**Digit Recognizer Projesi Raporu**

**ALPEREN MUTLU**

**202113172070 (İ.Ö)**

**Proje Amacı:**

Bu projede, **MNIST veri setine** benzeyen **Digit Recognizer** veri setini kullanarak, el yazısı rakamları (0-9) sınıflandırmak amaçlanmıştır. Modelin doğruluğu, çeşitli makine öğrenimi algoritmaları ile test edilmiştir. Sonuçlar, **Kaggle** platformunda belirtilen formatta sunulmuştur.

**Kullanılan Teknolojiler ve Kütüphaneler:**

1. **Python**: Proje, Python programlama dili kullanılarak geliştirilmiştir. Python, veri işleme, modelleme ve görselleştirme işlemleri için geniş bir kütüphane ekosistemine sahiptir.
2. **Jupyter Notebook**: Veri analizi, model geliştirme ve sonuç görselleştirmeleri için interaktif bir ortam sağlamak amacıyla Jupyter Notebook kullanılmıştır. Bu, kodların hücreler halinde yazılmasını ve kolayca çalıştırılmasını sağlar.
3. **Pandas**: Veri yükleme ve işleme için **Pandas** kütüphanesi kullanılmıştır. CSV formatındaki veri setleri **Pandas DataFrame** yapısına aktarılmış, bu yapı üzerinden veriye erişim ve manipülasyon işlemleri yapılmıştır.
4. **NumPy**: Veri setinin sayısal işleme ve manipülasyonu için **NumPy** kütüphanesi kullanılmıştır. NumPy, çok boyutlu dizilerle hızlı hesaplama yapabilme yeteneği sağlar.
5. **Matplotlib**: Veri görselleştirmeleri ve model sonuçlarının grafiksel olarak gösterilmesi için **Matplotlib** kütüphanesi kullanılmıştır. Özellikle resimlerin görselleştirilmesinde faydalı olmuştur.
6. **Scikit-learn**: Modelleme ve değerlendirme için **Scikit-learn** kütüphanesi kullanılmıştır. Modelin eğitilmesi, doğrulama yapılması ve sonuçların değerlendirilmesi bu kütüphane aracılığıyla gerçekleştirilmiştir.

**Kullanılan Makine Öğrenimi Algoritmaları:**

Proje için **Lojistik Regresyon** (Logistic Regression) algoritması seçilmiştir. Bu algoritma, sınıflandırma problemleri için yaygın olarak kullanılan ve hızlı bir şekilde çözümler elde edilebilen bir yöntemdir.

* **Lojistik Regresyon (Logistic Regression)**:
  + Lojistik Regresyon, doğrusal bir modeldir ve verinin sınıflarını (0-9 rakamları) tahmin etmek için kullanılır. Model, **softmax fonksiyonu** kullanarak her bir sınıf için olasılık değerleri üretir ve en yüksek olasılığa sahip olan sınıfı tahmin olarak verir.
  + **Multi-class sınıflandırma** için max\_iter=1000 parametresi ile modelin daha fazla iterasyon yaparak daha iyi sonuçlar elde etmesi sağlanmıştır. solver='lbfgs' parametresi ise daha iyi performans için kullanılan optimizasyon algoritmasını belirler.

**Adımlar ve Kodların İşlevleri:**

1. **Veri Setini Yükleme ve İnceleme**:
   * Eğitim (train.csv) ve test (test.csv) veri setleri, **Pandas** kütüphanesi kullanılarak yüklenmiştir.
   * İlk hücrede veri setinin boyutları ve ilk 5 satırı görüntülenerek veri hakkında bilgi edinilmiştir.

train\_data = pd.read\_csv('train.csv') # Eğitim veri seti

test\_data = pd.read\_csv('test.csv') # Test veri seti

1. **Veri Görselleştirme**:
   * **Matplotlib** kullanılarak veri setindeki ilk örneğin görselleştirilmesi yapılmıştır. Bu adım, verinin doğru bir şekilde temsil edilip edilmediğini anlamak için önemlidir.

plt.imshow(train\_data.iloc[0, 1:].values.reshape(28, 28), cmap='gray')

plt.title(f"Label: {train\_data.iloc[0, 0]}")

plt.show()

1. **Veri Hazırlığı**:
   * Eğitim veri seti özellikler (görüntü pikselleri) ve etiketlerden (rakamlar) ayrılmıştır.
   * Veriler **train\_test\_split** fonksiyonu ile eğitim ve doğrulama setlerine ayrılmıştır.
   * Görüntü verileri 0-255 arasında olduğu için, modelin daha verimli çalışabilmesi adına bu veriler **normalize** edilmiştir (0 ile 1 arasına dönüştürülmüştür).

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

X\_train = X\_train / 255.0

X\_val = X\_val / 255.0

1. **Model Eğitimi**:
   * **Lojistik Regresyon** modeli oluşturulmuş ve eğitim verisi ile eğitilmiştir. Bu adımda model, verilen veri üzerinden öğrenme sürecine başlamıştır.

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

1. **Model Değerlendirme**:
   * Modelin doğruluğu, doğrulama verisi üzerinde yapılan tahminlerle değerlendirilmiştir. **Scikit-learn**'ün accuracy\_score fonksiyonu ile doğruluk oranı hesaplanmış ve classification\_report ile daha detaylı bir değerlendirme yapılmıştır.

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

print("Doğruluk Skoru:", accuracy\_score(y\_val, y\_pred))

print("Sınıflandırma Raporu:\n", classification\_report(y\_val, y\_pred))

1. **Test Verisi Üzerinde Tahmin ve Kaggle Formatına Kaydetme**:
   * **Test veri seti** üzerinde tahminler yapılmış ve sonuçlar Kaggle formatında bir CSV dosyasına kaydedilmiştir.

submission = pd.DataFrame({'ImageId': range(1, len(test\_predictions) + 1), 'Label': test\_predictions})

submission.to\_csv('submission.csv', index=False)

**Sonuçlar ve Değerlendirme:**

Modelin doğruluğu, eğitim ve doğrulama verileri üzerinde yapılan değerlendirme ile hesaplanmıştır. Lojistik regresyonun temel güçlü yönlerinden biri, doğrusal sınıflandırmalarda hızlı ve etkili sonuçlar sunmasıdır. Ancak, daha karmaşık ve doğrusal olmayan veri yapıları için daha güçlü algoritmalar kullanılabilir.

Modelin son tahminleri **submission.csv** formatında kaydedilmiştir.

**Sonuç:**

Bu proje, el yazısı rakamları sınıflandırmak için **Lojistik Regresyon** algoritması kullanan temel bir sınıflandırma modelini içermektedir. Proje boyunca veri yükleme, görselleştirme, eğitim, tahmin yapma ve sonuçları kaydetme adımları takip edilmiştir. Bu projeyi geliştirerek daha karmaşık algoritmalar ve derin öğrenme modelleri de kullanılabilir.

Projeden bazı görseller:







