```
import os
import pickle
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report

# Güncel dataset kökü
DATASET_DIR = "dataset_cleaned"

texts = []
labels = []

# Kaç dosya kullanılacağını belirle
MAX_LIMITS = {
  "notifications": 3000,
  "reports": 3000,
  # contracts ve invoices için limit yok, hepsini kullan
}

# Her klasör için sayacı tut
counters = {}
```

```
for root, dirs, files in os.walk(DATASET_DIR):
  for file in files:
     if file.endswith(".txt"):
        label = os.path.basename(root)

        # Kategoriye göre limiti uygula
        if label in MAX_LIMITS:
           counters.setdefault(label, 0)
           if counters[label] >= MAX_LIMITS[label]:
              continue  # bu sınıf için limit doldu
           counters[label] += 1

        file_path = os.path.join(root, file)
        with open(file_path, "r", encoding="utf-8") as f:
           texts.append(f.read())
           labels.append(label)

print(f"Toplam {len(texts)} dosya yüklendi.")
print(f"Kullanılan örnek sayısı: {counters}")

# Veri setini ayır
X_train, X_test, y_train, y_test = train_test_split(
  texts, labels, test_size=0.2, random_state=42
)

# TF-IDF ve sınıflandırıcı
vectorizer = TfidfVectorizer(stop_words="english", max_features=5000)
```

```
X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)

classifier = LogisticRegression(max_iter=1000)
classifier.fit(X_train_vec, y_train)

y_pred = classifier.predict(X_test_vec)
print(classification_report(y_test, y_pred))

# Kaydet
os.makedirs("model", exist_ok=True)
with open("model/trained_model.pkl", "wb") as f:
  pickle.dump((vectorizer, classifier), f)

print("✅ Model ve vectorizer kaydedildi: model/trained_model.pkl")
```