Clasificación de anuncios en Revolico

Ernesto Luis Estevanell Valladares, Rodrigo Sua

Facultad de Matemática y Computación,
Universidad de la Habana, Habana, Cuba
{e.estevanell,r.garcia}@estudiantes.matcom.uh.cu

www.github.com/EEstevanell/Revolico-Category-Estimator

Resumen The abstract should summarize the contents of the paper and should contain at least 70 and at most 150 words. It should be written using the *abstract* environment.

Keywords: Clasificador, Machine Learning, Procesamiento de texto, Aprendizaje supervisado

1. Introducción

```
program Inflation (Output)
  {Assuming annual inflation rates of 7%, 8%, and 10%,...
   years};
   const
     MaxYears = 10;
   var
     Year: 0..MaxYears;
     Factor1, Factor2, Factor3: Real;
   begin
     Year := 0;
     Factor1 := 1.0; Factor2 := 1.0; Factor3 := 1.0;
     WriteLn('Year  7% 8% 10%'); WriteLn;
     repeat
       Year := Year + 1;
       Factor1 := Factor1 * 1.07;
       Factor2 := Factor2 * 1.08;
       Factor3 := Factor3 * 1.10;
       WriteLn(Year:5,Factor1:7:3,Factor2:7:3,Factor3:7:3)
     until Year = MaxYears
end.
```

2. Estado del Arte

3. Propuesta de Solución

Se propone una aplicación web que permite obtener, para un texto determinado, cuál sería su clasificación como anuncio en Revolico[1].

Dicha propuesta se compone de tres módulos principales:

- *Web app* (3.1)
- *Crawler* (3.2)
- *Engine* (3.3)

3.1. Web App

Se implementó un servicio Web utilizando **flask**. La interfaz de usuario propuesta permite la interacción del usuario con las herramientas presentes en los otros módulos.

Se ofrecen las siguientes funcionalidades:

- Iniciar un *crawler* para la recolección de anuncios de Revolico.
- Entrenar los modelos propuestos en *Engine*(3.3) con un corpus determinado.
- Obtener una clasificación para un texto determinado por cada estimador.

3.2. Crawler

Utilizando ***Scrapy***[2] se implementó una araña que permitiese la extracción de anuncios de Revolico con su correspondiente clasificación con el objetivo de construir un ***corpus***. Dicho ***dataset*** fue utilizado por el *Engine*(3.3) para entrenar el estimador implementado.

Se identificaron cinco categorías diferentes:

- Compra-Venta
- Empleo
- Computadoras
- Viviendas
- Autos

Se utilizó la versión *offline* de Revolico dadas las condiciones de conexión. Se pudieron observar algunas diferencias con el sitio real en cuanto a la estructura de las páginas pero, contando con la gran distribución de esta versión gracias al "paquete", se decidió seguir utilizándola.

Se implementó una política de ordenación **LIFO** y se estableció como **URL** semilla la raíz de Revolico. Una vez satisfecha la cantidad de anuncios de una categoría determinada, no se acceden a links de dicha categoría. De esta manera garantizamos una exploración balanceada por cada categoría, facilitando la recolección de anuncios.

[1] Popular sitio de compra y venta cubana

[2] *framework de python para crawlers*

3.3. Engine

Se implementó un *pipeline* de clasificación que permite variar sus componentes para lograr un rendimiento adecuado.

El *pipeline* propuesto consiste de las siguientes etapas:

- preprocesamiento
  1. tokenización
  2. *stemming*
  3. vectorización
- clasificación

3.4. Preprocesamiento

El preprocesamiento de los datos es una etapa fundamental de un *pipeline* de *machine learning*. Específicamente, al trabajar con texto, se necesitan realizar una serie de pasos para convertir los datos en una entrada válida para un método del dominio tratado.

Entre las componentes tratadas en nuestra propuesta se encuentran:

- tokenizador
- *stemmer*
- vectorizador

En la práctica, una selección incorrecta de un método en algún paso puede llevar a un rendimiento bajo del *pipeline* de clasificación. Además, existe una amplia gama de posibles técnicas para las funcionalidades anteriores, lo que hace de este un problema a lidiar.

Se presenta un estimador general que permite seleccionar, a modo de parámetro, un tokenizador o vectorizador a utilizar, dando facilidad para realizar un proceso de búsqueda sobre el espacio de decisiones. Se utilizó por defecto la función ***word_tokenize***[3] (**nltk**) como tokenizador mientras que ***CountVectorizer***[4] se seleccionó como vectorizador por defecto.

Se utilizó la popular clase de *stemming* ***SnowballStemmer*** de **nltk**.

3.5. Clasificación

Se propone el uso de tres clasificadores:

- Naive Bayes []
- Support Vector Machine (Linear kernel) []
- Logistic Regression []

[3] Vectorizador recomendado por los creadores para el caso general.

[4] Vectorizador que consiste en obtener una matriz Término-Documento con los valores tf computados

4. Comparando los clasificadores

Las pruebas se realizaron sobre dos datasets:

- Un dataset compuesto por 4000 anuncios con las 5 categorías contempladas en Revolico (3.2).
- El *dataset Movie Reviews* encontrado en ***nltk_data***.

Comparamos 3 *pipelines* distintos variando únicamente los clasificadores utilizados en cada caso (por simpleza nos referiremos a cada *pipeline* como un estimador). Cada estimador se compuso de la siguiente manera:

1. tokenizer: Función ***word_tokenize*** de **nltk**.
2. stemmer: ***SnowballStemmer*** de **nltk**.
3. vectorizer: ***CountVectorizer*** de **scikit_learn**.

Con cada estimador se realizaron pruebas en ambos *datasets*. Cada prueba consistía en obtener la precisión media mediante validación cruzada de *30-folds* como se muestra en la tabla 1. Las figuras 1, 2, 3 muestran las curvas de aprendizaje (mostrando precisión) de cada estimador con el dataset Revolico utilizando validación cruzada de *5-folds* por cada tamaño utilizado.

Dataset	**Naive Bayes**	**SVM (Linear kernel)**	**Logistic Regression**
Movie Reviews	7.99	78.48	0.07
Revolico	7.99	78.48	0.07

Tabla 1. Índice de precisión de clasificación medio de los estimadores sobre los *datasets Movie Review* y Revolico.

5. Resultados

Se aprecian resultados similares entre los estimadores para ambos *datasets*. Teniendo en cuenta que en el caso de Revolico existen 5 clasificaciones distintas, se han obtenido resultados muy satisfactorios.

Gracias a las curvas 1, 2, 3 podemos observar que los modelos presentan poco nivel de *bias* y un ligero nivel de *variance* que podría ser resuelto mediante la inclusión de más datos.

6. Conclusiones

Hemos logrado implementar 3 estimadores que alcanzan resultados muy satisfactorios en los *datasets* de prueba. Se construyó un crawler capáz de construir un corpus tageado a partir del sitio de anuncios clasificados Revolico. Además, se ofrece un servicio web que proporciona una interfaz de usuario intuitiva para la utilización de las herramientas implementadas.

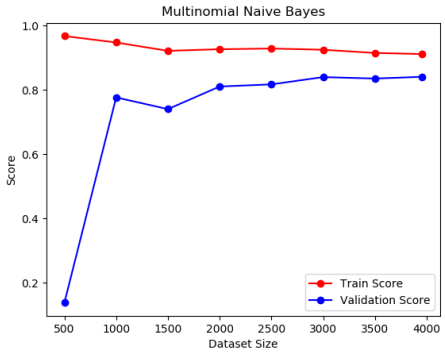

Figura 1. Curva de aprendizaje del estimador con Naive Bayes

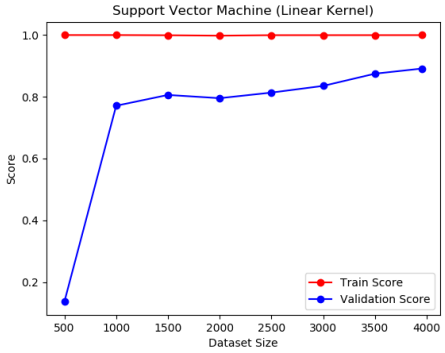

Figura 2. Curva de aprendizaje del estimador con SVM (Linear Kernel)

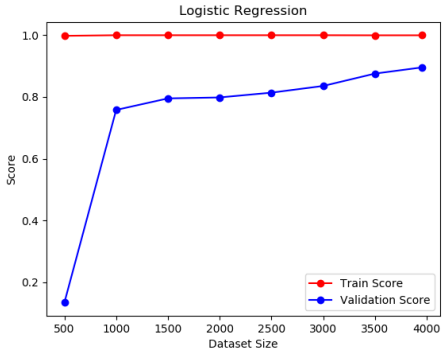

Figura 3. Curva de aprendizaje del estimador con Linear Regression

Referencias