Plan de Estudio: Machine Learning con Python en 1.5 Meses - Versión Mejorada

Módulo 1: Fundamentos y Primeros Pasos (Semanas 1-2)

Días 1-4: 1. Aprendizaje Supervisado y 2. K-Vecinos más Cercanos

- **Teoría**: Conceptos básicos de ML, tipos de aprendizaje. Entender KNN, cómo funciona, cuándo usarlo.
- **Actividad Conceptual**: Clasificación sencilla con un ejemplo de la vida real.
- **Ejercicio Práctico con Python**:
  - Cargar un dataset simple (ej. iris de sklearn).
  - Separar datos en entrenamiento y prueba.
  - Implementar KNeighborsClassifier y hacer una predicción básica.
  - Calcular la precisión del modelo.
  - **Nuevo**: Explorar el efecto de diferentes valores de k en el rendimiento.

Días 5-8: 3. Regresión Ridge y Lasso

- **Teoría**: Introducción a la regresión lineal. Regularización: qué es y por qué es importante. Ridge y Lasso: diferencias y aplicaciones.
- **Actividad Conceptual**: Interpretar coeficientes de un modelo.
- **Ejercicio Práctico con Python**:
  - Cargar un dataset de regresión (ej. boston o diabetes de sklearn).
  - Aplicar Ridge y Lasso regression.
  - Comparar sus resultados usando métricas como el Error Cuadrático Medio (MSE).
  - **Nuevo**: Usar GridSearchCV para encontrar el mejor parámetro de regularización.

Días 9-12: 4. Clasificador Bayesiano

- **Teoría**: Teorema de Bayes y su aplicación en clasificación. Naive Bayes: ventajas y desventajas.
- **Actividad Conceptual**: Ejercicio de probabilidad condicional.
- **Ejercicio Práctico con Python**:
  - Utilizar GaussianNB o MultinomialNB (dependiendo del tipo de datos).
  - Aplicar el clasificador a un dataset de texto (ej. clasificación de spam simplificada) o numérico.
  - Evaluar el rendimiento del modelo.

- **Nuevo**: Comparar diferentes variantes de Naive Bayes según el tipo de datos.

Módulo 2: Clasificadores Avanzados y Ensembles (Semanas 3-4)

Días 13-16: 5. Random Forest y XGBoost

- **Teoría**: Concepto de "ensemble learning". Árboles de decisión. Random Forest: bagging. XGBoost: boosting.
- **Actividad Conceptual**: Comparación de modelos.
- **Ejercicio Práctico con Python**:
  - Implementar RandomForestClassifier y XGBClassifier.
  - Comparar su precisión y tiempos de entrenamiento en un dataset.
  - Explorar la importancia de las características (feature_importances_).
  - **Nuevo**: Implementar validación cruzada para evaluar robustez de los modelos.

Días 17-20: 7. Máquinas de Vectores de Soporte (SVM)

- **Teoría**: Concepto de margen, vectores de soporte. Kernel trick: cómo SVM maneja datos no lineales.
- **Actividad Conceptual**: Visualización de separación de clases.
- **Ejercicio Práctico con Python**:
  - Usar SVC de sklearn.
  - Experimentar con diferentes kernels (linear, poly, rbf) y sus efectos en la frontera de decisión (si es posible visualizarla en 2D).
  - **Nuevo**: Optimizar hiperparámetros C y gamma usando GridSearchCV.

Módulo 3: Redes Neuronales y Temas Específicos (Semanas 5-6)

Días 21-25: 8. Redes Neuronales y Deep Learning

- **Teoría**: Neuronas artificiales, capas, activación. Aprendizaje profundo. Conceptos básicos de entrenamiento. **Técnicas de regularización: dropout, batch normalization**.
- **Actividad Conceptual**: Explicación de una red simple y efectos del overfitting.
- **Ejercicio Práctico con Python**:
  - Construir una red neuronal simple con scikit-learn (MLPClassifier) o, idealmente, con Keras/TensorFlow para un dataset de clasificación (ej. MNIST para números).
  - **Nuevo**: Implementar dropout y early stopping para prevenir overfitting.
  - Entrenar y evaluar la red.

- **Nuevo**: Experimentar con diferentes arquitecturas (número de capas y neuronas).

Días 26-30: 9. Computer Vision

- **Teoría**: Aplicación de ML a imágenes. Redes convolucionales (CNN). **Transfer Learning con modelos preentrenados**. Ejemplos.
- **Actividad Conceptual**: Identificación de componentes en una CNN y concepto de transfer learning.
- **Ejercicio Práctico con Python**:
  - Si ya estamos usando Keras/TensorFlow, construir una CNN básica para un dataset de imágenes (ej. fashion_mnist o cifar10).
  - Mostrar cómo las capas convolucionales extraen características.
  - **Nuevo**: Implementar transfer learning usando un modelo preentrenado (ej. VGG16 o ResNet) para un problema de clasificación de imágenes.

Módulo 4: Evaluación, Reducción de Dimensionalidad y Pipelines (Semanas 7-8 - Ajuste Final)

Días 31-34: 14. Análisis de Componentes Principales (PCA)

- **Teoría**: Concepto de reducción de dimensionalidad. PCA: cómo reduce las dimensiones y por qué es útil.
- **Actividad Conceptual**: ¿Cuándo aplicar PCA?
- **Ejercicio Práctico con Python**:
  - Aplicar PCA a un dataset.
  - Visualizar los datos después de la reducción de dimensionalidad (si se reduce a 2 o 3 componentes).
  - Verificar cómo afecta al rendimiento de un modelo downstream.
  - **Nuevo**: Determinar el número óptimo de componentes usando la varianza explicada acumulada.

Días 35-38: 15. Evaluación de Modelos (Mejorado)

- **Teoría**: Métricas de evaluación para clasificación (precisión, recall, F1-score, ROC-AUC). Métricas para regresión (MAE, MSE, R2). **Validación cruzada estratificada para datasets desbalanceados**.
- **Actividad Conceptual**: Interpretar un informe de métricas y entender cuándo usar cada métrica.
- **Ejercicio Práctico con Python**:
  - Calcular diferentes métricas de evaluación para un modelo de clasificación y uno de regresión.
  - Implementar validación cruzada (cross_val_score o KFold).

- **Nuevo**: Implementar validación cruzada estratificada para datasets desbalanceados.
- Generar una matriz de confusión.
- **Nuevo**: Crear curvas ROC y calcular AUC para problemas multiclase.
- **Nuevo**: Evaluar modelos con técnicas de bootstrap.

Días 39-42: 16. Cadenas de Algoritmos y Pipelines

- **Teoría**: Workflow de Machine Learning. Pipelines: cómo automatizar y estandarizar el proceso. **Prevención de data leakage**.
- **Actividad Conceptual**: Diseñar un pipeline simple en pseudocódigo y identificar posibles fuentes de data leakage.
- **Ejercicio Práctico con Python**:
  - Construir un Pipeline de sklearn que incluya pasos de preprocesamiento (ej. StandardScaler, SimpleImputer) y un modelo.
  - Entrenar y evaluar el pipeline completo.
  - **Nuevo**: Implementar GridSearchCV dentro de un pipeline para optimización de hiperparámetros.
  - **Nuevo**: Crear pipelines para diferentes tipos de variables (numéricas y categóricas) usando ColumnTransformer.

Días 43-45: 17. Apéndice y Repaso General

- **Repaso**: Revisión de temas menos claros, conceptos clave.
- **Actividad Práctica**: Proyecto integrador completo donde combines varios de los conceptos y algoritmos aprendidos:
  - **Nuevo**: Proyecto end-to-end que incluya:
    - Análisis exploratorio de datos (EDA)
    - Preprocesamiento y feature engineering
    - Selección de modelos con validación cruzada
    - Optimización de hiperparámetros
    - Evaluación final con métricas apropiadas
    - Interpretación de resultados y conclusiones

Recursos Necesarios para la Práctica

Software y Librerías:

- **Python**: Asegúrate de tener Python 3.8+ instalado.

- **Librerías esenciales**:
  - `numpy`, `pandas` - manipulación de datos
  - `scikit-learn` - ¡fundamental! algoritmos de ML
  - `matplotlib`, `seaborn` - visualización
  - **Nuevo**: `plotly` - visualizaciones interactivas
- **Para Redes Neuronales**: `tensorflow` o `keras`
- **Nuevos**:
  - `xgboost` - para algoritmos de boosting
  - `imbalanced-learn` - para manejo de datasets desbalanceados
  - `shap` - para interpretabilidad de modelos

Entorno de Desarrollo:

- **Entorno**: Jupyter Notebooks, Google Colab, o VS Code con extensión de Python
- **Nuevo**: Git para control de versiones de tus proyectos

Mejoras Adicionales Implementadas

1. Validación Cruzada Estratificada

- Incluida específicamente para datasets desbalanceados
- Enseña cuándo y por qué usar diferentes tipos de validación cruzada

2. Técnicas de Regularización en Deep Learning

- Dropout y early stopping para prevenir overfitting
- Batch normalization para estabilizar el entrenamiento

3. Transfer Learning

- Fundamental en computer vision moderna
- Uso práctico de modelos preentrenados

4. Evaluación Robusta

- Bootstrap sampling
- Curvas ROC para multiclase
- Métricas específicas para problemas desbalanceados

5. Pipelines Avanzados

- ColumnTransformer para diferentes tipos de variables
- GridSearchCV integrado en pipelines
- Prevención de data leakage

6. **Proyecto Integrador Completo**

- Experiencia end-to-end realista
- Incluye todas las fases de un proyecto de ML real

Este plan mejorado mantiene la estructura sólida original mientras incorpora técnicas y conceptos modernos esenciales para un practicante de ML actual.