Universidad de Buenos Aires
Facultad De Ingeniería
Año 2023 - 1er Cuatrimestre

Organización de Datos (75.06 /95.58)

Trabajo Practico 2: Críticas cinematográficas

INFORME FINAL
FECHA: 30/06/23

INTEGRANTES:

Gonzalez,Tomás	#108193
<togonzalez@fi.uba.ar>	
Moreno del Ruvo, Valentina	#107948
<vmdr18@gmail.com>	
Pol, Juan Manuel	#108448
<jpol@fi.uba.ar>	

1. Introducción

El objetivo del presente trabajo práctico es identificar dentro de una colección de críticas cinematográficas si la critica es positiva o negativa.

Se construirán diferentes modelos de clasificación, los cuales podrán detectar el *sentimiento* por lo menos en forma binaria.

Se realizo un pre-procesamiento de las críticas para que puedan ser analizadas, este pre-procesamiento consistió en reducir el conjunto de datos (se filtraron las stopwords, caracteres numéricos y otros que puedan confundir el algoritmo) para luego transformar los textos en vectores utilizando el método de *Bag of Words*

- Bayes Naive.
- Random Forest.
- XGBoost.
- Un modelo de red neuronal aplicando Keras y Tensor Flow.
- Un ensamble de al menos 3 modelos (Random Forest, SVM, XGBoost).

Para cada uno de estos modelos se realizo una búsqueda de hiperpartametros para poder optimizar el desempeño en training: el cual se vio reflejado al hacer los submits en Kaggle.

2. Modelos

- Bayes Naive.

  -Para este modelo se utilizo por un lado el algoritmo MultinomialNB. Se realizo una búsqueda exhaustiva de los mejores hiperparametros utilizando la técnica de Grid Search Cross Validation (CV) con un valor de k-fold de 10.

  Se exploraron diferentes valores para 'alpha' el cual controla la suavización de las probabilidades condicionales en el modelos de Bayes Naive. Se obtuvo que el mejor conjunto de hiperparametros encontrado fue 'alpha = 0.05'

  -Por otro lado se utilizo el algoritmo Bernoulli. Utilizando también la técnica de Cross Validation.

  Se exploraron diferentes valores para 'alpha' y para 'binarize', que determina el umbral para binarizar las características en el modelo. Se obtuvo el mejor conjunto de hiperparametros en: 'alpha'= 0.3, 'binarize'= 0.0.

- Random Forest

  Para este modelo se utilizo el algoritmo RandomForestClassifier. Se utilizo la tecnica de CV con un valor de k-fold de 5.

  Se consideraron dos hiperparametros principales: 'n_estimators' y 'max_depth'. Se exploraron los valores de [50, 100, 200] y de [None, 5, 10] respectivamente. Los mejores

hiperparametros encontrados fueron 'n_estimators= 200' y 'max_depth= None'.

- XGBoost

  Para este modelo se utilizo el algoritmo XGBClassifier. Se definieron y ajustaron los hiperparametros siguiendo una búsqueda aleatoria (Randomized Search) para encontrar la combinación óptima.

  Se exploraron varios hiperparametros para controlar la complejidad del modelo: 'gamma', 'learning_rate', 'max_depth', 'n_estimators', 'subsample'.

  La búsqueda aleatoria se llevo a cabo con un total de 5 iteraciones y una validación cruzada de 5 pliegues.Se utiliza ademas el valor 'random_state' 42.

  Los hiperparametros seleccionados finalmente fueron: 'gamma = 0.18', 'learning_rate = 0.31', 'max_depth = 4', 'n_estimators = 107', 'subsample = 0.83'.

- Modelo de red neuronal

  En el caso del modelo de red neuronal se utilizo la biblioteca Keras junto con TensorFlow. La arquitectura para el modelo contiene:

  - Capa de entrada: el tamaño de la capa de entrada se determina por la dimensión de X_train, esto asegura que la red neuronal pueda recibir adecuadamente los datos de entrada.
  - Capa oculta: 100 unidades neuronales y una función de activación 'relu', se utiliza para introducir no linealidad en el modelo y permitir que la red neuronal aprenda representaciones más complejas de los datos.
  - Capa de dropuot: para regularizar el modelo y ayuda a prevenir el 'sobreajuste' al apagar aleatoriamente una fracción de las neuronas durante el entrenamiento.
  - Capa de salida: una sola unidad neuronal y utiliza la función de activación 'sigmoid'. El problema de clasificación se reduce a una clasificación binaria donde se busca predecir la etiqueta de sentimiento positivo o negativo, la función de activación se utiliza para producir una probabilidad de pertenecer a la clase positiva.

  El modelo de red se compila utilizando la función de pérdida 'binary_crossentropy', el optimizador 'adam' y se registra la métrica de precisión ('accuracy').

  Se eligió esta arquitectura porque es una configuración básica para un problema de clasificación binaria. Permite capturar patrones y relaciones entre las características de entrada y generar una salida que se ajuste al problema de clasificación de sentimientos.

- Ensamble de 3 modelos.

  Se utilizan como hiperparametros para los 3 modelos, los valores por defecto o los aclarados. Como son modelos ya previamente analizados a este punto ya se tiene una idea de que parámetros funcionarán mejor. Por ende no se hace una nueva búsqueda de hiperparametros

Se utilizo un ensamble de Stacking Classifier con un meta modelo de regresión logística. Este ensamble combina las predicciones de los tres modelos base (Random Forest, SVM y XGBoost) y utiliza el meta modelo para realizar una clasificación final.

3. Entrenamiento y Resultados

Modelo	Precisión	Recall	$f1_{score}$(Testeo)	$f1_{score}$ (Kaggle)
Bayes Naive (Multinomial)	0.83	0.83	0.83	0.70633
Bayes Naive (Bernoulli)	0.83	0.83	0.83	0.71099
Random Forest	0.84	0.84	0.84	0.71874
XGBoost	0.84	0.83	0.83	0.77049
Red Neuronal	0.88	0.88	0.88	0.742
Ensambles	0.86	0.86	0.86	0.73812