Plan: Corregir Modelo de Segmentacion

Problemas Identificados

1. PROBLEMA CRITICO - Data Augmentation Desincronizado

Los generadores de imagen y mascara se crean separadamente sin sincronizar:

```
# PROBLEMA ACTUAL - Cada generador usa transformaciones aleatorias DIFERENTES
image_generator = datagen.flow_from_dataframe(brain_df_mask_train, x_col='image_path', ...)
mask_generator = datagen.flow_from_dataframe(brain_df_mask_train, x_col='mask_path', ...)
```

Resultado: La imagen se rota 10 grados a la izquierda, pero la mascara se rota 5 grados a la derecha. El modelo aprende correlaciones incorrectas.

2. Validacion con Augmentation

El generador de validacion usa el mismo `datagen` con augmentation, lo cual corrompe la evaluacion.

3. Metricas Enganosas

- Accuracy: ~96% (alta porque el fondo es >90% de pixeles)
- Dice: ~0.1 (muy bajo, indica que no detecta tumores)

Cambios Necesarios

Archivo: `notebooks/03_Deep_Learning_Segmentation_Model_Development.ipynb`

Celda del ImageDataGenerator (linea ~200):

Separar en dos generadores - uno con augmentation para entrenamiento, uno sin para validacion:

```
# Generador CON augmentation para entrenamiento
train_datagen = ImageDataGenerator(
    rescale=1./255.,
    rotation_range=10,
    width_shift_range=0.05,
    height_shift_range=0.05,
    shear_range=0.05,
    zoom_range=0.1,
    horizontal_flip=True,
    fill_mode='nearest'
)

# Generador SIN augmentation para validacion/test
val_datagen = ImageDataGenerator(rescale=1./255.)
```

Celda de generadores (linea ~220):

Sincronizar transformaciones usando el mismo seed:

```
SEED = 42

def train_generator_fn():
    image_generator = train_datagen.flow_from_dataframe(
        brain_df_mask_train,
        x_col='image_path',
        class_mode=None,
        color_mode="rgb",
        target_size=(256, 256),
        batch_size=BATCH_SIZE,
        seed=SEED,
        shuffle=True
    )

    mask_generator = train_datagen.flow_from_dataframe(
        brain_df_mask_train,
        x_col="mask_path",
        class_mode=None,
        color_mode="grayscale",
        target_size=(256, 256),
        batch_size=BATCH_SIZE,
        seed=SEED,  # MISMO SEED - sincroniza transformaciones
        shuffle=True
    )

    while True:
        img = next(image_generator)
        msk = next(mask_generator)
        # Binarizar mascaras (augmentation puede crear valores intermedios)
        msk = (msk > 0.5).astype(np.float32)
        yield (img, msk)

def val_generator_fn():
    image_generator_val = val_datagen.flow_from_dataframe(
        brain_df_mask_val,
        x_col='image_path',
        class_mode=None,
        color_mode="rgb",
        target_size=(256, 256),
        batch_size=BATCH_SIZE,
        seed=SEED,
        shuffle=False  # Sin shuffle para validacion
    )
```

```
mask_generator_val = val_datagen.flow_from_dataframe(
    brain_df_mask_val,
    x_col="mask_path",
    class_mode=None,
    color_mode="grayscale",
    target_size=(256, 256),
    batch_size=BATCH_SIZE,
    seed=SEED,
    shuffle=False
)

while True:
    img = next(image_generator_val)
    msk = next(mask_generator_val)
    msk = (msk > 0.5).astype(np.float32)
    yield (img, msk)
```

Celda de test_generator (linea ~1346):

Aplicar la misma logica - usar `val_datagen` sin augmentation y con seed sincronizado.

Resultado Esperado

Despues de estos cambios:

- Dice coefficient deberia aumentar significativamente (>0.5 es razonable)
- Las mascaras predichas deberian corresponder con las areas de tumor
- El modelo aprendera las correlaciones correctas entre imagen y mascara