Digit Recognizer

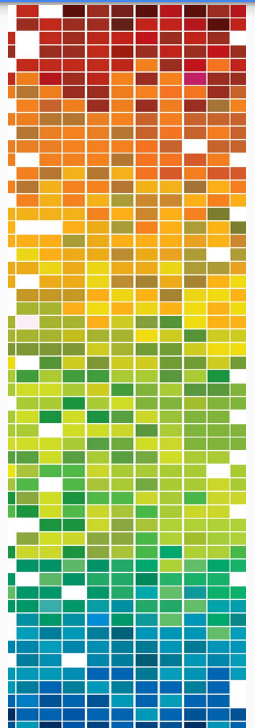
Andy Xie, Kevin Tong, Simon Bach, Angelo Onato, Christopher Ortiz

Project Description, Details, Goals

- Project is sourced from a Kaggle competition
- We shall be learning computer vision fundamentals with the famous MNIST (Modified National Institute of Standards and Technology) data
- We are to use machine learning algorithms to correctly identify digits from a dataset of tens of thousands of handwritten images
- We are to practice computer vision fundamentals including simple neural networks and classification methods such as SVM and KNN
- The goal is to take an image of a handwritten single digit, and determine what that digit is. For every image ID in our test set, we should predict the correct label

Data Details

- Training and testing data
  - Training: 42000 images
  - Testing: 28000 images
- Gray-scale images of 0 to 9
  - Hand drawn
- 784 pixels for each image
  - Pixel value range 0-255



KNN

- The KNN method of classification is one of the simplest methods in machine learning its most basic level of machine learning, it is essentially classification by finding the most similar data points in the training data, and making an educated guess based on their classifications

```
#K = 3 KNN Accuracy
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=42)
k = 3
knn = KNeighborsClassifier(n_neighbors=k)
knn.fit(X_train, y_train)
y_predict = knn.predict(X_test)
accuracy = accuracy_score(y_test, y_predict)
print("KNN (with k=3) Accuracy: ",accuracy)
```

```
KNN (with k=3) Accuracy:  0.9676190476190476
```

Random Forest

- Random forest is a supervised learning algorithm
- It creates a forest and makes tries to make it as random as possible which can be used for both classification and regression problems, used by many current machine learning systems

```
#Random-Forest estimators = 100
my_RandomForest = RandomForestClassifier(n_estimators = 100, bootstrap = True, random_state=3)
my_RandomForest.fit(X_train, y_train)
y_predict = my_RandomForest.predict(X_test)
score = accuracy_score(y_test, y_predict)
print("Random Forest Accuracy: ",score)
```

```
Random Forest Accuracy:  0.9612698412698413
```

Decision Tree

- Decision Trees are a very commonly used tool in data mining to achieve a particular goal or target at each node, widely used in machine learning
- Most of the time is training with the “bagging” method
- The idea of the “bagging” method is a combination of learning models to get a better overall result

```
#Decision Tree Randon State 5
my_decisiontree = DecisionTreeClassifier(random_state=5)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=4)
my_decisiontree.fit(X_train, y_train)
y_predict = my_decisiontree.predict(X_test)
score = accuracy_score(y_test, y_predict)
print("Decision Tree Randon State 5 Accuracy: ",score)
```

```
Decision Tree Randon State 5 Accuracy:  0.849047619047619
```

Logistic Regression

- Logistic regression is a very powerful algorithm it looks at the relationship between a training variables and testing variables
- Provided the right representation of the labels and features it can return very accurate results, which makes it a very accurate machine learning algorithm

```
# Logistic Regression
my_logreg = LogisticRegression()
my_logreg.fit(X, y)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=4)
my_logreg.fit(X_train, y_train)
y_predict = my_logreg.predict(X_test)
score = accuracy_score(y_test, y_predict)
print("Logistic Regression Accuracy: ", score)
```

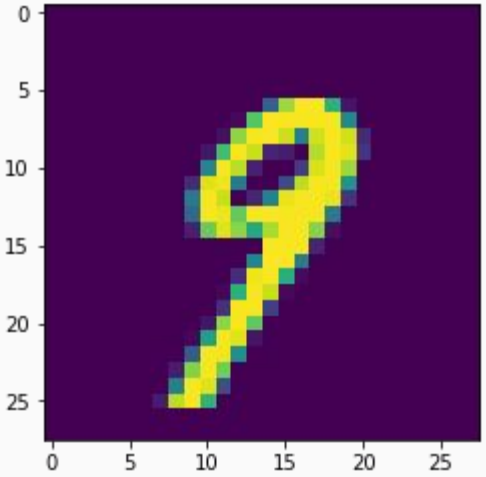
```
Logistic Regression Accuracy:  0.9045714285714286
```

Results

- Among the algorithms we used, there were a few that were very close to each other in terms of results
- Those algorithms were
  - Random forest with n-estimators of 100, 700, and 500 with bagging
  - KNN
- The best one ended up being KNN with 96.76% accuracy
  - k was 3, test size of 0.1, random state of 42
- The worst one ended up being the decision tree with 85.32% accuracy
  - test size of 0.25, random state of 4

Visualization

Picking a random number



Visualization

Plotting 1-10 from the data

Visualization

Plotting our number data