Facultad de Matemática, Física, Astronomía y Computación

Universidad Nacional de Córdoba

Aprendizaje Activo para la Extracción de Relaciones en Textos

Rodrigo Jaime

Director: Dr. Franco M. Luque

Resumen

A la hora de realizar un trabajo de aprendizaje automático, podemos encontrarnos con una fuente abundante de datos sin etiquetar y que su etiquetado manual sea un proceso costoso. Una técnica útil en estos casos es permitir que el algoritmo de aprendizaje consulte a un oráculo sobre las etiquetas de ciertos datos seleccionados que el algoritmo considere importantes. Este tipo de aprendizaje semi-supervisado iterativo se llama aprendizaje activo. Utilizando IEPY, un framework de extracción de información orientado a extracción de relaciones, construiremos un sistema de extracción de relaciones con aprendizaje activo. Realizaremos una serie de experimentos sobre selección de instancias y etiquetado de features sobre un corpus basándonos en el trabajo de Settles 2011 con el objetivo de acelerar inicialmente el desempeño del modelo.

Palabras claves: Aprendizaje Activo – Extracción de Información – Extracción de Relaciones – Procesamiento de Lenguaje Natural – Multinomial Naive Bayes - IEPY

Abstract

When working on some machine learning tasks, we may have to deal with large pool of unlabeled data and the labeling process to be manual and resource-intensive. A useful technique in these cases is to allow the learning algorithm to query an oracle about the labels of certain data selected as important by the algorithm. This kind of iterative semi-supervised learning is known as active learning. Using IEPY, an information extraction framework oriented towards relationship extraction, we will build a relationship extractor system with active learning. We will also perform a series of experiments about instance selection and feature labeling on a corpus, based on the work from Settles 2011 with the mission of accelerating the model's initial performance.

Keywords: Active Learning – Information Extraction – Relationship Extraction – Natural Language Processing – Multinomial Naive Bayes – IEPY

Agradecimientos

Se cierra un capítulo que considero que ha sido posible, más allá del esfuerzo personal, gracias a la gente que me ha acompañado en el camino. Les dedico un agradecimiento a todos ellos. Desde el comienzo hasta el final, mis padres y mi hermana dándome su apoyo y consejos; familia y amigos también presentes alentándome. A mis compañeros con los que he recorrido este camino. A mi novia que me dió aliento y acompañó en el final.

Quiero agradecer a mi director, Franco Luque, de quien he aprendido muchas cosas y quien con paciencia y siempre buena disposición me ha asistido para llevar a cabo este trabajo. Finalmente, a FaMAF por contar con excelentes profesores y personal no docente, brindando una formación académica de nivel destacable a sus alumnos.

Índice general

1. Introducción

El campo de la extracción de información contiene mucho potencial. Desde sus orígenes a finales de la década de 1970 ha ido ganando interés a medida que la disponibilidad de datos en formato digital crecía, llegando a volúmenes que nunca podrían ser procesados bajo métodos de revisión manual por humanos. Incluso han surgido conferencias basadas en competencias como MUC (Message Understanding Conferences) o ACE (Automatic Content Extraction) para las cuales se desarrollaron novedosas técnicas e ideas. Estas competencias se han repetido en múltiples ocasiones con distintos temas como: la extracción de entidades como personas y compañías y relaciones entre estas entidades como *"es-CEO-de"*, información sobre actividades terroristas en latinoamérica a partir de reportes en diarios, entre otros.

Una de las actividades dentro de la extracción de información es la extracción de relaciones. Ésta consiste en identificar menciones de ciertas relaciones en oraciones dadas. La extracción de relaciones tiene utilidad en problemas como:

Question Answering: tomemos de ejemplo *"¿Dónde nació Roger Federer?"*. Para poder contestar la pregunta, sin entrar en detalles de Question Answering, necesitaríamos haber capturado la relación *"nació-en"* entre PERSONA y LOCACIÓN a partir de un texto como el siguiente: *"El famoso tenista Roger Federer nació en Suiza, el 8 de agosto de 1981."*.

Construcción de bases de conocimiento: a partir de textos de biomedicina sobre proteínas se pueden descubrir interacciones entre proteínas, asociada a la relación *"se-une"* entre PROTEÍNA y PROTEÍNA, que permite la posibilidad de encontrar drogas nuevas.

Encontramos en Settles 2011 una serie de ideas y técnicas que permiten entrenar un modelo para extracción de información de manera más eficiente al requerir menos instancias de entrenamiento. Esto lo logra explotando la información que contienen tanto las instancias ya usadas de entrenamiento como las que aún no se utilizaron, junto con la interacción con un oráculo que asiste al algoritmo proveyéndole información. Este conjunto de técnicas es aplicable a nuestro tema de interés: la extracción de relaciones.

En este trabajo contaremos con un corpus que contiene candidatos de relación *"located-in"* entre entidades de tipo ORGANIZATION y LOCATION. Estos candidatos han sido etiquetados respecto a la ocurrencia de la relación (SÍ o NO) con un alto grado de

certeza. Por lo tanto aplicaremos las técnicas de Settles 2011 buscando aumentar el rendimiento obtenido del modelo respecto a la cantidad de instancias usadas en el entrenamiento en la etapa inicial. Realizaremos una serie de experimentos para ver los resultados de la aplicación de estas técnicas.

Respecto a la estructura de este trabajo, será desarrollado de la siguiente manera:

- el capítulo 2 describe preliminares y marco teórico sobre temas como extracción de información, extracción de relaciones, aprendizaje automático, métricas de rendimiento y una descripción sobre IEPY.
- en el capítulo 3 desarrollamos la adaptación de las ideas del trabajo de Settles 2011 sobre el algoritmo con aprendizaje activo y el uso de consultas respecto a instancias y features.
- en el capítulo 4 se plantean los experimentos principales a realizar junto con los resultados. También describimos su contexto, como el dataset a utilizar y la extracción de features.
- el capítulo 5 contiene una serie de experimentos complementarios y sus resultados, realizados con el fin de mejorar el rendimiento de los modelos y resolver obstáculos encontrados durante el desarrollo de los experimentos.
- en el sexto y último capítulo, presentaremos las conclusiones obtenidas a partir del desarrollo del trabajo y los experimentos. Finalmente plantearemos una serie de opciones para continuar el desarrollo del trabajo, como experimentos, modificaciones y el uso de otros corpus.

2. Preliminares

Comenzaremos introduciendo una serie de temas para sentar la base de conceptos relacionados con el trabajo que hemos llevado a cabo.

2.1. Extracción de Información

La Extracción de Información (EI o generalmente referida como IE, por Information Extraction) es una tarea del Procesamiento de Lenguaje Natural (PLN) que consiste en la extracción automática de información de un medio no estructurado obteniendo información estructurada como entidades, relaciones entre entidades y atributos que describen entidades. Tener la información estructurada nos permite operar con ella para realizar consultas, organizarla y analizarla. Es por esto que junto con la abundancia de la información no estructurada y su continua generación, cobra valor el desarrollo de la EI. Algunas de las tareas en las que es de utilidad son la Obtención de Información (Information Retrieval) y Pregunta-Respuesta (Question-Answering).

Los tipos de información para extraer generalmente de mayor interés son: entidades nombradas, relaciones entre entidades y eventos. Nos enfocaremos en los dos primeros.

2.1.1. Entidad nombrada

Una entidad nombrada (EN o NE por named entity) es generalmente una palabra o frase sustantiva compuesta por uno o pocos tokens que representa un objeto específico del mundo real. Tomemos como ejemplo la mención de una entidad nombrada, que en este caso será *Steven Spielberg* en la siguiente oración: *"Steven Spielberg dirigió Jurassic Park"*. Una mención de una entidad nombrada puede usar el nombre propio (*Steven Spielberg*), una forma nominal (*director*) o una forma pronominal (*él*).

Las entidades nombradas pueden categorizarse en algunos tipos genéricos como: PERSONA, ORGANIZACIÓN, LUGAR, FECHA, HORA, TELÉFONO, EMAIL, URL, CANTIDAD, etc. Existen otros tipos de dominio específico como: PROTEÍNAS, ENZIMAS, ambas del dominio de la biología. El Reconocimiento de Entidades Nombradas (REN) es una subtarea de EI encargada de identificar las ocurrencias de un tipo particular de EN en los documentos dados.

2.1.2. Relaciones entre entidades

Las relaciones se definen entre dos o más entidades relacionadas de una manera predefinida. Expresan la asociación entre dos o más fragmentos de texto que representan a las entidades. Formalmente podemos definirlas como una tupla $t = (e_1, e_2, \ldots, e_n)$ donde e_i son entidades en una relación predefinida r dentro de un documento D. Particularmente nos concentraremos en las relaciones binarias, es decir las relaciones entre sólo 2 entidades. Ejemplos de relaciones son:

- *"nació en"* como relación entre entidades de tipo PERSONA y LUGAR: *"nació en"*(*Tom Hanks*, *Estados Unidos*).
- *"recomienda libro a"* entre PERSONA, LIBRO, PERSONA. *"recomienda libro a"*(*Yo, Juego de Tronos, Keanu Reaves*).

2.2 Métodos de Extracción de Información

Podemos categorizar los métodos usados para la extracción de información bajo dos dimensiones:

- *codificados a mano* o *basados en aprendizaje*
- *basados en reglas* o *estadísticos*.

Nuestro interés se ubica dentro de la primera dimensión, en particular los *basados en aprendizaje*.

2.2.1 *Basados en aprendizaje*

Estos sistemas requieren ejemplos no estructurados que han sido etiquetados manualmente para entrenar los modelos de aprendizaje automático para extracción. Se requiere además experiencia en el dominio para la identificación y etiquetado de ejemplos que serán representativos del entorno real en el que se utilizará, así como también un buen entendimiento del aprendizaje automático para poder elegir entre los distintos modelos y características a extraer para la robustez del sistema ante datos no vistos previamente.

2.3. Métodos supervisados

Estos métodos requieren un conjunto de datos etiquetados donde cada par de menciones de entidad esté etiquetado con alguno de los tipos de relaciones predefinidos. Para los pares donde no corresponda ninguno de estos tipos de relación,

existe un tipo especial de relación NONE (NINGUNO). En general la extracción de relaciones se formula como un problema de clasificación multi clase donde cada clase corresponde a un tipo diferente de relación, incluyendo NONE también. Estos métodos suelen clasificarse en dos tipos: métodos basados en features y métodos basados en kernels.

Uno de los problemas de los métodos supervisados es la dificultad de contar en algunas ocasiones con un conjunto de datos de tamaño suficiente completamente etiquetado. Como hemos planteado anteriormente a veces es un proceso costoso, por ejemplo por tiempo o dificultad.

2.3.1. Métodos basados en features

En estos métodos tenemos que para cada instancia de relación (i.e. menciones de pares de entidades) en los datos etiquetados se genera un conjunto de features. Luego son presentados en forma de vector a un clasificador para su entrenamiento para así poder clasificar cualquier nueva instancia de relación. Una vez que los features son diseñados podemos elegir otros clasificadores para ser usados por estos métodos. La mayor parte de los esfuerzos en estos métodos son enfocados en el diseño del conjunto "correcto" de features. Puede ocurrir que algunos features sean buenos indicadores de la relación mientras que otros no tanto, por lo tanto para su selección se requiere un meticuloso análisis de la contribución de cada feature y el conocimiento de los fenómenos lingüísticos subyacentes. A continuación veremos algunos ejemplos de features de varios tipos:

<u>Palabras</u>

- Cada una de las palabras de la oración.
- Palabras de ambas menciones.
- Todas las palabras entre las menciones.

<u>Tipos de entidad</u>

- Tipos de entidad de ambas menciones.

<u>Nivel de mención</u>

- Tipos de mención (NOMBRE, NOMINAL o PRONOMBRE) de ambas menciones.

<u>Superposición</u>

- Cantidad de palabras separando las dos menciones.
- Cantidad de otras menciones entre las menciones.
- Indicadores de si las dos menciones pertenecen a la misma categoría sintáctica (NP o frase sustantiva, VP o frase verbal, o PP frase preposicional).

Dependencia

- Palabras, part of speech (POS) y chunk labels de palabras sobre las cuales las menciones son dependientes en el árbol de dependencias.
- Cantidad de links atravesados en el árbol de dependencias para ir de una de las menciones a otra.

Árbol de análisis sintáctico

- Camino de no-terminales conectando las dos menciones en el árbol de análisis sintáctico, y el camino anotado con las palabras principales.

2.3.2. Métodos basados en kernels

El rendimiento general de los métodos basados en features dependen en gran parte de la efectividad de los features diseñados. La principal ventaja de los métodos basados en kernel es que se evita esta ingeniería de features explícita. Las funciones de kernels son diseñadas para computar similaridades entre representaciones de alta dimensión de dos instancias de relación. Para la clasificación se utilizan SVM (Support Vector Machines). Distintos sistemas de extracción de relaciones basados en estos métodos utilizan diferentes representaciones para las instancias de relaciones tales como secuencias, árboles de análisis sintáctico, etc. La mayoría de las técnicas miden la similaridad entre dos representaciones cualesquiera (por ejemplo árboles) en relación con la cantidad de subrepresentaciones (subárboles) entre ellas.

2.4. Métodos semi-supervisados

Generar datos etiquetados para extracción de relaciones es una tarea intensiva en relación al costo, esfuerzo y tiempo que conlleva. Principalmente la motivación detrás de diseñar técnicas semi-supervisadas se desdobla en:

- reducir los esfuerzos manuales requeridos para generar datos etiquetados.
- darle mayor uso a los datos no etiquetados que se consiguen con mayor facilidad y menor esfuerzo.

Algunas de estas técnicas son:

Bootstrapping: estos algoritmos requieren un gran corpus de datos etiquetados y algunas instancias *semillas* del tipo de relación de interés.

Aprendizaje activo: el algoritmo aprendiz puede preguntar sobre las etiquetas verdaderas de algunas instancias no etiquetadas seleccionadas de manera inteligente. Nos concentraremos en esta técnica, la cual describiremos con mayor detalle.

Método de propagación de etiqueta: se basa en un grafo donde las instancias etiquetadas y no etiquetadas en los datos son representadas como nodos en el grafo donde sus lados reflejan la similaridad entre los nodos. La información de etiquetado para los nodos se propaga a los nodos cercanos a través de lados con peso de manera iterativa. Finalmente las etiquetas de los ejemplos no etiquetados se infieren cuando el proceso de propagación converge.

2.4.1. Aprendizaje Activo

El desempeño del modelo depende de encontrar una cantidad adecuada de datos etiquetados para el entrenamiento de los parámetros del modelo. Dado que la recolección de datos etiquetados para el entrenamiento suele ser un proceso costoso a nivel de esfuerzo, técnicas como el aprendizaje activo contribuyen a reducir este esfuerzo, aprovechando la cantidad limitada de instancias etiquetados y el gran conjunto de instancias no etiquetadas. El conjunto etiquetado nos provee los datos de entrenamiento para un obtener un clasificador inicial preliminar. El aprendedor activo analiza el conjunto no etiquetado en búsqueda de aquellas instancias que una vez etiquetadas mejoren el clasificador con la tasa más rápida posible. Éste es un problema conceptual importante en el área del aprendizaje automático con donde se trabaja continuamente en el desarrollo de métodos teóricamente robustos para el aprendizaje activo. Hay estudios como Probst et al. 2007 y Thompson et al. 1999 que han mostrado que ésta técnica puede reducir significativamente la cantidad de ejemplos que necesitan ser manualmente etiquetados.

2.5. Extracción de relaciones

La tarea de extracción de relaciones consiste en identificar menciones de las relaciones de interés dentro de las oraciones que pertenecen a ciertos documentos dados. Esta tarea será presentada como una tarea de clasificación y le daremos una breve descripción formal.

Dada una oración $S = w_1, w_2, \ldots, e_1, \ldots, w_j, \ldots, e_2, \ldots, w_n$, donde e_1 y e_2 son entidades y r la relación, definamos una función T_c que toma una oración y devuelve los features c extraídos a partir de S, y f_r como función que a partir de los features de la oración decide si la relación ocurre entre las entidades. En caso positivo tendremos que $f_r(T_c(S)) = 1$; caso contrario tendremos $f_r(T_c(S)) = -1$.

El problema de extracción de relaciones binarias se puede plantear en tres casos:

- las entidades se encuentran identificadas previamente en el texto no estructurado, y dados pares fijos de entidades necesitamos encontrar el tipo de relación que existe entre el par.
- tenemos una relación de tipo r y un nombre de entidad e, y nuestro objetivo es extraer las entidades con las cuales e tiene la relación r.
- tenemos un corpus no estructurado de gran tamaño y abierto como la web donde no podemos asumir que los pares de entidades están marcados. Dada una relación fija de tipo r, nuestro objetivo es extraer todas las instancias de los pares de entidades que tienen la relación r entre ellas, a través de técnicas de reconocimiento y un filtrado apropiados.

En este trabajo nos concentraremos en el primer caso, con el objetivo de determinar si ocurre o no una relación r fija dada.

2.5.1. Prediciendo la relación entre un par de entidades

Dado un conjunto fijo R de tipos de relaciones, con cada uno involucrando un par de tipos de entidades, nuestro objetivo es identificar todas las ocurrencias de las relaciones en R en un documento donde todas las entidades han sido marcadas. En nuestro caso R sólo contiene a la relación *"located-in"*. Típicamente, en la extracción de relaciones en textos en lenguaje natural, se asume que las dos entidades de argumento son parte de la misma oración o se encuentran en una acotada proximidad entre ellas. Por lo tanto el problema básico de reconocimiento se describe como:

dada una secuencia de texto x y dos entidades marcadas E_1 y E_2 en x, identificar si alguna de las relaciones de $\mathcal{Y}$ se da entre *E1* y *E2*. El conjunto $\mathcal{Y}$ incluye todos los tipos de relaciones de R y contiene además un elemento especial *"otra"* para el caso donde ninguna de las relaciones se aplican al par de entidades dado.

El problema de predicción en la extracción de relaciones se trata de relacionar dos palabras de entidades en una oración, lo cual requiere una hábil combinación de pistas con ruido, locales y no locales, de diversas estructuras sintácticas y semánticas en una oración. A continuación veremos algunos de los tipos de recursos más comunes que son útiles para la extracción de relaciones:

Tokens de superficie: los tokens alrededor y entre las dos entidades suele contener pistas fuertes para la extracción de relaciones. Por ejemplo, la relación *"creado-por"* entre una entidades de tipo PRODUCTO y COMPAÑÍA está fuertemente

vinculada a la presencia del token unigrama *desarrollado* o el token bigrama *desarrollado por* entre las dos entidades, en la oración:

El navegador ***<PRODUCTO> Firefox </PRODUCTO>*** *fue* ***desarrollado por*** *la* ***<COMPAÑIA> Fundación Mozilla </COMPAÑIA>*** *y publicado por primera vez el 9 de noviembre de 2004.*

Generalmente un token es generalizado a su raíz morfológica. Por ejemplo, *"desarrollado"* se lleva a *"desarrollar"*.

Etiquetas Part Of Speech (POS): las etiquetas POS juegan un rol importante en la extracción de relaciones. Los verbos en una oración son clave a definir la relación entre entidades, que son generalmente sustantivos o frases sustantivas.

Estructura de árbol de análisis sintáctico: un árbol de análisis agrupa palabras en una oración en tipos de frase prominentes como frases sustantivas, frases preposicionales y frases verbales, lo cual aporta mayor entendimiento de la relación entre entidades en una oración que las etiquetas POS.

Grafo de dependencia: los árboles de análisis completos son costosos de crear. Un grafo de dependencia que relaciona cada palabra con las palabras que dependen de ella suele ser tan adecuado como un árbol de análisis.

2.5.2. Métodos de extracción de relaciones

Los métodos presentados a continuación utilizan de diferentes maneras los recursos recientemente presentados para clasificar una entrada (x, E_1, E_2) en una de las clases de $\mathcal{Y}$. Asumamos que tenemos N ejemplos de entrenamiento de la forma (x^i, E_1^i, E_2^i, r^i) con $i \in \{1, \ldots, N\}$, donde $r^i \in \mathcal{Y}$ denota la relación que existe entre las entidades E_1^i y E_2^i en la oración x^i.

Un desafío importante es manejar la diversidad de las formas estructurales que las diferentes entradas representan. Por ejemplo, los tokens y las etiquetas POS forman una secuencia, el análisis de información es un árbol, y la estructura de dependencia es un grafo. Además puede haber errores en cualquiera de los recursos utilizados como entradas, dado que las librerías lingüísticas utilizadas para estas tareas no son perfectas. A pesar que la redundancia provee robustez ante errores, demasiada redundancia conlleva ruido y mayor tiempo de procesamiento.

Los métodos utilizados para extracción de relaciones pueden ser categorizados en uno de los siguientes 3 tipos principales:

Métodos basados en features, que extraen un conjunto plano de features a partir de la entrada y luego invocan un clasificador de los ya existentes como un SVM,

un árbol de decisión o en nuestro caso un Multinomial Naive Bayes. Utilizaremos este tipo de métodos.

Métodos basados en kernel que diseñan kernels especiales para capturar la similaridad entre estructuras como árboles y grafos.

Métodos basados en reglas que crean reglas proposicionales y de primer orden sobre estructuras alrededor de las dos entidades.

2.6. Métricas de rendimiento

Decimos que un problema de clasificación es binario cuando cada entrada es clasificada en una y sólo una clase entre 2 clases sin superposición. Generalmente se le asignan de nombre a estas clases: clase positiva y clase negativa. La clase positiva representará los casos cuando la relación efectivamente ocurra; la clase negativa cuando no suceda.

Cuando trabajamos con un conjunto de evaluación, conocemos las clases a las que pertenece cada elemento del conjunto. Por lo tanto luego de realizar la clasificación de los elementos de este conjunto al realizar la evaluación del modelo podemos categorizar cada resultado como:

- ***tp***: true positive o verdadero positivo, cuando el elemento pertenece a la clase positiva y éste fue clasificado como positivo.
- ***tn***: true negative o verdadero negativo, cuando el elemento pertenece a la clase negativa y éste fue clasificado como negativo.
- ***fp***: false positive o falso positivo, cuando el elemento pertenece a la clase negativa y éste fue clasificado como positivo.
- ***fn***: false negative o falso negativo, cuando el elemento pertenece a la clase positiva y éste fue clasificado como negativo.

A partir de estos valores se han definido métricas como las siguientes:

- **accuracy**: en general, mide el porcentaje de predicciones correctas sobre el total de predicciones.

$$\frac{tp + tn}{tp + fp + fn + tn}$$

- **precision**: mide el porcentaje de predicciones correctas de la clase positiva sobre el total de predicciones asignadas a la clase positiva.

$$\frac{tp}{tp + fp}$$

- **recall**: mide el porcentaje de predicciones correctas de la clase positiva sobre el total de los elementos que pertenecen a la clase positiva.

$$\frac{tp}{tp + fn}$$

- **F-score**: representa la media armónica entre los valores de precision y recall. La métrica general lleva β como parámetro, llamándose Fβ-Score. Generalmente se utiliza con $\beta = 1$.

$$F\beta\text{-}score : \frac{(1 + \beta^2) * precision * recall}{\beta^2 * precision + recall} \qquad F1\text{-}score : \frac{2 * precision * recall}{precision + recall}$$

2.7. IEPY: un framework para la Extracción de Información

IEPY[1] es un framework de código abierto para desarrollar proyectos de Extracción de Información, en particular orientado a Extracción de Relaciones. Está basado principalmente en python como lenguaje de programación y *django*, un famoso framework web de python, como uno de sus componentes principales. IEPY permite realizar tanto experimentos de EI desde un punto de vista más científico, como proyectos que requieran extraer información a partir de grandes conjuntos de datos.

Provee:

- Herramientas para la anotación de corpus mediante una interfaz web
- Extracción de Relaciones utilizando Aprendizaje Activo
- Extracción de Relaciones basada en reglas

El flujo de trabajo general para un proyecto es el siguiente:

- Preprocesamiento de los datos originales y su carga en la base de datos
- Otros preprocesamientos, por ejemplo lingüístico.
- Creación de Relaciones
- Anotación de datos
- Ejecución de Extracción de Relaciones

Un proyecto de IEPY consta de los siguientes componentes:

[1]http://iepy.machinalis.com/

Base de datos
A través de *django*, IEPY ofrece una interfaz de base de datos configurable y compatible con varias implementaciones populares de sistemas tanto SQL como NoSQL.

Administrador web
Permite manejar los objetos de la base de datos y etiquetar la información necesaria para el módulo de Extracción de Relaciones.

Módulos de proyecto
Estos se encargan de implementar el flujo de trabajo del proyecto. Inicialmente se incluye un módulo de cargado de datos genérico y uno de preprocesamiento de datos con rutinas de análisis de lenguaje natural de Stanford.

2.7.1. Modificaciones de IEPY en este trabajo

El framework ofrece un módulo de preprocesamiento de corpus con una funcionalidad base para importar desde el corpus las instancias que se convertirán en documentos de la base de datos. Antes de insertar el documento se corre una serie de pasos de preprocesamiento. Estos incluyen la ejecución de tareas de PLN como tokenización, separación de oraciones, etiquetado part of speech (POS tagging), reconocimiento de entidades nombradas (NER), junto con otras tareas particulares que dependan del corpus elegido.

Con las siguientes modificaciones pudimos reducir el tiempo de ejecución del preprocesamiento significativamente y simplificar las dependencias del proyecto:

- Dentro del módulo que ejecuta el pipeline de estas tareas de preprocesamiento, se agregó de manera opcional la posibilidad de correr el pipeline dentro de una transacción atómica con la base de datos, funcionalidad provista por *django*.

- Se agregó un módulo alternativo para la ejecución de las principales tareas de PLN ya nombradas. IEPY viene con el módulo StanfordPreprocessor que utiliza CoreNLP, la popular librería desarrollada por la universidad de Stanford, implementada en Java. Se lo utiliza mediante una interfaz en python ya implementada por IEPY. Este módulo introduce una dependencia del proyecto con el JRE (Java Runtime Environment). Se decidió implementar las mismas funcionalidades requeridas utilizando otro framework de PLN llamado SpaCy, que tiene un buen desempeño y está implementado en Cython. De esta manera podemos simplificar las dependencias del proyecto introducidas por IEPY, eliminando el requerimiento del JRE, manteniendo un desempeño similar.

3. Algoritmo con aprendizaje activo para extracción de relaciones

Este trabajo propone implementar técnicas presentadas en Settles 2011 basadas en el uso de métodos semi-supervisados con aprendizaje activo utilizando consultas para la selección de instancias y etiquetado de features.

Originalmente estas técnicas se utilizaron sobre documentos con una etiqueta, como por ejemplo:

- artículos periodísticos donde cada uno tiene un tema etiquetado (adquisiciones, maíz, ganancias, etc)
- mensajes de grupos de discusión online con un tema etiquetado (hockey, baseball, etc)
- reseñas de películas categorizadas como positivas o negativas en cuanto al sentimiento como etiqueta.

En este trabajo aplicaremos estas técnicas para la extracción de relaciones. Los documentos serán oraciones que contienen un candidato de una determinada relación con la etiqueta que indica si ocurre o no efectivamente la relación.

3.1. Modelo

El clasificador utilizado en estas técnicas es Multinomial Naive Bayes (MNB). Algunas de sus características que lo hacen atractivo como elección es su simpleza, velocidad (nos interesa tener bajos tiempos de ejecución), y su desempeño en relación a aplicaciones de lenguaje natural y clasificación de texto.

MNB modela la distribución de features como una distribución multinomial, que es la generalización de la distribución binomial, lo que permite tener una cantidad finita k de posibles categorías de resultados para cada evento. En los experimentos de extracción de relaciones realizados en este trabajo sólo tendremos 2 posibles categorías: la relación analizada efectivamente ocurre o no ocurre. Por lo tanto k=2, i.e. $j \in \{0, 1\}$.

Los documentos son vistos como secuencias de palabras donde se asume de manera ingenua que las palabras en cada posición son generadas independientemente. Bajo estas características, podemos calcular la probabilidad de que un documento x tenga clase y_j como:

$$P_\theta(x|y_j) = P(|x|) \prod_k (\theta_{jk})^{f_k(x)}$$

Donde:

θ : vector que representa la parametrización del modelo.

θ_j : $P_\theta(y_j)$, la probabilidad de la clase y_j .

θ_{jk} : $P_\theta(f_k|y_j)$, la probabilidad de generar la palabra f_k

$f_k(x)$: frecuencia de conteo de la palabra f_k en el documento x.

A partir de esto, podemos utilizar la regla de Bayes para obtener la probabilidad de la clase a partir del documento. Nótese que podemos descartar $P(|x|)$ si asumimos que su distribución es independiente de la clase y la longitud del documento es fija.

$$P_\theta(y_j|x) = \frac{P_\theta(y_j)P_\theta(x|y_j)}{P_\theta(x)} = \frac{\theta_j \prod_k (\theta_{jk})^{f_k(x)}}{Z(x)}$$

Donde:

$Z(x)$: reemplazo de $P_\theta(x)$, la probabilidad de generar el documento x. Es una constante de normalización sumando sobre todas las posibles etiquetas de clase.

3.2. Estimación de parámetros del modelo

Para el entrenamiento del clasificador contaremos con un conjunto de instancias etiquetadas $\mathcal{L}$ donde cada elemento puede verse como $\langle x^{(l)}, y^{(l)} \rangle$ siendo x la instancia, y la etiqueta, y $l \in \{1, \ldots, L\}$. Para calcular la probabilidad de alguna de las 2 clases, por ejemplo, que la relación ocurra efectivamente en una instancia dada necesitamos obtener: θ_{jk} y θ_j .

Para θ_{jk} calculamos la proporción de veces que la palabra f_k ocurre en el conjunto de instancias etiquetadas bajo la clase y_j . A su vez incluimos un prior m_{jk} sobre las ocurrencias contadas de f_k bajo la clase y_j para una versión suavizada del estimador de máxima verosimilitud:

$$\theta_{jk} = \frac{m_{jk} + \sum_i P(y_j|x^{(i)}) f_k(x^{(i)})}{Z(f_k)}$$

Donde:

$P(y_j|x^{(i)})$: vale 1 cuando la verdadera etiqueta de $x^{(i)}$ corresponde a la clase y_j . Caso contrario vale 0.

$Z(f_k)$: es una constante de normalización sumando sobre todas las palabras en el vocabulario.

m_{jk}: definido de la siguiente manera $m_{jk} = 1 + \alpha\ \Phi(f_k, y_j)$, con $\Phi(f_k, y_j) \in \{0,\ 1\}$ valiendo 1 cuando f_k es etiquetado bajo y_j, 0 caso contrario.

Con esta fórmula utilizando α ($\alpha > 0$) incrementamos la probabilidad P($f_k | y_j$) (o θ_{jk}) de que la palabra f_k aparezca en un documento de clase y_j cuando se etiqueta f_k bajo y_j. Determinaremos más adelante qué valores de α utilizaremos para ver cómo afecta al experimento.

Para θ_j, de manera similar a θ_{jk}, contamos la proporción de documentos etiquetados con la clase y_j, incluyendo un prior m_j sobre las ocurrencias contadas de estos documentos etiquetados bajo la clase y_j. La estimamos como:

$$\theta_j = \frac{m_j + \sum_i P(y_j|x^{(i)})}{Z(\mathcal{L})}$$

Donde:

$Z(\mathcal{L})$: es una constante de normalización sumando sobre los documentos de $\mathcal{L}$.

m_j: un prior con distribución uniforme, prior de Laplace, donde siempre vale 1.

3.3. Consultas sobre instancias y features

Originalmente el modelo ya descrito es utilizado dentro de un sistema interactivo con aprendizaje activo mediante la selección de los documentos y features más informativos respecto al aprendizaje y la presentación al usuario para obtener una respuesta, esperando mejorar el rendimiento del modelo. A diferencia del trabajo original, no tendremos como oráculo a un humano de manera interactiva en tiempo real para las respuestas a las selecciones de instancias y features hechas por el sistema. En su lugar las instancias seleccionadas serán las próximas a ser utilizadas para el entrenamiento del modelo, y las etiquetas de los features elegidos serán determinadas por una heurística que utilizará la información disponible completa sobre todo el conjunto de entrenamiento.

3.3.1. Selección de instancias

Utilizaremos para el análisis de la información que aportan las instancias la estrategia de muestreo por incertidumbre basado en entropía. Con él obtendremos para las instancias en $\mathcal{U}$ (conjunto de instancias no etiquetadas) el valor de la entropía de la clase posterior bajo el modelo:

$$H_\theta(Y|x) = -\sum_j P_\theta(y_j|x)\ log P_\theta(y_j|x)$$

Cabe aclarar que en nuestro caso no tenemos instancias sin etiquetar, i.e. todas las instancias están etiquetadas, y nos referiremos al conjunto $\mathcal{U}$ como aquel conformado, en cada iteración de entrenamiento, por las instancias aún no utilizadas en el entrenamiento del modelo. De todas formas para este cálculo no utilizamos la etiqueta de clase de las instancias.

Luego el usuario, que en nuestro caso es el sistema, selecciona las *K* instancias con mayor valor, por lo tanto con mayor aporte de mayor información. Esta heurística es una aproximación a consultar la instancia con mayor ganancia de información (dado que la entropía de clase se vuelve cero cuando es etiquetada) bajo la suposición de que cada x es representativa de la distribución natural subyacente de los datos. Una ventaja de esta heurística es su alta velocidad para computar, útil tanto en sistemas interactivos en tiempo real como en este trabajo, lo cual disminuye los tiempos de procesado.

3.3.2. Selección y etiquetado de features

Para seleccionar los features a consultar ordenaremos los features en base a su ganancia de información usando la siguiente fórmula:

$$IG(f_k) = \sum_{\mathbf{I}_k} \sum_j P(\mathbf{I}_k, y_j)\ log \frac{P(\mathbf{I}_k, y_j)}{P(\mathbf{I}_k)P(y_j)}$$

Donde:

$\mathbf{I}_k \in \{0, 1\}$, es una variable que indica la presencia (1) o ausencia (0) de un feature.

Éste es el método general de selección de features para identificar los features más destacados en la clasificación de textos (Sebastiani 2002). En la técnica original para computar $IG(f_k)$ se utilizan las instancias en $\mathcal{L}$ con sus etiquetas reales y las

instancias etiquetadas probabilísticamente de $\mathcal{U}$ (conjunto de instancias no etiquetadas) para reflejar lo que el modelo cree haber aprendido. En este trabajo utilizaremos las etiquetas reales de todas las instancias del conjunto de entrenamiento (incluyendo a las instancias aún no utilizadas en el entrenamiento).

Una vez calculados los valores de IG de cada feature tomamos las mejores *V* features y asociaremos cada uno de ellos f_k con la clase y_j cuando f_k ocurra al menos un 75% del total de ocurrencias en instancias de clase y_j. Caso contrario f_k no tendrá etiqueta alguna de clase. De esta manera consultamos por los features que el modelo cree que son más informativos y les asignamos una etiqueta de clase cuando parezcan estar mayormente correlacionados con ella.

4. Experimentos

4.1. Dataset

El corpus con el cual se realizaron los experimentos está compuesto por 1850 documentos en inglés, los cuales poseen candidatos de la relación *"located-in"* del tipo ORGANIZATION-LOCATION. Nos referiremos al corpus con el nombre ORGLOC. Cada documento contiene oraciones con una o más ocurrencias de relación. Nos referiremos a las ocurrencias de relación junto a su oración como instancias. De estas instancias 2993 son positivas y 10210 negativas. Aquí podemos ver algunos ejemplos del corpus:

This water is then treated at the **Jones Island Water Reclamation Facility** in **Milwaukee**, **Wisconsin** with microbes to digest nutrients that are found in it.

Jones Island Water Reclamation Facility - **Milwaukee** = Positivo
Jones Island Water Reclamation Facility - **Wisconsin** = Positivo

Defense Supply Center, **Richmond**, or DSCR, serves as the Aviation Demand and Supply Chain manager for **Defense Logistics Agency**.

Defense Supply Center - **Richmond** = Positivo
Defense Logistics Agency - **Richmond** = Negativo

4.2. Preparación de los conjuntos de entrenamiento y de evaluación

Para la obtención de los conjuntos se realizaron los siguientes pasos:

- Se decide para el conjunto de evaluación que sus ocurrencias tengan igual proporción de positivas y negativas, es decir 50% cada una. Para el conjunto de entrenamiento las proporciones estarán dadas por la elección de instancias al azar, luego de realizar el proceso de filtrado adecuado que describiremos más adelante.
- Se elige el tamaño del conjunto de evaluación como el 20% de las instancias disponibles.
- Se procede a elegir las instancias para el conjunto de evaluación eligiendo las instancias con una "dificultad" alta, dada por los features: i) distancia entre las entidades del candidato a relación y ii) cantidad de entidades nombradas de la instancia.

- Las instancias de entrenamiento se eligen entre las restantes, aplicando la restricción de no compartir el documento de origen de la instancia con las instancias que se encuentran elegidas para evaluación. De esta manera evitamos filtrar información del conjunto de evaluación en el modelo a entrenar. Al aplicar la restricción queda modificada la proporción original de 80% de instancias para entrenamiento y 20% para evaluación, debido a la exclusión de ciertas instancias del conjunto de entrenamiento. La proporción final queda como 62.37% (4376 instancias) para entrenamiento y 37,63% (2640 instancias) para evaluación.

4.3 Features

La extracción de features está dada por los siguientes extractores:

- Del tipo *sparse*:
    - *bag of words*
    - *bag of word bigrams*
    - *bag of words in between*
    - *bag of word bigrams in between*
- Del tipo *dense*:
    - *entity order*
    - *entity distance*
    - *other entities in between*

Aclaraciones sobre features *sparse*:

- Los features del tipo 'bag of ***e***' consisten en el conjunto de ocurrencias de elementos ***e*** en el texto, eliminando elementos duplicados.
- 'word bigrams' se refiere a cada par de palabras consecutivas. Por ejemplo en *"El día está lluvioso"* tenemos tres word bigrams: *(El, día)*, *(día, está)*, *(está, lluvioso)*.
- La característica 'in between' restringe la extracción sobre el texto al fragmento que se encuentra entre las entidades que pertenecen al candidato de relación.
- En todos los casos mencionados se excluyen stop words, como sugiere Manning 2008, es decir palabras sin significado como artículos, pronombres, etc. y símbolos de puntuación.

Aclaraciones sobre features *dense*:

- 'entity order' devuelve 0 o 1 según el orden de ocurrencia de los tipos de las entidades del candidato de relación.
- 'entity distance' devuelve la cantidad de tokens que se encuentran entre las entidades del candidato de relación.
- 'other entities in between' devuelve la cantidad de otras entidades que ocurren entre las entidades del candidato de relación.

4.4. Experimentos

Los experimentos a realizar se hicieron tomando como base las ideas de Settles 2011, donde se plantea la aplicación de selección instancias y anotación de features en el marco de un entrenamiento semi-supervisado con aprendizaje activo. Utilizamos el clasificador ya mencionado Multinomial Naive Bayes (MNB), con el cual podemos aplicar el parámetro α para la anotación de features, permitiéndonos influir en el peso de algunos features elegidos con algunas clases según consideremos que existe una correlación entre ellos.

La estructura principal de los experimentos será la siguiente:

- Se realiza bajo un procedimiento de aprendizaje activo, por lo que iremos entrenando el modelo en múltiples iteraciones, con una cierta cantidad de instancias por iteración hasta consumir el conjunto de entrenamiento. Al final de cada iteración se utilizará el modelo sobre el conjunto de evaluación para obtener las métricas de desempeño del modelo. Obtendremos los valores de: *tp*, *fp*, *tn*, *fn,* con los que calcularemos la métrica F1-Score.
- Comenzamos eligiendo al azar *N* instancias (*N*=10) para entrenar al clasificador. En las siguientes iteraciones se obtendrán *K* instancias por iteración elegidas bajo algún criterio. El valor de *K* dependerá de la iteración (*I*) del momento e irá incrementándose. Esto nos permite tener mayor detalle en las mediciones iniciales y luego reducir la cantidad de iteraciones finales al ir aumentando *K.* Queda K definido de la siguiente manera para nuestro conjunto de entrenamiento de 4376 instancias:
    - $1 < I < 10$: K=25
    - $I < 16$: K=50
    - $I < 23$: K=75
    - $I < 28$: K=100
    - $I < 33$: K=200
    - $I \geq 33$: K=300

- Cada experimento se realiza L veces (L=3) y sus resultados son unificados aplicando micro promedio, es decir acumulamos los *tp*, *fp*, *tn*, *fn* de las distintas ejecuciones y calculamos las métricas al final. El objetivo esta acumulación es para evitar perturbaciones que puedan provocar alguno de los valores elegidos al azar, como por ejemplo el conjunto inicial de ocurrencias elegidas para entrenar el modelo, dado que luego el modelo puede tener un rol en la selección de las siguientes ocurrencias dependiendo del criterio de selección.

- Para cada una de las L repeticiones de los experimentos, tenemos un estado diferente de aleatoriedad. Todos los experimentos utilizan los mismos estados de aleatoriedad, según la repetición en la que se encuentren. De esta manera intentamos que todos los experimentos partan del mismo estado inicial, por ejemplo al momento de elegir al azar las primeras N instancias.

Para el experimento de base las K ocurrencias a elegir en cada iteración son obtenidas al azar. Utilizaremos la leyenda *random* para denotar esta métrica de selección de instancias. Aquí presentamos su resultado:

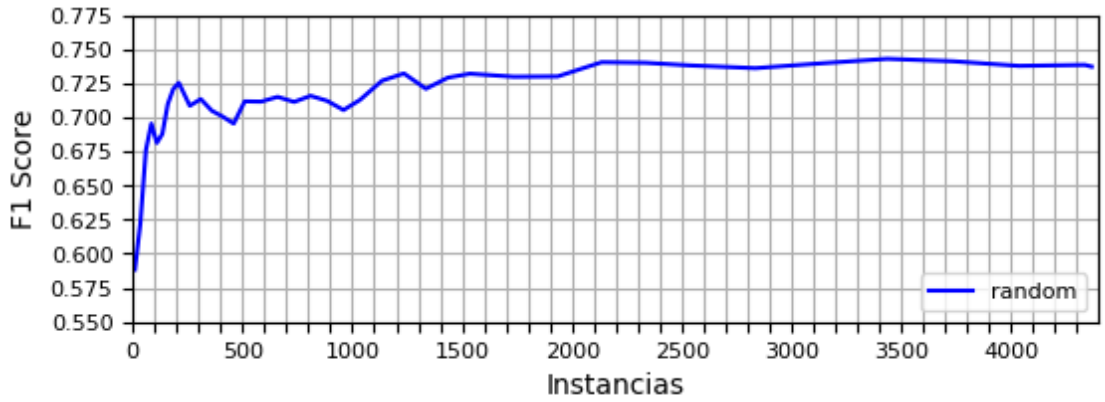

Figura 1. Experimento base.

Se realizaron los siguientes experimentos principales, que comparten las condiciones previamente nombradas a excepción de posibles modificaciones explicitadas en su descripción:

- Selección de instancias
- Etiquetado de features
- Selección de instancias y etiquetado de features

4.4.1. Selección de instancias

Para cada iteración posterior a la primera, las *K* instancias a elegir del conjunto de entrenamiento son seleccionadas en base a alguno de los criterios sobre las instancias que se mencionan luego.

Comencemos denominando $P_\theta(y_0|x)$ (siendo y_0 la clase negativa) como *p0* y $P_\theta(y_1|x)$ (siendo y_1 la clase positiva) como *p1*.

Uncertainty Sampling (o muestreo por incertidumbre) es una de las estrategias de selección de instancias de Settles 2009 que utilizaremos. Es una estrategia simple que prioriza aquellas instancias sobre las cuales el modelo está menos seguro sobre la clase con cual etiquetarlas. En nuestro caso que contamos con 2 clases, podemos definirla como aquella instancia que minimice |*p1* - 0.5| (o |*p1* - *p0*|).

Una variante de esta estrategia se llama Muestreo por incertidumbre basado en entropía, que ya fue definida en el capítulo anterior de este trabajo.

Los criterios de selección que utilizaremos son:

- Muestreo por certeza de clase negativa (*p0* descendente).
- Muestreo por certeza de clase positiva (*p0* ascendente).
- Muestreo por certeza (|*p1* - *p0*| descendente).
- Muestreo por incertidumbre (|*p1* - *p0*| ascendente).
- Muestreo por incertidumbre basado en entropía (IG descendente).
- Muestreo por certeza basado en entropía (IG ascendente).

A continuación veremos los resultados de este experimento bajo las distintas métricas.

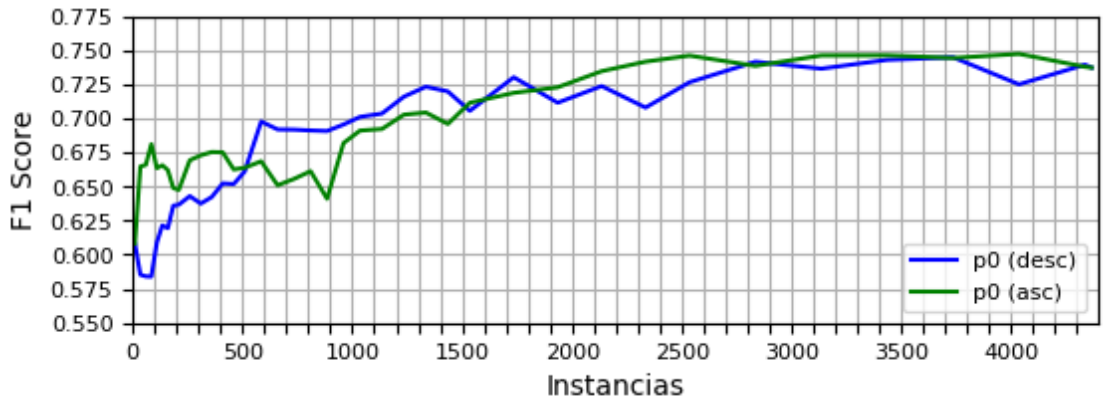

Figura 2. Experimento 4.4.1.: Selección de instancias. Métricas de selección de instancias: *p0* descendente y *p0* ascendente.

En la Figura 2 podemos ver que tiene un mejor comienzo el muestreo por certeza de clase positiva (*p0* desc), aunque luego pierde temporalmente para finalmente tener resultados similares, aunque con leve ventaja *p0* descendente.

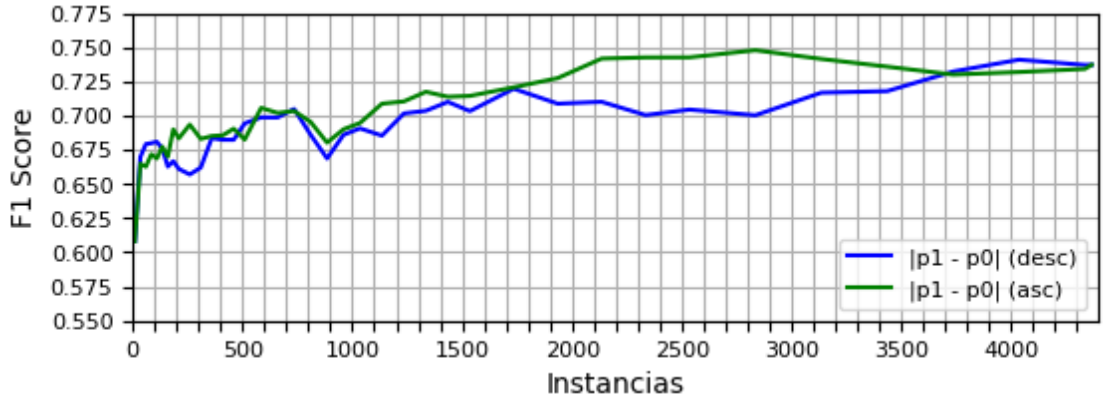

Figura 3. Experimento 4.4.1.: Selección de instancias. Métricas de selección de instancias: *|p1 - p0|* descendente y *|p1 - p0|* ascendente.

Aquí podemos ver que el muestreo por incertidumbre tiene una ventaja general, a pesar de arrancar por debajo con una leve diferencia en contra en las primeras 100 instancias.

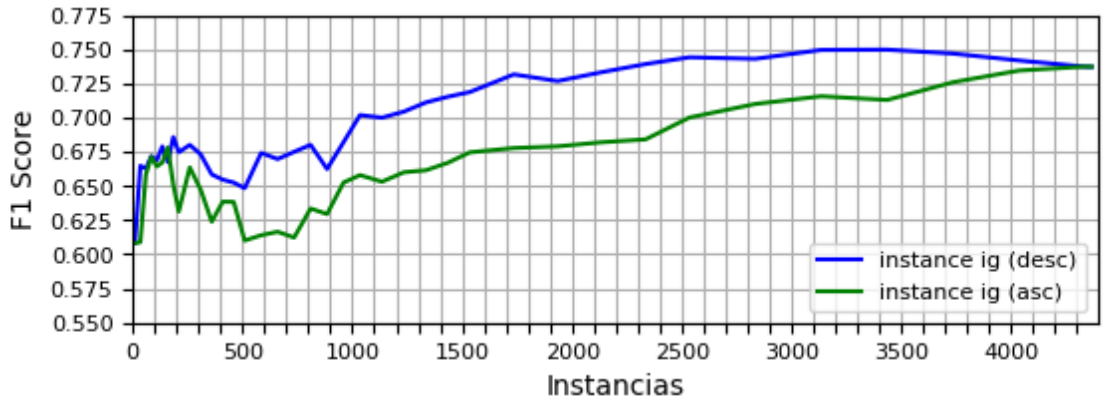

Figura 4. Experimento 4.4.1.: Selección de instancias. Métricas de selección de instancias: *IG* descendente y *IG* ascendente.

Si bien arrancan similar durante las primeras 200 instancias, luego el muestreo por incertidumbre basado en entropía supera con ventaja de varios puntos a la métrica alternativa.

4.4.2. Etiquetado de features

Este proceso consiste en seleccionar los features más informativos y darle mayor peso en el modelo para la clase (positiva o negativa) con la que estén mayormente correlacionados. Para esto utilizaremos el parámetro α, el cual es un valor numérico mayor a 0 que se utiliza para el smoothing aditivo en el modelo de MNB. Para modificar los pesos del clasificador utilizaremos una matriz que tendrá como dimensiones *cf[i]* x |*Y*|, donde *cf[i]* es la cantidad de features del conjunto de instancias elegidas del conjunto de entrenamiento hasta la iteración *i*, e *Y* es el conjunto de clases, por lo tanto |*Y*|=2 en nuestro caso. Si queremos darle peso a un feature hacia una clase, pondremos como valor α=75 en la posición correspondiente a este feature en la matriz. Los valores de la matriz serán sumados al conteo de ocurrencias de cada feature para cada clase.

Para determinar si un feature se asocia favorablemente a alguna clase, nos fijaremos en el conteo de ocurrencias del feature por clase, y si se cumple que la proporción de ocurrencia supera el 75% para alguna clase, lo asociaremos a ésta. Caso contrario descartaremos la selección de ese feature para el etiquetado.

Nuestro objetivo es seleccionar los features que provean al modelo con la mayor información posible para optimizar la relación esfuerzo y rendimiento, i.e. queremos obtener un mejor rendimiento y acortar el esfuerzo requerido para ello obteniendo mayor información de las instancias. Para ello calcularemos el Information Gain de todos los features, que ya definimos en el capítulo anterior.

Al final de la primera iteración se calcula el Information Gain para todos los features de todas las instancias del conjunto de entrenamiento, tanto las ya utilizadas en el entrenamiento como las que no, utilizando las etiquetas de clase real de las instancias.

A partir de la segunda iteración inclusive, previo al entrenamiento del modelo, se elegirán los *M* features para ser etiquetados bajo las siguientes condiciones:

- la cantidad de features candidatos por iteración (*M*) será el 5% de la cantidad de instancias usadas hasta el momento para el entrenamiento.
- de los M features candidatos se elegirán los que tengan mayor IG, pero tendrá prioridad que la proporción de features asociados a la clase positiva sea como mínimo de 40%.

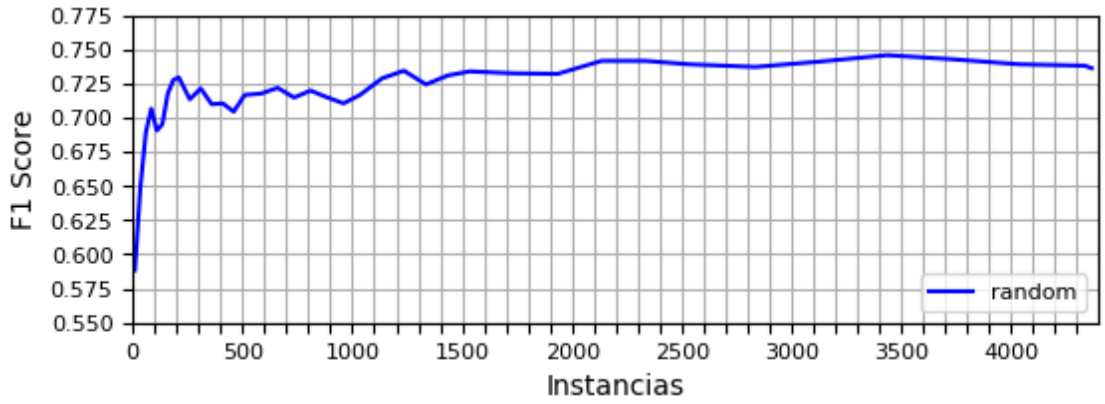

Figura 5. Experimento 4.4.2.: Etiquetado de features.

Por el momento sólo diremos que se asemeja a la Figura 1. del experimento base. Dejaremos para más adelante el análisis más detallado y las comparaciones.

4.4.3. Selección de instancias y etiquetado de features

Se combinan los experimentos anteriores de selección de instancias y etiquetado de features, aplicándolos simultáneamente.

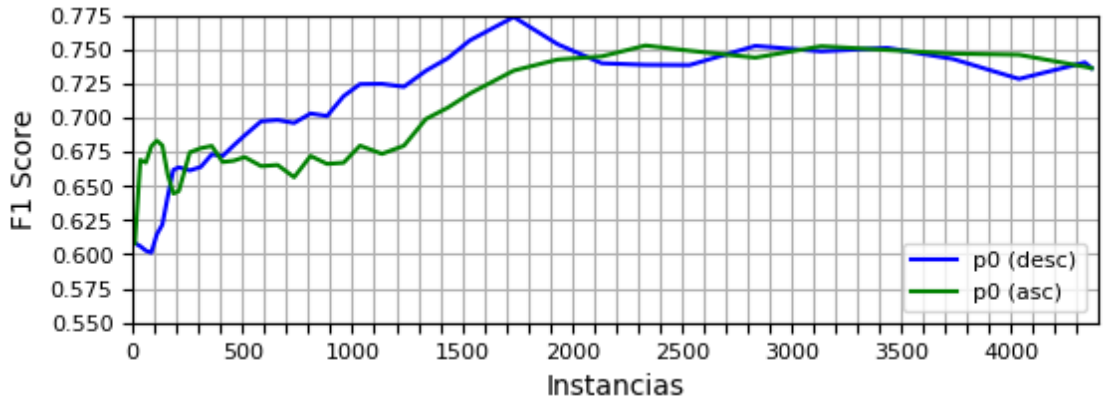

Figura 6. Experimento 4.4.3.: Selección de instancias y etiquetado de features. Métricas de selección de instancias: *p0* descendente y *p0* ascendente con etiquetado de features.

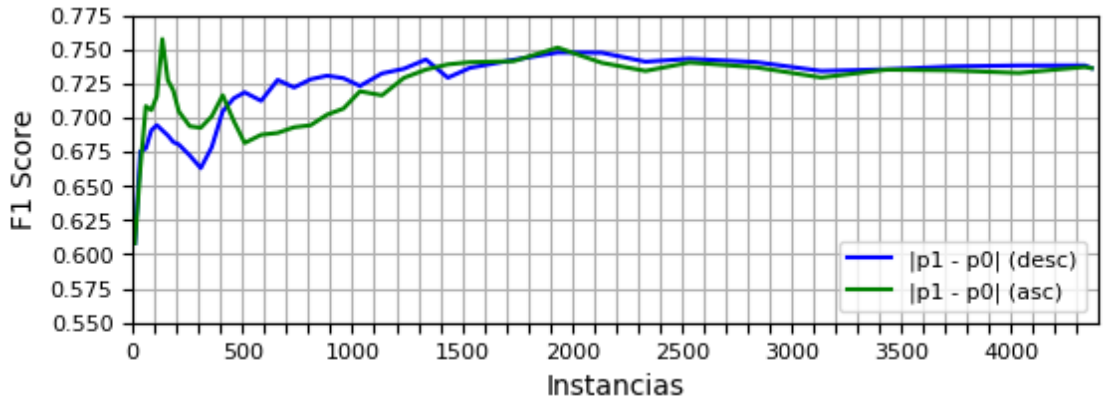

Figura 7. Experimento 4.4.3.: Selección de instancias y etiquetado de features. Métricas de selección de instancias: |*p1 - p0*| descendente y |*p1 - p0*| ascendente con etiquetado de features.

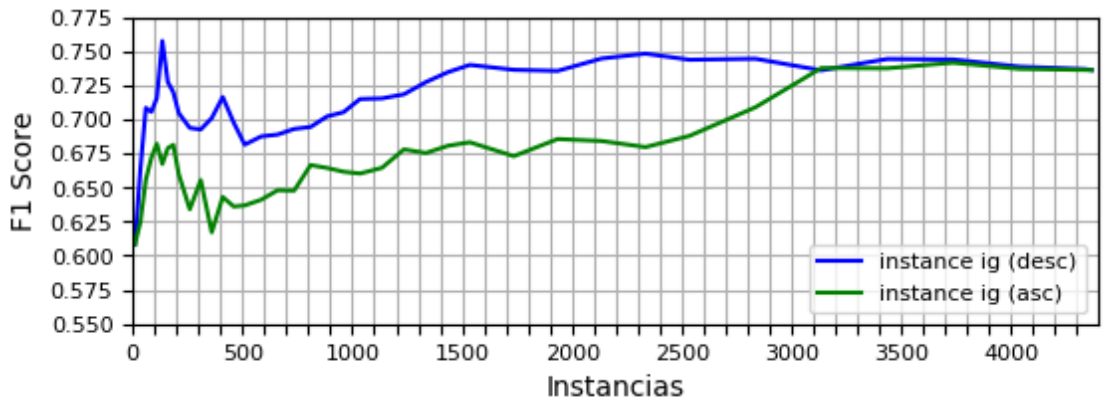

Figura 8. Experimento 4.4.3.: Selección de instancias y etiquetado de features. Métricas de selección de instancias: *IG* descendente y *IG* ascendente con etiquetado de features.

4.5. Análisis de resultados

A continuación podremos analizar los distintos experimentos y realizar observaciones y comparaciones en base a los siguientes gráficos.

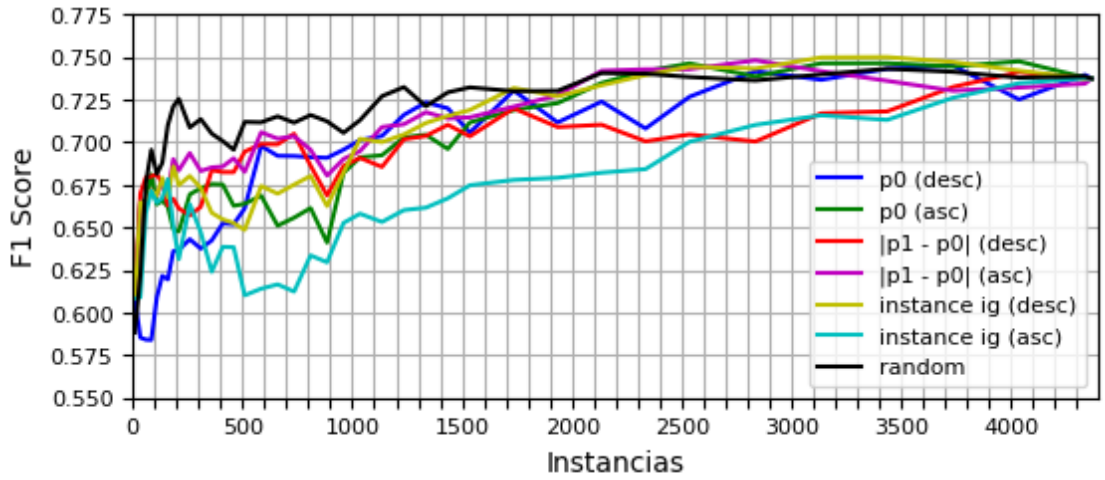

Figura 9. Experimento base y 4.4.1. Comparación de todas las métricas de selección de instancias.

En la Figura 9. tenemos todas las métricas del experimento base y 4.4.1. De esta manera podemos observar lo siguiente:

- la métrica *random* encabeza el ranking de rendimiento durante la primera mitad del entrenamiento y luego es levemente superada por algunas métricas. De todas maneras podríamos tomarla como ganadora.

- otra métrica destacable en rendimiento es la de muestreo por incertidumbre (|*p1* - *p0*| asc) la cual podríamos decir que mantiene el 2do lugar en general, aunque en las últimas 1000 instancias finales de entrenamiento decae levemente.

- las métricas que peor andan son muestreo por certeza de la clase negativa (*p0* desc) y muestreo por certeza basado en entropía (*IG* asc). Esto parecería tener sentido si tenemos en cuenta que tener certeza en que no ocurre la relación no aporta mucha utilidad, y la otra métrica son las que menos ganancia de información aportan.

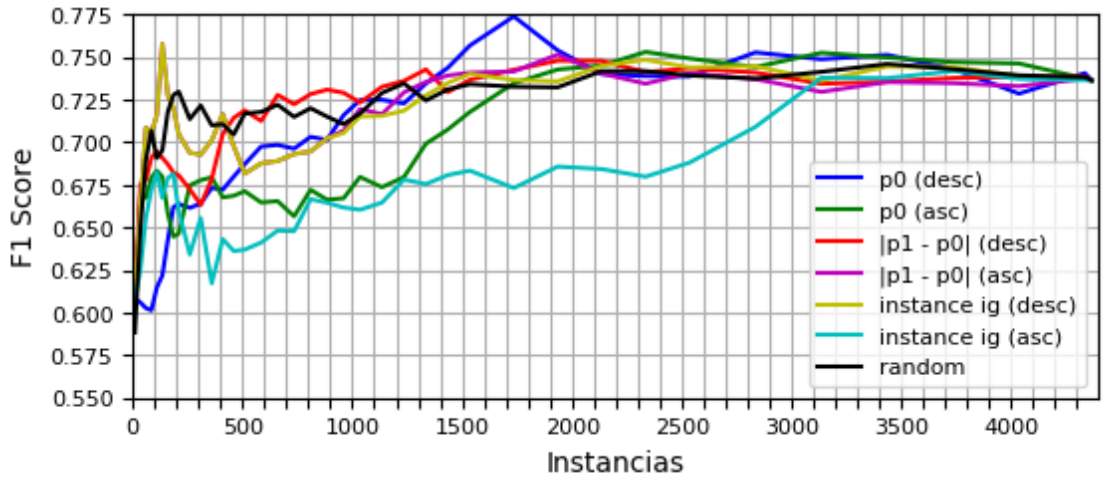

Figura 10. Experimento 4.4.2 y 4.4.3. Comparación de todas las métricas de selección de instancias con etiquetado de features.

De la Figura 10. podemos observar:

- se destacan inicialmente el muestreo por incertidumbre (|p1 - p0| asc) y el muestreo por incertidumbre basado en entropía (*IG* desc), quienes tienen curvas bastante similares. Obtienen los mejores rendimientos durante casi las 200 primeras instancias. Luego desde las 400 instancias hasta las 1000, se desempeñan por debajo de las mejores. A partir de allí tienen un desempeño similar a las mejores.
- la métrica *random* entra en el conjunto de las métricas con buen desempeño.
- similar al análisis de la Figura 9, tenemos que las métricas de muestreo por certeza de la clase negativa (*p0* desc) y muestreo por certeza basado en entropía (*IG* asc) no tienen buen desempeño. En este caso también podemos incluir al muestreo por certeza de la clase positiva (*p0* asc) en este grupo. Un detalle es que a pesar de tener mal desempeño en general, *p0* desc alcanza el máximo rendimiento momentáneamente cerca de la mitad del entrenamiento.

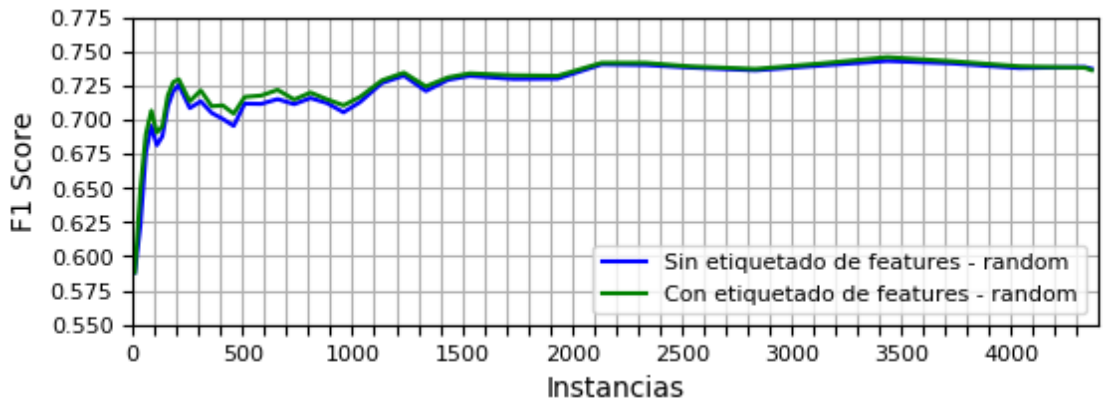

Figura 11. Comparación entre experimento base y 4.4.2. Métrica de selección de instancias *random*, sin y con etiquetado de features.

Se puede observar una leve mejora en rendimiento en la métrica *random* cuando utilizamos etiquetado de features, sin embargo no es muy significativa.

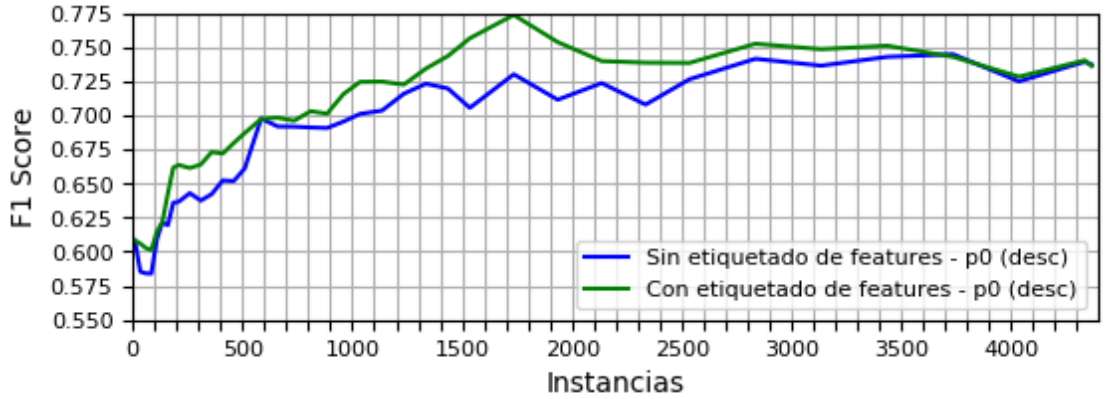

Figura 12. Comparación entre experimento 4.4.1 y 4.4.3. Métrica de selección de instancias *p0* descendente, sin y con etiquetado de features.

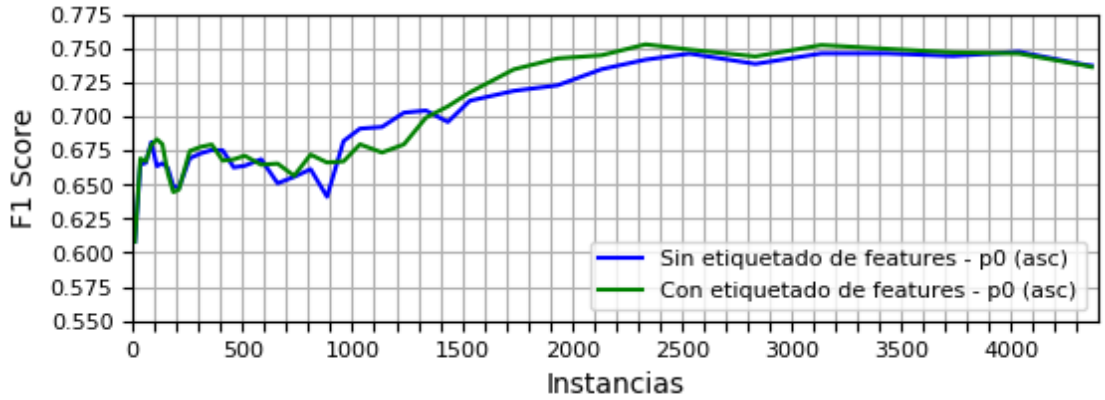

Figura 13. Comparación entre experimento 4.4.1 y 4.4.3. Métrica de selección de instancias *p0* ascendente, sin y con etiquetado de features.

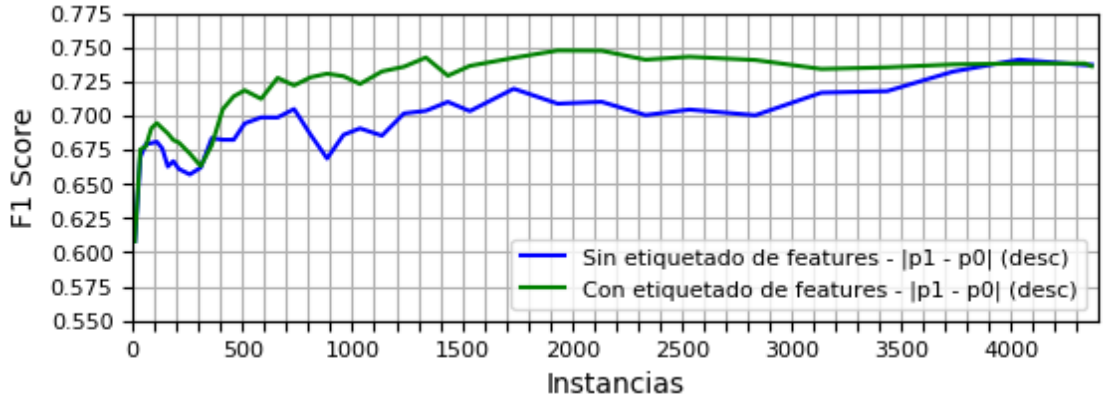

Figura 14. Comparación entre experimento 4.4.1 y 4.4.3. Métrica de selección de instancias |*p1 - p0*| descendente, sin y con etiquetado de features.

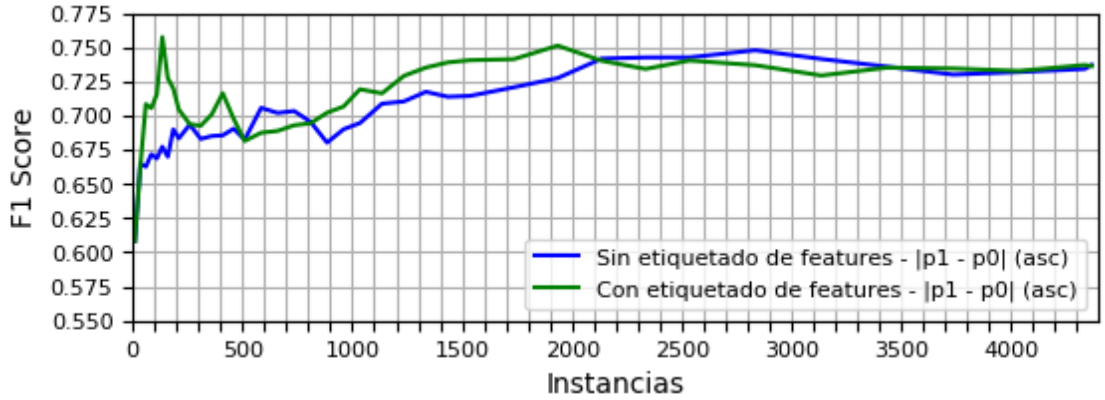

Figura 15. Comparación entre experimento 4.4.1 y 4.4.3. Métrica de selección de instancias |*p1 - p0*| ascendente, sin y con etiquetado de features.

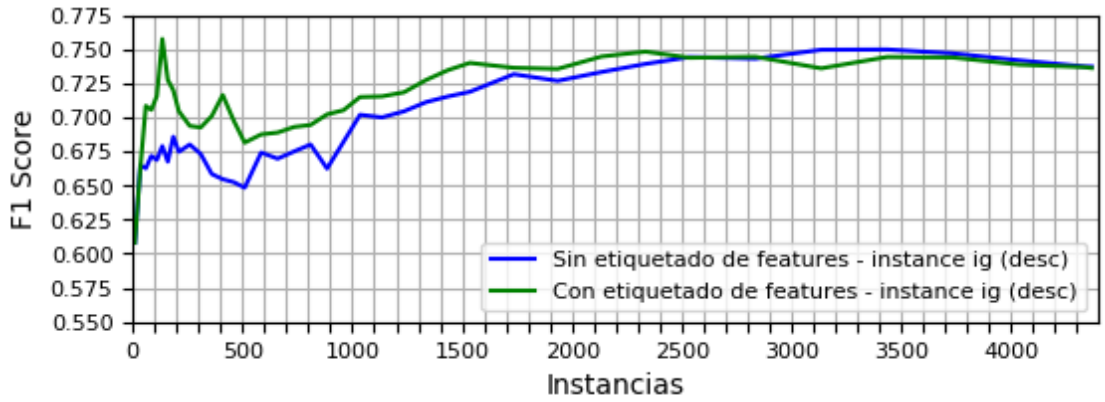

Figura 16. Comparación entre experimento 4.4.1 y 4.4.3. Métrica de selección de instancias *IG* descendente, sin y con etiquetado de features.

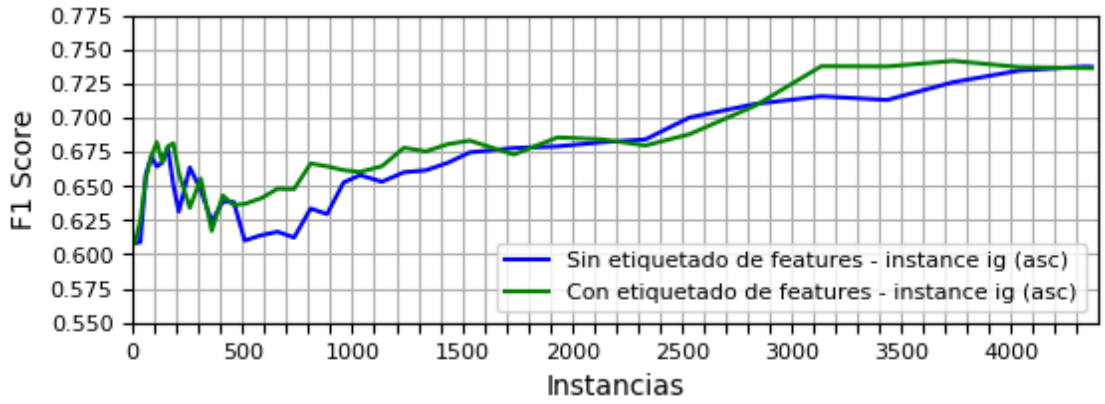

Figura 17. Comparación entre experimento 4.4.1 y 4.4.3. Métrica de selección de instancias *IG* ascendente, sin y con etiquetado de features.

En el resto de las figuras (desde la Figura 12 hasta la Figura 17), podemos observar una mejora significativa en general a grosso modo de un 2.5%. La curva relacionada al etiquetado de features es casi siempre superior a la curva sin etiquetado.

5. Experimentos complementarios

Junto a los experimentos principales anteriormente nombrados se han realizado algunas modificaciones con el fin de solucionar inconvenientes y obtener mejores resultados. A continuación describiremos los experimentos complementarios relacionados.

5.1. Experimento complementario 1

Exclusión de instancias negativas que comparten la oración con instancias positivas

Descripción

Al realizar los experimentos se encontró el siguiente problema: hay casos donde instancias negativas y positivas se dan sobre la misma oración. Por lo tanto los features obtenidos de ellas son similares. Esto provocaba que ciertos features potencialmente indicadores de la ocurrencia positiva o negativa relación, como *('is', 'located')-(bag of words bigram)* para el caso positivo, no tuvieran un sesgo claro hacia la clase correspondiente. Dicha situación dificultaba el análisis de la hipótesis sobre la mejora de rendimiento con el etiquetado de features.

Veamos un caso de ejemplo:

The academy is located in the center of the town of **Wilbraham**, 1.5 hours from **Boston** and 3 hours from **New York City**.

The academy - **Wilbraham** = Positivo
The academy - **Boston** = Negativo
The academy - **New York City** = Negativo

Para remover este inconveniente se decidió excluir todas las instancias negativas cuyas oraciones también tengan al menos 1 instancia positiva.

Resultados

El resultado de esta modificación dejó al corpus con 2993 instancias positivas y 6782 instancias negativas. La nueva proporción de clases sigue siendo válida, por lo tanto no presenta un problema. La proporción entre el conjunto de entrenamiento y de evaluación queda como 69.07% (4366 instancias) para entrenamiento y 30,93% (1955 instancias) para evaluación.

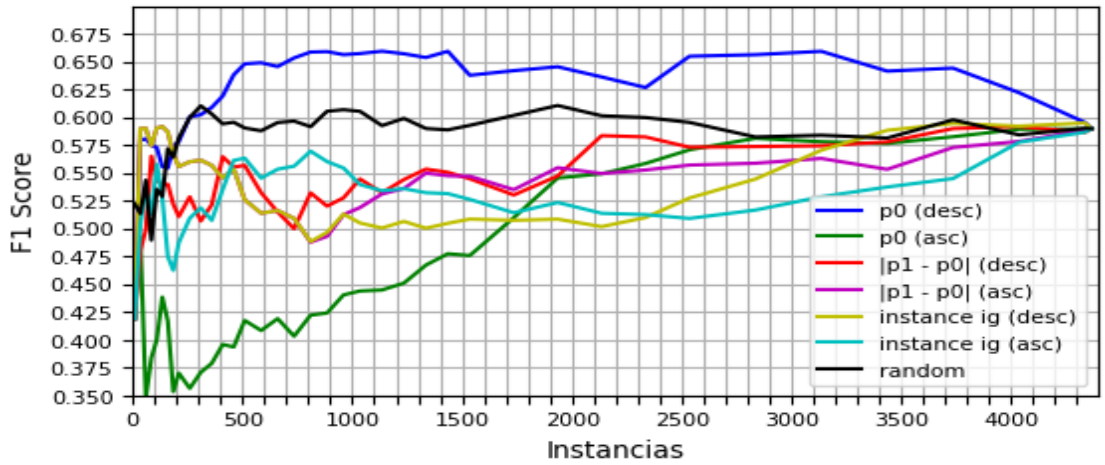

Figura 18. Resultados del experimento base y 4.4.1 utilizando el dataset sin excluir dichas instancias negativas.

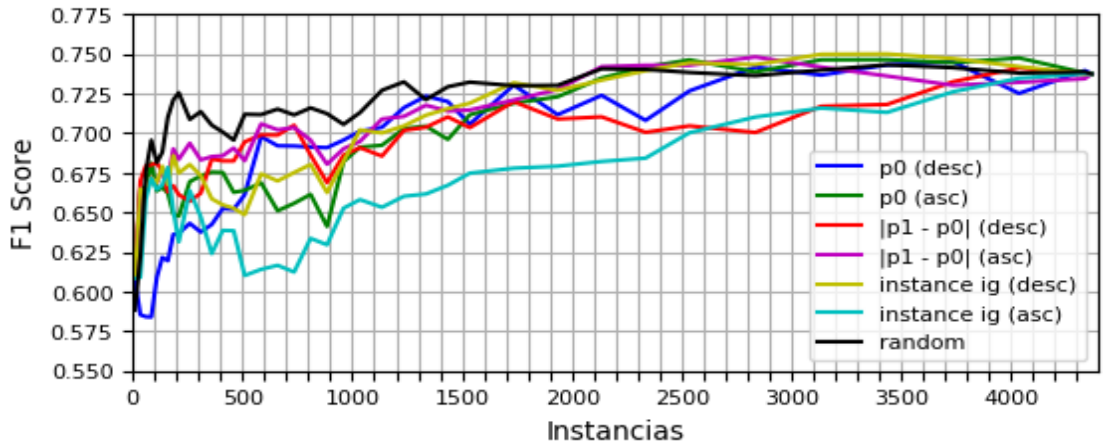

Figura 19. Resultados del experimento base y 4.4.1 utilizando el dataset excluyendo dichas instancias negativas.

Ya se puede apreciar una diferencia significativa en el rendimiento de ambos casos, favoreciendo la exclusión de las instancias. Haremos las comparaciones por métrica para simplificar el análisis visual.

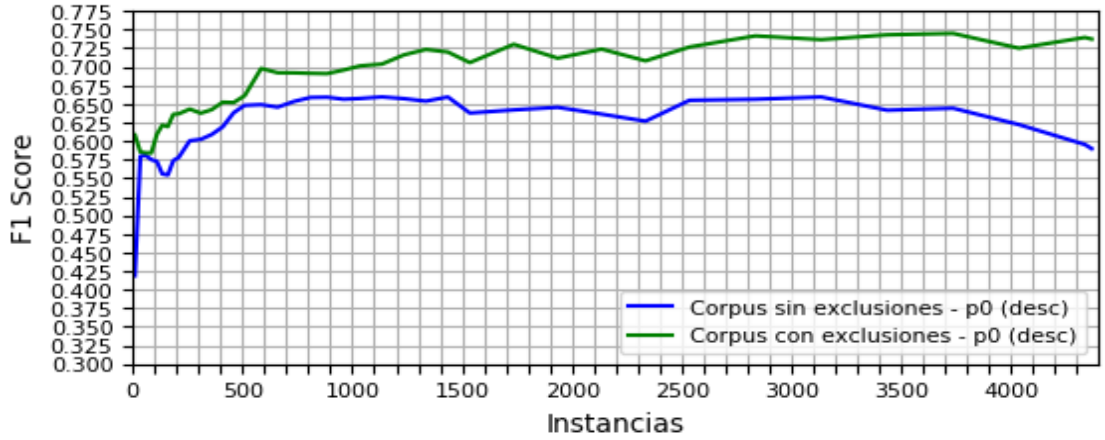

Figura 20. Resultados del experimento 4.4.1 utilizando la métrica *p0* descendente para el corpus sin exclusiones y con exclusiones.

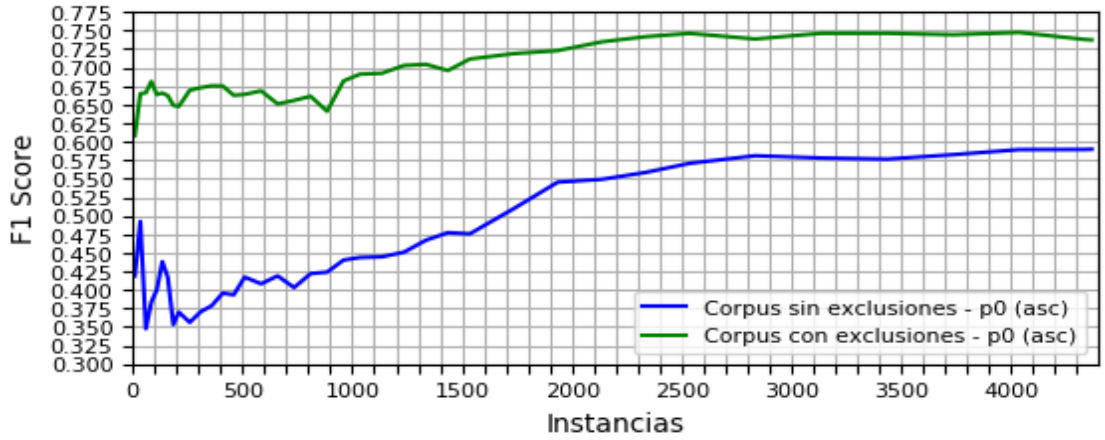

Figura 21. Resultados del experimento 4.4.1 utilizando la métrica *p0* ascendente para el corpus sin exclusiones y con exclusiones.

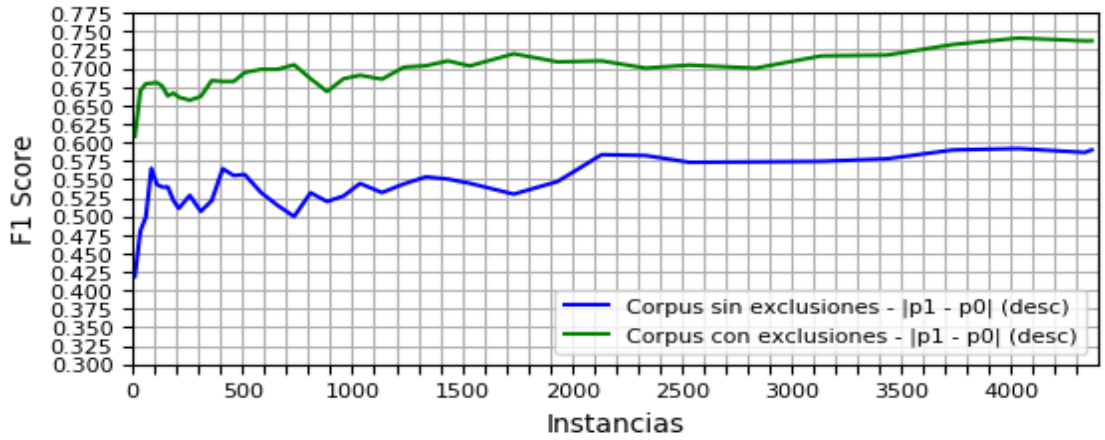

Figura 22. Resultados del experimento 4.4.1 utilizando la métrica |*p1* - *p0*| descendente para el corpus sin exclusiones y con exclusiones.

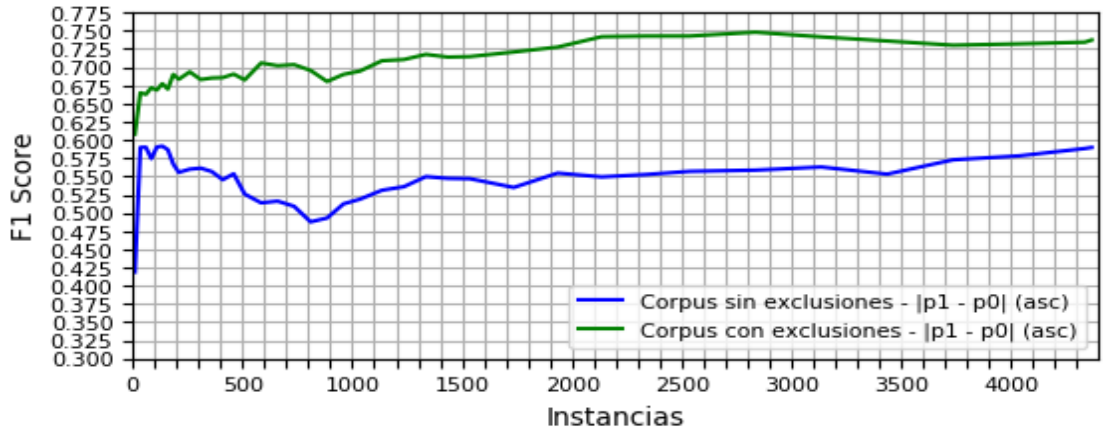

Figura 23. Resultados del experimento 4.4.1 utilizando la métrica |*p1* - *p0*| ascendente para el corpus sin exclusiones y con exclusiones.

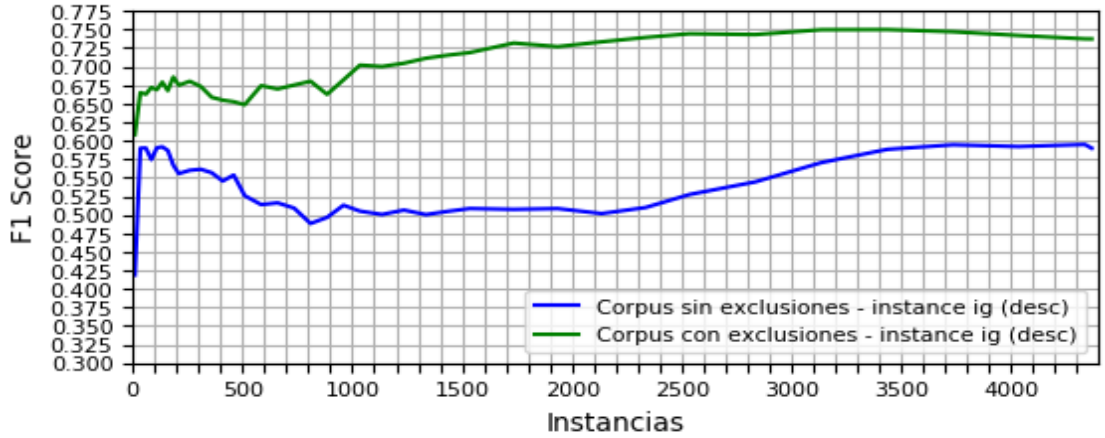

Figura 24. Resultados del experimento 4.4.1 utilizando la métrica *IG* descendente para el corpus sin exclusiones y con exclusiones.

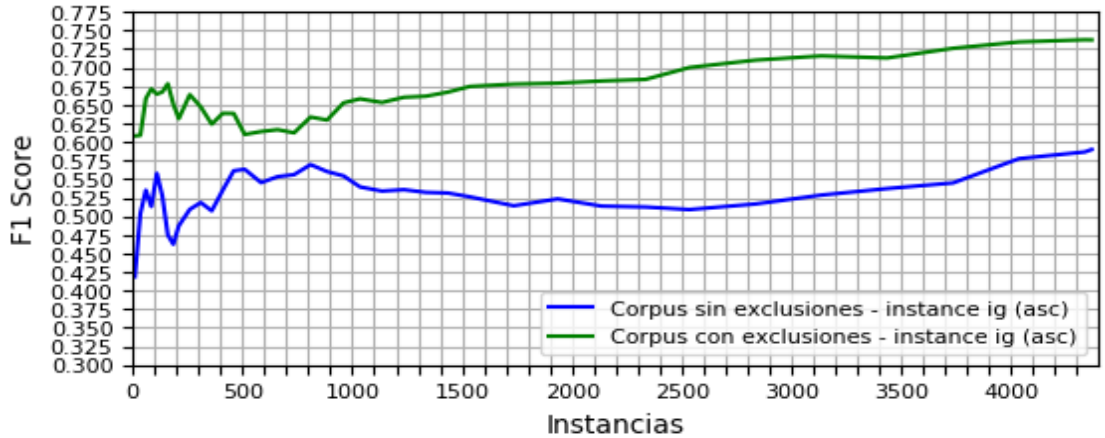

Figura 25. Resultados del experimento 4.4.1 utilizando la métrica *IG* ascendente para el corpus sin exclusiones y con exclusiones.

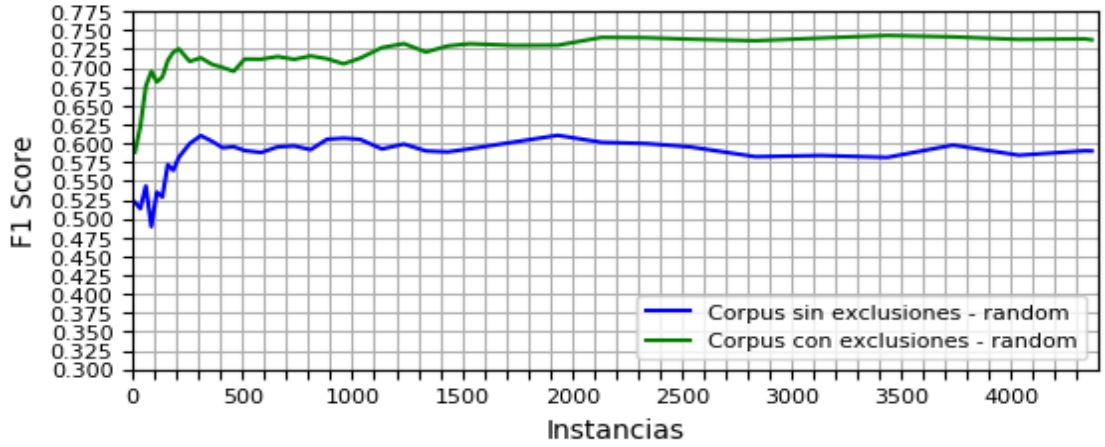

Figura 26. Resultados del experimento base utilizando la métrica *random* para el corpus sin exclusiones y con exclusiones.

Se muestran en todas estas figuras las mejoras individuales de cada métrica.

Hasta el momento vimos los cambios referidos al experimento sin etiquetado de features. Veamos a continuación en concreto los detalles de estos cambios en el proceso de selección y etiquetado de features originales del experimento 4.4.2.

Referencias:

- BOW: bag_of_words
- BIGR: bigrams
- no_SW: no_stopwords
- no_PU: no_punctuation

Experimento 4.4.2. Etiquetado de features - **Sin exclusiones** - Iteración 5 de entrenamiento:

- Features candidatos elegidos **positivos**:
  - *('school', 'located')* - 'BOW_BIGR_in_between_no_SW_no_PU' - NO: 16 - YES: 52
  - *('public', 'high')* - 'BOW_BIGR_in_between_no_SW_no_PU' - NO: 7 - YES: 34
  - *('company', 'founded')* - 'BOW_BIGR_no_SW_no_PU' - NO: 1 - YES: 6
  - *'covance'* - 'BOW_no_SW_no_PU' - NO: 0 - YES: 4

- Features candidatos elegidos **negativos**:
  - *"'s"* - 'BOW_no_SW_no_PU' - NO: 628 - YES: 68
  - *'including'* - 'BOW_no_SW_no_PU' - NO: 173 - YES: 5
  - *"'s"* - 'BOW_in_between_no_SW_no_PU' - NO: 215 - YES: 12
  - *'army'* - 'BOW_no_SW_no_PU' - NO: 146 - YES: 7

    - *'service'* - 'BOW_no_SW_no_PU' - NO: 144 - YES: 9

- Features candidatos **no elegidos**:
    - *'located'* - 'BOW_in_between_no_SW_no_PU' - NO: 193 - YES: 209
    - *'located'* - 'BOW_no_SW_no_PU' - NO: 311 - YES: 229
    - *'school'* - 'BOW_in_between_no_SW_no_PU' - NO: 123 - YES: 140
    - *'school'* - 'BOW_no_SW_no_PU' - NO: 520 - YES: 243
    - *('school', 'located')* - 'BOW_BIGR_no_SW_no_PU' - NO: 92 - YES: 98

Experimento 4.4.2. Etiquetado de features - **Con exclusiones** - Iteración 5 de entrenamiento:

- Features candidatos elegidos **positivos**:
    - *'located'* - 'BOW_in_between_no_SW_no_PU' - NO: 60 - YES: 303
    - *'located'* - 'BOW_no_SW_no_PU' - NO: 110 - YES: 333
    - *('school', 'located')* - 'BOW_BIGR_no_SW_no_PU' - NO: 8 - YES: 142
    - *'based'* - 'BOW_in_between_no_SW_no_PU' - NO: 19 - YES: 131
    - *'based'* - 'BOW_no_SW_no_PU' - NO: 50 - YES: 170
    - *'headquartered'* - 'BOW_in_between_no_SW_no_PU' - NO: 3 - YES: 81
    - *'headquartered'* - 'BOW_no_SW_no_PU' - NO: 12 - YES: 98

- Features candidatos elegidos **negativos**:
    - *"'s"* - 'BOW_no_SW_no_PU' - NO: 659 - YES: 100
    - *'including'* - 'BOW_no_SW_no_PU' - NO: 219 - YES: 5

- Features candidatos **no elegidos**:
    - *'school'* - 'BOW_in_between_no_SW_no_PU' - NO: 91 - YES: 229
    - *'school'* - 'BOW_no_SW_no_PU' - NO: 366 - YES: 367

En el experimento sin exclusiones podemos ver que en esta iteración no asocia como positivos a features como *'located',* dado que aparece en una proporción bastante pareja entre ambas clases. En el experimento con exclusiones posee mayor cantidad de asociaciones positivas que negativas, lo cual permite etiquetarlo como positivo. A su vez la cantidad de features candidatos relevantes rechazados es menor y los features elegidos también son más útiles. Es a partir de estos resultados donde mejora el rendimiento del modelo y soluciona el inconveniente mencionado entre features y clases, que decidimos utilizar el corpus con las exclusiones de dichas instancias negativas en los experimentos principales.

5.2. Experimento complementario 2

Selección del conjunto de evaluación basado en heurísticas

Descripción

El conjunto de evaluación original estaba formado por las primeras 1955 instancias del conjunto, manteniendo una proporción de 50% de instancias con etiqueta positiva y negativa. Se propuso un conjunto de evaluación alternativo manteniendo igual cantidad de instancias y proporción de clases. Para obtenerlo tomamos las instancias que obtengan mayor puntaje basado en la siguiente heurística: *distancia entre entidades* + *cantidad total de entidades*. De esta manera apuntamos a disminuir la cantidad de instancias sencillas en el conjunto de evaluación y reforzar el aprendizaje del modelo, es decir ver que el rendimiento aumente con el tiempo.

Resultados

Veamos primero en general los rendimientos de cada conjunto de evaluación siendo usados en el experimento base y el 4.4.1.

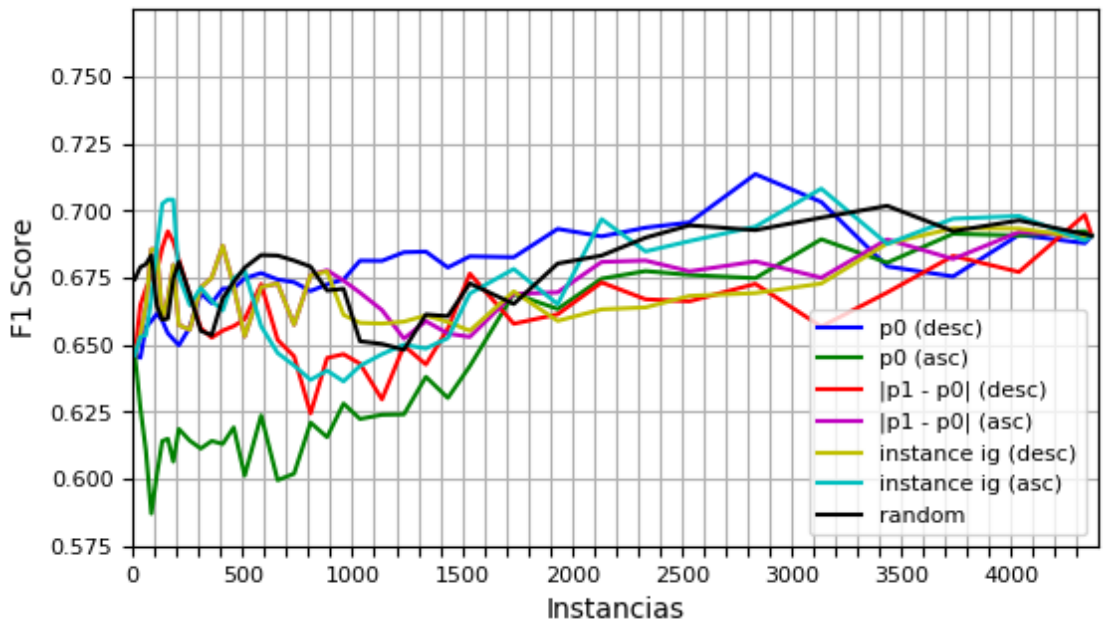

Figura 27. Resultados del experimento base y 4.4.1 utilizando el conjunto de test formado por las primeras 1955 instancias del corpus.

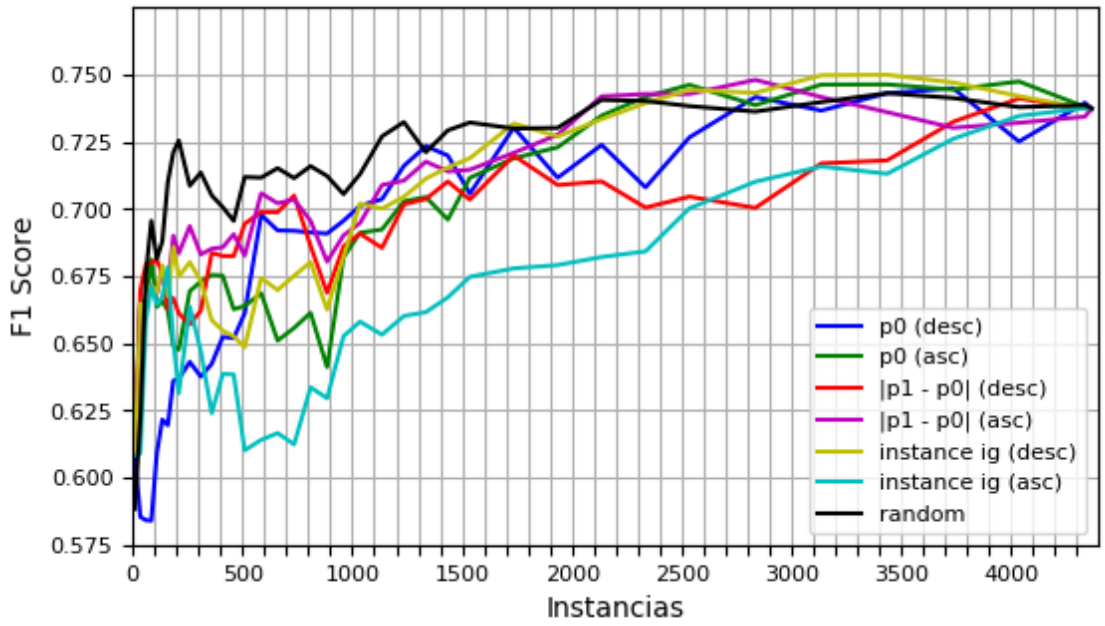

Figura 28. Resultados del experimento base y 4.4.1 utilizando el conjunto de test formado por las primeras 1955 instancias del corpus basadas en heurísticas.

Podemos notar una diferencia en la trayectoria de las curvas en ambos casos, teniendo una pendiente de crecimiento mayor en el caso del conjunto alternativo propuesto, lo cual es positivo.

Veamos ahora la comparación de a una métrica a la vez.

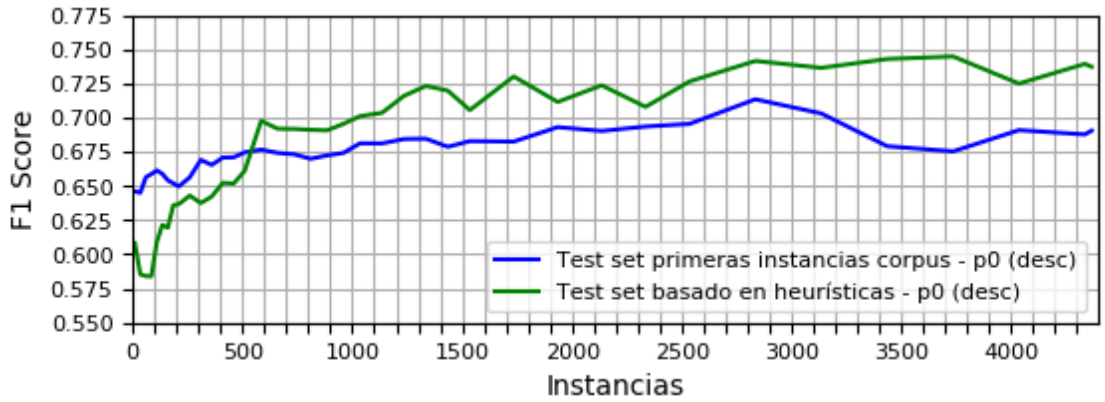

Figura 29. Resultados del experimento 4.4.1 utilizando la métrica *p0* descendente para ambos conjuntos de evaluación.

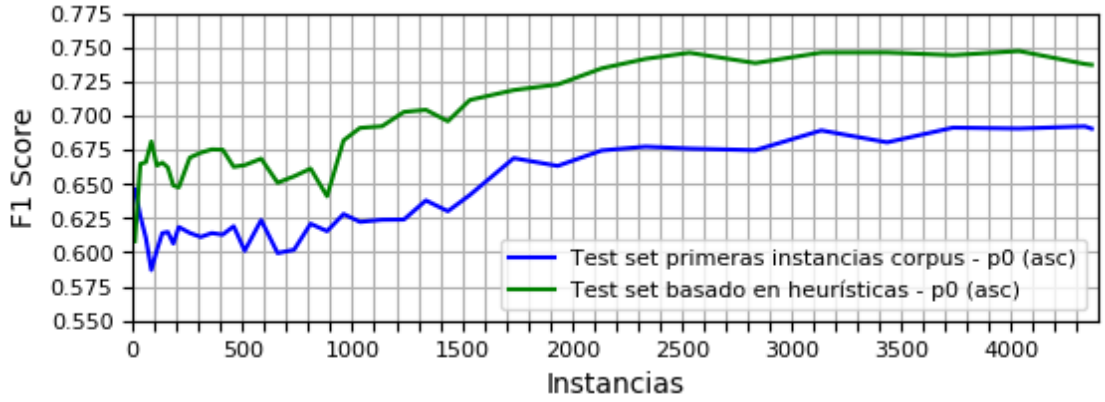

Figura 30. Resultados del experimento 4.4.1 utilizando la métrica *p0* ascendente para ambos conjuntos de evaluación.

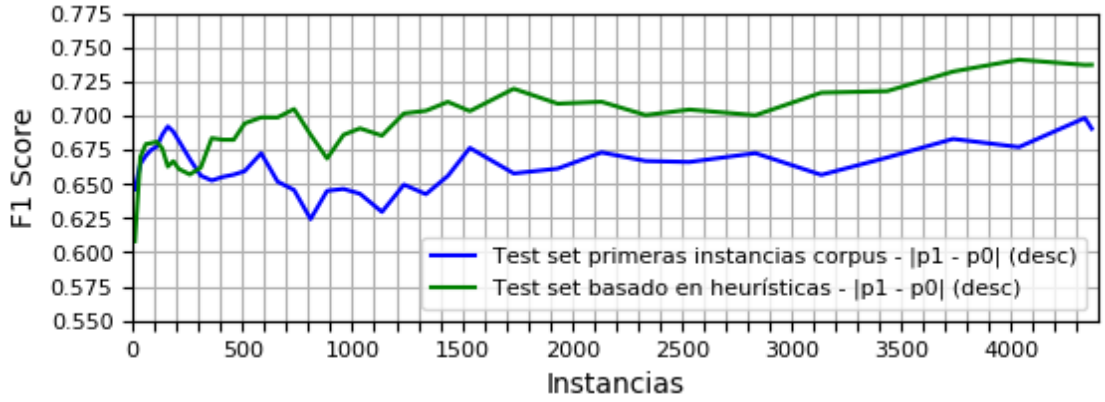

Figura 31. Resultados del experimento 4.4.1 utilizando la métrica |*p1* - *p0*| descendente para ambos conjuntos de evaluación.

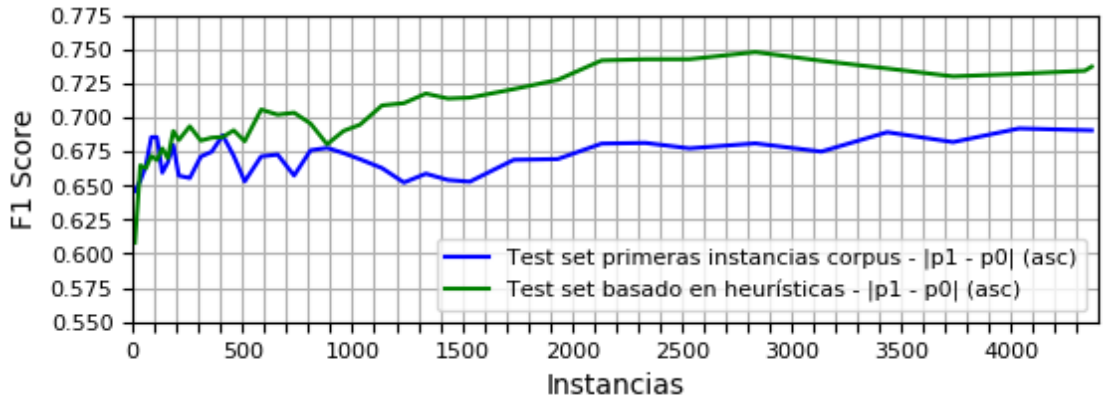

Figura 32. Resultados del experimento 4.4.1 utilizando la métrica |*p1* - *p0*| ascendente para ambos conjuntos de evaluación.

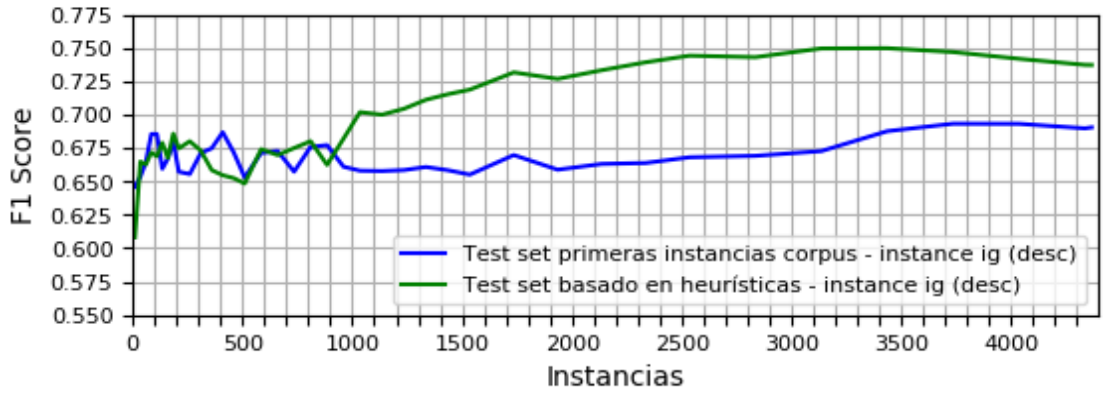

Figura 33. Resultados del experimento 4.4.1 utilizando la métrica *IG* descendente para ambos conjuntos de evaluación.

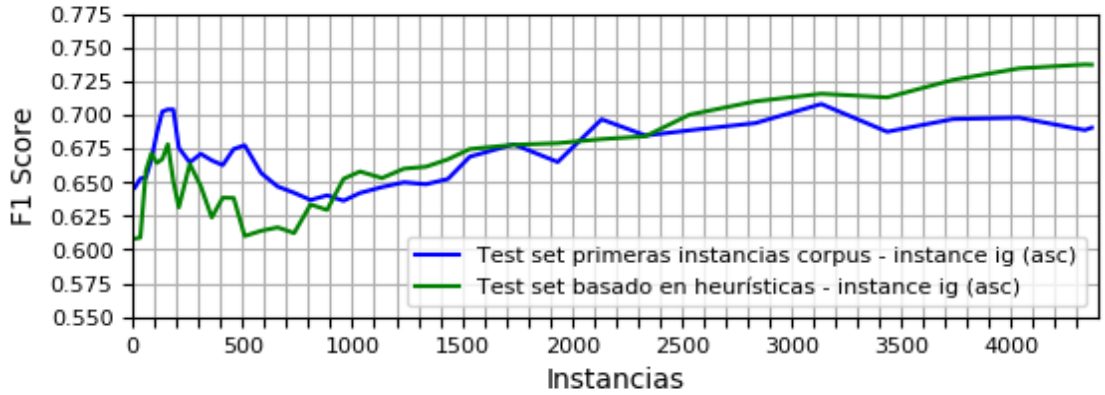

Figura 34. Resultados del experimento 4.4.1 utilizando la métrica *IG* ascendente para ambos conjuntos de evaluación.

Figura 35. Resultados del experimento base utilizando la métrica *random* para ambos conjuntos de evaluación.

Debido a las mejoras dadas en las curvas del conjunto basado en heurísticas, este conjunto será utilizado para la evaluación en los experimentos principales.

5.3. Experimento complementario 3

Etiquetado de features: elección del valor de α a utilizar

Descripción

Uno de los parámetros dentro de los experimentos con etiquetado de features es α. Cuando etiquetamos un feature f_k bajo la etiqueta y_j, éste parámetro nos permite modificar el prior m_{jk} que usamos en el cálculo de la probabilidad $P(f_k|y_j)$ para, por ejemplo, enfatizar la asociación entre la palabra f_k y la clase de documento y_j.
Recordemos que m_{jk} está definido como:

$$m_{jk} = 1 + \alpha\, \Phi(f_k, y_j)$$

con $\Phi(f_k, y_j) \in \{0, 1\}$ valiendo 1 cuando f_k es etiquetado bajo y_j, 0 caso contrario.

En Settles 2011 se midió la precisión de su modelo utilizando valores de α que iban desde 1 a 4096. Notó que la precisión se mantiene relativamente estable utilizando α < 100. En particular, se utilizó α=50 para sus experimentos.

En este trabajo probaremos los siguientes valores para α: 10, 50, 75, 150, 300, 500.

Resultados

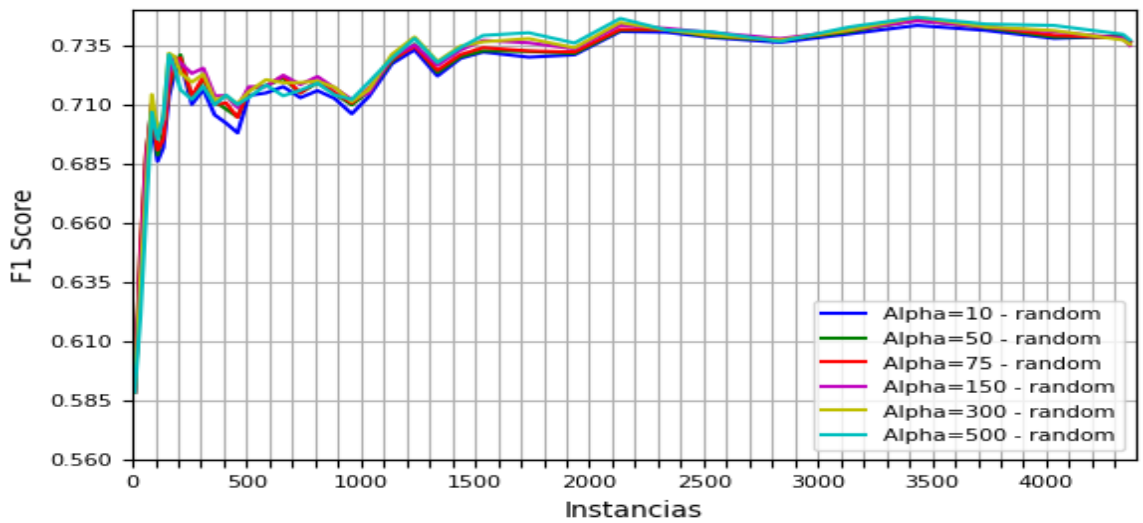

Figura 36. Resultados del experimento 4.4.2. con los distintos valores de α.

Mostramos a continuación la parte relevante de análisis con mayor detalle.

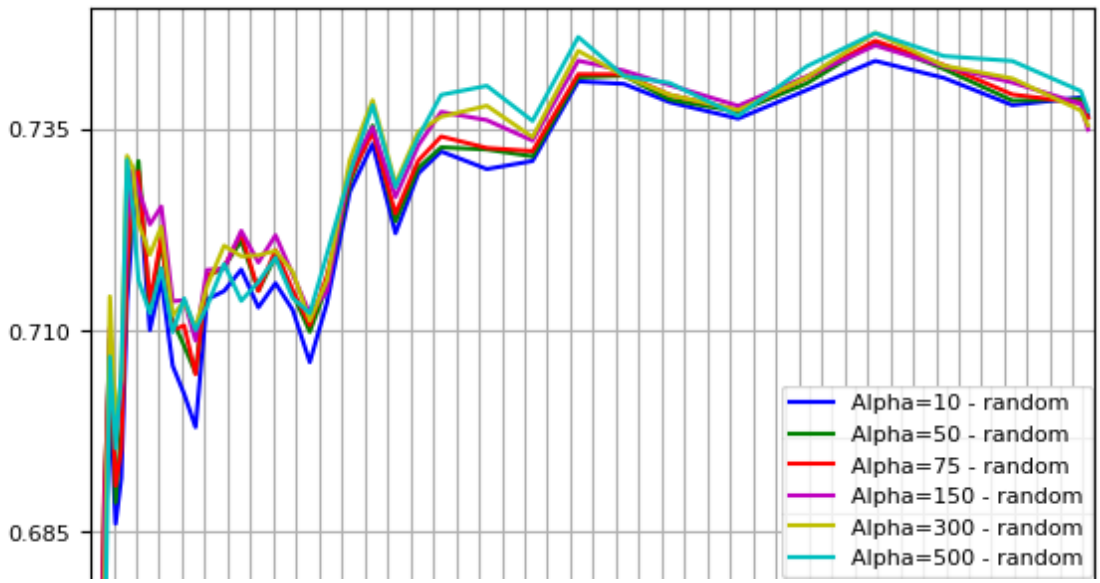

Figura 37. Resultados del experimento 4.4.2. con los distintos valores de α, sección relevante del gráfico con mayor detalle.

La diferencia de rendimiento no es muy significativa entre los distintos valores de α. En general α=10 tiene un desempeño ligeramente peor que el resto por lo que queda descartado. Entre las opciones restantes tienen un comportamiento similar salvo en algunas franjas pequeñas. Por una cuestión de estabilidad respecto a la influencia de los pesos del modelo utilizaremos el valor de α**=75**.

6. Conclusiones y trabajo futuro

Gracias al desarrollo de los experimentos que hemos planteado en este trabajo pudimos observar el impacto en el rendimiento de un modelo, generado por un clasificador MNB en el marco de un sistema de aprendizaje activo aplicando ciertas técnicas de selección de instancias y etiquetado de features. Sobre los experimentos sin etiquetado de features, podemos decir que las métricas de selección de instancias no han cumplido nuestras expectativas, comparándolas con la métrica *random*. Sin embargo, en los experimentos con etiquetado de features encontramos varios resultados positivos. En general la aplicación de las etiquetas en los features ha mejorado levemente el rendimiento en todas las métricas de selección de instancias, incluyendo *random* aunque en este caso es poco significativa la mejora. Particularmente, bajo esta condición, las métricas de selección de instancias de muestreo por incertidumbre (|*p1* - *p0*| ascendente) y de muestreo por incertidumbre basado en entropía (*IG* descendente) han superado a la métrica *random* en el periodo inicial del entrenamiento lo cual es positivo y no había sucedido en los experimentos sin etiquetado de features.

Para realizar los experimentos se desarrolló una serie de módulos dependientes de IEPY encargados de realizar todas las funcionalidades necesarias para un sistema con estas características de aprendizaje activo. Entre estas funcionalidades tenemos la manipulación del conjunto de datos, el cálculo y aplicación de las distintas métricas y la ejecución de la evaluación y las métricas de rendimiento, por nombrar algunas de las más relevantes. Hemos también agregado módulos de preprocesamiento de corpus con menores dependencias y mejores tiempos de ejecución (SpaCy preprocessor). Dentro de las posibilidades de continuación de la base de este trabajo tenemos la utilización de otros corpus y relaciones distintas, la implementación de nuevas métricas de selección de instancias u otra configuración de los features a extraer de las instancias. También podría probarse con un corpus en otro idioma, como el español.

Bibliografía

Burr Settles. *Closing the Loop: Fast, Interactive Semi-Supervised Annotation With Queries on Features and Instances*. Empirical Methods in Natural Language Processing (EMNLP), pages 1467-1478, 2011.

Sunita Sarawagi. *Information Extraction*. Foundations and Trends in Databases: Vol. 1: No. 3, pp 261-377, 2008.

Sachin Pawar, Girish K. Palshikar and Pushpak Bhattacharyya. *Relation Extraction: A Survey*. CoRR abs/1712.05191. 2017.

Nguyen Bach, Sameer Badaskar. *A Review of Relation Extraction*. Literature review for Language and Statistics II, 2007.

Burr Settles. *Active learning literature survey*. Computer Sciences Technical Report 1648, University of Wisconsin-Madison, 2009.

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, pp. 2825-2830, 2011.

Sokolova, M., and G. Lapalme. *A systematic analysis of performance measures for classification tasks*. Information Processing and Management, 45, pp. 427-437, 2009.

Hossin, M. and Sulaiman, M.N. *A Review on Evaluation Metrics for Data Classification Evaluations*. International Journal of Data Mining & Knowledge Management Process (IJDKP) Vol.5, No.2, 2015.

J. Baldridge and M. Osborne. *Active learning and the total cost of annotation*. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 9–16. ACL Press, 2004.

J.D. Rennie, L. Shih, J. Teevan, and D. Karger. *Tackling the poor assumptions of naive bayes text classifiers*. In Proceedings of the International Conference on Machine Learning (ICML), pages 285–295, 2003.

David D. Lewis, Jason Catlett. *Heterogeneous Uncertainty Sampling for Supervised Learning*. In Proceedings of the Eleventh International Conference on Machine

Learning, pages 148–156, 1994.

Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

F. Sebastiani. *Machine learning in automated text categorization*. ACM Computing Surveys, 34(1):1–47, 2002.

Manning, Christopher D., Surdeanu, Mihai, Bauer, John, Finkel, Jenny, Bethard, Steven J., and McClosky, David. *The Stanford CoreNLP Natural Language Processing Toolkit*. In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55-60, 2014.

Oliphant T. E.. *A guide to NumPy*. Vol. 1. Trelgol Publishing USA, 2006.

J. D. Hunter. *Matplotlib: A 2D Graphics Environment*. Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.