Poster minería de datos

Iker Martinez-Ayo Iñurria

Euskal Herriko Unibertsitatea (December 11, 2022)

Objetivos

- Ddo un Tweet (texto) predecir si padece sentimientos positivos o negativos.
- Investigar:
  1. ¿Cúal es mejor representación?
     1. TF-IDF
     2. Document-embeddings
  2. ¿Cúal es mejor clasificador?
     1. Red neuronal
     2. Regresion logística

Tarea y datos

- Dado un Tweet como input, predecir si tiene sentimientos ligados al suicidio.
- Data source: https://www.kaggle.com/datasets/kazanova/sentiment1
- Cada instancia esta representada por 6 atributos. (Target, id, date, flag, username y text)

Representación del texto

- **Pre-procesamiento** Partiamos de un documento ya pre-procesado, el cual obtuvimos en el proyecto grupal.

Pre	Instancias	Vocabulario	Clases
Raw	1600000	-	3
TF-IDF	10000	45	2
Document embeddings	10000	301	2

Table: Descripción de los datos

- **Representacion** Con Document embeddings decidí usar 300 dimensiones por ser un valor recomendado por muchos usuarios y sencillo. Por otra parte, en TF-IDF use la función TfidfVectorizer,que nos ofrece un punto de calculo simple de calcular con un coste computacional bastante bajo, y el resultado fue un espacio vectorial de 45.

Classifier 1: Regresion logistica

La regresión logística resulta útil para los casos en los que se desea predecir un resultado según los valores de un conjunto de datos. De esta forma, podemos calcular la relación entre un atributo y su clase, prediciendo la probabilidad mayor de pertenencia.

Representación de la clase

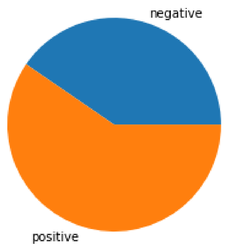


Figure: Proporción de las instancias en el conjunto de datos

Classifier 2: Perceptrón simple

He tenido problemas al generar redes neuronales asi que al final me decidí por usar el perceptrón simple, el cual es la base de las redes neuronales. Las redes neuronales pueden ayudarnos a tomar decisiones inteligentes sin necesidad de interpretacion humana, ya que pueden aprender y modelar las relaciones entre los datos de entrada y salida, para asi obtener una predicción de salida habiendo obtenido un dato de entrada.

WordCloud

Figure: Palabras mas repetidas en los tweets

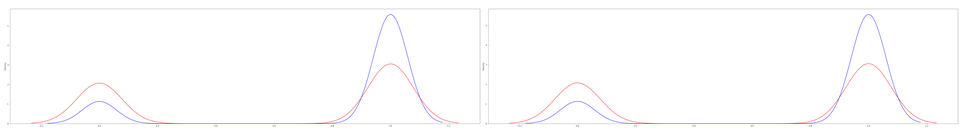
Matriz de las 4 combinaciones

Combinación	Resultado
Regresión logística + TF-IDF	0,6779
Regresión logística + Document embeddings	0,5558
Red neuronal + TF-IDF	0,54
Red neuronal + Document embeddings	0,67

Table: Descripción de los datos

Regresión logística

Graficas del mejor clasificador medio de los dos.



(a) TF-IDF + Logistic regresion

(b) TF-IDF + Document embeddings

Experimental Results: best approach results

Resultados finales obtenidos con la mejor combinación.

```
model = Perceptron(max_iter=1000, eta0=1.0)
model.fit(x2_train, y2_train)
```

```
Perceptron()
```

```
y2_pred = model.predict(x2_test)
```

```
classificationReport = classification_report(y2_test, y2_pred)
```

```
print(classificationReport)
```

```
              precision    recall  f1-score   support

           0       0.59      0.53      0.56       786
           1       0.71      0.77      0.74      1214

    accuracy                           0.67      2000
   macro avg       0.65      0.65      0.65      2000
weighted avg       0.67      0.67      0.67      2000
```

Figure: Matriz de confusión con medidas de precisión

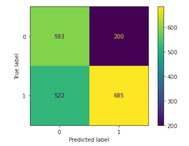


Figure: Mapa de calor

Discusión sobre los resultados

- He obtenido mejores resultados medios con TF-IDF.
- Al principio estimaba que la regresión logística iba a tener mejores resultados que el perceptron simple, y ha sido de este modo.
- He mejorado al random guesser intentando tener un software con bajo coste computacional.

Conclusiones

- He aprendido como implementar perceptrones simples y regresiones logísticas a datasets reales.
- El software mejora las predicciones de un random guesser con un coste computacional no muy alto.
- Esperaba mejores resultados, auque al menos las 4 aproximaciones mejoran al random guesser.