Clasificación de Péptidos Antimicrobianos
Universidad Nacional de Colombia

Integrantes:
Edwin Villarraga Ossa – Andrés M. Zapata Rincón - Lesty Delgado Palacios - Santiago Giraldo Escobar - Stephany Gloria Moreno

METODOLOGÍA

PARTE 1: CLASIFICACIÓN SIN SELECCIÓN

Cargue de datos

Inicialmente se observa que los datos suministrados contienen tres archivos. En el archivo DatosPositivos2 están los péptidos de la clase 1, es decir, que son antimicrobianos, en el archivo DatosNegativos2 se tienen los péptidos de la clase 0, es decir, que no son antimicrobianos y en el archivo DatosValidacion2 se facilita un conjunto de datos para validar los modelos seleccionados.

Análisis exploratorio de datos

En este punto se analiza que tipo de datos contiene la muestra, se hace un análisis descriptivo de los datos y se determina que variables pueden ser descartadas. A partir de este, se encuentra lo siguiente:

- El archivo con datos positivos tiene 8322 filas y 1761 columnas, se destaca la primera columna porque no tiene nombre, en la descripción realizada por Python se evidencia que es un conteo, es decir, corresponde a un índice, razón por la cual se elimina del conjunto de datos. El archivo con datos negativos tiene 13679 filas y 1760 columnas, no se elimina ninguna columna.
- Se verifica que los nombres de los encabezados son los mismos para ambos conjuntos de datos, por lo que se unen para formar una base de datos única.
- Las columnas *sequence*, *length* y *class* no son flotantes como las demás. La primera es del tipo object y las otras dos son del tipo int64. La columna *sequence* es la representación de la cadena de aminoácidos que representa cada péptido. La columna *length* corresponde a la cantidad de aminoácidos en la cadena. Las otras columnas parecen estar relacionadas con propiedades biofísicas o químicas de los péptidos. Dado que *lenght* y *class* son variables enteras que no tienen mucha variación, es posible inferir que son del tipo categóricas.
- Se encuentra que ambos conjuntos tienen cadenas de péptidos entre 6 y 35 aminoácidos, y que la columna *class* es la que diferencia o etiqueta ambos conjuntos de datos entre los péptidos positivos (1) y negativos (0).
- Se evalúa que existen secuencias duplicadas (81) en el conjunto de datos completo, por esta razón se eliminan estos registros, independiente de si las demás variables son distintas, con esto se busca evitar un sesgo o distorsión en la muestra.
- Todas las variables son numéricas excepto 1, *sequence*, la cual es una cadena de letras que identifica la molécula.

Normalización

En este punto se hace un preprocesamiento adicional a la muestra de datos, haciendo un arreglo aleatorio de los índices del dataframe para evitar que haya un sesgo en el momento de partir los datos en los subconjuntos de entrenamiento y pruebas, también se elimina la columna secuencia

del dataframe ya que no es una variable numérica, se separa la clase como etiqueta del conjunto de datos y el resto como las variables independientes del modelo. Por último, se escalan los datos de las variables independientes de manera estándar o normal con la función StandardScaler() de skit-learn.

Creación de conjuntos de entrenamiento y prueba
Se divide el conjunto de datos en 2 subconjuntos: un conjunto con el 80% de las muestras para el entrenamiento (training) y otro con el 20% restante como conjunto de pruebas (testing) para entrenar y probar los modelos.

Selección de modelos de clasificación
Se entrenan 6 algoritmos de clasificación: Regresión logística, Bosques aleatorios, SVM, Adaboost, Xgboost e Histogram-based Gradient Boosting Classification Tree.

Inicialmente se realiza ajuste de hiperparámetros con la clase GridSearchCV disponible en scikit-learn y se obtienen los mejores valores de los parámetros para cada clasificador (Ver **Tabla 3**). GridSearchCV, permite evaluar y seleccionar de forma sistemática los parámetros de un modelo, indicándole el modelo y los parámetros a probar, puede evaluar el rendimiento del primero en función de los segundos mediante validación cruzada.

Luego, se entrenan los clasificadores con los datos de entrenamiento de la partición y se realiza validación cruzada para la medición del desempeño, se emplea validación cruzada de k iteraciones (KFold), en Python la función sklearn.model_selection.KFold, con n_splits=10, ya que lo más común es utilizar la validación cruzada de 10 iteraciones y se usan los valores de los parámetros hallados previamente para cada clasificador. Los resultados obtenidos se muestran en la ***Tabla 1*** se puede observar que los mejores modelos son Histogram-based Gradient Boosting Classification Tree, SVM y Bosques Aleatorios.

Modelo	Accuracy
Regresión Logística	88.96%
Bosques Aleatorios	**90.15%**
Máquina de Vectores de Soporte	**91.01%**
Adaboost	88.45%
XGBoost	89.52%
Histogram-based Gradient Boosting Classification Tree	**93.04%**

Tabla 1. Resultados utilizando validación cruzada sobre los datos de entrenamiento

Posteriormente se usa el conjunto de prueba (correspondiente al 20% de los datos), para evaluar los clasificadores. Para cada caso las métricas usadas son la matriz de confusión, precisión, recall y F1-score.

Los resultados obtenidos se muestran en la ***Tabla 2*** **s**e ve que los mejores modelos siguen siendo Histogram-based Gradient Boosting Classification Tree y SVM, teniendo en cuenta todas las medidas de desempeño. En cuanto al recall, se ve que los modelos son mejores prediciendo la clase negativa que la clase positiva, la precisión de la clase 0 y 1 son muy parejos y en general f1-Score es mejor para la clase negativa en cada modelo.

Modelo	Accuracy	Precision Class=0	Recall class=0	f1-Score class=0	Precision Class=1	Recall class=1	f1-Score class=1
Regresión Logística	89.8%	0.90	0.94	0.92	0.89	0.83	0.86
Bosques Aleatorios	89.8%	0.87	0.98	0.92	0.96	0.76	0.85
Máquina de Vectores de Soporte	**91.7%**	**0.90**	**0.97**	**0.94**	**0.95**	**0.83**	**0.88**
Adaboost	88.5%	0.89	0.92	0.91	0.87	0.82	0.84
XGBoost	89.4%	0.88	0.96	0.92	0.92	0.79	0.85
Hist Gradient Boosting Classifier	**94.7%**	**0.94**	**0.98**	**0.96**	**0.96**	**0.90**	**0.93**

Tabla 2. Resultados sobre los datos de prueba.

La **Tabla 3** muestra para cada clasificador, los valores de los hiperpárametros hallados con GridSearchCV y los tiempos de entrenamiento, entrenamiento con validación cruzada y evaluación con el conjunto de prueba.

Modelo	Mejores parámetros	Tiempo train	Tiempo train con CV	Tiempo test
Regresión Logística	'C': 0.01, 'penalty': 'l2', 'solver': 'newton-cg'	00:03:48.68	00:00:40.13	00:00:05.61
Bosques Aleatorios	criterion': 'gini', 'max_depth': 15, 'min_samples_leaf': 2, 'n_estimators': 350	00:22:44.62	00:02:30.10	00:01:20.09
SVM	C': 0.0001, 'gamma': 1	00:00:10.93	00:14:29.23	00:07:43.62
Adaboost	'algorithm': 'SAMME.R', 'learning_rate': 1.0, 'n_estimators': 150	00:38:55.49	00:05:51.32	00:03:05.61
XGBoost	'learning_rate': 0.01, 'max_depth': 5, 'n_estimators': 200, 'subsample': 1.0	01:51:27.05	00:07:23.82	00:03:55.67
Hist Gradient Boosting Classifier	learning_rate': 0.01, 'max_depth': 5, 'max_leaf_nodes': 2	00:27:45.84	00:00:28.18	00:00:34.36

Tabla 3. Valores de los Hiperparámetros y tiempos de ejecución.

Cabe resaltar, que por alguna razón el SVM encuentra un mejor resultado con kernel='rbf', max_iter=10000, sin los parámetros C y gamma (pasa de un accuracy de 62.38% a uno de 91.01%), por este motivo no se toma la solución de la malla de búsqueda y se deja la configuración sin especificar dichos parámetros. Así mismo, Histogram-based Gradient Boosting Classification Tree da un mejor resultado sin la parametrización obtenida (pasa de un accuracy de 79.17% a uno de 93.04%).

Respeto a los tiempos, se ve que el entrenamiento con validación cruzada y de prueba son mucho menores que el tiempo de entrenamiento de la participación sin validación cruzada. Además, el modelo **XGBoost** es el que toma más tiempo en el entrenamiento (1 hora 51 minutos y 27 segundos). En la Parte 2 a continuación, se hace selección de características y se valida en el nuevo conjunto con reducción de variables, se entrena y se prueba, para finalmente hacer un análisis general de los resultados obtenidos en cada uno de los pasos desarrollados en la parte 1 y 2.

PARTE 2: CLASIFICACIÓN CON SELECCIÓN

Selección y Reducción de Variables

Se crea un modelo de regresión LASSO y se ajusta a los datos de entrenamiento escalados, con una semilla=0, con la función SelectFromModel de scikit-learn para identificar las características susceptibles de ser seleccionadas, es decir, las más predictivas. Entre mayor alpha más reducción de variables. Se obtienen 163 variables significativas, es decir se excluyen 1595 variables de un total de 1758, equivalente a una reducción del 90.7%.

Aplicación de los clasificadores sobre el nuevo conjunto de características

En esta parte, se entrenan los clasificadores con los datos de entrenamiento de la partición (del nuevo conjunto de características), se realiza ajuste de los hiperparámetros (ver **Tabla 6**) , en este caso se agrega validación cruzada usando la función GridSearchCV, con cv=10, lo cual supone el mismo proceso del KFold realizado en la parte 1.

Los resultados para el entrenamiento se muestran en la **Tabla 4**, se ve que el mejor es el modelo Histogram-based Gradient Boosting Classification Tree, para el caso del modelo de Máquina de Vectores de Soporte la parametrización no mejora el accuracy por tanto en la prueba no se tienen en cuenta.

Modelo	Accuracy
Regresión Logística	88.73%
Bosques Aleatorios	**89.98%**
Máquina de Vectores de Soporte	**37.62%**
Adaboost	88.44%
XGBoost	89.06%
Histogram-based Gradient Boosting Classification Tree	**90.96%**

Tabla 4. Resultados sobre los datos de entrenamiento con reducción de variables.

Posteriormente, se evalúan los modelos con el conjunto de prueba. Los resultados se muestran en la **Tabla 5**, de acuerdo con las medidas de desempeño, el modelo Histogram Based Gradient Boosting Classification Tree es el mejor en los datos de prueba, le sigue el SVM. Se ve que para las clases 0 y 1, el recall y f1-Score sigue siendo mejor para la clase 0, sin embargo, para la clase 1 no son malos y la precisión clase 0 y 1 muy pareja para cada modelo.

Modelo	Accuracy	Precision Class=0	Recall class=0	f1-Score class=0	Precision Class=1	Recall class=1	f1-Score class=1
Regresión Logística	88.2%	0.88	0.93	0.91	0.88	0.80	0.84
Bosques Aleatorios	90.3%	0.88	0.98	0.93	0.95	0.78	0.86
SVM	**91.0%**	**0.96**	**0.90**	**0.93**	**0.82**	**0.93**	**0.87**
Adaboost	88.3%	0.89	0.92	0.91	0.87	0.82	0.84
XGBoost	89.3%	0.88	0.96	0.92	0.93	0.78	0.85
Hist Gradient Boosting Classifier	**93.2%**	**0.93**	**0.97**	**0.95**	**0.94**	**0.88**	**0.91**

Tabla 5. Resultados utilizando validación cruzada sobre los datos de prueba con reducción de variables.

La **Tabla 6**, muestra los valores de los hiperparámetros seleccionados mediante GriedSearchCV y el tiempo de evaluación con el conjunto de prueba para cada modelo.

Modelo	Mejores parámetros	Tiempo train	Tiempo test
Regresión Logística	'C': 0.1, 'penalty': 'l2', 'solver': 'newton-cg'	00:00:28.15	00:00:00.48
Bosques Aleatorios	criterion': 'entropy', 'max_depth': 15, 'min_samples_leaf': 2, 'n_estimators': 350	00:16:41.76	00:00:33.33
SVM	C': 0.0001, 'gamma': 1	00:00:02.24	00:00:33.74
Adaboost	algorithm': 'SAMME.R', 'learning_rate': 1.0, 'n_estimators': 150	00:07:19.96	00:00:16.43
XGBoost	learning_rate': 0.01, 'max_depth': 5, 'n_estimators': 200, 'subsample': 0.5	00:22:59.57	00:00:23.53
Hist Gradient Boosting Classifier	learning_rate': 1, 'max_depth': 8, 'max_leaf_nodes': 2	01:31:26.65	00:00:02.93

Tabla 6. Valores de los Hiperparámetros y tiempos de ejecución en el conjunto de datos reducido.

Como se mencionó, en el caso del modelo SVM, igual que en la parte 1, el resultado es mejor sin los parámetros seleccionados con el GridSearchCV (encuentra un mejor resultado con kernel='rbf', max_iter=10000, pasa de un accuracy de 62.2% a uno de 91.0%), así mismo, para el modelo Histogram Based Gradient Boosting Classification Tree (pasa de un accuracy de 89.4% a uno de 93.2%), por lo que los modelo se dejan sin ajuste de los hiperparámetros.

Respecto a los tiempos, se ve que los de entrenamiento son mayores dado que evalúa el 80% de los datos mientras que los de prueba son mucho menores (prueba el 20% de los datos). Excepto en el SVM dado que el entrenamiento se corre con los parámetros ajustados mientras que en la prueba no se tienen en cuenta, dado que no mejoran el rendimiento.

EVALUACIÓN CON EL CONJUNTO DE VALIDACIÓN

Inicialmente se cargan los datos de validación (datos que no han conocido los modelos), se hace un procesamiento a la base de datos (se borran las tres primeras columnas) y se normalizan los datos. Se usan las variables seleccionadas por la regresión LASSO, puesto que no hubo una variación significativa de la precisión de los modelos al reducir el número de variables. También se usan los valores de los parámetros hallados que mejoran el rendimiento de los clasificadores.

Al evaluar los modelos se obtienen los resultados mostrados en la **Tabla 7**, en general, el accuray obtenido disminuye significativamente respecto a la evaluación con los datos de prueba, lo que puede indicar que hubo sobre entrenamiento, sin embargo los valores de recall class=1 mejoran lo que nos indica que son buenos prediciendo los casos positivo, los mejores modelos en esta etapa son los bosque aleatorios y SVM (Máquina de Vectores de Soporte) teniendo en cuenta todas las medidas de desempeño.

Modelo	Accuracy	Precision Class=0	Recall class=0	f1-Score class=0	Precision Class=1	Recall class=1	f1-Score class=1
Regresión Logística	77.4%	0.98	0.74	0.85	0.42	0.93	0.58
Bosques Aleatorios	**85.2%**	**1.00**	**0.82**	**0.90**	**0.53**	**0.99**	**0.69**
SVM	**79.3%**	**0.99**	**0.76**	**0.86**	**0.44**	**0.96**	**0.61**
Adaboost	70.7%	0.98	0.66	0.79	0.36	0.95	0.52
XGBoost	74.7%	0.98	0.71	0.83	0.39	0.91	0.54
Hist Gradient Boosting Classifier	74.4%	0.70	0.99	0.82	0.97	0.39	0.56

Tabla 7. Resultados usando el conjunto de validación con reducción de variables.

A continuación, se hace una comparación de los resultados y se dan a conocer los hallazgos y conclusiones obtenidos.

Análisis, discusión de resultados y conclusiones

La **Tabla 8** muestra los resultados obtenidos por los diferentes modelos durante las etapas de train y test sin reducción y con reducción de variables.

Modelo	Accuracy Train	Accuracy Test	Accuracy Test	Accuracy Validation Data	Recall Class = 1
	Cross-Validation	All Variables	Reduced Variables	Reduced Variables	
Regresión Logística	88.96%	89.8%	88.2%	77.4%	93%
Bosques Aleatorios	90.15%	89.8%	90.2%	**85.2%**	**99%**
SVM	**91.01%**	**91.7%**	**91.0%**	**79.3%**	**96%**
Adaboost	89.52%	88.5%	88.3%	70.7%	95%
XGBoost	90.17%	89.4%	89.0%	74.7%	91%
HGBCT[1]	**93.04%**	**94.7%**	**93.2%**	**74.4%**	39%

Tabla 8. Comparación de resultados obtenidos.

El modelo Histogram Based Gradient Boosting Classificarion Tree presentó resultados muy favorables durante la etapa de cross validation y pruebas, con todas las variables y con variables reducidas. Sin embargo, en los datos de validación, su accuracy y recall class=1, disminuyó mucho, lo cual indica que el modelo tuvo overfitting en los datos de prueba y que tal vez los datos de validación tienen alguna diferencia en sus distribuciones de datos respecto a los datos de entrenamiento.

El modelo de Bosques Aleatorios no presentó un desempeño superior a los demás modelos durante la etapa de entrenamiento, pero en los datos de validación tuvo el mejor desempeño con accuracy 85,2% y recall class=1 de 99% lo cual indica que es muy bueno prediciendo los casos de péptidos que son verdaderamente antimicrobianos.

El modelo de Clasificación de **Máquinas de Vectores de Soporte (SVM)** presentó el segundo mejor desempeño, en las etapas de entrenamiento y validación, por lo que presentó resultados más consistentes. Se elegiría este modelo para implementar en producción.

Sin embargo, dado el desbalance de las clases en los datos de validación, la medida de accuracy es engañosa, no es una buena medida para evaluar el rendimiento de los modelos. Para el problema en cuestión, se busca predecir péptidos con capacidad antimicrobiana y en los datos de validación hay pocos casos con esta característica, 269, frente a 1354 casos de péptidos sin propiedades antimicrobianas, esta puede ser la razón por la que en los datos de validación el accuracy no está siendo tan bueno.

En este caso la principal medida de desempeño debería ser el recall para los casos de class=1, es decir, la capacidad del modelo para identificar los verdaderos casos de péptidos antimicrobianos. Bajo la medida de Recall, también los modelos SVM y Bosques aleatorios presentan un mejor desempeño que los demás. La capacidad predictiva condicional respecto al hecho de que un péptido sea antimicrobiano es del **99%** y **96%** respectivamente, lo cual representa una herramienta que mejora la eficiencia en la búsqueda de potenciales péptidos antimicrobianos.

[1] Hace referencia al modelo Histogram Based Gradient Boosting Classification Tree.