# CS 6375 HW 2

Neural Networks, K-nearest neighbors, and SVM

Parker Whitehead

pmw180000@utdallas.edu

In [ ]:
```
import numpy as np
import pandas as pd
```

In [ ]:
```
from sklearn.datasets import fetch_openml
# Load data from https://www.openml.org/d/554
X, y = fetch_openml('mnist_784', version=1, return_X_y=True)
X = X / 255.

# rescale the data, use the traditional train/test split
# (60K: Train) and (10K: Test)
X_train, X_test = X[:60000], X[60000:]
y_train, y_test = y[:60000], y[60000:]
```

## SKLearn SVM

Parameters Tested:

```
    All combinations of:

    kernel_values: linear, poly, rbf, and sigmoid.

    penalty values: 0.5, 1, and 2.
```

Total tested: 10

In [ ]:
```
from sklearn.svm import SVC
import gc

kernel_values = ['linear','poly','rbf','sigmoid']
penalty_values = [.5,1,2]

for ker in kernel_values:
    for pen in penalty_values:
        svm = SVC(kernel=ker,C=pen)
        gc.collect()
        svm.fit(X_train,y_train)
        results = svm.predict(X_test)
        err = 1 - (sum([int(prediction == actual) for prediction, actual in zip(result
        print(f'Kernel used: {ker}')
```

```
        print(f'Penalty  used: {pen}')
        print(f'Error Rate: {err:.5f}\n')
```

```
Kernel used: linear
Penalty  used: 0.5
Error Rate: 0.05700

Kernel used: linear
Penalty  used: 1
Error Rate: 0.05960

Kernel used: linear
Penalty  used: 2
Error Rate: 0.06190

Kernel used: poly
Penalty  used: 0.5
Error Rate: 0.02690

Kernel used: poly
Penalty  used: 1
Error Rate: 0.02290

Kernel used: poly
Penalty  used: 2
Error Rate: 0.02150

Kernel used: rbf
Penalty  used: 0.5
Error Rate: 0.02410

Kernel used: rbf
Penalty  used: 1
Error Rate: 0.02080

Kernel used: rbf
Penalty  used: 2
Error Rate: 0.01690

Kernel used: sigmoid
Penalty  used: 0.5
Error Rate: 0.21030

Kernel used: sigmoid
Penalty  used: 1
Error Rate: 0.22410

Kernel used: sigmoid
Penalty  used: 2
Error Rate: 0.22970
```

## MLP Classifier

Parameters Tested:

```
    All combinations of:

    solvers: adam, sgd, and lbfgs
```

activations: identity, logistic, tanh, and relu

Total Tested: 12

In [ ]:
```
from sklearn.neural_network import MLPClassifier

solvers = ['adam','sgd','lbfgs']
activations = ['identity','logistic','tanh','relu']

for sol in solvers:
    for act in activations:
        mlp = MLPClassifier(solver=sol,activation=act)
        gc.collect()
        mlp.fit(X_train,y_train)
        results = mlp.predict(X_test)
        err = 1 - (sum([int(prediction == actual) for prediction, actual in zip(result
        print(f'Solver used: {sol}')
        print(f'Activation function used: {act}')
        print(f'Error Rate: {err:.5f}\n')
```

```
Solver used: adam
Activation function used: identity
Error Rate: 0.07630

Solver used: adam
Activation function used: logistic
Error Rate: 0.02310

Solver used: adam
Activation function used: tanh
Error Rate: 0.02180

Solver used: adam
Activation function used: relu
Error Rate: 0.02380

c:\Users\pmw99\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\neur
al_network\_multilayer_perceptron.py:692: ConvergenceWarning: Stochastic Optimizer: M
aximum iterations (200) reached and the optimization hasn't converged yet.
  warnings.warn(
Solver used: sgd
Activation function used: identity
Error Rate: 0.07550

c:\Users\pmw99\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\neur
al_network\_multilayer_perceptron.py:692: ConvergenceWarning: Stochastic Optimizer: M
aximum iterations (200) reached and the optimization hasn't converged yet.
  warnings.warn(
Solver used: sgd
Activation function used: logistic
Error Rate: 0.06420
```

```
c:\Users\pmw99\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:692: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and the optimization hasn't converged yet.
  warnings.warn(
Solver used: sgd
Activation function used: tanh
Error Rate: 0.03230

c:\Users\pmw99\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:692: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and the optimization hasn't converged yet.
  warnings.warn(
Solver used: sgd
Activation function used: relu
Error Rate: 0.02970

c:\Users\pmw99\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:549: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
  self.n_iter_ = _check_optimize_result("lbfgs", opt_res, self.max_iter)
Solver used: lbfgs
Activation function used: identity
Error Rate: 0.07430

Solver used: lbfgs
Activation function used: logistic
Error Rate: 0.02680

Solver used: lbfgs
Activation function used: tanh
Error Rate: 0.02760

Solver used: lbfgs
Activation function used: relu
Error Rate: 0.02440
```

## K-nearest Neighbors

Parameters Tested:

```
    All combinations of:

    num_neighbors: 3, 5, 7, 11, and 21

    weight_options: uniform and distance
```

Total tested: 10

In [ ]:
```
from random import uniform
from sklearn.neighbors import KNeighborsClassifier
import gc

num_neighbors = [3,5,7,11,21]
weight_options = ['uniform','distance']

for wei in weight_options:
    for nei in num_neighbors:
        knn = KNeighborsClassifier(n_neighbors=nei,weights=wei)
        gc.collect()
        knn.fit(X_train,y_train)
        results = knn.predict(X_test)
        err = 1 - (sum([int(prediction == actual) for prediction, actual in zip(result
        print(f'Number of neighbors used: {nei}')
        print(f'Weight option used: {wei}')
        print(f'Error Rate: {err:.5f}\n')
```

```
Number of neighbors used: 3
Weight option used: uniform
Error Rate: 0.02950

Number of neighbors used: 5
Weight option used: uniform
Error Rate: 0.03120

Number of neighbors used: 7
Weight option used: uniform
Error Rate: 0.03060

Number of neighbors used: 11
Weight option used: uniform
Error Rate: 0.03320

Number of neighbors used: 21
Weight option used: uniform
Error Rate: 0.03700

Number of neighbors used: 3
Weight option used: distance
Error Rate: 0.02830

Number of neighbors used: 5
Weight option used: distance
Error Rate: 0.03090

Number of neighbors used: 7
Weight option used: distance
Error Rate: 0.03000

Number of neighbors used: 11
Weight option used: distance
Error Rate: 0.03220

Number of neighbors used: 21
Weight option used: distance
Error Rate: 0.03680
```

## Best Error Rate for Each of the Three Classifiers:

**SVM Classifier:**

```
Kernel used: rbf
Penalty  used: 2
Error Rate: 0.0169
```

**MLP Classifier:**

```
Solver used: adam
Activation function used: tanh
Error Rate: 0.0218
```

**K-Nearest Neighbors Classifier:**

```
Number of neighbors used: 3
Weight option used: distance
Error Rate: 0.0283
```