**Predicting Employee Departures Using Machine Learning: Model Development**

**1. تقسيم البيانات (Train-Test Split)**

**لماذا؟**

* لفصل البيانات إلى مجموعة تدريب لتعليم النموذج، ومجموعة اختبار لتقييم أداء النموذج على بيانات غير مألوفة.

**كيف؟**

* قم بتقسيم البيانات إلى جزأين:
  + **80%** بيانات تدريب (Training Data).
  + **20%** بيانات اختبار (Testing Data).

**الكود باستخدام Python:**

from sklearn.model\_selection import train\_test\_split

# افترض أن df هو اسم DataFrame الخاص بك

X = df.drop('Attrition', axis=1) # حذف العمود المستهدف

y = df['Attrition'] # العمود المستهدف

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

**2. اختيار النماذج (Model Selection)**

**أفضل النماذج لهذه المشكلة (تصنيف):**

1. **Logistic Regression:** بسيط وسريع، مفيد كبداية لفهم العلاقات بين المتغيرات.
2. **Decision Tree:** شجرة اتخاذ القرار لتفسير النتائج بسهولة.
3. **Random Forest:** يعمل بشكل أفضل للبيانات غير الخطية.
4. **Gradient Boosting (مثل XGBoost):** نموذج قوي ومناسب للبيانات ذات العلاقات المعقدة.

**ما الذي يمكن أن تفعله؟**

* ابدأ باستخدام نموذج بسيط (مثل Logistic Regression)، ثم جرب نماذج أكثر تقدمًا لتحديد الأفضل.

**3. تدريب النماذج (Training the Model)**

**كيف يتم التدريب؟**

* يتم تمرير بيانات التدريب إلى النموذج ليتمكن من تعلم الأنماط.

**الكود باستخدام Logistic Regression كمثال:**

from sklearn.linear\_model import LogisticRegression

# إنشاء النموذج

model = LogisticRegression()

# تدريب النموذج على بيانات التدريب

model.fit(X\_train, y\_train)

**4. تقييم الأداء الأساسي (Baseline Performance)**

**لماذا؟**

* لفهم الأداء الأولي للنموذج ومقارنته بنماذج أكثر تعقيدًا لاحقًا.

**كيف؟**

* استخدم مجموعة الاختبار للتنبؤ وقم بحساب الدقة (Accuracy).

**الكود:**

from sklearn.metrics import accuracy\_score

# التنبؤ باستخدام مجموعة الاختبار

y\_pred = model.predict(X\_test)

# حساب الدقة

accuracy = accuracy\_score(y\_test, y\_pred)

print(f"Accuracy: {accuracy}")

**5. تحسين النموذج (Hyperparameter Tuning)**

**لماذا؟**

* النماذج تحتوي على معلمات تحتاج إلى ضبط لتحسين الأداء.

**كيف؟**

* **Grid Search:** تجربة كل تركيبة ممكنة من المعلمات.
* **Random Search:** تجربة تركيبات عشوائية من المعلمات.

**مثال مع Random Forest:**

from sklearn.ensemble import RandomForestClassifier

from sklearn.model\_selection import GridSearchCV

# إنشاء النموذج

rf\_model = RandomForestClassifier(random\_state=42)

# إعداد الشبكة لتجربة المعلمات

param\_grid = {

'n\_estimators': [50, 100, 200],

'max\_depth': [None, 10, 20],

'min\_samples\_split': [2, 5, 10]

}

# استخدام GridSearchCV

grid\_search = GridSearchCV(rf\_model, param\_grid, cv=3, scoring='accuracy')

grid\_search.fit(X\_train, y\_train)

# أفضل معلمات

best\_params = grid\_search.best\_params\_

print(f"Best Parameters: {best\_params}")

**6. التقييم الشامل (Comprehensive Evaluation)**

**ما الذي تفعله؟**

* قياس الأداء باستخدام مقاييس متعددة مثل:
  + **Precision:** دقة التنبؤات الإيجابية.
  + **Recall:** قدرة النموذج على اكتشاف الإيجابيات الحقيقية.
  + **F1-Score:** متوسط مرجح بين Precision وRecall.
  + **Confusion Matrix:** لتحليل الأخطاء.

**الكود:**

from sklearn.metrics import classification\_report, confusion\_matrix

# تقرير الأداء

print(classification\_report(y\_test, y\_pred))

# مصفوفة الارتباك

conf\_matrix = confusion\_matrix(y\_test, y\_pred)

print(conf\_matrix)

**7. مقارنة النماذج (Model Comparison)**

**ما الذي يمكن فعله؟**

* جرب العديد من النماذج (مثل Logistic Regression وRandom Forest وXGBoost).
* استخدم نفس المقاييس لتحديد النموذج الأفضل.

**8. تحليل النتائج (Insights from the Model)**

**ما الذي يمكن استنتاجه؟**

* حدد الميزات الأكثر تأثيرًا باستخدام:
  + **Feature Importance** (لـ Random Forest).
  + **Coefficients** (لـ Logistic Regression).

**مثال مع Random Forest:**

import matplotlib.pyplot as plt

# أهمية الميزات

importances = grid\_search.best\_estimator\_.feature\_importances\_

plt.barh(X\_train.columns, importances)

plt.xlabel('Importance')

plt.ylabel('Features')

plt.title('Feature Importance')

plt.show()

**9. نشر النموذج (Model Deployment)**

**كيف؟**

* حول النموذج إلى خدمة ويب باستخدام Flask أو FastAPI.
* أنشئ واجهة يمكنها قبول إدخال بيانات موظف وإرجاع التنبؤ.

**نصيحة إضافية**

* إذا أردت شرحًا أكثر عمقًا لأي خطوة أو مساعدة بالكود، يمكنك طلب ذلك مباشرة!