Name:	Elad Siman Tov
Course:	ME 036049
Homerwork:	No. 1

Reading

I hereby declare that I have read chapter 1 from the Course textbook - Understanding Deep Learning.

Theory

1. Option C: Machine Learning is a branch of artificial intelligence that enables systems to learn from data and make predictions or decision.
2. Option C: The main goal of supervised learning is minimizing the error between predicted and actual outputs
3. Option C: An example of unsupervised learning is Customer segmentation.
4. Option B: The purpose of a validation set in machine learning is to measure the model's performance on unseen data and tune hyper-parameters.
5. Option D: Over-fitting in machine learning is when the model memorizes the training data and performs poorly on unseen data.
6. Option C: The purpose of feature scaling in machine learning is to standardize or normalize the numerical features to a common scale.
7. Option B: The appropriate evaluation metrics for a binary classification problem is the F1-Score.
8. Option A: The role of a loss function in a machine learning model is to measure the distance between predicted and actual values.
9. Option D: The purpose of cross-validation in machine learning is to prevent over-fitting by using multiple validation sets.
10. Option A: The machine learning algorithms used for both classification and regression are Random Forests, KNN and SVM (Under some modifications)

Prerequisites

```python
import numpy as np
```

Question 1 - Dot and Cross

```
u = np.array([2, 4, -1])
v = np.array([1, -3, 5])
np.dot(u, v)
```

```
-15
```

Question 2 - Matrix Multiplications

```
A = np.array([[3, -1],
              [2, 4]])
B = np.array([[5, 2],
              [-1, 0]])
print(np.matmul(A, B))
print(f'{np.linalg.det(A):.0f}')
```

```
[[16  6]
 [ 6  4]]
14
```

Question 3 - Probability

```
print("3 / 6")
```

```
3 / 6
```

Question 4 - Probability

```
print("26 / 26")
```

```
26 / 26
```

Question 5 - Prime Numbers

```
first_five_primes = [2, 3, 5, 7, 11]
first_five_primes
```

```
[2, 3, 5, 7, 11]
```

Question 6 - Even / Odd

```
def is_even(num: int) -> str:
    return "Odd" if num % 2 else "Even"

print(f"The number 2 is {is_even(2)}")
print(f"The number 11 is {is_even(11)}")
```

```
The number 2 is Even
The number 11 is Odd
```

Question 7 - NumPy Matrices

```
randarray = np.random.rand(3, 3)
print(randarray)
transposed = randarray.T
print(transposed)
```

```
[[0.70393914 0.22099322 0.14898089]
 [0.07371348 0.7440815  0.90432532]
 [0.4300282  0.84015148 0.85990781]]
[[0.70393914 0.07371348 0.4300282 ]
 [0.22099322 0.7440815  0.84015148]
 [0.14898089 0.90432532 0.85990781]]
```

Question 8 - Simulation

```
theoretical = 1 / 6
monte = [0 for _ in range(6)]
for inx in range(100):
    carlo = np.random.randint(1, 6)
    monte[carlo] += 1

print(f"The simulated value is {monte[3] / 100:.4f}")
print(f"The theoretical value is {theoretical:.4f}")
print(f"The error is approx. {(100 * (theoretical - monte[3] / 100) / theoretical):.0f}%")
```

```
The simulated value is 0.2200
The theoretical value is 0.1667
The error is approx. -32%
```

Computation

Deep Learning With Tensorflow and Keras

The following output was recieved after cloning the project and running mnist.py file:

... After some log ...

```
Epoch 185/200 375/375 [==============================] -
1s 3ms/step - loss: 0.2777 - accuracy: 0.9227 - val_loss: 0.2760 - val_accuracy:
0.9231 Epoch 186/200 375/375 [==============================]
- 1s 3ms/step - loss: 0.2776 - accuracy: 0.9225 - val_loss: 0.2759 - val_accuracy:
0.9234 Epoch 187/200 375/375 [==============================]
- 1s 3ms/step - loss: 0.2774 - accuracy: 0.9225 - val_loss: 0.2759 - val_accuracy:
0.9236 Epoch 188/200 375/375 [==============================]
- 1s 3ms/step - loss: 0.2773 - accuracy: 0.9226 - val_loss: 0.2758 - val_accuracy:
0.9232 Epoch 189/200 375/375 [==============================]
- 1s 3ms/step - loss: 0.2772 - accuracy: 0.9227 - val_loss: 0.2757 - val_accuracy:
0.9233 Epoch 190/200 375/375 [==============================]
- 1s 4ms/step - loss: 0.2770 - accuracy: 0.9227 - val_loss: 0.2756 - val_accuracy:
0.9230 Epoch 191/200 375/375 [==============================]
- 1s 4ms/step - loss: 0.2769 - accuracy: 0.9228 - val_loss: 0.2756 - val_accuracy:
0.9234 Epoch 192/200 375/375 [==============================]
- 1s 3ms/step - loss: 0.2768 - accuracy: 0.9229 - val_loss: 0.2756 - val_accuracy:
0.9233 Epoch 193/200 375/375 [==============================]
- 1s 4ms/step - loss: 0.2767 - accuracy: 0.9227 - val_loss: 0.2755 - val_accuracy:
0.9233 Epoch 194/200 375/375 [==============================]
- 2s 4ms/step - loss: 0.2765 - accuracy: 0.9229 - val_loss: 0.2753 - val_accuracy:
0.9233 Epoch 195/200 375/375 [==============================]
- 1s 4ms/step - loss: 0.2764 - accuracy: 0.9229 - val_loss: 0.2753 - val_accuracy:
0.9231 Epoch 196/200 375/375 [==============================]
- 1s 3ms/step - loss: 0.2763 - accuracy: 0.9230 - val_loss: 0.2752 - val_accuracy:
0.9237 Epoch 197/200 375/375 [==============================]
- 1s 3ms/step - loss: 0.2762 - accuracy: 0.9229 - val_loss: 0.2751 - val_accuracy:
0.9237 Epoch 198/200 375/375 [==============================]
- 1s 4ms/step - loss: 0.2760 - accuracy: 0.9230 - val_loss: 0.2751 - val_accuracy:
0.9237 Epoch 199/200 375/375 [==============================]
- 1s 3ms/step - loss: 0.2759 - accuracy: 0.9229 - val_loss: 0.2750 - val_accuracy:
0.9235 Epoch 200/200 375/375 [==============================]
- 1s 4ms/step - loss: 0.2758 - accuracy: 0.9230 - val_loss: 0.2750 - val_accuracy:
0.9237 313/313 [==============================] - 1s
2ms/step - loss: 0.2770 - accuracy: 0.9225 Test accuracy: 0.9225000143051147
313/313 [==============================] - 1s 2ms/step
```

Generative Deep Learning

The output after running mlp.ipynb cells will be attached in a separate pdf file

Google Colab

I hereby declare that I have completed all the tasks in this section, and I will attach the TODO tasks in a separate pdf.