Universidad de los Andes
Ingeniería de Sistemas y Computación
ISIS 3301 Inteligencia de Negocios

Proyecto 1
Machine learning: Analítica de Textos

Integrantes:

- Juan Micolás Mononaños – 201911676 - Random forest
- Juan Pablo Sarmiento – 201914487 - Regresión logística
- Juan José Ochoa – 2013913552 - Naive Bayes

Tabla de contenido

1 Comprensión del Negocio y Enfoque Analítico

Oportunidad / Problema del negocio	El problema que el negocio está interesado en resolver es determinar el éxito y la aceptación que va a tener un alimento en base a los comentarios que este recibe.
Descripción del requerimiento desde el punto de vista de machine learning	El negocio requiere de modelos de machine learning capaces de predecir la aceptación de un alimento en base a los comentarios que este recibe, entendiendo por buena aceptación tener buenas calificaciones por parte de los usuarios. Es por esto por lo que vamos a hacer uso de algoritmos NLP (Natural Language Processing) con el fin de predecir los sentimientos que se generan en los clientes tomando como base los textos de las reviews que ellos escriben. Para que un modelo de su respectivo algoritmo se considere que satisface estos requerimientos debe cumplir que:

Detalles de la actividad de minería de datos:

Tarea	Técnica	Algoritmo e hiperparámetros
Preprocesado de datos	Filtración de caracteres html	Regular expresions (substring)
Preprocesado de datos	**Filtración de caracteres de puntuación**	**Regular expresions (substring)**
Preprocesado de datos	**Stemming de las palabras**	**NLTK Snowball Stemmer**
Preprocesado de datos	Eliminación de palabras vacías	NLTK stopwords
Modelamiento	Regresión	Logistic Regression

Modelamiento	Clasificación	Random Forest (criterion=gini, max_depth=none)
Modelamiento	Predicción	Naive Bayes (alpha = 0.01)

2 Comprensión y preparación de los datos

En la Figura 1 se puede apreciar las columnas más importantes de todo nuestro dataset el cual consta de 568454 filas, dichas variables son ‘Score’ y ‘Text’, ya que con base a estas se va a realizar la predicción del éxito de un producto alimenticio según los comentarios realizados.

Score	Time	Summary	Text
2	1349222400	Shorted amount	I received two boxes of the Emeril's coffee an...
4	1199232000	Best coffee for the Keurig	I like strong coffee and I have found this is ...
5	1283644800	Dingo Stix	My dog loves these stix, she takes it and then...
1	1278115200	Better do it from scratch	As much as I LOVE NordicWare pans, I was compl...
5	1303948800	Excellent Tea!	An excellent tea! This is an elegant green tea...
4	1339113600	Sojos original	I didn't mean to order a 40-pound bag. But I d...
4	1301529600	Healthy and delicious, wish they were cheap.	For the cost this is not a good snack but if y...
5	1233792000	Virgin Coconut Oil	I buy this regularly at Whole Foods for about ...
5	1308182400	the best!!!!!!!	i have had a really hard time finding these ol...
5	1203120000	Love it....	This stuff is great. Yeah, it still has calor...

Figura 1. Columnas más importantes

Luego de realizar un conteo de todos los puntajes ingresados a los productos, se pudo destacar que los datos no están balanceados, ya que la gran mayoría tienen un puntaje alto, lo cual va a impactar más adelante los resultados de los modelos implementados.

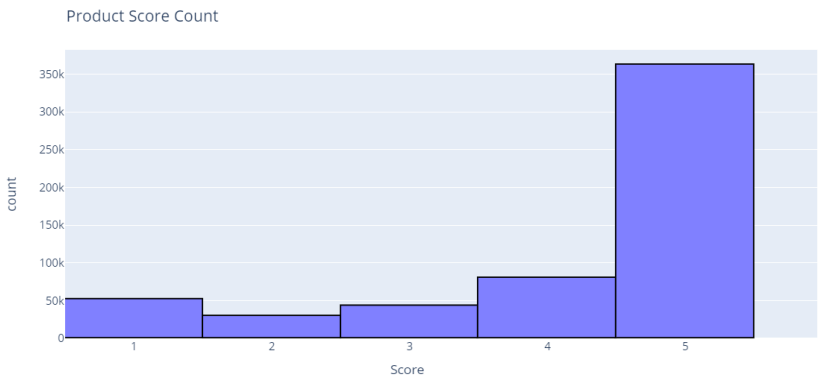

Figura 2. Conteo de puntaje en reviews

Por otro lado, se tomó la decisión de eliminar las reviews de 3 estrellas para centrarse en dividir las reviews en positivas (4 y 5 de score) y negativas (1 y 2 de score). Ya que esto facilitaría la creación de una nueva variable binaria para trabajar con los modelos, dicha columna se definió como ‘rating’, donde un valor de “0” representa una review negativa y un valor de “1” representa una review positiva. En la Figura 3 se puede apreciar la definición y distribución de la nueva variable “rating”.

```
data_clean['rating'] = data_clean['Score'].apply(lambda rating : 1 if rating > 3 else 0)
positive = data_clean[data_clean['rating'] == 1]
negative = data_clean[data_clean['rating'] == 0]
```

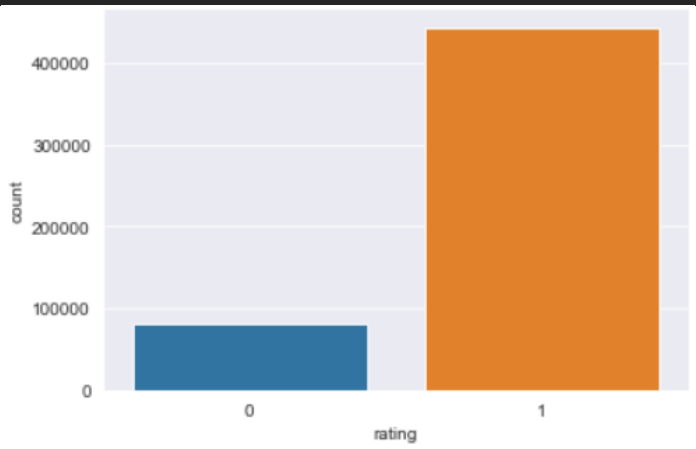

Figura 3. Definición y distribución de la variable rating

En la Figura 4 está el código utilizado para el preprocesamiento del texto de las reviews. Dentro de este proceso se limpió el texto de caracteres de puntuación y caracteres ajenos al abecedario inglés, se eliminaron palabras ‘vacías’ como artículos o pronombres junto con contracciones del inglés, se convirtieron todos los caracteres a minúsculas y finalmente se realizó el proceso de stemming, que consiste en reducir las palabras a su raíz o ‘stem’. Todos estos pasos son vitales para la analítica de textos y van a mejorar considerablemente el desempeño y métricas de las tareas realizadas por los modelos de machine learning utilizados.

```
sno = nltk.stem.SnowballStemmer('english')
stops = set(stopwords.words("english"))
def cleanhtml(sentence):
    cleanr = re.compile('<.*?>')
    cleantext = re.sub(cleanr, ' ', sentence)
    return cleantext

def cleanpunc(sentence):
    cleaned = re.sub(r'[?|!|\'|"|#]',r'',sentence)
    cleaned = re.sub(r'[.|,|)|(|\|/]',r' ',cleaned)
    return  cleaned

def final_sentence(text):

    text = text.split()
    text = [cleanhtml(x) for x in text]
    text = [cleanpunc(x) for x in text]

    def test(word):
        if word.isalpha() and len(word) > 2 and word.lower() not in stops:
            s=(sno.stem(word.lower()))
            return s
        else:
            pass

    text = [test(x) for x in text if test(x)]


    return ' '.join(text)
```

Figura 4. Preprocesamiento del texto

Score	Time	Summary	Text	rating	CleanedText
5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...	1	bought sever vital can dog food product found ...
1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...	0	product arriv label jumbo salt peanut actual s...
4	1219017600	"Delight" says it all	This is a confection that has been around a fe...	1	confect around pillowi citrus gelatin nut case...
2	1307923200	Cough Medicine	If you are looking for the secret ingredient i...	0	look secret ingredi robitussin believ found go...
5	1350777600	Great taffy	Great taffy at a great price. There was a wid...	1	great taffi great wide assort yummi deliveri t...

Figura 5. Columnas CleanedText y Rating

Finalmente, después de todo el preprocesamiento realizado a los datos de todo el dataframe queda como resultado las dos variables que van a ser utilizadas para las tareas de los modelos de machine learning. La variable ‘rating’ la cual indica si la review es negativa o positiva y la variable ‘CleanedText’ la cual es la colección de palabras ya procesadas del texto de cada review. Esto se puede visualizar en la Figura 5.

3 Modelado y Evaluación

3.1 Regresión Logística

A pesar de su nombre no es un algoritmo para aplicar en problemas de regresión en los que se busca un valor continuo, sino que es un método para problemas de clasificación, en los que se obtienen un valor binario entre 0 y 1, esto nos favorece debido a la transformación del puntaje en la nueva columna binaria “Rating”. A partir de matrices de pesos aleatorias ajustadas las cuales representan el pool de palabras con todas sus características y un vector de tokens los cuales se multiplican entre si basado en un punto dado para obtener un valor escalar, el cual es luego pasado a una función sigma en la cual se produce un valor binario para clasificar dicho punto.

```
data_lr = data_clean.copy()
X_train, X_test, y_train, y_test = train_test_split(data_lr['CleanedText'], data_lr['rating'], test_size = 0.2)
cv = CountVectorizer(token_pattern=r'\b\w+\b')
X_train =  cv.fit_transform(X_train)
X_test = cv.transform(X_test)
```

```
lr = LogisticRegression()
lr.fit(X_train,y_train)
```

```
LogisticRegression()
```

Figura 6. Implementación código del modelo

El modelo de Regresión Logística tiene una precisión del 64% para reviews negativas y un 97% para reviews positivas, el accuracy total del modelo es del 92%.

```
print('Classification report:\n\n', classification_report(predictions,y_test))
```

```
Classification report:

              precision    recall  f1-score   support

           0       0.64      0.81      0.71     13076
           1       0.97      0.93      0.95     92082

    accuracy                           0.92    105158
   macro avg       0.81      0.87      0.83    105158
weighted avg       0.93      0.92      0.92    105158
```

```
skplt.plot_confusion_matrix(y_test,predictions)
```

```
<AxesSubplot:title={'center':'Confusion Matrix'}, xlabel='Predicted label', ylabel='True label'>
```

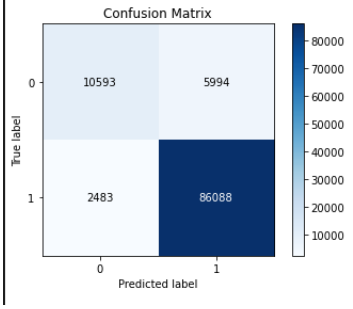

Figura 7. Resultados de Regresión Logística

3.2 Random Forest

Un algoritmo de Random Forest consta de muchos árboles de decisión. El "bosque" generado por el algoritmo de Random Forest se entrena mediante agrupamiento o agregación de arranque. El ensacado es un meta-algoritmo de conjunto que mejora la precisión de los algoritmos de aprendizaje automático. El algoritmo establece el resultado en función de las predicciones de los árboles de decisión. Predice tomando el promedio o la media de la salida de varios árboles. El aumento del número de árboles aumenta la precisión del resultado.

```
X = data_rf['CleanedText']
y = data_rf['rating']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

```
vectorizer = TfidfVectorizer()
vectorizer_tfidf = vectorizer.fit(X_train)
X_train = vectorizer_tfidf.fit_transform(X_train)
X_test = vectorizer.transform(X_test)
```

```
clf = RandomForestClassifier()
clf.fit(X_train, y_train)
preds = clf.predict(X_test)
```

Figura 8. Implementación código del modelo

El modelo de Random Forest tiene una precisión del 50% para reviews negativas y un 100% para reviews positivas, el accuracy total del modelo es del 92%. Se presume que con una muestra más grande de reviews con 2 estrellas o menos en el dataset, la precisión en las predicciones de las reviews negativas subiría. Otro factor para considerar es que las reviews positivas tienden a ser mucho más genéricas, y la gran mayoría utiliza los mismos calificativos, mientras que las reviews negativas tienen descripciones mucho más detalladas dificultando el proceso de clasificación a partir de la analítica de textos.

```
print('Classification report:\n\n', classification_report(preds,y_test))
```

```
Classification report:

              precision    recall  f1-score   support

           0       0.50      0.98      0.66      8326
           1       1.00      0.92      0.96     96832

    accuracy                           0.92    105158
   macro avg       0.75      0.95      0.81    105158
weighted avg       0.96      0.92      0.93    105158
```

```
skplt.plot_confusion_matrix(y_test,preds)
```

```
<AxesSubplot:title={'center':'Confusion Matrix'}, xlabel='Predicted label', ylabel='True label'>
```

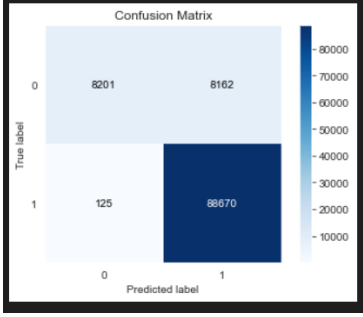

Figura 7. Resultados de Random Forest

3.3 Naive Bayes

En estos modelos se asume que las variables predictoras son independientes entre sí. Son una manera sencilla de construir modelos de analítica de textos con un comportamiento muy bueno debido a su simplicidad. Lo consiguen proporcionando una forma de calcular la probabilidad condicional de que ocurra un evento A, dadas probabilidades de eventos anteriores. Para este algoritmo se convierten los datos en una tabla de frecuencias y se utiliza la ecuación Naive Bayes para calcular la probabilidad de la ocurrencia de los eventos, la predicción seleccionada es aquella con la mayor probabilidad. Este algoritmo es apropiado para problemas de clasificación binarios como el descrito por el negocio.

```
def Text_Into_Vector(model,data):
    model_vect = model(ngram_range=(1,2)) #in scikit-learn
    final_array = model_vect.fit_transform(data.values)

    return model_vect, final_array

def Split_data(x_vec, y_vec):
    X_train, X_test, Y_train, Y_test = train_test_split(x_vec, y_vec, test_size=.33, random_state=0)
    X_tr, X_cv, Y_tr, Y_cv = train_test_split(X_train, Y_train, test_size=.33, random_state=0)
    return X_tr, X_cv, X_test, Y_tr, Y_test, Y_cv, X_train, Y_train

def Normalization(train, cv, test):
    train=preprocessing.normalize(train)
    cv=preprocessing.normalize(cv)
    test=preprocessing.normalize(test)

    return train, cv, test
```

Figura 9. Conversión de los datos a vectores

```
def Multinomial_NB(X_train,X_cv,Y_train,Y_cv):
    best_alpha=0
    max_roc_auc=-1
    pred_cv = []
    pred_train = []
    alpha=[10000,5000,1000,500,100,50,10,5,1,0.5,0.1,0.05,0.01,0.005,0.001,0.0005,0.0001,0.00005,0.00001]

    for i in alpha:
        mulbnb = MultinomialNB(alpha=i)
        mulbnb.fit(X_train,Y_train)
        probs = mulbnb.predict_proba(X_cv)[:,1]
        prob = mulbnb.predict_proba(X_train)[:,1]

        auc_score_cv = roc_auc_score(Y_cv,probs)
        auc_score_train = roc_auc_score(Y_train,prob)

        pred_cv.append(auc_score_cv)
        pred_train.append(auc_score_train)

        if(max_roc_auc<auc_score_cv):
            max_roc_auc=auc_score_cv
            best_alpha=i
    return best_alpha

def Testing_model(X_train,Y_train,X_test,Y_test,best_alpha):

    bnb = MultinomialNB(alpha = best_alpha, fit_prior=True, class_prior=None)
    bnb.fit(X_train,Y_train)
    probs = bnb.predict_proba(X_test)[:,1]

    roc_auc = roc_auc_score(Y_test,probs)

    prediction=bnb.predict(X_test)
    skplt.plot_confusion_matrix(Y_test,prediction)

    print("macro f1 score for data :",metrics.f1_score(Y_test, prediction, average = 'macro'))
    print("micro f1 score for data:",metrics.f1_score(Y_test, prediction, average = 'micro'))
    print("hamming loss for data:",metrics.hamming_loss(Y_test,prediction))
    print("\n")
    print("Precision recall report for data:\n",metrics.classification_report(Y_test, prediction))
    print("\n")

    return bnb,roc_auc
```

Figura 10. Aplicación de la ecuación de Naive Bayes

El modelo de Naive Bayes tiene una precisión del 84% para reviews negativas y un 94% para reviews positivas, el accuracy total del modelo es del 93% convirtiendo en el mejor modelo de los 3 utilizados.

Precision recall report for data:

	precision	recall	f1-score	support
0	0.84	0.67	0.74	27017
1	0.94	0.98	0.96	146494
accuracy			0.93	173511
macro avg	0.89	0.82	0.85	173511
weighted avg	0.92	0.93	0.92	173511

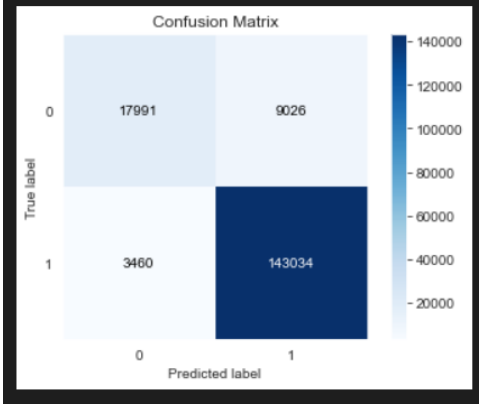

Figura 11. Resultados de Naive Bayes

4 Resultados

Para comenzar, en la fase de perfilamiento de los datos realizamos un wordcloud para ver las palabras más recurrentes del dataset, excluyendo obviamente las palabras vacías y signos de puntuación. Esto con el objetivo de darnos una idea del impacto que puede llegar a tener cada una al momento de aplicar cada uno de los modelos propuestos. El resultado se puede apreciar en la figura 12.

Figura 12. Resultados Generales

Ahora bien, en la fase de preparación de datos, como ya se había mencionado anteriormente, se realiza una división en reviews positivas y negativas basado en su respectivo score. Se realizó un wordcloud para cada conjunto con el propósito de visualizar las palabras más recurrentes para tanto reviews positivas como negativas (figura 13 y figura 14 respectivamente) y así tener un panorama sobre el impacto de las palabras al momento de predecir el sentimiento de su respectiva review cuando se apliquen los modelos propuestos.

Figura 13. Resultados Positivos

Figura 14. Resultados Negativos

El mejor modelo de los 3 seleccionados es Naive Bayes, esto se debe a la forma en cómo funciona; donde divide el dataset en dos partes (en nuestro caso reviews positivas y negativas) y calcula la probabilidad de ver las palabras positivas en esos reviews, y lo mismo con las palabras negativas. Cuando le llegue un String determina la probabilidad de ver cada palabra de la cadena en uno de los dos grupos para así al final clasificar el nuevo texto como review mala o buena. Por otro lado, se presume que con una muestra más grande de reviews con 2 estrellas o menos en el dataset, la precisión en las predicciones de las reviews negativas subiría en los modelos de Regresión Lineal y Random Forest. Sin embargo, también es necesario tener en cuenta que varía en gran medida las palabras utilizadas para referirse a una review negativa en comparación a una positiva. Otro factor para considerar es que las reviews positivas tienden a ser mucho más genéricas, y la gran mayoría utiliza los mismos calificativos, mientras que las reviews negativas tienen descripciones mucho más detalladas dificultando el proceso de clasificación a partir de la analítica de textos. Debido a esto, tanto Regresión Lineal y Random Forest son muy eficientes para predecir cuándo un comentario va a representar una buena acogida para el producto, pero, puede llegar a tener un poco más de dificultades prediciendo cuando va a ser una mala recepción.

5 Trabajo en Equipo

Cada uno de los miembros del equipo se encargó de la implementación de uno de los diferentes modelos y algoritmos de machine learning. La persona que asumió el rol de líder del proyecto fue Nicolás Bolaños, el líder de negocio es Juan José Ochoa y el líder de datos y analítica es Juan Pablo Sarmiento.

Los mayores retos enfrentados en el desarrollo del proyecto fue en primer lugar la inexperiencia con el uso e implementación de los algoritmos NLP, en segundo lugar, debido al gran volumen de los datos fue difícil la implementación de un preprocesamiento óptimo de los datos y por la naturaleza del problema que se quería resolver era necesario hacer varias modificaciones a los datos para que los modelos funcionaran de forma óptima.

Adicionalmente los elevados tiempos de ejecución de los modelos hicieron que fuera muy costoso implementar correcciones y hacer comparaciones entre resultados para su optimización. Cada estudiante se encargó de la implementación de su algoritmo correspondiente y cada uno destine mayor tiempo a ciertas tareas según su rol asignado. Nicolás estuvo más pendiente al final en la logística de la totalidad del proyecto, sus entregables y que cada uno cumpliera con sus funciones. Juan José estuvo más activo en las partes que era necesario justificar al negocio los resultados y problemáticas. Finalmente, Juan Pablo aportó más a la parte de preprocesamiento de datos y la implementación de los algoritmos NLP. En promedio cada integrante aportó un promedio de 5.5 horas al desarrollo del proyecto.

Repartición de puntos:

- Juan Nicolas Bolaños: 35
- Juan José Ochoa: 31
- Juan Pablo Sarmiento: 34

Lo anterior tiene justificación en la proporción de horas invertidas en el proyecto y el grado de dificultad de cada una de las tareas asignadas.