“PREDICCIÓN DE LA PRECIPITACIÓN USANDO MACHINE LEARNING 2025 LIMA, PERÚ”

Autor/Autores: Perez Nina Divano Estefano - 20221481, Ascama Rojo Flavio - 20212426, Mora Pilares Martin Fernando - 20200874, Mamani Yufra Reynaldo Alonso - 20221386, Palomino Villalobos Gabriel - 20200497

Resumen-El presente trabajo aborda la necesidad de una predicción temprana y precisa de la precipitación en la región de Junín, dado el impacto que este fenómeno tiene sobre la vida y la economía local. El objetivo es desarrollar un modelo que anticipe la ocurrencia de lluvia a partir de dataset proporcionado por la Plataforma Nacional de Datos Abiertos- Estación Meteorológica Automática para variables atmosféricas. Para ello, se evaluarán diversas técnicas de balanceo con el fin de reducir el sesgo en la data. Los resultados obtenidos permitirán identificar patrones relevantes en la dinámica climática y apoyar la toma de decisiones en la gestión del riesgo ambiental.

1. Introducción

En la región de Junín, la precipitación representa un factor crítico para la agricultura, las estructuras, la seguridad vial y la planificación hídrica. Entre 2024 y 2025 fueron aproximadamente 100 000 personas en riesgo en 1900 comunidades, con pérdidas económicas de alrededor de 689 millones (Cenepred, 2025). Es por ello que una predicción oportuna de precipitaciones permitiría mitigar riesgos y optimizar decisiones en sectores dependientes del clima.

Ante esta problemática surge la hipótesis:

¿Es posible entrenar un modelo de Machine Learning para predecir la ocurrencia de precipitación en la región de Junín a partir de variables atmosféricas teniendo un desbalance del target en la data evaluada?

Para responder a esta incógnita se desarrollará un modelo de aprendizaje automático para predecir si habrá precipitación en un determinado momento, usando variables como temperatura, humedad, presión, velocidad del viento, entre otras.

Objetivos:

- Desarrollar un modelo de Machine Learning que anticipe la ocurrencia de precipitaciones en la región de Junín.
- Implementar y comparar técnicas de balanceo de clases para reducir el sesgo en la clase minoritaria.
- Comparar el rendimiento de sobre-muestreos y sub-muestreos para distintos tamaños de datasets.

2. Trabajos relacionados

Se revisaron trabajos relacionados al manejo del balanceo para mejorar el rendimiento de algoritmos de Machine Learning.

Imbalanced Data problem in Machine Learning: A review

Este artículo cubre técnicas a nivel de datos (como SMOTE, Borderline-SMOTE, ADASYN), técnicas a nivel de algoritmo (aprendizaje sensible al costo, modificaciones en SVM y Random Forest), y métodos híbridos (como SMOTE-ENN, GANs, o Stacked CNN+RNN). La investigación enfatiza la necesidad de usar métricas adecuadas como AUC-ROC, F1-Score y G-Mean para evaluar modelos en contextos desbalanceados.

The effects of data balancing approaches: A case study

Este estudio explora estrategias de re muestreo y selección de características para mejorar la clasificación de muestras de orina bovina tratadas con hormonas de crecimiento. El conjunto de datos incluye un desbalance del 5%. Para corregir esto, se probaron algoritmos de clasificación como Logistic Regression, Gaussian Process Classifier y Linear Discriminant Analysis, combinados con SMOTE y ADASYN para sobre-muestreo.

El modelo más efectivo fue Logistic Regression con SMOTE o ADASYN, seguido por Gaussian Process Classifier con SMOTE y Linear Discriminant Analysis. Se concluye que la combinación de transformación logarítmica y eliminación recursiva de características mejora significativamente la precisión del modelo.

Mitigating Imbalance: Cost-Sensitive Learning for Enhanced Weather Prediction in Imbalanced Datasets

Este artículo emplea aprendizaje sensible al costo, ajustando los pesos en modelos como Árboles de Decisión, Regresión Logística, Ridge Classifier y SVM, logrando una mejor representación de eventos climáticos poco frecuentes. La investigación destaca cómo técnicas de balanceo, como la calibración de pesos y la optimización de hiper parámetros, pueden mejorar significativamente métricas como Balanced Accuracy y F1-Score, fortaleciendo la capacidad predictiva de los modelos.

3. Metodología

El objetivo es predecir si un día habrá precipitación o no, a partir de datos registrados en la Estación Meteorológica Automática (EMA) Huayao, Junín, Perú.

Para la predicción se usarán algoritmos de clasificación. Todos los experimentos se realizan con python usando Google Colab.

A. Descripción de datos

El archivo IGP_EstacionEMA_2018-2024_Dataset.csv contiene aproximadamente 61000 registros y 12 variables las cuales son:

Variable	Descripción
FECHA_CORTE	Fecha de corte de información
UBIGEO	Código de Ubicación Geográfica que denotan "DDppdd" (Departamento, provincia,distrito)
YY	Year, año de registro
MM	Month, mes de registro
DY	Day, dia de registro
HH	Hour, hora promedio del registro
TT	Air Temperature 2 meters (temperatura del aire a 2 metros) - [°C]
HR	Relative Humidity (Humedad Relativa) - [%]
RR	Precipitation (precipitación) - [mm]
PP	Atmospheric pressure (presión atmosférica) - [hPa]
FF	Wind speed (velocidad del aire) - [m/s]
DD	Wind direction (dirección del aire) - [grados]

Tabla 1: Diccionario de datos del dataset de precipitaciones en Junín.
Fuente: Instituto Geofísico del Perú (2025)

Todas las variables son numéricas menos UBIGEO que es Alfanumérica.

B. Procesamiento de datos

Para el preprocesamiento del dataset primero se eliminaron las columnas de UBIGEO y FECHA_CORTE al no ser relevantes para el modelo.
Se elimina la columna YY porque se espera que el modelo sea capaz de predecir información de nuevos años.
Para MM, DY y HH usamos una codificación cíclica o radial, de esta manera el modelo no pensara que los valores extremos, como lunes y domingo, son lejanos.

C. Balanceo

Se realizará transformación logarítmica para mejorar la distribución de las variables numéricas, especialmente cuando están fuertemente sesgadas y se usará validación cruzada con 10 folds para evaluar la capacidad de generalización de un modelo de machine learning.

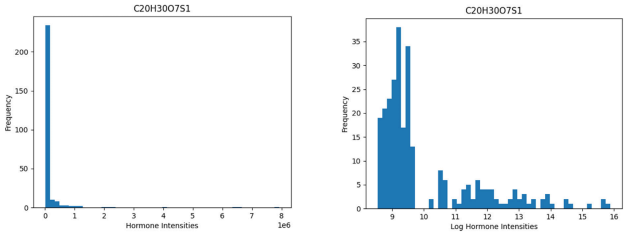


Figura 1. Transformación logarítmica.

Fuente: Moojiman, P., Catal, C., Tekinerdogan, B., Lommen, A. & Blokland, M. (2022)

Luego se aplicarán técnicas de balanceo. El estudio evaluará tanto métodos de sobremuestreo como submuestreo, así como sus combinaciones.

Sobremuestreo (que generan nuevas muestras sintéticas de la clase minoritaria):

- SMOTE (Synthetic Minority Oversampling Technique)
- Borderline-SMOTE
- ADASYN (Adaptive Synthetic Sampling)
- Random Oversampling (ROS)
- SVM-SMOTE

Submuestreo (que eliminan ruido o redundancia de la clase mayoritaria):

- Random Undersampling (RUS)
- Tomek Links
- Edited Nearest Neighbors (ENN)
- One-Sided Selection (OSS)
- Neighborhood Cleaning Rule (NCL)
- Near Miss

Se espera mejorar el rendimiento de los algoritmos de clasificación al ser combinados con las técnicas de balanceo, reduciendo falsos negativos y aumentando la representatividad de la clase de interés.

D. Algoritmos de clasificación

Se probarán algoritmos como Logistic Regression, KNeighbours classifier, Decision trees, Adaboost, Gradient Boosting, Linear Discriminant Analysis y Random Forest. De los cuales, este último parece ser el más prometedor debido a pruebas preliminares. En cuanto a los demás, tuvieron una alta representatividad en los trabajos anteriores revisados.

Modelo Random Forest
Un algoritmo basado en el método de ensamblado conocido como bagging. Recomendado específicamente cuando se manejan datos desbalanceados (Khoshgoftaar, Golawala & Van Hulse, 2007).

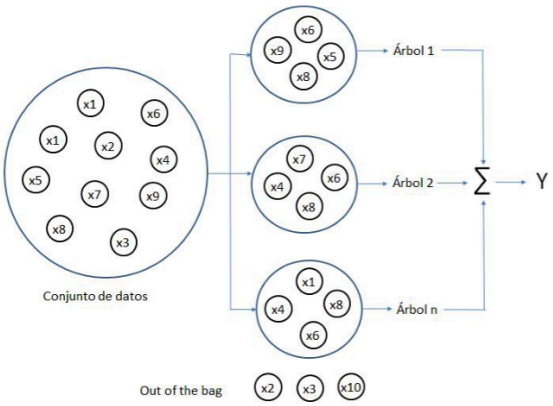


Figura 2: Funcionamiento de Random Forest
Fuente: Espinosa, J. (2020)

- El modelo construye múltiples árboles de decisión durante el entrenamiento, para que durante la clasificación, se tome la clase que representa la mayoría de las predicciones de los árboles individuales.
- **Parámetro class_weight='balanced':** Esto indica que el modelo debe ajustar los pesos inversamente proporcionales a las frecuencias de las clases en los datos de entrada. Esto ayuda al modelo a manejar los conjuntos con datos desbalanceados

E. Evaluación de algoritmos

La evaluación del desempeño estará determinada por la matriz de confusión:

		clase real	
		positiva	negativa
predicción	positiva	verdadero positivo (tp)	falso positivo (fp)
	negativa	falso negativo (fn)	verdadero negativo (tn)

Figura 3. Matriz de confusión.
Fuente: Acevedo, M. & Vargas, K. (2017)

Considerando el recall (por categoría):

$$R = \frac{TP}{FN + TP}$$

y el Balanced-accuracy (score global):

$$BA = 0.5\left(\frac{TN}{FP + TN} + \frac{TP}{FN + TP}\right)$$

Se espera que estos valores tiendan a 1.

4. Experimentación y Resultados

Datos usados
Se utilizó un conjunto de datos meteorológicos horarios de la estación EMA(Estación Meteorológica Automática) de la región de

Junín, Perú, proporcionados por el Servicio Nacional de Meteorología e Hidrología (SENAMHI). Los datos abarcan desde enero de 2018 hasta diciembre de 2024.

Métricas de Evaluación
Dada la naturaleza desbalanceada del problema, se priorizaron métricas robustas a la asimetría de clases:

- **Precisión (Precision):** Proporción de predicciones positivas correctas.
- **Exhaustividad (Recall):** Proporción de casos positivos reales detectados.
- **Puntuación F1 (F1-Score):** Media armónica de precisión y exhaustividad.
- **Exactitud Balanceada (Balanced Accuracy):** Media de exhaustividad por clase.
- **Matriz de Confusión:** Para visualizar falsos positivos/negativos.

Experimentos Realizados
a) Estrategia de Validación

- División de datos:
  - 80% entrenamiento (48,637 muestras).
  - 20% validación (12,160 muestras).
  - Mecanismo: Partición aleatoria estratificada (train_test_split de Scikit-learn) con semilla (random_state = 7) para garantizar repetibilidad.

b)Técnicas de Remuestreo Probadas

Tipo	Técnica	Parámetros Clave
Sobremuestreo	SMOTE	k_neighbors = 5
	Borderline-SMOTE	kind = "borderline-1"
	ADASYN	n_neighbors = 5
	Random Oversampling	Muestreo aleatorio con reposición
	SVMSMOTE	Kernel SVM para generar muestras
Submuestreo	Random Undersampling	Reducción aleatoria de la clase mayoritaria
	Tomek Links	Eliminación de pares "confusos"
	Edited Nearest Neighbours (ENN)	n_neighbors = 3
	One-Sided Selection (OSS)	Combinación de Tomek Links y CNN
	Neighbourhood Cleaning Rule (NCL)	Limpieza basada en vecindades
	NearMiss	Selección basada en distancia a minoritaria

Modelos y Parámetros

Algoritmos:

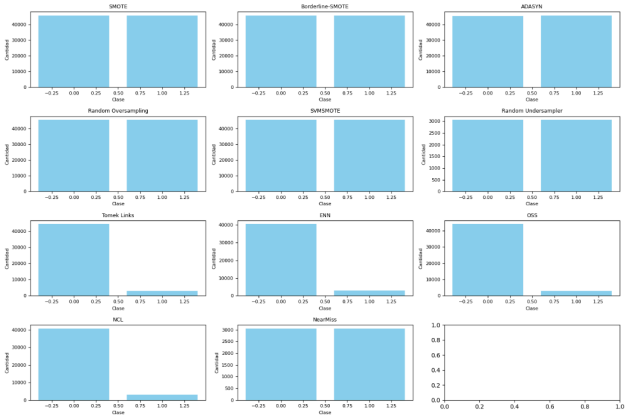
- Regresión Logística (LogisticRegression)
- K-Nearest Neighbors con 5 y 10 vecinos (KNeighborsClassifier)
- Árboles de decisión (DecisionTreeClassifier)
- Random Forest (RandomForestClassifier)
- AdaBoost (AdaBoostClassifier)
- Gradient Boosting (GradientBoostingClassifier)
- Análisis Discriminante Lineal (LinearDiscriminantAnalysis)

Ajuste de Hiper Parámetros
Búsqueda en cuadrícula (Grid Search) con validación cruzada de 5 pliegues. Parámetros clave: max_depth, n_estimators, C y kernel

Escenarios Comparados

1. Línea base: Modelos entrenados sin remuestreo
2. Sobremuestreo: Cada técnica aplicada individualmente al entrenamiento
3. Submuestreo: Idéntico enfoque para técnicas de reducción
4. Híbridos: Combinación de SMOTE + Tomek Links

Distribución de clases tras aplicar técnicas de resampling



Pruebas de algoritmos + sobre - muestreo

Las técnicas evaluadas fueron:

- SMOTE
- Borderline-SMOTE
- SVM SMOTE
- ADASYN
- RandomOverSampler (ROS)

Para cada una de estas técnicas, se entrenaron y evaluaron distintos algoritmos de clasificación, entre ellos: LogReg, 5NN, 10NN, CART, Random Forest, AdaBoost, GradientBoosting, y LDA. La métrica utilizada fue el balanced accuracy, con validación cruzada (k-fold).

Se observa que la mejor combinación, según los resultados de validación cruzada, fue **Random Forest + Random OverSampler (ROS)**, alcanzando un accuracy balanceado de 0.9932. Posteriormente, se validó esta combinación sobre el conjunto de validación, obteniendo:

```
Balanced Accuracy score: 0.6593270460358056

Matriz de Confusión:
[[11277   147]
 [  492   244]]

Reporte de clasificacion en conjunto de validacion:

              precision    recall  f1-score   support

         0.0       0.96      0.99      0.97     11424
         1.0       0.62      0.33      0.43       736

    accuracy                           0.95     12160
   macro avg       0.79      0.66      0.70     12160
weighted avg       0.94      0.95      0.94     12160
```

Aunque el modelo logró una alta precisión general (95%) y un rendimiento muy elevado para la clase mayoritaria (clase 0), los resultados muestran que la clase minoritaria (clase 1) aún presenta desafíos: el recall de 0.33 indica que el modelo sólo logra identificar correctamente un tercio de los ejemplos positivos.

Pruebas de algoritmos + sub - muestreo

Las técnicas evaluadas fueron:

- Random UnderSampler (RUS)
- Tomek Links
- Edited Nearest Neighbours (ENN)
- One-Sided Selection (OSS)
- Neighbourhood Cleaning Rule (NCL)
- NearMissAl igual que en el caso anterior, se entrenaron y evaluaron distintos algoritmos de clasificación sobre los datos submuestreados: LogReg, 5NN, 10NN, CART, Random Forest, AdaBoost, GradientBoosting, y LDA, utilizando validación cruzada y la métrica de balanced accuracy.

Se observa que la más efectiva combinación fue **Random Forest + Random Undersampler**, que logró un balanced accuracy promedio de 0.8291 en la validación cruzada.

```
Reporte de clasificacion en conjunto de validacion:

              precision    recall  f1-score   support

         0.0       0.99      0.81      0.89     11424
         1.0       0.22      0.85      0.35       736

    accuracy                           0.81     12160
   macro avg       0.61      0.83      0.62     12160
weighted avg       0.94      0.81      0.86     12160
```

El modelo logró una mejora significativa en la detección de la clase minoritaria (1), con un recall de 0.85, es decir, fue capaz de identificar correctamente el 85% de los casos positivos. Esto refleja una mejor sensibilidad que en el caso del sobre-muestreo.

Pruebas con pesos

En esta sección se evaluó el impacto de asignar pesos personalizados a las clases dentro del modelo de Random Forest, con el objetivo de compensar el desbalance sin alterar el conjunto de datos mediante técnicas de remuestreo. Se probaron distintas configuraciones del parámetro class_weight, incluyendo el valor 'balanced' y pesos definidos manualmente para dar mayor importancia a la clase minoritaria.

1- Resultados para: class_weight = 'balanced'

```
Balanced Accuracy score: 0.6593270460358056

Matriz de Confusión:
[[11277   147]
 [  492   244]]

Reporte de clasificacion en conjunto de validacion:

              precision    recall  f1-score   support

         0.0       0.96      0.99      0.97     11424
         1.0       0.62      0.33      0.43       736

    accuracy                           0.95     12160
   macro avg       0.79      0.66      0.70     12160
weighted avg       0.94      0.95      0.94     12160
```

2- Resultados para: class_weight = {0: 1, 1: 13.25}

```
Balanced Accuracy score: 0.6502976190476191

Matriz de Confusión:
[[11288   136]
 [  506   230]]

Reporte de clasificacion en conjunto de validacion:

              precision    recall  f1-score   support

         0.0       0.96      0.99      0.97     11424
         1.0       0.63      0.31      0.42       736

    accuracy                           0.95     12160
   macro avg       0.79      0.65      0.69     12160
weighted avg       0.94      0.95      0.94     12160
```

3- Resultados para: class_weight = {0: 1, 1: 0.04}

```
Balanced Accuracy score: 0.6971973572037511

Matriz de Confusión:
[[11242   182]
 [  434   302]]

Reporte de clasificacion en conjunto de validacion:

              precision    recall  f1-score   support

         0.0       0.96      0.98      0.97     11424
         1.0       0.62      0.41      0.50       736

    accuracy                           0.95     12160
   macro avg       0.79      0.70      0.73     12160
weighted avg       0.94      0.95      0.94     12160
```

La mejor configuración dentro de las pruebas con pesos fue {0: 1, 1: 0.04}, la cual logró un mayor recall para la clase minoritaria. No obstante, comparado con técnicas como el sub-muestreo o el sobre-muestreo, no presentó mejoras significativas a nivel general.

Pruebas usando escaladores

Se evaluó el impacto de aplicar técnicas de escalamiento (MinMaxScaler) como parte de un pipeline de clasificación, en combinación con los algoritmos ya utilizados y técnicas de remuestreo (ROS y RUS). Se construyeron pipelines que incluían escalamiento previo al entrenamiento de modelos como Random Forest, LogReg, SVM, AdaBoost, entre otros.

Las combinaciones más destacadas fueron nuevamente Random Forest + ROS y Random Forest + RUS, por lo que se procedió a validar estos modelos. Los resultados fueron:

- RF + ROS: Balanced accuracy

```
Balanced Accuracy score: 0.6667123371087565

Matriz de Confusión:
[[11275   149]
 [  481   255]]

Reporte de clasificacion en conjunto de validacion:

              precision    recall  f1-score   support

         0.0       0.96      0.99      0.97     11424
         1.0       0.63      0.35      0.45       736

    accuracy                           0.95     12160
   macro avg       0.80      0.67      0.71     12160
weighted avg       0.94      0.95      0.94     12160
```

- RF + RUS: Balanced accuracy

```
Balanced Accuracy score: 0.8278623949579832

Matriz de Confusión:
[[9276 2148]
 [ 115  621]]

Reporte de clasificacion en conjunto de validacion:

              precision    recall  f1-score   support

         0.0       0.99      0.81      0.89     11424
         1.0       0.22      0.84      0.35       736

    accuracy                           0.81     12160
   macro avg       0.61      0.83      0.62     12160
weighted avg       0.94      0.81      0.86     12160
```

Aunque el uso de escaladores no generó mejoras consistentes en todas las combinaciones, en el caso de Random Forest + RUS + MinMaxScaler, se logró un recall sobresaliente de 0.84 para la clase minoritaria, junto con un buen balance general (balanced accuracy de 0.8279). Aún así, no fue suficiente para superar el recall sin escaladores.

Pruebas con distintos tamaños de datasets (manteniendo la desproporción de desbalance)

Se realizaron pruebas con fracciones del dataset de 10% y 2% del total de la data, obteniendo los siguientes resultados:

10% de la data original con sobremuestreo: RF + ROS

```
Balanced Accuracy score: 0.686969696969697

Matriz de Confusión:
[[1127   23]
 [  40   26]]

Reporte de clasificacion en conjunto de validacion:

              precision    recall  f1-score   support

         0.0       0.97      0.98      0.97      1150
         1.0       0.53      0.39      0.45        66

    accuracy                           0.95      1216
   macro avg       0.75      0.69      0.71      1216
weighted avg       0.94      0.95      0.94      1216
```

10% de la data original con submuestreo: LDA + RUS

```
Balanced Accuracy score: 0.8328063241106719

Matriz de Confusión:
[[870 280]
 [  6  60]]

Reporte de clasificacion en conjunto de validacion:

              precision    recall  f1-score   support

         0.0       0.99      0.76      0.86      1150
         1.0       0.18      0.91      0.30        66

    accuracy                           0.76      1216
   macro avg       0.58      0.83      0.58      1216
weighted avg       0.95      0.76      0.83      1216
```

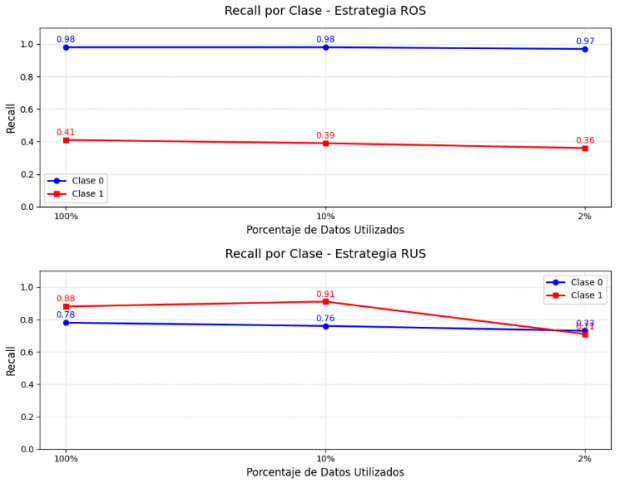
2% de la data original con sobremuestreo: RF + ROS

```
Balanced Accuracy score: 0.665527950310559

Matriz de Confusión:
[[224   6]
 [  9   5]]

Reporte de clasificacion en conjunto de validacion:

              precision    recall  f1-score   support

         0.0       0.96      0.97      0.97       230
         1.0       0.45      0.36      0.40        14

    accuracy                           0.94       244
   macro avg       0.71      0.67      0.68       244
weighted avg       0.93      0.94      0.94       244
```

2% de la data original con submuestreo: RF + RUS

```
Balanced Accuracy score: 0.7245341614906833

Matriz de Confusión:
[[169  61]
 [  4  10]]

Reporte de clasificacion en conjunto de validacion:

              precision    recall  f1-score   support

         0.0       0.98      0.73      0.84       230
         1.0       0.14      0.71      0.24        14

    accuracy                           0.73       244
   macro avg       0.56      0.72      0.54       244
weighted avg       0.93      0.73      0.80       244
```

Se puede comparar los recalls de la data gráficamente:



El resultado evidencia un decaimiento del rendimiento de las técnicas de submuestreo a medida que existe menos data disponible, esto se debe a que al aplicarlas se pierde representatividad de la data, por lo que se está eliminando información importante para el modelo. En cambio, el sobremuestreo es más estable:

Disminución del rendimiento en porcentaje:

- Sobremuestreo: - 4.29%
- Submuestreo: - 13.25%

5. Conclusión

1. Se pudo mejorar el resultado de predicción de ocurrencias de lluvia usando técnicas de submuestreo y sobremuestreo para un dataset desbalanceado.

2. Utilizar técnicas de submuestreo sin tener mucha data genera pérdidas de representatividad.

3. La asignación de pesos ayudó a mejorar las predicciones.

4. El escalamiento de la data no ayudó de manera significativa a mejorar las predicciones.

6. Sugerencias de trabajos futuros

Podría experimentarse con distintas proporciones de desbalance y distintos datasets, para tener una comparativa más amplia del rendimiento de técnicas de balanceo.

Aplicar el modelo a distintas regiones del Perú con variabilidad climática significativa, como la costa, sierra y selva, permitirá evaluar su capacidad de generalizar patrones y adaptar su rendimiento a diferentes contextos meteorológicos mejorando su robustez en escenarios reales.

7. Implicancias éticas

Sesgo en el modelo: Si no se corrige el desbalance de clases, el modelo podría subestimar eventos importantes como lluvias reales, afectando decisiones críticas. Esto se mitiga con técnicas de balanceo, métricas adecuadas y monitoreo constante.

Privacidad y uso de datos: Aunque los datos actuales son públicos, futuras fuentes podrían contener información sensible. Es necesario implementar anonimización, recolectar solo lo indispensable y garantizar consentimiento informado.

Automatización sin supervisión: Confiar sin cuestionar en las predicciones puede derivar en decisiones erróneas. Se debe asegurar la supervisión humana y utilizar el modelo como apoyo, no como única guía.

8. Link del repositorio del trabajo

https://github.com/DivanoPerezNina/InformeIA/tree/main

El documento final es: Proyecto_final_grupo.ipynb

9. Declaración de contribución de cada integrante

Divano Perez Nina: Informe, el ppt y la experimentación.
Reynaldo Mamani Yufra: Introducción, algunas partes de procesamiento y experimentación en el Informe, ppts de experimentación y pruebas en colab.
Gabriel Palomino Villalobos: Informe, experimentacion en colab, ppt.
Martin Mora Pilares: Informe y la experimentación.
Flavio Ascama Rojo: Informe, aportes en el ppt y pruebas en colab.

10. Referencias

[1] Khoshgoftaar, T. M., Golawala, M., & Van Hulse, J. (2007). *An empirical study of learning from imbalanced data using Random Forest*. En *Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)* (Vol. 2, pp. 310–317). IEEE.

[2]. Moojiman, P., Catal, C., Tekinerdogan, B., Lommen, A. & Blokland, M. (2022) The effects of data balancing approaches: A case study. Applied Soft Computing. Países Bajos, 2023, vol. 132, art. 109853, pp. 1–31.

[3]. Altalhan, M., Algarni, A. & Alouane, T. (2024), Imbalanced Data problem in Machine Learning: A review. IEEE Access.

[4].S. D and G. N. Nimeshika, "Mitigating Imbalance: Cost-Sensitive Learning for Enhanced Weather Prediction in Imbalanced Datasets," 2024 10th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 2024, pp. 1695-1698