Other	1	0.93	0.96	27178
accuracy			0.94	32485
Macro avg	0.72	0.95	0.8	32485
Weighted avg	0.96	0.94	0.94	32485
MAE	3.63398491611513			

Ilustración 6.1: Tabla de resultados NB muestras tamaño short

Support Vector Machine Short Samples

	precision	recall	f-score	support
Comercializadora	1	0.99	0.99	742
ConsumoEnergia	1	1	1	98
CUPS	1	1	1	98
Direccion	0.99	1	0.99	1157
Distribuidora	1	0.99	1	168
FechaEmision	1	1	1	111
FechaCargo	1	1	1	79
FechaFin	0.89	1	0.94	119
FechaInicio	1	1	1	111
FinContrato	1	0.84	0.91	93
ImporteAlquiler	1	1	1	134
ImporteEnergia	1	1	1	83
ImporteImpuestos	0.99	1	1	134
ImportePotencia	0.99	1	0.99	89
ImporteTotal	1	1	1	248
NIF	1	1	1	122
Nombre	1	1	1	619
NumeroFactura	1	1	1	98
PeajeAcceso	1	1	1	108
PotenciaContratada	1	1	1	122
Poblacion	0.96	0.95	0.96	389
CP	1	0.99	1	122
NumeroContrato	1	1	1	111
Provincia	0.98	0.91	0.95	152
Other	1	1	1	27178
accuracy			1	32485
macro avg	0.99	0.99	0.99	32485
weighted avg	1	1	1	32485
MAE	0.08339233492381098			

Ilustración 6.2: Tabla de resultados SVM muestras tamaño short

Stochastic Gradient Descent Short Samples

	precision	recall	f-score	support
Comercializadora	0.96	0.81	0.88	742
ConsumoEnergia	0.95	0.92	0.93	98
CUPS	0.94	0.84	0.89	98
Direccion	0.96	0.88	0.92	1157
Distribuidora	0.89	0.98	0.93	168
FechaEmision	1	1	1	111
FechaCargo	1	1	1	79
FechaFin	0.97	0.91	0.94	119
FechaInicio	1	1	1	111
FinContrato	0.88	1	0.93	93
ImporteAlquiler	0.99	1	0.99	134
ImporteEnergia	1	0.45	0.62	83
ImporteImpuestos	0.97	0.87	0.92	134
ImportePotencia	0.99	0.75	0.85	89
ImporteTotal	0.94	0.92	0.93	248
NIF	1	0.92	0.96	122
Nombre	0.98	0.8	0.88	619
NumeroFactura	0.96	0.93	0.94	98
PeajeAcceso	0.96	0.87	0.91	108
PotenciaContratada	1	0.91	0.95	122
Poblacion	0.72	0.79	0.76	389
CP	0.82	1	0.9	122
NumeroContrato	0.87	1	0.93	111
Provincia	0.88	0.6	0.71	152
Other	0.98	0.99	0.99	27178
accuracy			0.97	32485
macro avg	0.94	0.89	0.91	32485
weighted avg	0.97	0.97	0.97	32485
MAE	1.5655841157457289			

Ilustración 6.3: Tabla de resultados SGD muestras tamaño short

De los datos obtenidos durante el entrenamiento con el menor tamaño de muestras, el tamaño short, que se muestran en las tablas 6.1, 6.2 y 6.3 podemos concluir que el mejor clasificador es el SVM aunque también es el que más recursos consume, donde NB es el que menos consume de los 3 y al mismo tiempo el más sencillo es el que menos resultados correctos ofrece durante el entrenamiento.

Esta misma tendencia se mantiene para el resto de tamaños de muestra donde SVM da el mejor resultado y NB el peor, con la diferencia de que en términos generales todos los clasificadores dan mejor resultado a mayor sea el tamaño de muestra dando menos errores en el tamaño de muestra large.

Naive Bayes Medium Samples

	precision	recall	f-score	support
Comercializadora	0.72	0.99	0.83	2168
ConsumoEnergia	0.68	0.89	0.77	299
CUPS	0.66	0.9	0.76	300
Direccion	0.87	0.98	0.92	3325
Distribuidora	0.75	1	0.86	489
FechaEmision	0.87	1	0.93	268
FechaCargo	0.8	1	0.89	235
FechaFin	0.89	0.91	0.9	343
FechaInicio	0.9	0.92	0.91	353
FinContrato	0.88	1	0.94	306
ImporteAlquiler	0.83	1	0.9	349
ImporteEnergia	0.45	0.94	0.61	256
ImporteImpuestos	0.51	0.97	0.66	345
ImportePotencia	0.65	1	0.79	308
ImporteTotal	0.81	1	0.89	784
NIF	0.92	1	0.95	389
Nombre	0.66	0.99	0.79	1781
NumeroFactura	0.53	0.93	0.68	325
PeajeAcceso	0.67	1	0.8	298
PotenciaContratada	0.87	1	0.93	363
Poblacion	0.74	0.71	0.72	1127
CP	0.83	1	0.91	387
NumeroContrato	0.83	1	0.91	320
Provincia	0.52	0.8	0.63	458
Other	1	0.94	0.97	81718
accuracy			0.95	97294
macro avg	0.75	0.95	0.83	97294
weighted avg	0.96	0.95	0.95	97294
MAE	3.1588689950048305			

Ilustración 6.4: Tabla de resultados NB muestras tamaño medium

Support Vector Machine Medium Samples

	precision	recall	f-score	support
Comercializadora	1	1	1	2168
ConsumoEnergia	1	1	1	299
CUPS	1	1	1	300
Direccion	0.99	1	1	3325
Distribuidora	1	1	1	489
FechaEmision	1	1	1	268
FechaCargo	1	1	1	235
FechaFin	0.91	1	0.95	343
FechaInicio	1	1	1	353
FinContrato	1	0.89	0.94	306
ImporteAlquiler	1	1	1	349
ImporteEnergia	1	1	1	256
ImporteImpuestos	1	1	1	345
ImportePotencia	1	1	1	308
ImporteTotal	0.98	1	0.99	784
NIF	1	1	1	389
Nombre	1	1	1	1781
NumeroFactura	1	1	1	325
PeajeAcceso	1	1	1	298
PotenciaContratada	1	1	1	363
Poblacion	0.98	0.97	0.98	1127
CP	1	1	1	387
NumeroContrato	1	1	1	320
Provincia	0.98	0.96	0.97	458
Other	1	1	1	81718
accuracy			1	97294
macro avg	0.99	0.99	0.99	97294
weighted avg	1	1	1	97294
MAE	0.046837420601475936			

Ilustración 6.5: Tabla de resultados SVM muestras tamaño medium

Stochastic Gradient Descent Medium Samples

	precision	recall	f-score	support
Comercializadora	0.85	0.95	0.9	2168
ConsumoEnergia	0.92	0.98	0.95	299
CUPS	1	0.91	0.95	300
Direccion	0.82	1	0.9	3325
Distribuidora	0.96	0.93	0.94	489
FechaEmision	1	1	1	268
FechaCargo	0.99	1	0.99	235
FechaFin	0.95	0.91	0.93	343
FechaInicio	1	1	1	353
FinContrato	0.91	1	0.95	306
ImporteAlquiler	1	0.92	0.96	349
ImporteEnergia	0.97	0.93	0.95	256
ImporteImpuestos	1	0.75	0.86	345
ImportePotencia	1	0.67	0.8	308
ImporteTotal	0.99	0.9	0.94	784
NIF	0.98	0.99	0.98	389
Nombre	0.99	0.78	0.88	1781
NumeroFactura	0.99	0.92	0.96	325
PeajeAcceso	0.99	0.81	0.89	298
PotenciaContratada	1	0.97	0.99	363
Poblacion	0.88	0.69	0.77	1127
CP	0.96	1	0.98	387
NumeroContrato	0.91	0.92	0.91	320
Provincia	0.97	0.47	0.63	458
Other	0.99	0.99	0.99	81718
accuracy			0.98	97294
macro avg	0.96	0.9	0.92	97294
weighted avg	0.98	0.98	0.97	97294
MAE	1.430365695726355			

Ilustración 6.6: Tabla de resultados SGD muestras tamaño medium

Naive Bayes Large Samples

	precision	recall	f-score	support
Comercializadora	0.74	0.99	0.85	7279
ConsumoEnergia	0.71	0.9	0.79	1042
CUPS	0.71	0.9	0.8	989
Direccion	0.88	0.99	0.93	10974
Distribuidora	0.83	1	0.91	1901
FechaEmision	0.98	1	0.99	865
FechaCargo	0.86	1	0.93	841
FechaFin	0.93	0.92	0.93	1206
FechaInicio	0.92	0.99	0.96	1174
FinContrato	0.88	1	0.94	911
ImporteAlquiler	0.84	1	0.91	1208
ImporteEnergia	0.49	0.96	0.65	802
ImporteImpuestos	0.63	0.99	0.77	1169
ImportePotencia	0.67	1	0.8	898
ImporteTotal	0.83	1	0.91	2413
NIF	0.99	1	0.99	1182
Nombre	0.68	0.99	0.81	5729
NumeroFactura	0.66	0.92	0.77	1100
PeajeAcceso	0.72	1	0.84	969
PotenciaContratada	0.87	1	0.93	1280
Poblacion	0.75	0.74	0.75	3659
CP	0.95	1	0.98	1166
NumeroContrato	0.9	1	0.95	1134
Provincia	0.53	0.79	0.63	1463
Other	1	0.95	0.98	272845
accuracy			0.96	324199
macro avg	0.8	0.96	0.87	324199
weighted avg	0.97	0.96	0.96	324199
MAE	2.6060012523172498			

Ilustración 6.7: Tabla de resultados NB muestras tamaño large

Support Vector Machine Large Samples

	precision	recall	f-score	support
Comercializadora	1	1	1	7279
ConsumoEnergia	1	1	1	1042
CUPS	1	1	1	989
Direccion	1	1	1	10974
Distribuidora	1	1	1	1901
FechaEmision	1	1	1	865
FechaCargo	1	1	1	841
FechaFin	1	0.92	0.96	1206
FechaInicio	1	1	1	1174
FinContrato	0.9	1	0.95	911
ImporteAlquiler	1	1	1	1208
ImporteEnergia	1	1	1	802
ImporteImpuestos	1	1	1	1169
ImportePotencia	1	1	1	898
ImporteTotal	1	1	1	2413
NIF	1	1	1	1182
Nombre	1	1	1	5729
NumeroFactura	1	1	1	1100
PeajeAcceso	1	1	1	969
PotenciaContratada	1	1	1	1280
Poblacion	0.99	0.98	0.99	3659
CP	1	1	1	1166
NumeroContrato	1	1	1	1134
Provincia	0.99	0.98	0.99	1463
Other	1	1	1	272845
accuracy			1	324199
macro avg	1	1	0.99	324199
weighted avg	1	1	1	324199
MAE	0.020000061690504906			

Ilustración 6.8: Tabla de resultados SVM muestras tamaño large

Stochastic Gradient Descent Large Samples

	precision	recall	f-score	support
Comercializadora	0.92	0.88	0.9	7279
ConsumoEnergia	0.96	1	0.98	1042
CUPS	1	0.89	0.94	989
Direccion	0.91	0.98	0.94	10974
Distribuidora	0.97	0.96	0.97	1901
FechaEmision	1	1	1	865
FechaCargo	0.93	1	0.96	841
FechaFin	0.99	0.92	0.95	1206
FechaInicio	1	1	1	1174
FinContrato	0.9	1	0.95	911
ImporteAlquiler	1	0.89	0.94	1208
ImporteEnergia	0.95	0.96	0.95	802
ImporteImpuestos	0.97	0.93	0.95	1169
ImportePotencia	1	0.76	0.86	898
ImporteTotal	0.99	0.91	0.95	2413
NIF	1	0.99	0.99	1182
Nombre	0.96	0.88	0.92	5729
NumeroFactura	0.97	0.92	0.94	1100
PeajeAcceso	0.97	0.84	0.9	969
PotenciaContratada	1	0.98	0.99	1280
Poblacion	0.86	0.73	0.79	3659
CP	0.98	1	0.99	1166
NumeroContrato	0.81	1	0.89	1134
Provincia	0.88	0.62	0.73	1463
Other	0.99	0.99	0.99	272845
accuracy			0.98	324199
macro avg	0.96	0.92	0.94	324199
weighted avg	0.98	0.98	0.98	324199
MAE	1.1704662876813314			

Ilustración 6.9: Tabla de resultados SGD muestras tamaño large

Gracias a la librería Seaborn podemos generar mapas de calor a usando la matriz de confusión obtenida de la clase metrics, esta matriz de confusión es generada durante el entrenamiento e indica en qué campos está el clasificador etiquetando los datos extraídos (eje x) con respecto a la clase a la que realmente pertenecen (eje y). De esta forma es mucho mas fácil ver qué campos son los que están ocasionando más fallos durante el entrenamiento de los clasificadores como explicamos anteriormente.

En el mapa de calor de NB entrenado con el mayor tamaño de muestras 6.10 el clasificador tiene problemas etiquetando muestras pertenecientes a los campos Poblacion y Provincia etiquetándolos como dirección, al igual con campos como distribuidora y el campo other que es destinado a toda la información inútil extraída del texto.

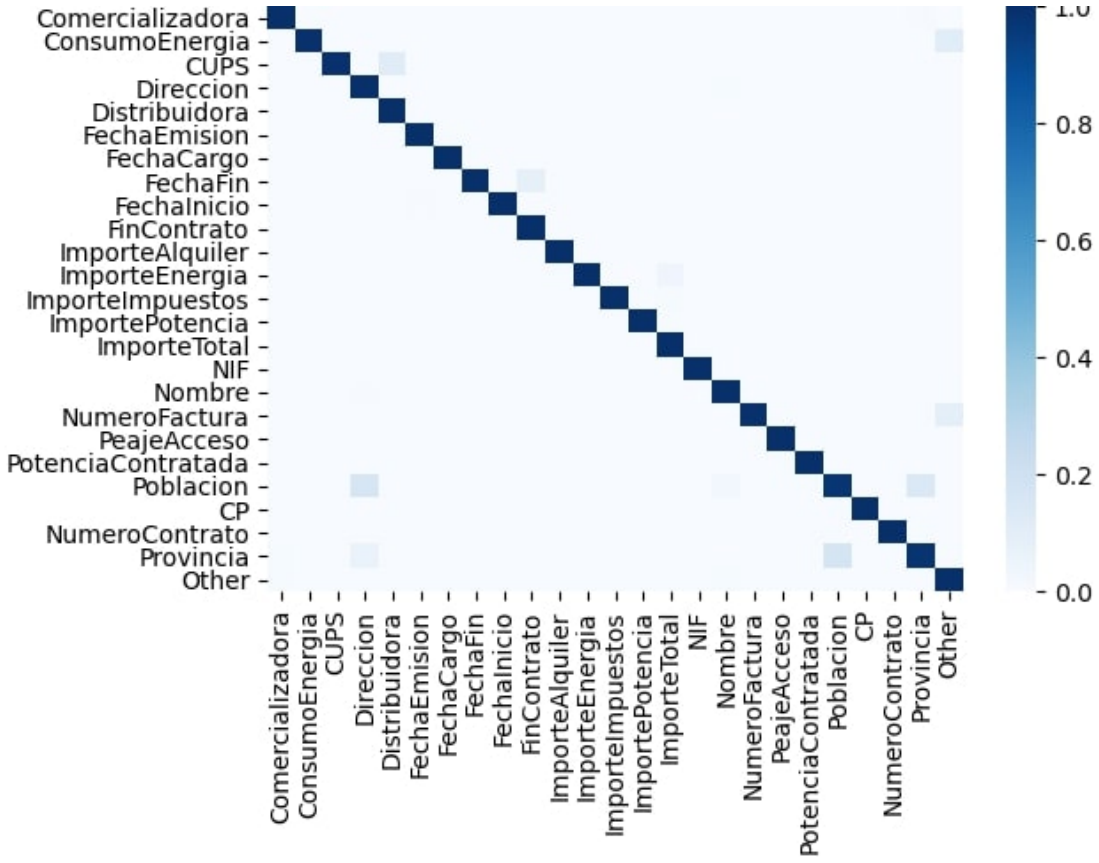

Ilustración 6.10: Mapa de calor NB muestras tamaño large

En caso del clasificador SVM 6.11, con el mismo número de muestras solo tiene problemas destacables con el campo FechaFin etiquetándolos erróneamente en el de FinContrato. Este clasificador a pesar de ser el que mejor resultados ofrece, tiene una gran coste en tiempo de entrenamiento(24h aproximadamente).

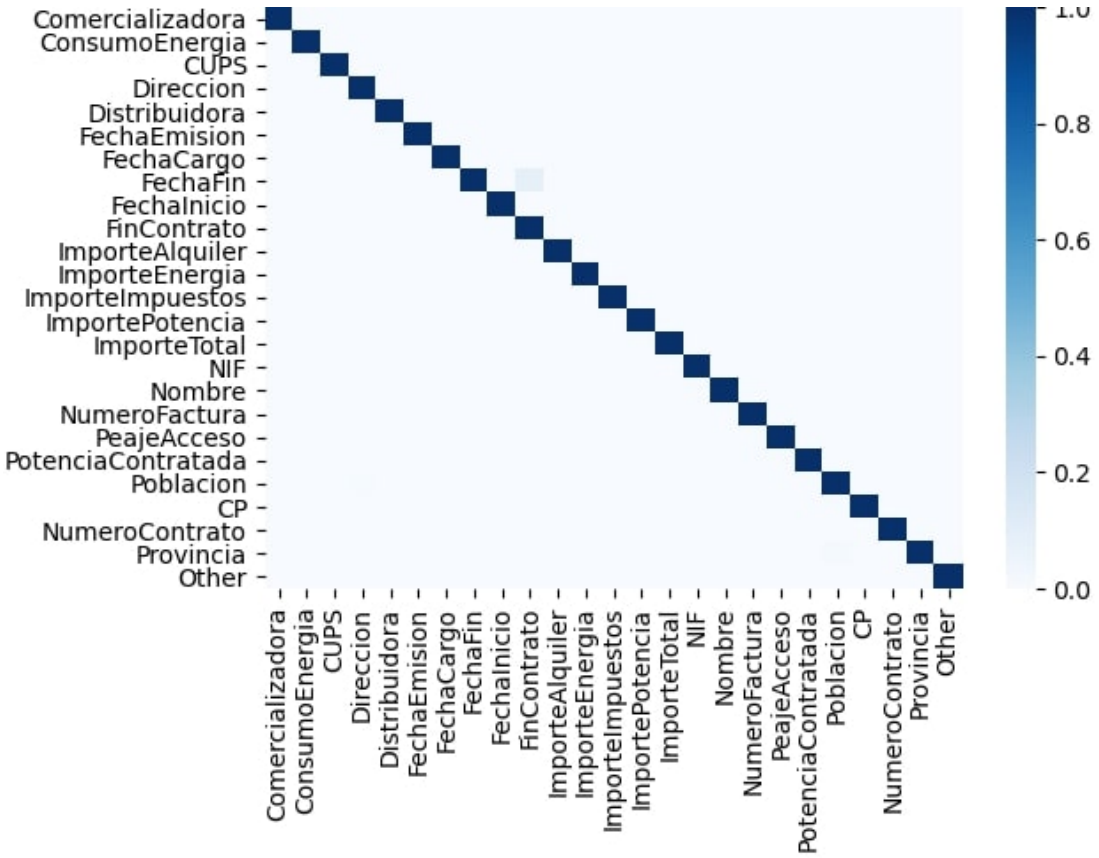

Ilustración 6.11: Mapa de calor SVM muestras tamaño large

El clasificador SGD 6.9 tiene una taza menor de error que el NB 6.7 sin embargo comete mas errores en el etiquetado del campo other 6.12 donde etiqueta muchas muestras que no corresponden con ese campo a diferencia del resto.

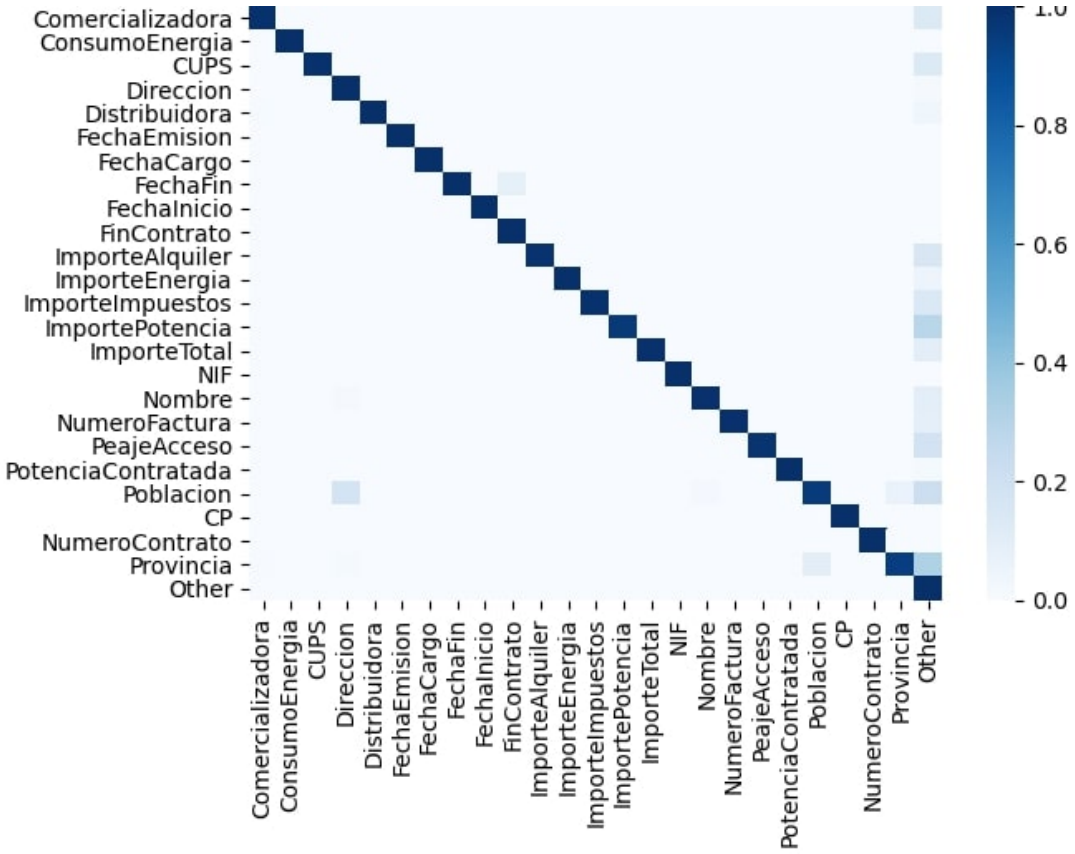

Ilustración 6.12: Mapa de calor SGD muestras tamaño large

Capítulo 7
Conclusiones y trabajo futuro

7.1. Conclusiones y problemas encontrados

Una vez se había concluido que el clasificador que mejor resultado ofrecía era el SVM se probó con una plantilla de factura realista4.84.9 para ver que tal se desenvolvía en un caso real, dando el siguiente resultado:

'Nombre': ['DE', 'LA'], 'Poblacion': ['Madrid'], 'Provincia': ['Madrid'], 'NumeroFactura': ['1NSN191000529711'], 'Comercializadora': ['EDP', 'COMERCIALIZADORA,', 'S.A.U.'], 'FechaInicio': ['26.08.2019-24.10.2019'], 'PotenciaContratada': ['4,40', '4,4'], 'PeajeAcceso' : ['2.0A'], 'ImporteAlquiler': ['1,57'], 'ImporteTotal': ['77,12', '23,57'], 'CUPS': ['ES0021000018WW'], 'Distribuidora': ['IBERDROLA', 'DISTRIBUCION', 'ELECTRICA,', 'S.A.'], 'ImporteEnergia': ['65789']

Se concluye que el clasificador tiene problemas a la hora de obtener algunos campos mientras que otros como PeajeAcceso y NumeroFactura los obtiene adecuadamente. Para solucionar estos problemas se plantea usar facturas realistas generadas aleatoriamente por un script sustituyendo los campos etiquetados en las facturas de plantilla como se explicó en la sección 4.2. Sin embargo debido al impedimento de escribir en los ya mencionados textbox que se generaban al convertir un pdf a documento docx este último paso para tratar de entrenar con las facturas realistas los clasificadores para asegurar la máxima eficacia no pudo llevarse acabo por las limitaciones de la librería Python-Docx.

Otro problema en la realización del TFT fue la memoria, ya que el autor no estaba preparado para manejar la semejante cantidad de datos ni contaba con experiencia previa en la redacción de documentos formales de gran cantidad de páginas y explicaciones detalladas, a diferencia de el desarrollo de software o código en lenguajes de programación. Este problema alargó el proceso de finalización del presente TFT. Además de esto, no se tenía conocimiento previo de los métodos de aprendizaje automático, aunque esto era igualmente un aliciente a entrar en el proyecto con fines de experimentación y aprendizaje de un área ampliamente desconocida. El aprendizaje de los métodos de aprendizaje automático si bien fue instructivo y alentador no dejaban de ser algoritmos matemáticos con un nivel de complejidad significativo, esto no pudo impedir la realización del presente TFT pero inevitablemente alargó los procesos de desarrollo y redacción, además de la revisión bibliográfica que constaba de artículos en inglés muy técnico sobre una materia complicada como ya se ha explicado, no obstante estas dificultades no dejan de ser un aprendizaje continuo sobre áreas y situaciones que no se han experimentado en la titulación y que aportan valor al desarrollo personal.

7.2. Consecución de Objetivos Alcanzados

En el estado final del proyecto no se ha conseguido proporcionar el objetivo final de proporcionar un objeto JSON que extraerá toda la información relevante de una factura eléctrica sino solo parcial, no se ha conseguido generar facturas realistas automáticamente en lugar de solo en formato SpaCy. Se descartó el emplear el algoritmo de estimación en un servidor restful como se llegó a plantear en un inicio siendo esto un añadido más costoso a un proyecto donde no se habían logrado los objetivos justo antes descritos y gastando e invirtiendo mucho tiempo en ellos.

Los objetivos que se han alcanzado son los de revisión bibliográfica, anotación de las facturas, la generación automática de facturas en formato SpaCy y la implementación de métodos de aprendizaje automático así como su evaluación y refinamiento este último usando GridSearchCV de la librería SKLearn pero no se ha conseguido desarrollar el método combinando debido que ya ofrecía el SVM un resultado preliminar muy alto y al no poder generar facturas realistas como muestras para entrenar a los clasificadores teniendo que limitarse a usar las que ya se había conseguido generar en formato spacy.

7.3. Trabajo Futuro

Para mejorar el proyecto es necesario un aumento de la detección del SVM para generar un objeto JSON con toda la información relevante y además de preparar un servidor retful para usar el proyecto como servicio web donde el usuario enviando una factura recibe el objeto JSON con toda la información de la factura. Además se puede explorar otros métodos de estimación como las IAs generativas o las ya conocidas redes neuronales que para este proyecto ya se quedaban bastante grandes y complejas.

En la propuesta inicial existía como objetivo crear un servidor restful para que el método diera servicio a usuarios en línea, este a su vez se dividió en dos apartados en la planificación que obviamente también fueron eliminados de la planificación, uno crear un servicio web API Rest y el otro era configurar el servidor para que los usuarios pudieran enviar sus facturas y que se extrajera la información devolviendo un objeto JSON. Finalmente se decidió, junto con el tutor, eliminarlos aunque como hemos explicado se proponen como posibles ampliaciones en un trabajo futuro.

Apéndice A
Competencias específicas y aportaciones del trabajo

A.1. Competencias

"CII02: Capacidad para planificar, concebir, desplegar y dirigir proyectos, servicios y sistemas informáticos en todos los ámbitos, liderando su puesta en marcha y su mejora continua y valorando su impacto económico y social."(Competencias de GII de la EII)

Este proyecto me interesó debido a la ayuda que puede prestar a la hora de automatizar facturas eléctricas de esta manera ayudando a empresas o particulares en la gestión de estos datos.

"CII03: Capacidad para comprender la importancia de la negociación, los hábitos de trabajo efectivos, el liderazgo y las habilidades de comunicación en todos los entornos de desarrollo de software."(Competencias de GII de la EII)

Se realizaron reuniones con el tutor explicando los problemas que ocurrían y las modificaciones que se debían realizar en la planificación inicial del proyecto.

"CII06: Conocimiento y aplicación de los procedimientos algorítmicos básicos de las tecnologías informáticas para diseñar soluciones a problemas, analizando la idoneidad y complejidad de los algoritmos propuestos."(Competencias de GII de la EII)

Se probaron varios algoritmos verificando su utilidad en base al problema que se quería solucionar teniendo en cuenta factores como el rendimiento de la solución y su calidad.

"CII010: Conocimiento de las características, funcionalidades y estructura de los Sistemas Operativos y diseñar e Implementar aplicaciones basadas en sus servicios." (Competencias de GII de la EII)

Se utiliza la consola del sistema operativo windows 10 para ejecutar la aplicación y sus rutas de archivos y ficheros.

´´CII015: Conocimiento y aplicación de los principios fundamentales y técnicas básicas de los sistemas inteligentes y su aplicación práctica." (Competencias de GII de la EII)

Se utilizan en el proyecto técnicas de aprendizaje automático para solucionar el problema de las extracción de información de textos de facturas eléctricas.

A.2. Aportaciones del proyecto

Este proyecto busca aportar la automatización de la digitalización de la información relevante de una factura eléctrica a empresas o particulares, en la manera en que con solo una factura se creará un objeto JSON de la información relevante de la factura como la potencia contratada, la fechas de contrato y su finalización y sus costes, además será más fácil de almacenar y manipular digitalmente que una imagen o pdf de la factura.

Bibliografía

[1] Sistemas basados en reglas. http://eia.udg.es/~iitap/monografia/esp/arl4_8.html.

[2] Expresiones regulares. https://www2.iib.uam.es/bioinfo/curso/perl/tutoriales/cicei/cap6.htm.

[3] 1.1. linear models — scikit-learn 1.5.1 documentation. https://scikit-learn.org/stable/modules/linear_model.html.

[4] 1.10. decision trees — scikit-learn 1.5.1 documentation. https://scikit-learn.org/stable/modules/tree.html.

[5] Pandas : La biblioteca de python dedicada a la data science. https://datascientest.com/es/pandas-python.

[6] scikit-learn: machine learning in python — scikit-learn 1.5.1 documentation. https://scikit-learn.org.

[7] 1.9. naive bayes — scikit-learn 1.5.1 documentation. https://scikit-learn.org/stable/modules/naive_bayes.html.

[8] 1.4. support vector machines — scikit-learn 1.4.2 documentation. https://scikit-learn.org/stable/modules/svm.html.

[9] 1.5. stochastic gradient descent — scikit-learn 1.4.2 documentation. https://scikit-learn.org/stable/modules/sgd.html.

[10] All you need to know about support vector machines. https://www.spiceworks.com/tech/big-data/articles/what-is-support-vector-machine/.

[11] When do support vector machines fail? — by wei hao khoong — towards data science. https://towardsdatascience.com/when-do-support-vector-machines-fail-3f23295ebef2.

[12] Alejandro Alija. Spacy: mucho más que una librería para crear proyectos reales de procesamiento del lenguaje natural, 2022.

[13] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[14] Hui Han, C Lee Giles, Eren Manavoglu, Hongyuan Zha, Zhenyue Zhang, and Edward A Fox. Automatic document metadata extraction using support vector machines. In *2003 Joint Conference on Digital Libraries, 2003. Proceedings.*, pages 37–48. IEEE, 2003.

[15] Juan Carlos Lanas Ocampo. Conoce un poco spacy con python. https://es.linkedin.com/pulse/conoce-un-poco-spacy-con-python-juan-carlos-lanas-ocampo.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[17] John Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical Report MSR-TR-98-14, Microsoft, April 1998.

[18] Ranjini Srinivas. Managing large data sets using support vector machines. 2010.

[19] J. Sánchez, A. Salgado, A. García, and otros. IDSEM, an invoices database of the Spanish electricity market. *Sci Data*, 9(786), 2022.

[20] Harry Zhang. The optimality of naive bayes. *Aa*, 1(2):3, 2004.