Objetivo del proyecto

Con la creciente exposición que tenemos frente a las noticias online, consideramos que es importante el estudio de los artículos que se publican y qué factores hacen que unos artículos se compartan más que otros, haciendo así que unos sean populares o no.

Con la data que tenemos queremos (explicada más en detalle en el siguiente apartado) predecir si un artículo va a ser *popular* o no. Para determinar si un artículo es *popular* o no sobre el dataset que tenemos, marcamos un threshold de 1400 shares a partir del cual un artículo se considera *popular*. Este threshold se ha determinado así porque hace que el input data quede bastante balanceado, por lo que evitamos que el algoritmo aprenda mejor para la clase mayoritaria que para la minoritaria y cometer muchos errores del tipo 1, falsos positivos.

Descripción del proyecto

Data understanding

Este dataset se ha sacado del machine learning repository: *https://archive.ics.uci.edu/ml/datasets.php.* Consiste en 39797 artículos de la compañía Mashable publicados durante un periodo de 2 años. Cada instancia es un artículo con features obtenidas por distintas técnicas de NLP (Natural Language Processing).
Estas features se pueden clasificar en distintas categorías para entenderlas mejor:

- Número de palabras en distintas secciones.
  Indicando cómo de largo es el título, el contenido, cuantas stopwords hay, etc.
- Links y referencias.
  Indicando cuantos links hay y además, como de relevantes son los artículos citados.
- Digital media: Cuántas imágenes y videos se incluyen en el artículo
- Cuándo se publicó: Entre semana, fin de semana, qué día, etc.
- Keywords: Análisis de las palabras clave que aparecen y cómo se relevantes son, usando las shares que ha tenido en otros artículos.
- Sentiment and subjectivity analysis. Esta parte se engloban ratios de positividad, negatividad, *polarity*, *sentiment*, etc. Además, con Latent Dirichlet Allocation (LDA) se han obtenido los cinco topics más relevantes.

(Ver tabla en la imagen 1 del anexo.)

Metodología usada para la clasificación

Para hacer la predicción hemos usado cinco modelos de clasificación:

- Decision Tree
- Random Forest
- Support Vector Machines (LinearSVC)
- K-Nearest Neighbor (KNN)
- Logistic Regression

Por cada modelo que entrenamos vamos a usar GridSearchCV para encontrar los mejores hiper-parámetros dentro de un rango marcado por nosotras manualmente post-análisis del modelo y de los hiper-parámetros predeterminados por defecto. Además vamos a usar las

mismas métricas para todos para que los resultados sean comparables y poder determinar cuál es el que mejor resultado ha obtenido.

GridSearchCV

A la hora entrenar un modelo, uno de los puntos cruciales es la selección de hiper-parámetros. Desde un punto de vista analítico, hemos hecho un trabajo de entendimiento de todos los hiper-parámetros que puede tener cada uno de los modelos y selección de posibles entradas óptimas que resulten en un buen entrenamiento.
Sin embargo, no tiene sentido ir probando todos estos valores preseleccionados y todas las combinaciones posibles de forma manual ya que implica un tiempo innecesario; es por ello que usamos Grid Search Cross Validation.

Además, para no obtener directamente el valor óptimo que nos devuelve la cross validation, hemos hecho una gráfica que nos muestra como varía la accuracy por cada valor del hiper-parámetro probado. De esta forma podemos ver si hay algún otro valor que obtiene una accuracy muy similar aunque un poco más baja (por eso no ha sido retornada por el grid search) pero que hace el modelo más simple y más generalizable.

Métricas utilizadas

Para poder comparar la performance de los cinco modelos usamos en todos los casos las mismas métricas (las típicas usadas para clasificación) y marcamos cuales son las más importantes para nosotras de optimizar.

- *Precisión*. Nos indica de todos los que hemos cogido, cuántos realmente son populares.
- *Recall*: sensibilidad. Ya que nos centramos en coger tantos populares como sea posible de todos los que hay, aunque nos equivoquemos en algunos artículos diciendo que serán populares y en realidad no.
- *Accuracy*: optimizando este valor seleccionamos los hiper-parámetros en el GridSearch Cross-Validation
- *AUC*: área bajo la curva ROC (representación de la proporción de verdaderos positivos, frente a los falsos. Cuanto más se acerque este valor a 1, mejor es el modelo.

VotingClassifier

Una vez hemos tenido todos los modelos entrenados, hemos decidido probar a mejorar los resultados por medio del *VotingClassifier*.
Lo que hacemos con esto es crear un nuevo modelo de clasificación que lo que hace es agregar las predicciones de cada uno de los otros clasificadores y predecir la clase que obtiene la mayor votación. Esto es lo que sucede cuando usamos voting = *hard*.
En cambio, también hay otra posibilidad y es poner voting = soft. En este caso lo que sucede es que no funciona por voto simple, sino que predice la clase con mayor probabilidad usando la media de todos los otros clasificadores. Para usar esto tenemos que asegurarnos que todos los clasificadores tienen el método de predict_proba()

Usando esto podemos obtener mejor *accuracy* incluso que el mejor clasificador involucrado en el *ensembling*. Si paremos de un 51% (practicamente clasificación aleatoria) podemos obtener hasta un 75% de accuracy (en el mejor de los casos). Sin embargo, como se puede ver en el siguiente apartado, este no ha sido nuestro caso.

Resultados del proyecto

Model	Accuracy (test)	Precision	Recall	AUC
Random Forest	0.672	0.68	0.74	0.73
Decision Tree	0.644	0.66	0.68	0.69
LinearSVC	0.605	0.62	0.68	-
Logistic Regression	0.604	0.62	0.68	0.65
Knn	0.597	0.62	0.66	0.626

Tras hacer el Voting Classifier explicado previamente obtenemos:

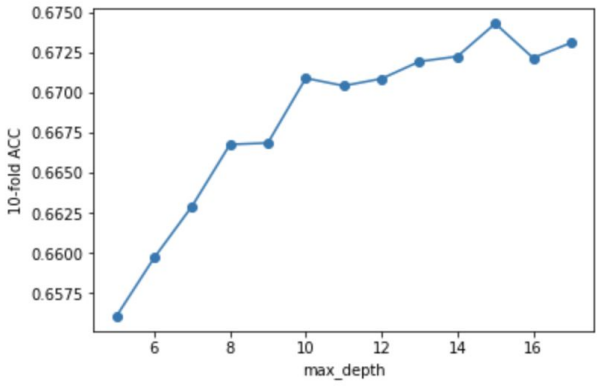
- Para voting = soft → accuracy = 0.656
- Para voting = hard → accuracy = 0.629

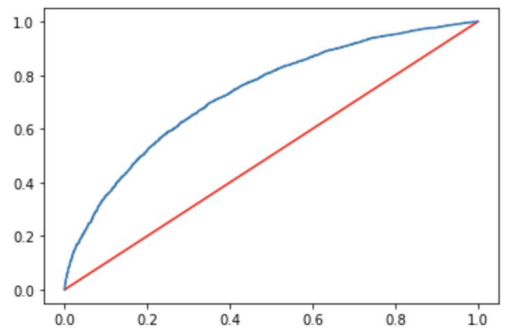
Analizando las métricas de clasificación como hemos explicado en la sección previa, vemos que claramente el modelo que mejor ha entrenado sobre la data que tenemos es RandomForest; incluso mejor que el voting classifier, lo cual nos ha sorprendido.

Los hiper-parámetros que se han modificado conforme a lo que se incluye por default son:

- n_estimators = 200
- max_features = 'sqrt'
- max_depth = 15

Este últimos hiper-parámetro ha sido seleccionado por GridSearchCV usando 10 folds y dando un param_grid de un rango entre 5 y 18. Por cada valor que este algoritmo de model_selection ha usado para max_depth podemos ver en la siguiente gráfica a la izquierda que accuracy obtenemos; y comprobamos que efectivamente la mejor accuracy con diferencia es en el valor 15.
En la gráfica de la derecha observamos la ROC curve con respecto a la línea base que muestra el hipotético caso de que la decisión de si el artículo es *popular* o *unpopular* fuera random (prob = 0.5 para cada clase), es cuál sería el peor escenario. Con el valor de AUC bastante diferenciado de 0.5, podemos ver que el modelo tiene una buena performance.





The AUC score is: 0.7337269752113043

Visualización

La función principal de nuestra visualización en Tableau es que un usuario final (escritor de un artículo o cualquiera de las personas que revisan el artículo antes de su publicación) pueda mirar este dashboard y ver las características principales que puede moldear del artículo (por ejemplo, subjetividad) y cómo afectan a si el artículo va a ser *popular* o no. También puede ver que cómo afecta el día de publicación y así poder evaluar si quiere publicarlo en un día entre semana o en fin de semana o si no es relevante.

Para decidir qué features incluir en el dashboard nos hemos inspirado en el "feature importance" obtenido por el Random Forest, ya que es el modelo más accurate y nos dice cómo de relevantes son las variables para determinar si el artículo será *popular* o no.

El dashboard puede visualizarse a través del siguiente enlace: https://public.tableau.com/views/Articlespopularityanalysis/Dashboard1?:language=es&:display_count=y&publish=yes&:origin=viz_share_link y a continuación procedemos a su análisis.

En primer lugar, podemos ver que en general los artículos categorizados como artículos de tecnología son más populares y que los de social media tienden a ser más populares que no populares. Por el contrario vemos que los artículos de entretenimiento tienden a ser no populares, lo que significa que tienden a tener menos de 1400 shares.

Si nos centramos en **cuándo** debe publicarse el artículo para tener más shares, hemos visto que no es muy relevante qué día de la semana se publica o qué día de fin de semana se publica. Sin embargo, sí que es relevante que en el fin de semana es más probable que el artículo sea *popular*, ya que la diferencia es bastante significativa visualmente; o lo que es lo mismo, el fin de semana parece ser más difícil que tu artículo sea "*no popular*".

Otro aspecto que el usuario puede analizar es cómo influye en la popularidad la **subjetividad** de su artículo. ¿Es mejor que sea un artículo objetivo y de opinión? Pues conforme a la distribución que se ve en este dashboard, parece ser que aquellos con valor de subjectivity entre 0 y 0.4, hay mayor tendencia a que el artículo sea *unpopular*. Si la subjetividad es intermedia (entre 0.4 y 0.6); es decir, deja entrever la opinión del autor pero no hace una crítica muy explícita, entonces claramente se aprecia que al lector le gusta más y lo comparte más, por lo que hay más popularidad.

Por último el usuario puede analizar la **calidad de las referencias** que tiene en su artículo. Podemos ver en la cola de la distribución (a partir de 10k de self_reference_avg_shasres) que cuanto mayor es la calidad de tus referencias, más raro es que tu artículo sea *unpopular*. Por calidad en tus referencias entendemos que los artículos a los que citas (propios de Mashable) sean buenos artículos, es decir, con muchas shares. Si en tu artículo solo referencias otros que no son populares, la probabilidad de que el tuyo sea *popular* o no parece estar bastante más igualada.

Este análisis se ha hecho con respecto a las tendencias generales; es decir, teniendo en cuenta todos los artículos de todos los *channels*. Sin embargo, el usuario final que usará este dashboard normalmente no puede cambiar el *channel* de su artículo. Es por ello, que puede hacer click sobre el *channel* que quiere analizar en el gráfico de burbujas y todo el dashboard se adaptará usando la tendencia de los artículos específicos.

ANEXO

IMAGEN 1. FEATURES

En esta tabla podemos ver una clasificación más detallada de las variables que tenemos en las categorías explicadas en la sección de “Data Understanding”.

WORDS	LINKS	KEYWORDS	NATURAL LANGUAGE PROCESSING	
number of words in title	number of links	number of keywords	closeness to top 5 LDA topic	positive words rate among non-neutral words
number words in article	number of mashable articles links	worst keyword (min/avg/max) shares	title subjectivity	negative words rate among non-neutral words
average word length	min, avg and max number of shares of mashable links	average keyword (min/avg/max) shares	article text subjectivity	polarity of positive words (min/avg/max)
rate of non-stop words	**TIME**	best keyword (min/avg/max) shares	title sentiment polarity	polarity of negative words (min/avg/max)
rate of unique words	day of the week	article category	rate of positive and negative words	article text polarity
rate of unique non-stop words	if published in weekend			

IMAGEN 2. IMPORTANCE FEATURES

Esta es la gráfica donde se pueden ver las 25 variables con más relevancia en el training del Random Forest. Estas son a las que se hace referencia en el apartado de “visualización”