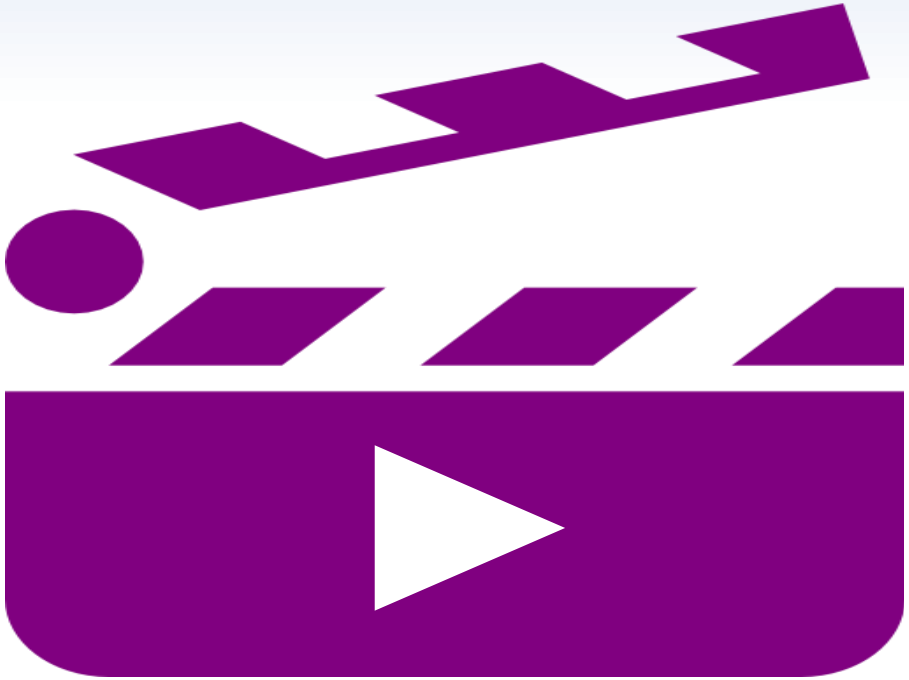
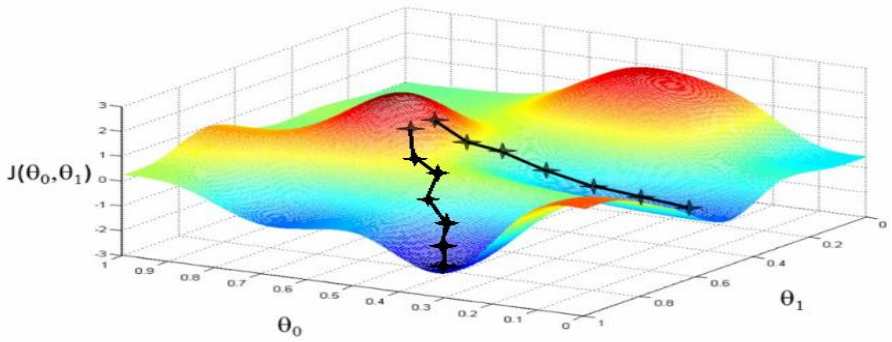


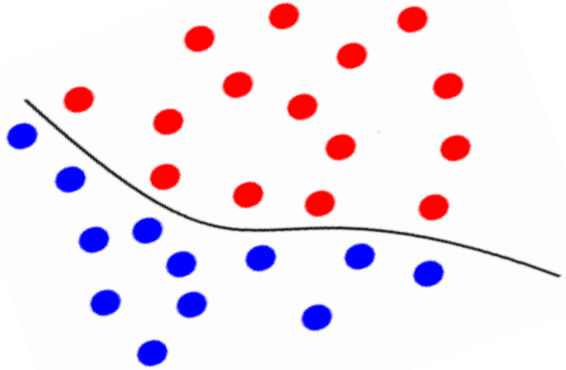
לא לשכוח להפעיל הקלטה!

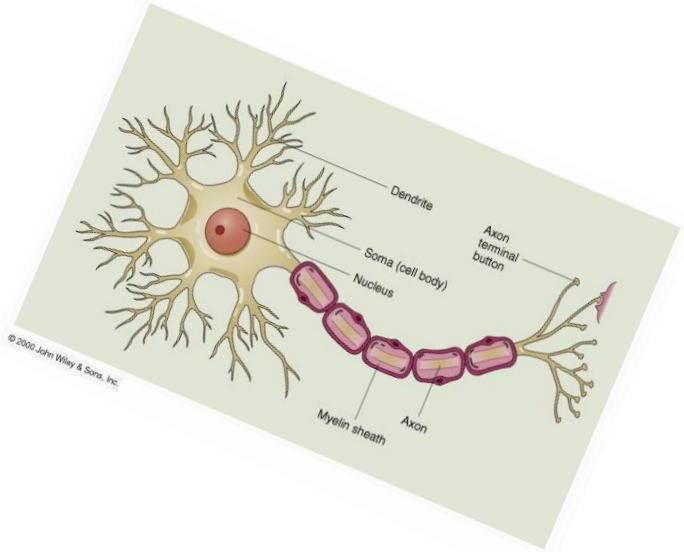
מכון ויצמן למדע
WEIZMANN INSTITUTE OF SCIENCE

מכון דוידסון לחינוך מדעי
DAVIDSON INSTITUTE OF SCIENCE EDUCATION

קלסיפיקציה ברשתות נוירונים







שיעור סינכרוני

מעל"ה – מדע חישובי פיזיקה

מה אנחנו צריכים כבר לדעת?

- למידה מפוקחת
- רגרסיה לוגיסטית ופונקציית מעבר
- מבנה הרשת
- רשת נוירונים לבעיות קלסיפיקציה עם מאפיינים סטטיסטיים

מה נלמד בשיעור זה?

- ✓ ייצוג של ערכי תמונות כערכים מספריים בקלט
- ✓ תהליך אימון הרשת עמוקה לסיווג תמונות

- **רגרסיה** – הערכת הפונקציה בין הקלט לפלט.
  - הפלט הוא מספר ממשי (או כמה מספרים).
- **קלסיפיקציה (סיווג)** – ניבוי לאיזו קבוצה שייכת כל דוגמא.
  - הפלט הוא התוית של הקבוצה.

ייצוג של הערכים בקלט

- ייצוג תמונות כמאפיינים מספריים

Mnist data-set

הנתונים הם תמונות
של כתב יד
ספרות 0-9
סה"כ – 70,000
דוגמאות

Mnist data-set

יבוא הספרייה

```
from tensorflow.keras.datasets import mnist
```

יבוא נתונים לסט האימון וסט הטסט

```
# Load the MNIST dataset
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()

# Display the shape of training data
print("Training data shape:", train_images.shape)
print("Training labels shape:", train_labels.shape)
```

60,000 דוגמאות בסט האימון
כל דוגמא היא מטריצה
28 * 28

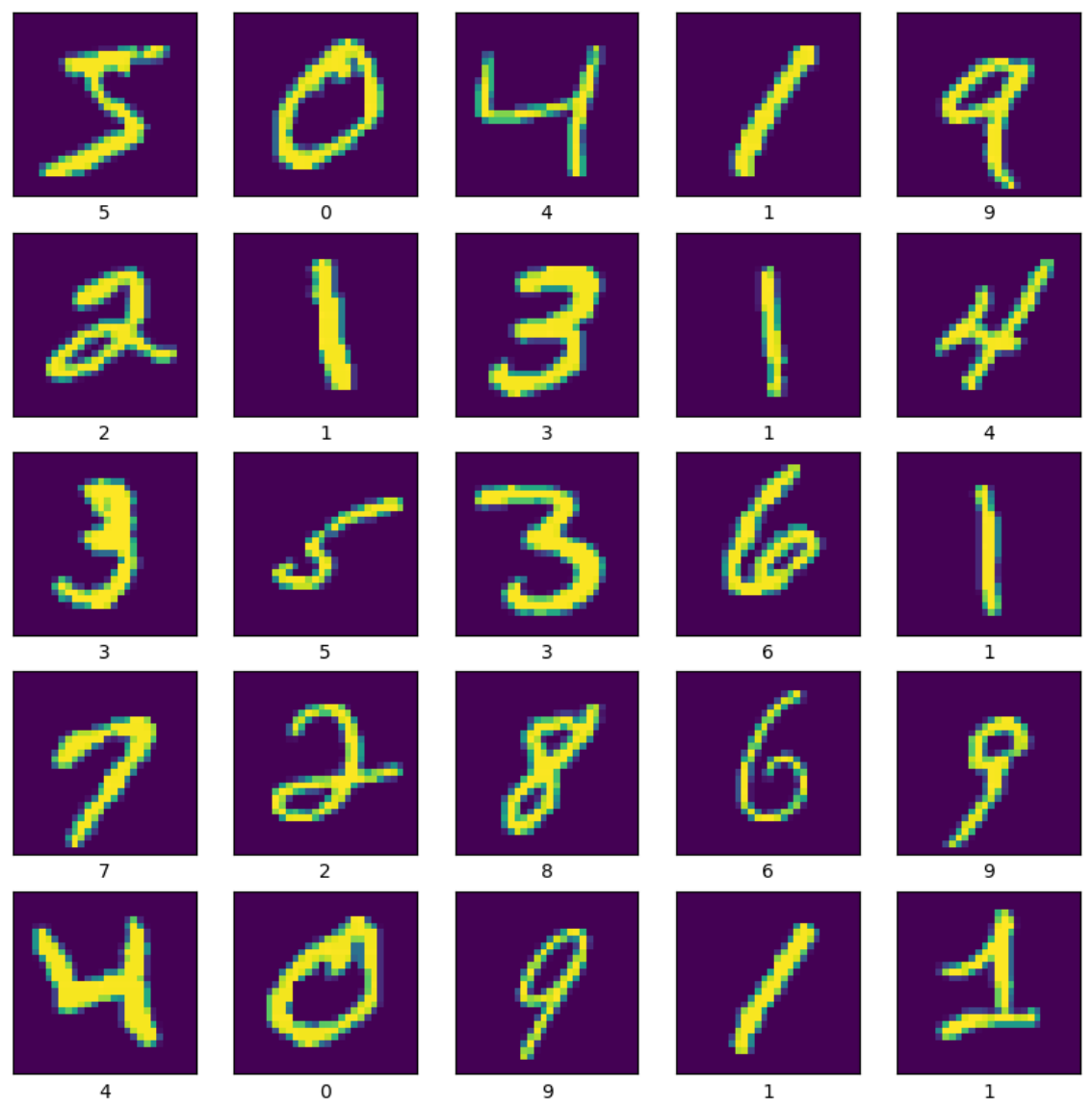
```
Training data shape: (60000, 28, 28)
Training labels shape: (60000,)
```

נירמול הנתונים

```
# Normalize the images to have values between 0 and 1
train_images = train_images / 255.0
test_images = test_images / 255.0
```

Mnist data-set

```
plt.figure(figsize=(10,10))
for i in range(25):
  plt.subplot(5,5,i+1)
  plt.xticks([])
  plt.yticks([])
  plt.grid(False)
  plt.imshow(train_images[i])
  plt.xlabel(train_labels[i])
```



fit & Compile & ANN sequential model - אימון המודל

```python
# Define the model
model = models.Sequential([
  layers.Flatten(input_shape=(28, 28)),
  layers.Dense(128, activation='relu'),
  layers.Dense(10, activation='softmax')
])
```

```python
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

```python
history = model.fit(train_images, train_labels, epochs=10, validation_split=0.2)
```

תוצאת אימון המודל

```
history.history
```

```
{'loss': [0.2863345444202423,
  0.12704862654209137,
  0.08615195006132126,
  0.06442221254110336,
  0.04853751137852669,
  0.0369839072227478,
  0.028473563492298126,
  0.024302255362272263,
  0.019546745344996452,
  0.015179120004177094],
 'accuracy': [0.9183541536331177,
  0.9622083306312561,
  0.973479151725769,
  0.9805416464805603,
  0.9848333597183228,
  0.9882291555404663,
  0.9912499785423279,
  0.9925833344459534,
  0.9942916631698608,
  0.9953749775886536],
 'val_loss': [0.14866235852241516,
  0.11108218133449554,
  0.0964551791548729,
  0.09157851338386536,
  0.08225511759519577,
  0.08600004017353058,
  0.0911000445485115,
  0.08661239594221115,
  0.09789513051509857,
  0.09421538561582565],
 'val_accuracy': [0.9575833082199097,
  0.9664999842643738,
  0.971916675567627,
  0.971833348274231,
  0.9760833382606506,
  0.9755833148956299,
  0.9760000109672546,
  0.9766666889190674,
  0.9755833148956299,
  0.9764166474342346]}
```

val ו - accuracy
לסט האימון ולסט הבדיקה

תוצאת אימון המודל - prediction

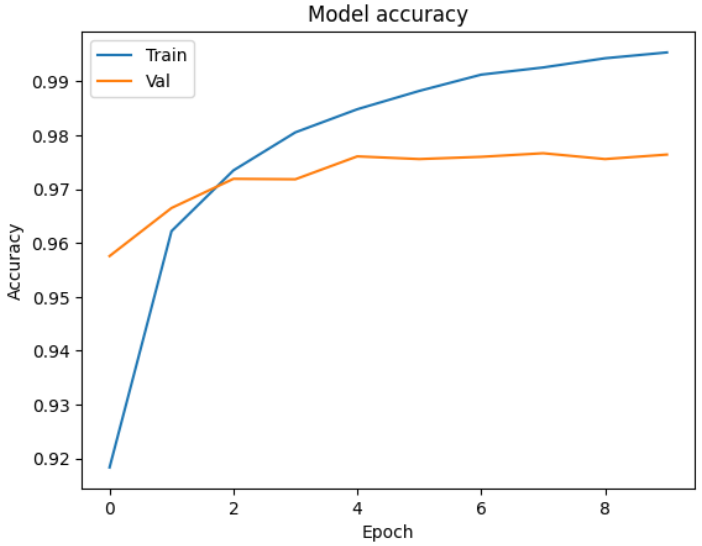
```
[26] # Predict the probabilities for each class
     y_pred_probs = model.predict(test_images)
     print(y_pred_probs.shape)
     print(y_pred_probs[:3])
     # Convert probabilities to class labels
     y_pred = np.argmax(y_pred_probs, axis=1)
     print(y_pred.shape)
     print(y_pred[:3])
```

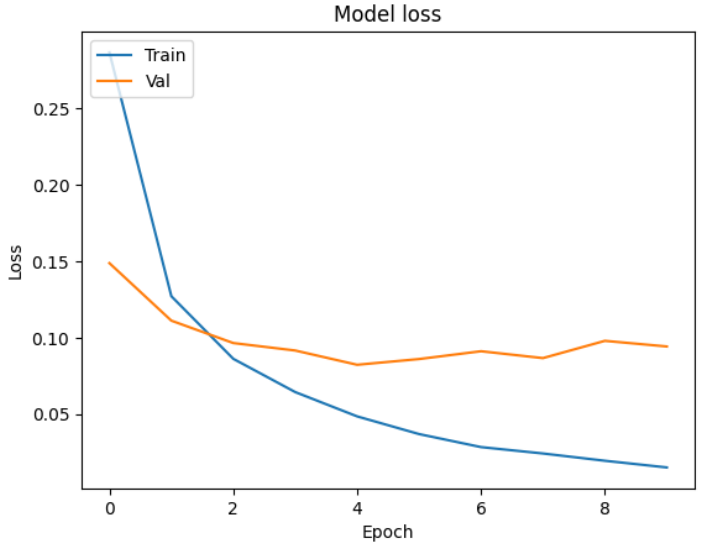
```
313/313 [==============================] - 1s 4ms/step
(10000, 10)
[[0.        0.        0.        0.000008 0.        0.        0.        0.999992 0.        0.        ]
 [0.        0.        0.999997 0.        0.        0.        0.        0.        0.000003 0.        ]
 [0.        0.999612 0.000045 0.        0.000002 0.        0.000002 0.000078 0.00026  0.        ]]
(10000,)
[7 2 1]
```

תוצאת אימון המודל - ויזואליזציה

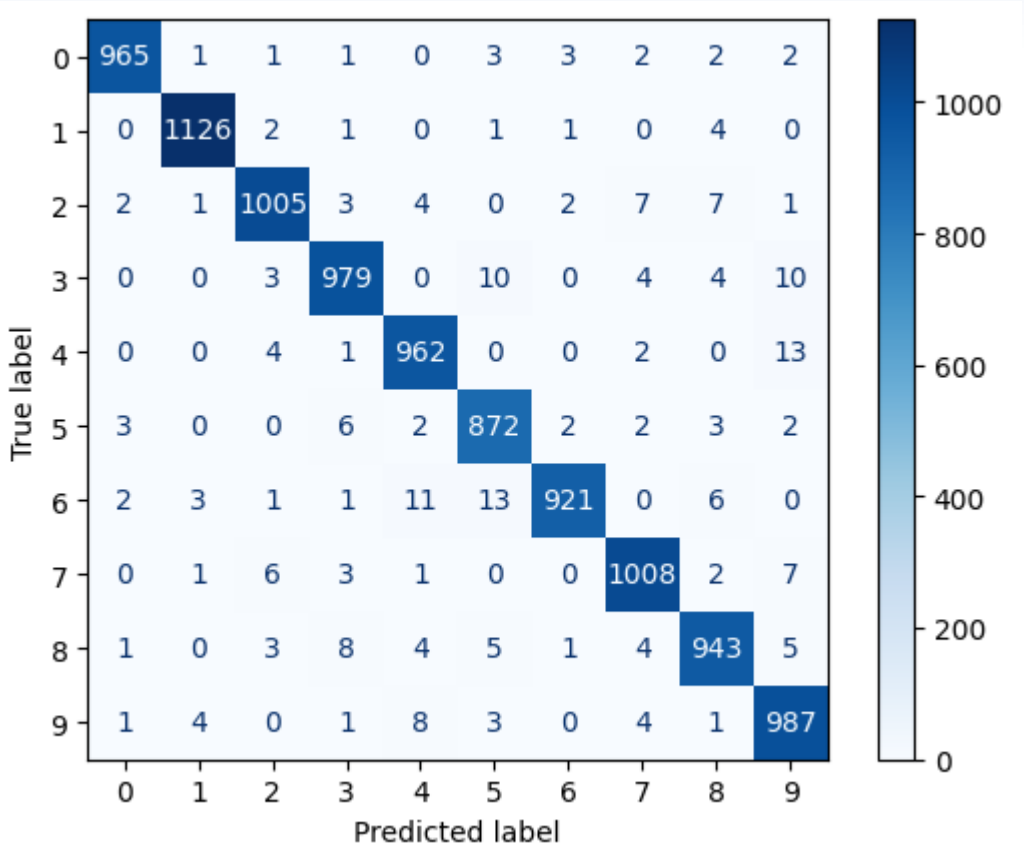
```python
# Plot training & validation accuracy values
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper left')
plt.show()

# Plot training & validation loss values
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper left')
plt.show()
```





תוצאת אימון המודל – Confusion matrix display



```
# Display the confusion matrix
ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=np.unique(test_labels)).plot(values_format='d', cmap='Blues')
plt.show()
```

תוצאת אימון המודל – Classification report

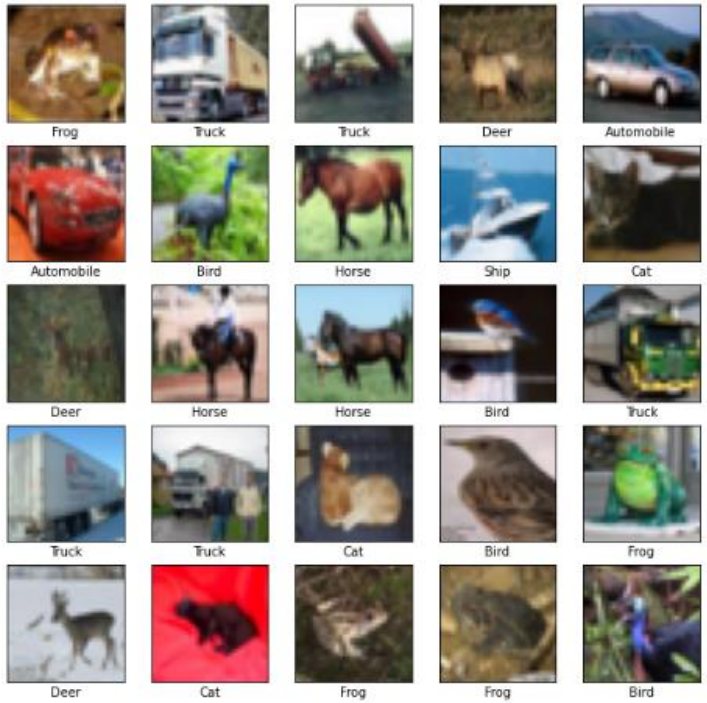
```
print(classification_report(y_true=test_labels,y_pred = y_pred,digits=4))
```

```
              precision    recall  f1-score   support

           0     0.9908    0.9847    0.9877       980
           1     0.9912    0.9921    0.9916      1135
           2     0.9805    0.9738    0.9772      1032
           3     0.9751    0.9693    0.9722      1010
           4     0.9698    0.9796    0.9747       982
           5     0.9614    0.9776    0.9694       892
           6     0.9903    0.9614    0.9756       958
           7     0.9758    0.9805    0.9782      1028
           8     0.9702    0.9682    0.9692       974
           9     0.9611    0.9782    0.9695      1009

    accuracy                         0.9768     10000
   macro avg     0.9766    0.9765    0.9765     10000
weighted avg     0.9769    0.9768    0.9768     10000
```

תרגיל – סיווג cifar10



- תמונות של:

```
['Airplane', 'Automobile', 'Bird', 'Cat', 'Deer',
 'Dog', 'Frog', 'Horse', 'Ship', 'Truck']
```

צור מחברת בשם:

Myname_cifar10

כלול במחברת:

- ✓ ייבוא של סיפריות
- ✓ ייבוא של נתונים
- ✓ הצגת הנתונים
- ✓ אימון המודל
- ✓ חיזוי
- ✓ הערכת ביצועים

בהצלחה !!!