CA2

February 28, 2024

1 CA2 - Supervised machine learning classification pipeline - applied to medical data

1.1 Part I: Data loading and data exploration

1.1.1 Import necessary libraries/modules:

[ ]:
```
# Insert your code below
# ======================
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from mlxtend.classifier import Perceptron, Adaline, LogisticRegression
import seaborn as sns
```

1.1.2 Loading and exploring data

1. Load the dataset `fetal_health.csv` with `pandas`. Use the first column as the row index.
2. Check for missing data, report on your finding and remove samples with missing data, if you find any.
3. Display the raw data with appropriate plots/outputs and inspect it. Describe the distributions of the values of feature `"baseline value"`, `"accelerations"`, and the target variable `"fetal_health"`.
4. Will it be beneficial to scale the data? Why or why not?
5. Is the data linearly separable using a combination of any two pairs of features? Can we expect an accuracy close to 100% from a linear classifier?

[ ]:
```
# Insert your code below
# ======================
# 1
df = pd.read_csv('assets/fetal_health.csv', index_col= 0) # set the first␣
 ↪column as row index
df.head() # view first five rows
```

[ ]:

	baseline value	accelerations	prolongued_decelerations	\
1584	132.0	0.000	0.0	
942	136.0	0.003	0.0	
1376	121.0	0.006	0.0	

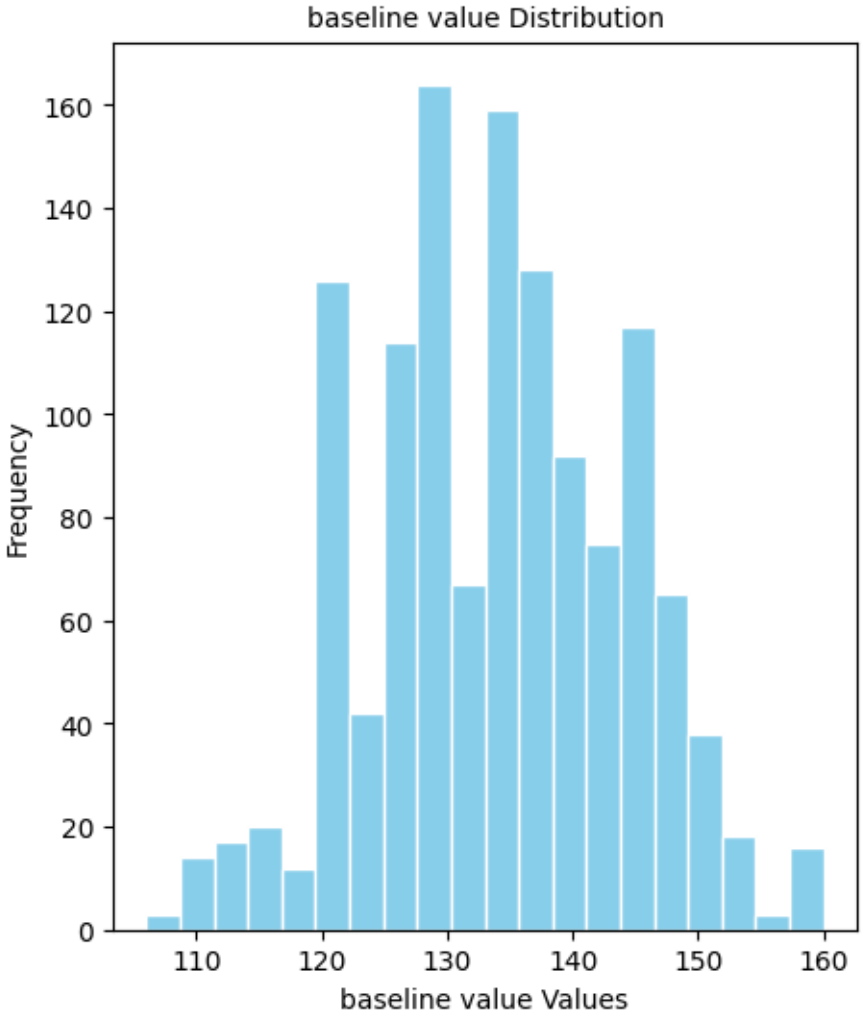
```
169            116.0          0.001                        0.0
1293           115.0          0.006                        0.0

      abnormal_short_term_variability  mean_value_of_short_term_variability  \
1584                             35.0                                   1.3
942                              54.0                                   0.7
1376                             24.0                                   1.7
169                              46.0                                   0.7
1293                             19.0                                   1.7

      percentage_of_time_with_abnormal_long_term_variability  histogram_mean  \
1584                                                  0.0                135.0
942                                                  15.0                141.0
1376                                                  0.0                121.0
169                                                  39.0                120.0
1293                                                  0.0                121.0

      histogram_variance  fetal_health
1584                 9.0             0
942                  8.0             0
1376                25.0             0
169                  1.0             0
1293                 9.0             0
```

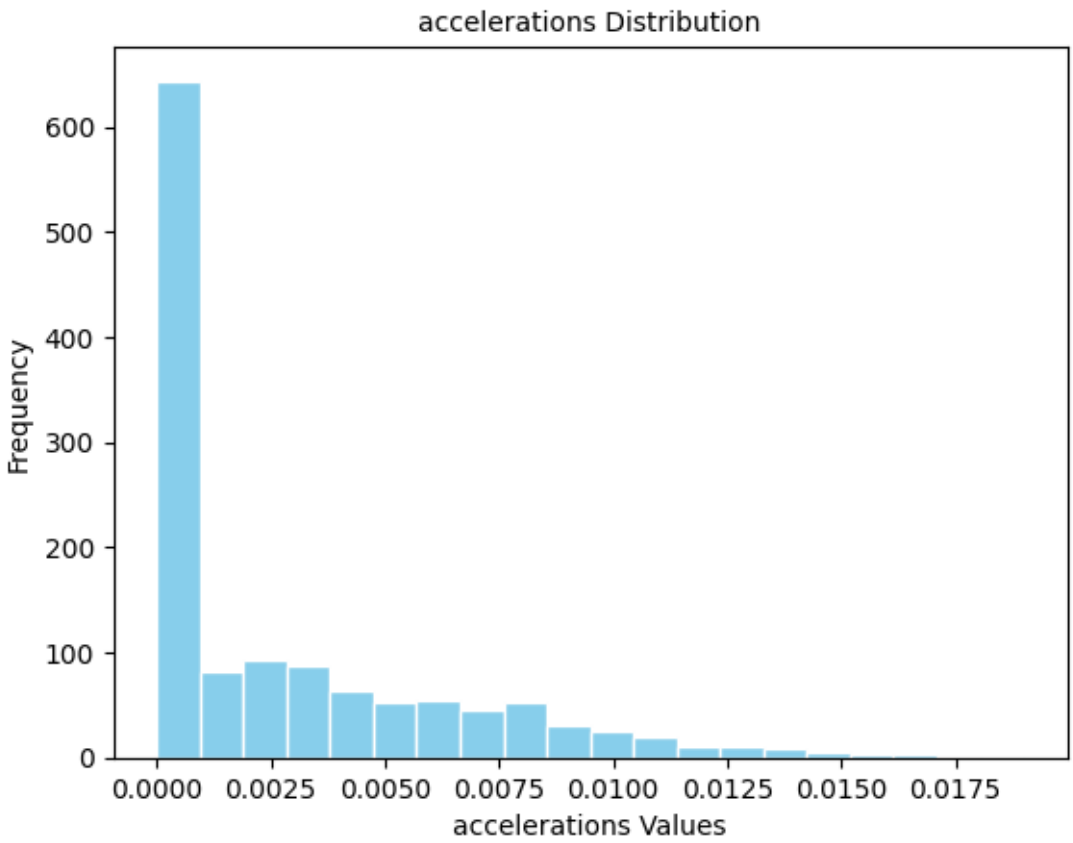
[ ]:
```
# 2
df_NaN = df.isna() # rows containing missing data
print(df_NaN.sum()) # view missing data -> there is None
```

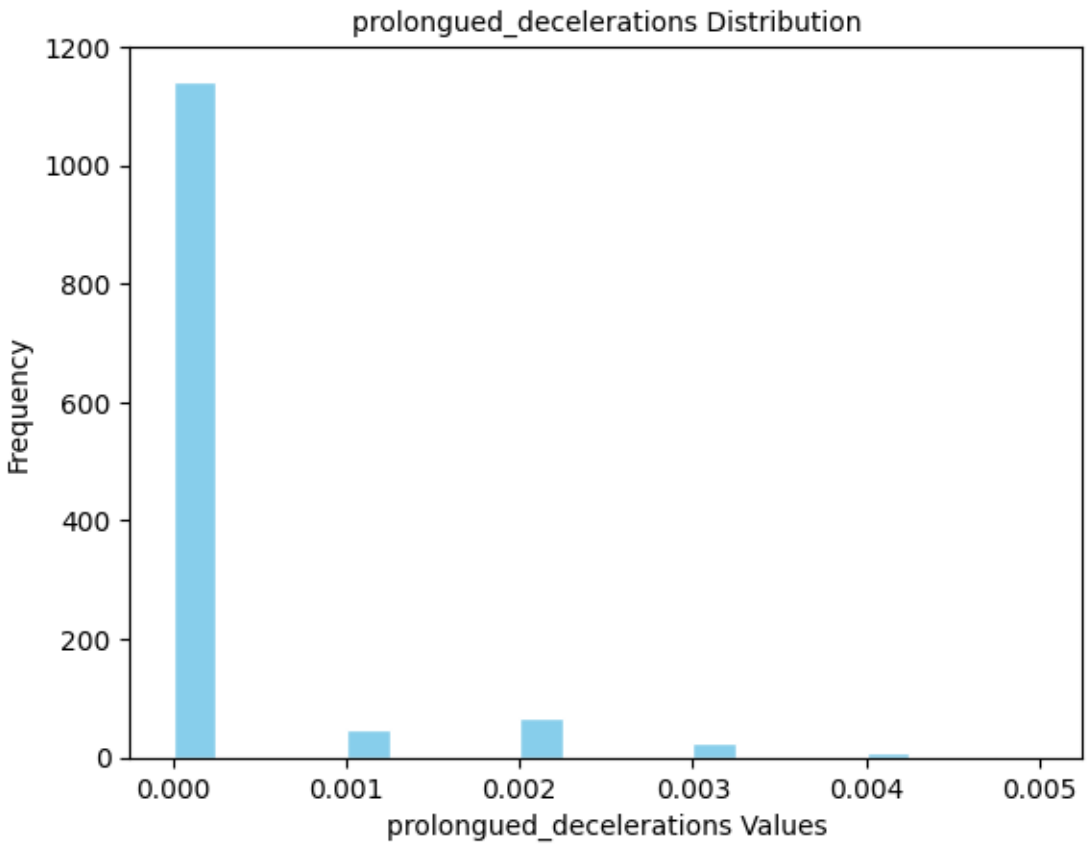
```
baseline value                                            0
accelerations                                             0
prolongued_decelerations                                  0
abnormal_short_term_variability                           0
mean_value_of_short_term_variability                      0
percentage_of_time_with_abnormal_long_term_variability    0
histogram_mean                                            0
histogram_variance                                        0
fetal_health                                              0
dtype: int64
```

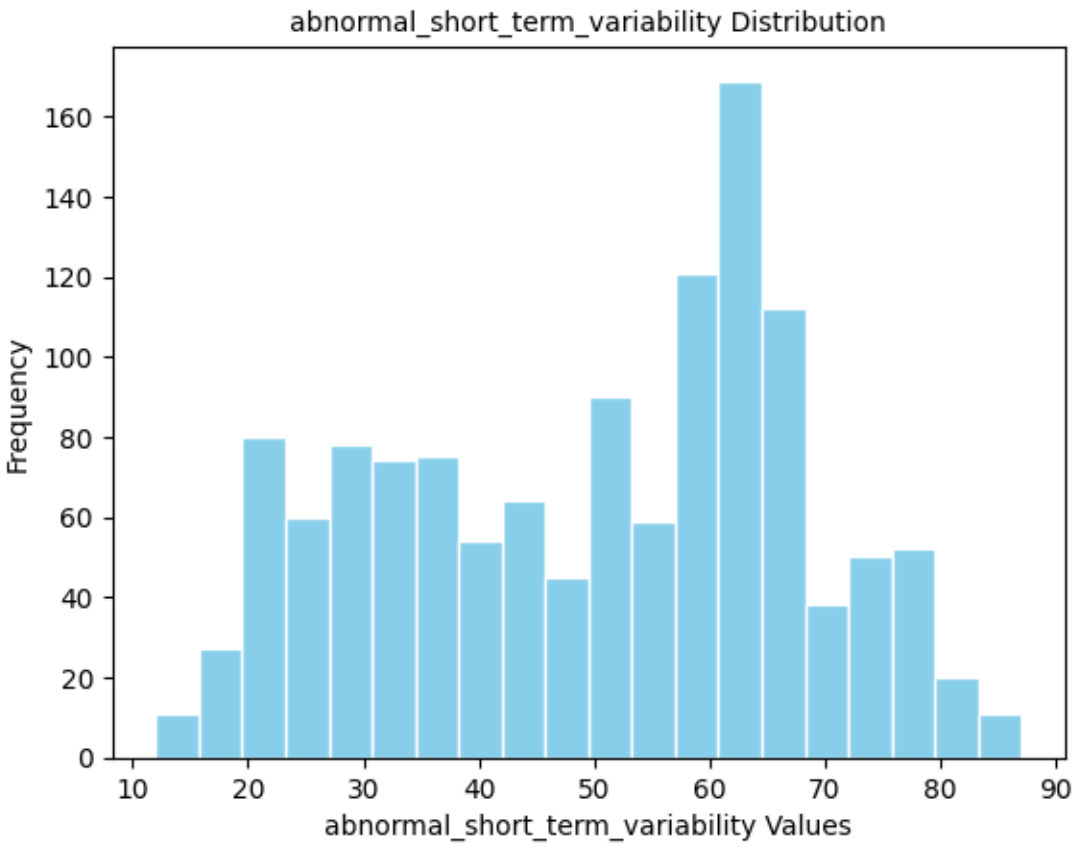
[ ]:
```
# 3
plt.figure(figsize = (5,6)) # alter the size, to makes the output look nicer

titels = list(df.columns) #makes a list with the headers from the dataframe
for idx, colname in enumerate(titels):
    plt.hist(df[colname], bins=20, color="skyblue", edgecolor="white") #␣
 ↪changed color and added edgecolor to make it nicer
    plt.title(f'{colname} Distribution', fontsize = 10) # Title
```

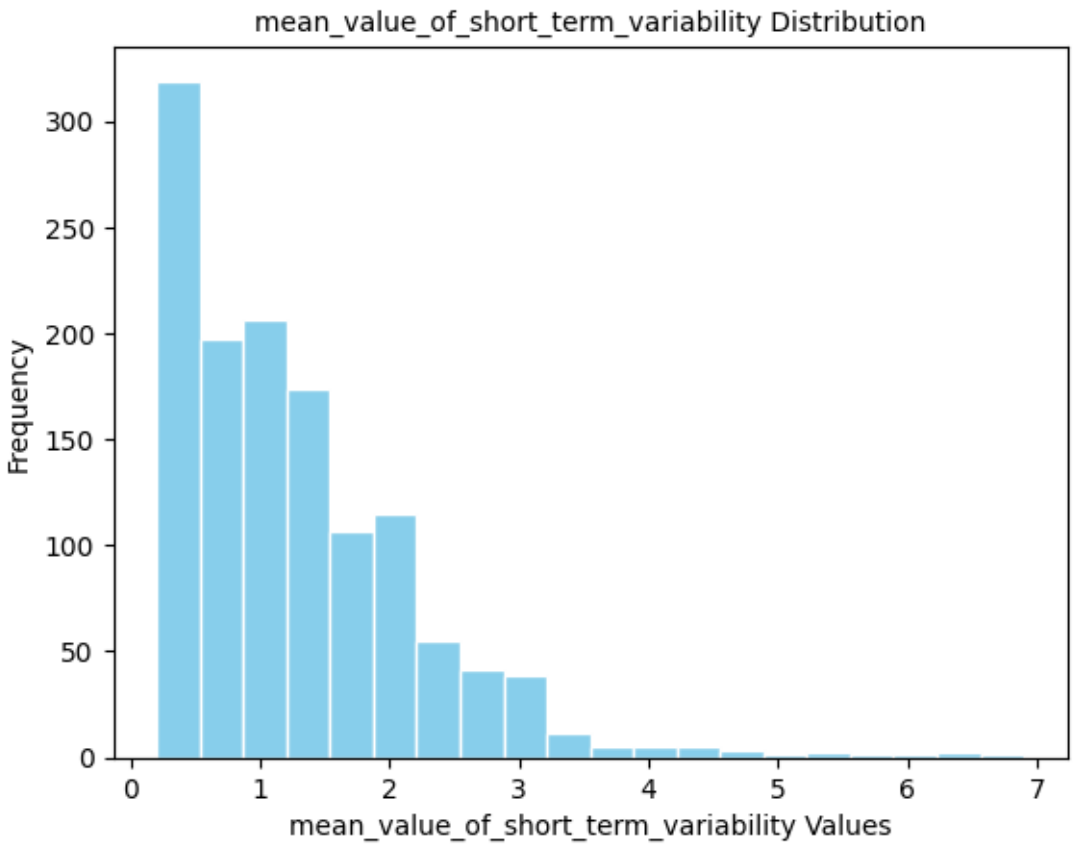
```
    plt.xlabel(f'{colname} Values')# x-axis title w
    plt.ylabel('Frequency') # y-axis title
    plt.show()
```

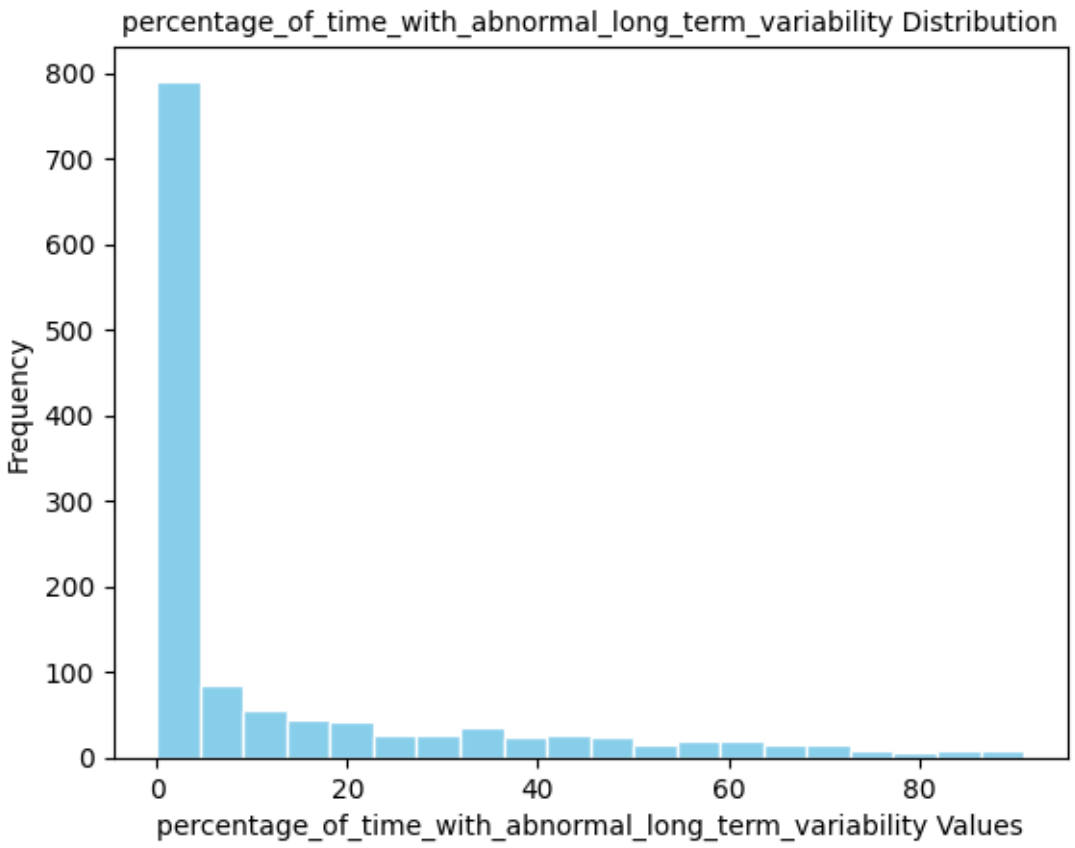


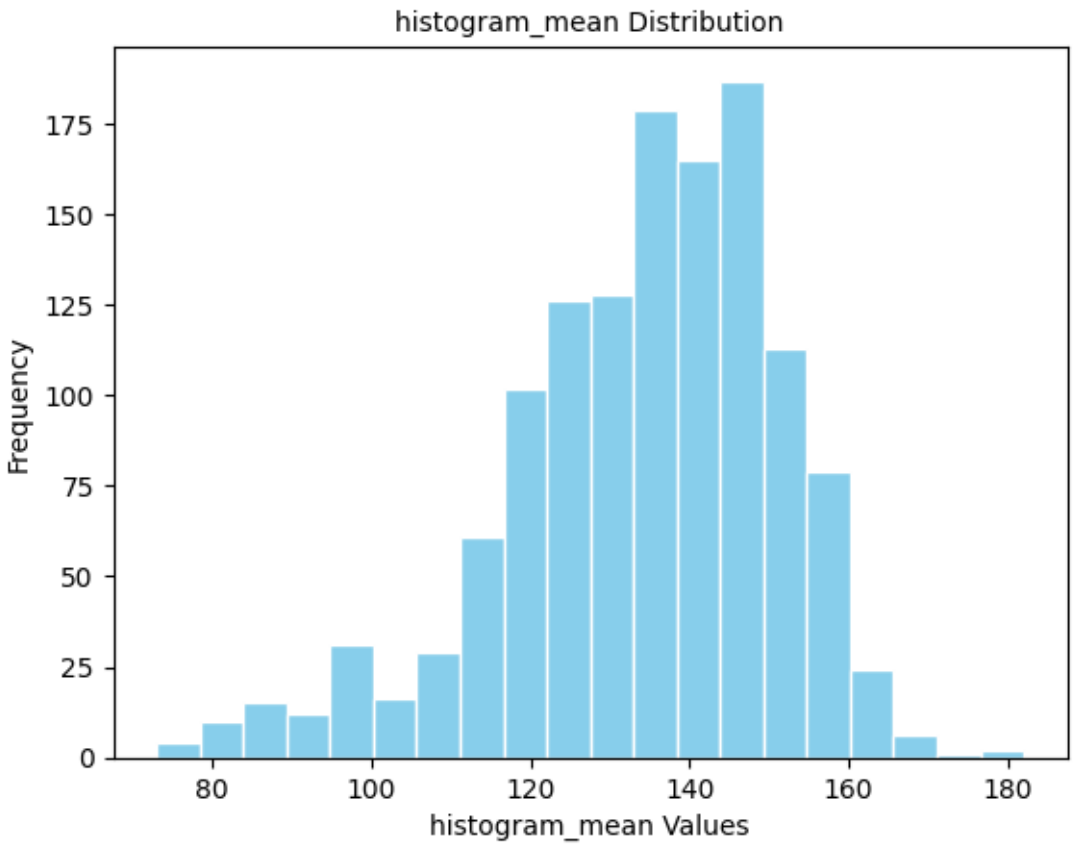

accelerations Distribution
Frequency
600
500
400
300
200
100
0
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
accelerations Values

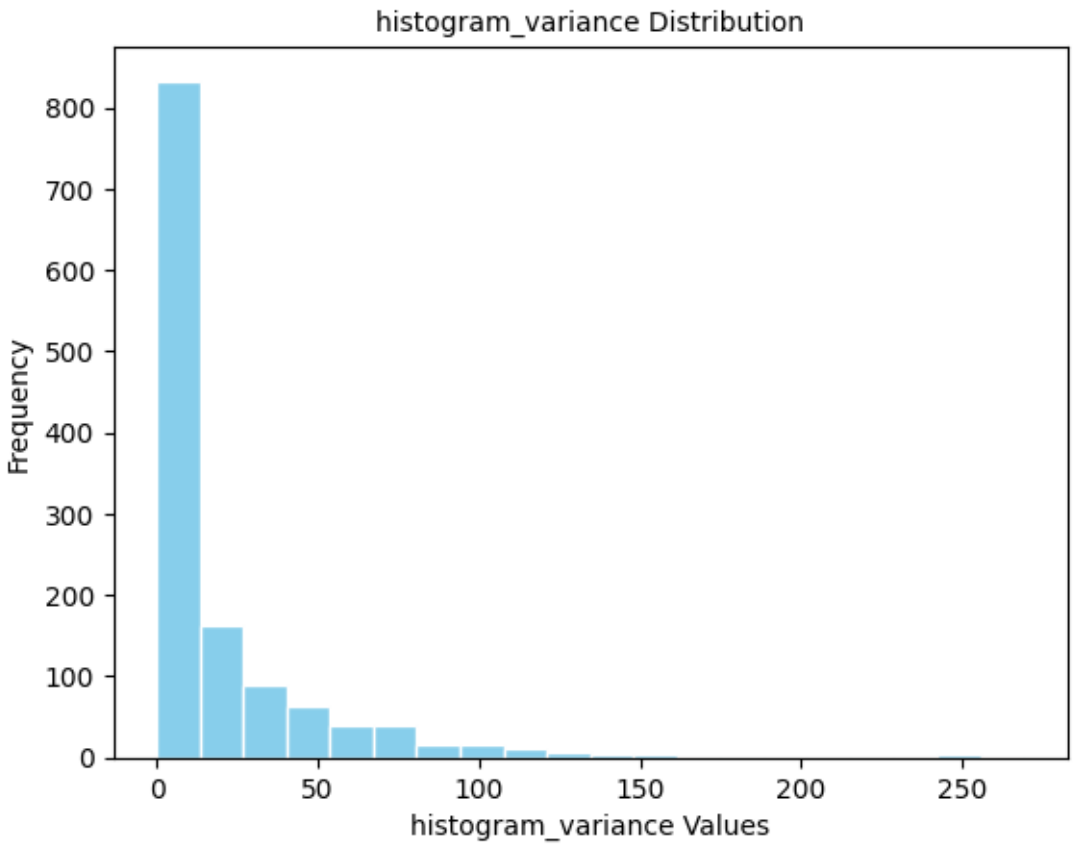

prolongued_decelerations Distribution
1200
1000
800
600
400
200
0
Frequency
0.000
0.001
0.002
0.003
0.004
0.005
prolongued_decelerations Values

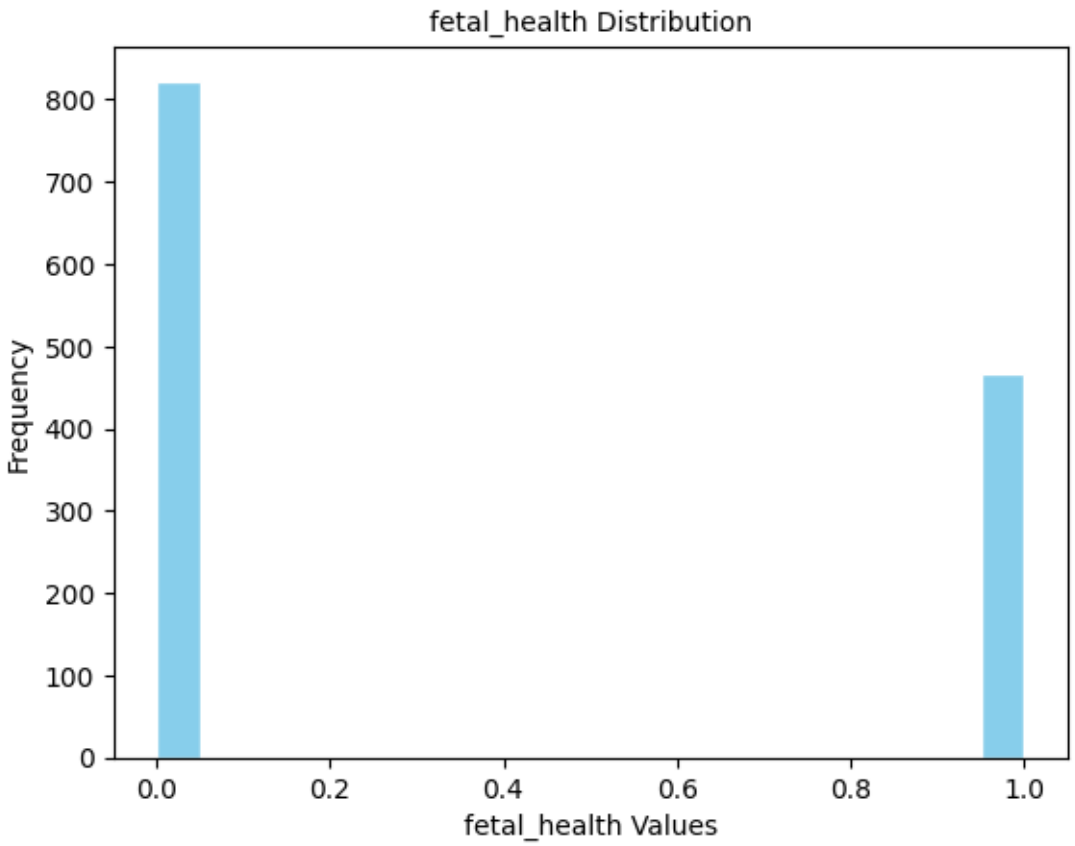

abnormal_short_term_variability Distribution
Frequency
0
20
40
60
80
100
120
140
160
10
20
30
40
50
60
70
80
90
abnormal_short_term_variability Values


mean_value_of_short_term_variability Distribution
Frequency
300
250
200
150
100
50
0
0
1
2
3
4
5
6
7
mean_value_of_short_term_variability Values


percentage_of_time_with_abnormal_long_term_variability Distribution
Frequency
800
700
600
500
400
300
200
100
0
0
20
40
60
80
percentage_of_time_with_abnormal_long_term_variability Values


histogram_mean Distribution
Frequency
0
25
50
75
100
125
150
175
80
100
120
140
160
180
histogram_mean Values


histogram_variance Distribution
Frequency
800
700
600
500
400
300
200
100
0
0
50
100
150
200
250
histogram_variance Values

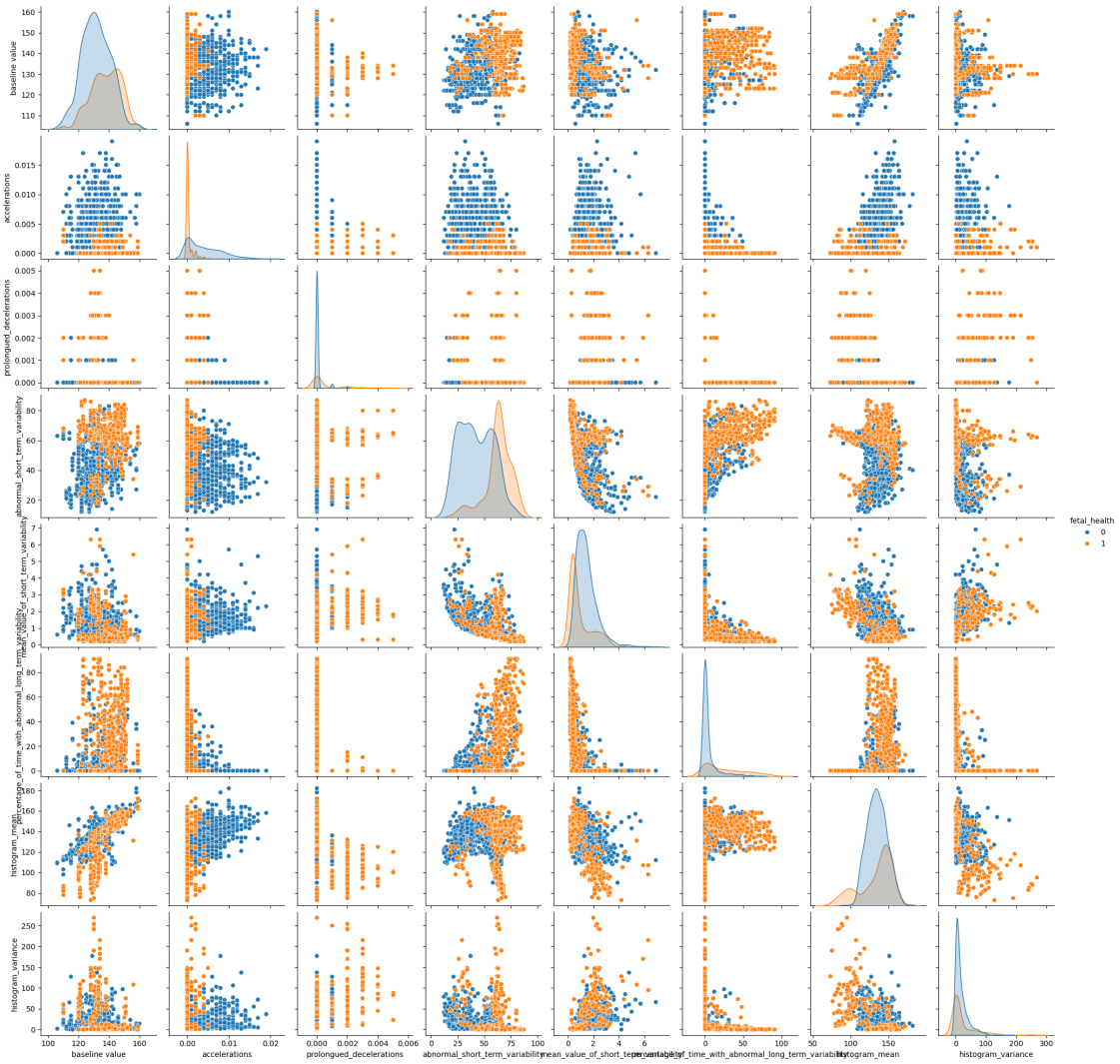


3.
   - Baseline value : describes the fetal heart rate in beats per minute. it is distrubuted almost as a normal distribution.
   - Acceleration: number of acceleration per minute, the distrubution is skewed right and the majority obseerved where 0.0.
   - Fetal health : the fetal health categorized into 0(normal) og 1(not normal). Most of them are 0

4. Will it be beneficial to scale the data? Why or why not?
   - Yes, it would be beneficical to scale the data. this ois beacuse it makes the ontra variance smaller and for example the baseline, to nomalize this, the extreme values could be -1 and 1, versus 110 and 160. The calculations of Wx +b will be much easier then.

5. Is the data linearly separable using a combination of any two pairs of features? Can we expect an accuracy close to 100% from a linear classifier?
   - It is not perfect linearly separable. Looking at the plot below, you can see that its hard to draw a line that separates the classes completaly, therefor we can not expect an accuancy close to 100% from linear classifier.

[ ]:
```
# 5
sns.pairplot(data=df, hue = 'fetal_health')
plt.show()
```



1.2 Part II: Train/Test Split

Divide your dataset into training and testing subsets. Follow these steps to create the split:

1. **Divide the dataset into two data sets, each data set only contains samples of either class 0 or class 1:**

- Create a DataFrame `df_0` containing all data with `"fetal_health"` equal to 0.
- Create a DataFrame `df_1` containing all data with `"fetal_health"` equal to 1.

2. **Split into training and test set by randomly sampling entries from the data frames:**

- Create a DataFrame `df_0_train` containing by sampling `75%` of the entries from `df_0` (use the `sample` method of the data frame, fix the `random_state` to `42`).
- Create a DataFrame `df_1_train` using the same approach with `df_1`.
- Create a DataFrame `df_0_test` containing the remaining entries of `df_0` (use `df_0.drop(df_0_train.index)` to drop all entries except the previously extracted ones).

- Create a DataFrame `df_1_test` using the same approach with `df_1`.

3. **Merge the datasets split by classes back together:**

- Create a DataFrame `df_train` containing all entries from `df_0_train` and `df_1_train`. (Hint: use the `concat` method you know from CA1)
- Create a DataFrame `df_test` containing all entries from the two test sets.

4. **Create the following data frames from these splits:**

- `X_train`: Contains all columns of `df_train` except for the target feature `"fetal_health"`
- `X_test`: Contains all columns of `df_test` except for the target feature `"fetal_health"`
- `y_train`: Contains only the target feature `"fetal_health"` for all samples in the training set
- `y_test`: Contains only the target feature `"fetal_health"` for all samples in the test set

5. **Check that your sets have the expected sizes/shape by printing number of rows and colums ("shape") of the data sets.**

- (Sanity check: there should be 8 features, almost 1000 samples in the training set and slightly more than 300 samples in the test set.)

6. **Explain the purpose of this slightly complicated procedure. Why did we first split into the two classes? Why did we then split into a training and a testing set?**

7. **What is the share (in percent) of samples with class 0 label in test and training set, and in the intial data set?**

[ ]:
```
# Insert your code below
# ======================

# 1
df_0 = df[df['fetal_health'] == 0]
df_1 = df[df['fetal_health'] == 1]
```

[ ]:
```
# 2 Split into training and test set by randomly sampling entries from the data␣
 ↪frames:
df_0_train = df_0.sample(frac = 0.75, random_state = 42)
df_1_train = df_1.sample(frac = 0.75, random_state = 42)

df_0_test = df_0.sample(frac = 0.75, random_state = 42)
df_1_test = df_1.sample(frac = 0.75, random_state = 42)

df_0_test = df_0.drop(df_0_train.index)
df_1_test = df_1.drop(df_1_train.index)
```

[ ]:
```
# 3. Merge the datasets split by classes back together:
df_train = pd.concat([df_0_train, df_1_train])
df_test = pd.concat([df_0_test, df_1_test])
```

[ ]:
```
# 4
X_train = df_train.drop(columns=["fetal_health"])
X_test = df_test.drop(columns=["fetal_health"])
y_train = df_train['fetal_health']
y_test = df_test['fetal_health']
```

[ ]:
```
# 5 shape of test and train sets
print(X_train.shape)
print(X_test.shape)

print(y_train.shape)
print(y_test.shape)
```

```
(967, 8)
(323, 8)
(967,)
(323,)
```

6. The purpose of first splitt into two classes is to undertand the patterns in the data and train the model on a portion of the data, and evaluate the preformance on unseen data to check and assess the capasity to generalize.

[ ]:
```
# 7 Calculate the share of samples with class 0

train_0_share = (y_train == 0).mean() * 100
test_0_share = (y_test == 0).mean() * 100
initial_0_share = (df['fetal_health'] == 0).mean() * 100


print("{:.1f}%".format(train_0_share))
print("{:.1f}%".format(test_0_share))
print("{:.1f}%".format(initial_0_share))
```

```
63.8%
63.8%
63.8%
```

1.2.1 Convert data to numpy arrays and shuffle the training data

Many machine learning models (including those you will work with later in the assignment) will not accept DataFrames as input. Instead, they will only work if you pass numpy arrays containing the data. Here, we convert the DataFrames `X_train`, `X_test`, `y_train`, and `y_test` to numpy arrays `X_train`, `X_test`, `y_train`, and `y_test`.

Moreover we shuffle the training data. This is important because the training data is currently ordered by class. In Part IV, we use the first n samples from the training set to train the classifiers. If we did not shuffle the data, the classifiers would only be trained on samples of class 0.

Nothing to be done here, just execute the cell.

```
[ ]: # convert to numpy arrays
X_train = X_train.to_numpy()
X_test = X_test.to_numpy()
y_train = y_train.to_numpy()
y_test = y_test.to_numpy()

# shuffle training data
np.random.seed(42) # for reproducibility
shuffle_index = np.random.permutation(len(X_train)) # generate random indices
X_train, y_train = X_train[shuffle_index], y_train[shuffle_index] # shuffle␣
 ↪data by applying reordering with the random indices
```

1.3 Part III: Scaling the data

1. Standardize the training *and* test data so that each feature has a mean of 0 and a standard deviation of 1.
2. Check that the scaling was successful
   - by printing the mean and standard deviation of each feature in the scaled training set
   - by putting the scaled training set into a DataFrame and make a violin plot of the data

Hint: use the `axis` argument to calculate mean and standard deviation column-wise.

Important: Avoid data leakage!

More hints:

1. For each column, subtract the mean (μ) of each column from each value in the column
2. Divide the result by the standard deviation (σ) of the column

(You saw how to do both operations in the lecture. If you don't remember, you can look it up in Canvas files.)

Mathematically (in case this is useful for you), this transformation can be represented for each column as follows:

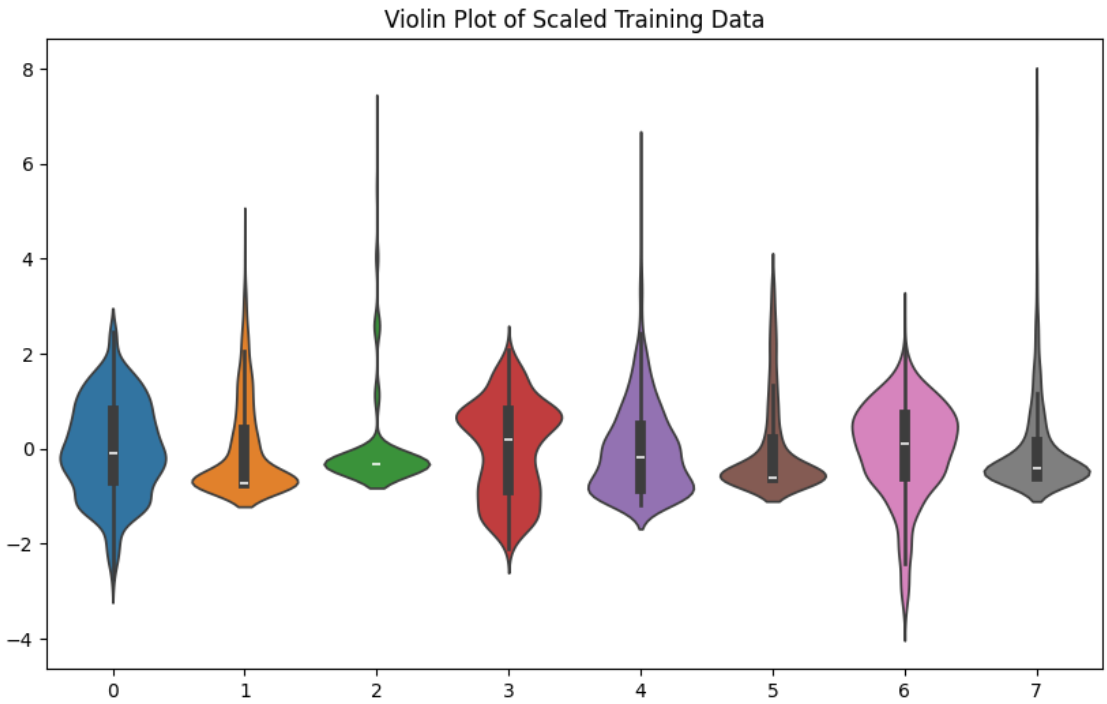
$$X_{\text{scaled}} = \frac{(X - \mu)}{\sigma}$$

where: - (X_{scaled}) are the new, transformed column values (a column-vector) - (X) is the original values - (μ) is the mean of the column - (σ) is the standard deviation of the column

```
[ ]: # Insert your code below
# ======================

# 1 Standarize the training and test data
X_train_scaled = (X_train - np.mean(X_train, axis=0)) / np.std(X_train, axis=0,␣
 ↪ddof=1)
X_test_scaled = (X_test - np.mean(X_train, axis=0)) / np.std(X_train, axis=0,␣
 ↪ddof=1)

# Double-check mean and std of the scaled X_train
```

```
X_train_sc_mean = np.mean(X_train_scaled, axis=0)
X_train_sc_std = np.std(X_train_scaled, axis=0, ddof=1)

print(f'Mean of each feature in the scaled training set: \n{X_train_sc_mean}')
print()
print(f'Standard deviation of each feature in the scaled training set:␣
  ↪{X_train_sc_std}')

df_scaled_train = pd.DataFrame(X_train_scaled)
# Making a violin plot
plt.figure(figsize=(10, 6))
plt.title('Violin Plot of Scaled Training Data')
sns.violinplot(data=df_scaled_train)
plt.show()
```

```
Mean of each feature in the scaled training set:
[-1.05396560e-16  3.84375974e-15 -3.17337791e-16  1.30195751e-16
 -6.05915410e-17 -3.18026658e-16 -3.16763736e-16 -3.23767211e-17]

Standard deviation of each feature in the scaled training set: [1. 1. 1. 1. 1.
1. 1. 1.]
```



1.4 Part IV: Training and evaluation with different dataset sizes and training times

Often, a larger dataset size will yield better model performance. (As we will learn later, this usually prevents overfitting and increases the generalization capability of the trained model.) However, collecting data is usually rather expensive.

In this part of the exercise, you will investigate

- how the model performance changes with varying dataset size
- how the model performance changes with varying numbers of epochs/iterations of the optimizer/solver (increasing training time).

For this task (Part IV), use the `Adaline`, `Perceptron`, and `LogisticRegression` classifier from the `mlxtend` library. All use the gradient descent (GD) algorithm for training.

Important: Use a learning rate of `1e-4` (`0.0001`) for all classifiers, and use the argument `minibatches=1` when initializing `Adaline` and `LogisticRegression` classifier (this will make sure it uses GD). For all three classifiers, pass `random_seed=42` when initializing the classifier to ensure reproducibility of the results.

1.4.1 Model training

Train the model models using progressively larger subsets of your dataset, specifically: first 50 rows, first 100 rows, first 150 rows, …, first 650 rows, first 700 rows (in total 14 different variants).

For each number of rows train the model with progressively larger number of epochs: 2, 7, 12, 17, …, 87, 92, 97 (in total 20 different model variants).

The resulting $14 \times 20 = 280$ models obtained from the different combinations of subsets and number of epochs. An output of the training process could look like this:

```
Model (1) Train a model with first 50 rows of data for 2 epochs
Model (2) Train a model with first 50 rows of data for 7 epochs
Model (3) Train a model with first 50 rows of data for 12 epochs
...
Model (21) Train a model with first 100 rows of data for 2 epochs
Model (22) Train a model with first 100 rows of data for 7 epochs
...
Model (279) Train a model with first 700 rows of data for 92 epochs
Model (280) Train a model with first 700 rows of data for 97 epochs
```

1.4.2 Model evaluation

For each of the 280 models, calculate the **accuracy on the test set** (do **not** use the score method but compute accuracy yourself). Store the results in the provided 2D numpy array (it has 14 rows and 20 columns). The rows of the array correspond to the different dataset sizes, and the columns correspond to the different numbers of epochs.

1.4.3 Tasks

1. Train the 280 Adaline classifiers as mentioned above and calculate the accuracy for each of the 280 variants.

2. Generalize your code so that is doing the same procedure for all three classifiers: `Perceptron`, `Adaline`, and `LogisticRegression` after each other. Store the result for all classifiers. You can for example use an array of shape $3 \times 14 \times 20$ to store the accuracies of the three classifiers.

Note that executing the cells will take some time (but on most systems it should not be more than 5 minutes).

[ ]:
```
# Train and evaluate all model variants
# Insert your code below

# Define parameters
eta = 0.0001
minibatches = 1
random_seed = 42
dataset_sizes = np.arange(50, 701, 50) # 14 rows
num_epochs = np.arange(2, 98, 5) # 20 columns
model_idx = 1

# Initialize classifiers
all_accuracies = np.zeros((3, len(dataset_sizes), len(num_epochs))) # Stores␣
 ↪the accuracies

# Loop over dataset sizes
for size_idx, size in enumerate(dataset_sizes):
    X_subset = X_train_scaled[:size]
    y_subset = y_train[:size]
    # Loop over number of epochs
    for epoch_idx, epochs in enumerate(num_epochs):
        perceptron = Perceptron(eta=eta, random_seed=random_seed, epochs =␣
 ↪epochs)
        perceptron.fit(X_subset, y_subset) # Train the classifier
        y_pred = perceptron.predict(X_test_scaled)   # Make predictions on the␣
 ↪test set
        accuracy = np.mean(y_pred == y_test)  # Calculates the accuracy
        all_accuracies[0][size_idx][epoch_idx] = accuracy  # Stores the␣
 ↪accuracy in all_accuracies
        print(f"Model ({model_idx}) - Percetron - Train a model with first␣
 ↪{size} rows of data for {epochs} epochs. Accuracy: {accuracy:.2f}")
        model_idx += 1

        adaline = Adaline(eta=eta, minibatches=minibatches,␣
 ↪random_seed=random_seed, epochs=epochs)
        adaline.fit(X_subset, y_subset) # Train the classifier
        y_pred = adaline.predict(X_test_scaled)   # Make predictions on the␣
 ↪test set
        accuracy = np.mean(y_pred == y_test)  # Calculates the accuracy
```

```
        all_accuracies[1][size_idx][epoch_idx] = accuracy  # Stores the␣
↪accuracy in all_accuracies
        print(f"Model ({model_idx}) - Adaline - Train a model with first {size}␣
↪rows of data for {epochs} epochs. Accuracy: {accuracy:.2f}")
        model_idx += 1

        LR = LogisticRegression(eta=eta, minibatches=minibatches,␣
↪random_seed=random_seed, epochs = epochs)
        LR.fit(X_subset, y_subset) # Train the classifier
        y_pred = LR.predict(X_test_scaled)   # Make predictions on the test set
        accuracy = np.mean(y_pred == y_test)  # Calculates the accuracy
        all_accuracies[2][size_idx][epoch_idx] = accuracy  # Stores the␣
↪accuracy in all_accuracies

        print(f"Model ({model_idx}) - Linear Regression - Train a model with␣
↪first {size} rows of data for {epochs} epochs. Accuracy: {accuracy:.2f}")
        model_idx += 1 # update model ID
```

```
Model (1) - Percetron - Train a model with first 50 rows of data for 2 epochs.
Accuracy: 0.81
Model (2) - Adaline - Train a model with first 50 rows of data for 2 epochs.
Accuracy: 0.80
Model (3) - Linear Regression - Train a model with first 50 rows of data for 2
epochs. Accuracy: 0.78
Model (4) - Percetron - Train a model with first 50 rows of data for 7 epochs.
Accuracy: 0.86
Model (5) - Adaline - Train a model with first 50 rows of data for 7 epochs.
Accuracy: 0.85
Model (6) - Linear Regression - Train a model with first 50 rows of data for 7
epochs. Accuracy: 0.82
Model (7) - Percetron - Train a model with first 50 rows of data for 12 epochs.
Accuracy: 0.86
Model (8) - Adaline - Train a model with first 50 rows of data for 12 epochs.
Accuracy: 0.87
Model (9) - Linear Regression - Train a model with first 50 rows of data for 12
epochs. Accuracy: 0.85
Model (10) - Percetron - Train a model with first 50 rows of data for 17 epochs.
Accuracy: 0.87
Model (11) - Adaline - Train a model with first 50 rows of data for 17 epochs.
Accuracy: 0.86
Model (12) - Linear Regression - Train a model with first 50 rows of data for 17
epochs. Accuracy: 0.85
Model (13) - Percetron - Train a model with first 50 rows of data for 22 epochs.
Accuracy: 0.88
Model (14) - Adaline - Train a model with first 50 rows of data for 22 epochs.
Accuracy: 0.86
Model (15) - Linear Regression - Train a model with first 50 rows of data for 22
```

```
epochs. Accuracy: 0.86
Model (16) - Percetron - Train a model with first 50 rows of data for 27 epochs.
Accuracy: 0.88
Model (17) - Adaline - Train a model with first 50 rows of data for 27 epochs.
Accuracy: 0.86
Model (18) - Linear Regression - Train a model with first 50 rows of data for 27
epochs. Accuracy: 0.87
Model (19) - Percetron - Train a model with first 50 rows of data for 32 epochs.
Accuracy: 0.89
Model (20) - Adaline - Train a model with first 50 rows of data for 32 epochs.
Accuracy: 0.86
Model (21) - Linear Regression - Train a model with first 50 rows of data for 32
epochs. Accuracy: 0.87
Model (22) - Percetron - Train a model with first 50 rows of data for 37 epochs.
Accuracy: 0.90
Model (23) - Adaline - Train a model with first 50 rows of data for 37 epochs.
Accuracy: 0.86
Model (24) - Linear Regression - Train a model with first 50 rows of data for 37
epochs. Accuracy: 0.86
Model (25) - Percetron - Train a model with first 50 rows of data for 42 epochs.
Accuracy: 0.90
Model (26) - Adaline - Train a model with first 50 rows of data for 42 epochs.
Accuracy: 0.86
Model (27) - Linear Regression - Train a model with first 50 rows of data for 42
epochs. Accuracy: 0.86
Model (28) - Percetron - Train a model with first 50 rows of data for 47 epochs.
Accuracy: 0.89
Model (29) - Adaline - Train a model with first 50 rows of data for 47 epochs.
Accuracy: 0.86
Model (30) - Linear Regression - Train a model with first 50 rows of data for 47
epochs. Accuracy: 0.86
Model (31) - Percetron - Train a model with first 50 rows of data for 52 epochs.
Accuracy: 0.89
Model (32) - Adaline - Train a model with first 50 rows of data for 52 epochs.
Accuracy: 0.86
Model (33) - Linear Regression - Train a model with first 50 rows of data for 52
epochs. Accuracy: 0.86
Model (34) - Percetron - Train a model with first 50 rows of data for 57 epochs.
Accuracy: 0.89
Model (35) - Adaline - Train a model with first 50 rows of data for 57 epochs.
Accuracy: 0.86
Model (36) - Linear Regression - Train a model with first 50 rows of data for 57
epochs. Accuracy: 0.86
Model (37) - Percetron - Train a model with first 50 rows of data for 62 epochs.
Accuracy: 0.87
Model (38) - Adaline - Train a model with first 50 rows of data for 62 epochs.
Accuracy: 0.86
Model (39) - Linear Regression - Train a model with first 50 rows of data for 62
```

```
epochs. Accuracy: 0.85
Model (40) - Percetron - Train a model with first 50 rows of data for 67 epochs.
Accuracy: 0.89
Model (41) - Adaline - Train a model with first 50 rows of data for 67 epochs.
Accuracy: 0.86
Model (42) - Linear Regression - Train a model with first 50 rows of data for 67
epochs. Accuracy: 0.85
Model (43) - Percetron - Train a model with first 50 rows of data for 72 epochs.
Accuracy: 0.89
Model (44) - Adaline - Train a model with first 50 rows of data for 72 epochs.
Accuracy: 0.86
Model (45) - Linear Regression - Train a model with first 50 rows of data for 72
epochs. Accuracy: 0.85
Model (46) - Percetron - Train a model with first 50 rows of data for 77 epochs.
Accuracy: 0.87
Model (47) - Adaline - Train a model with first 50 rows of data for 77 epochs.
Accuracy: 0.87
Model (48) - Linear Regression - Train a model with first 50 rows of data for 77
epochs. Accuracy: 0.85
Model (49) - Percetron - Train a model with first 50 rows of data for 82 epochs.
Accuracy: 0.88
Model (50) - Adaline - Train a model with first 50 rows of data for 82 epochs.
Accuracy: 0.87
Model (51) - Linear Regression - Train a model with first 50 rows of data for 82
epochs. Accuracy: 0.86
Model (52) - Percetron - Train a model with first 50 rows of data for 87 epochs.
Accuracy: 0.85
Model (53) - Adaline - Train a model with first 50 rows of data for 87 epochs.
Accuracy: 0.87
Model (54) - Linear Regression - Train a model with first 50 rows of data for 87
epochs. Accuracy: 0.85
Model (55) - Percetron - Train a model with first 50 rows of data for 92 epochs.
Accuracy: 0.89
Model (56) - Adaline - Train a model with first 50 rows of data for 92 epochs.
Accuracy: 0.87
Model (57) - Linear Regression - Train a model with first 50 rows of data for 92
epochs. Accuracy: 0.85
Model (58) - Percetron - Train a model with first 50 rows of data for 97 epochs.
Accuracy: 0.89
Model (59) - Adaline - Train a model with first 50 rows of data for 97 epochs.
Accuracy: 0.87
Model (60) - Linear Regression - Train a model with first 50 rows of data for 97
epochs. Accuracy: 0.86
Model (61) - Percetron - Train a model with first 100 rows of data for 2 epochs.
Accuracy: 0.83
Model (62) - Adaline - Train a model with first 100 rows of data for 2 epochs.
Accuracy: 0.83
Model (63) - Linear Regression - Train a model with first 100 rows of data for 2
```

```
epochs. Accuracy: 0.80
Model (64) - Percetron - Train a model with first 100 rows of data for 7 epochs.
Accuracy: 0.88
Model (65) - Adaline - Train a model with first 100 rows of data for 7 epochs.
Accuracy: 0.87
Model (66) - Linear Regression - Train a model with first 100 rows of data for 7
epochs. Accuracy: 0.86
Model (67) - Percetron - Train a model with first 100 rows of data for 12
epochs. Accuracy: 0.88
Model (68) - Adaline - Train a model with first 100 rows of data for 12 epochs.
Accuracy: 0.87
Model (69) - Linear Regression - Train a model with first 100 rows of data for
12 epochs. Accuracy: 0.88
Model (70) - Percetron - Train a model with first 100 rows of data for 17
epochs. Accuracy: 0.88
Model (71) - Adaline - Train a model with first 100 rows of data for 17 epochs.
Accuracy: 0.87
Model (72) - Linear Regression - Train a model with first 100 rows of data for
17 epochs. Accuracy: 0.87
Model (73) - Percetron - Train a model with first 100 rows of data for 22
epochs. Accuracy: 0.88
Model (74) - Adaline - Train a model with first 100 rows of data for 22 epochs.
Accuracy: 0.88
Model (75) - Linear Regression - Train a model with first 100 rows of data for
22 epochs. Accuracy: 0.87
Model (76) - Percetron - Train a model with first 100 rows of data for 27
epochs. Accuracy: 0.89
Model (77) - Adaline - Train a model with first 100 rows of data for 27 epochs.
Accuracy: 0.88
Model (78) - Linear Regression - Train a model with first 100 rows of data for
27 epochs. Accuracy: 0.87
Model (79) - Percetron - Train a model with first 100 rows of data for 32
epochs. Accuracy: 0.89
Model (80) - Adaline - Train a model with first 100 rows of data for 32 epochs.
Accuracy: 0.88
Model (81) - Linear Regression - Train a model with first 100 rows of data for
32 epochs. Accuracy: 0.87
Model (82) - Percetron - Train a model with first 100 rows of data for 37
epochs. Accuracy: 0.86
Model (83) - Adaline - Train a model with first 100 rows of data for 37 epochs.
Accuracy: 0.88
Model (84) - Linear Regression - Train a model with first 100 rows of data for
37 epochs. Accuracy: 0.87
Model (85) - Percetron - Train a model with first 100 rows of data for 42
epochs. Accuracy: 0.87
Model (86) - Adaline - Train a model with first 100 rows of data for 42 epochs.
Accuracy: 0.88
Model (87) - Linear Regression - Train a model with first 100 rows of data for
```

```
42 epochs. Accuracy: 0.87
Model (88) - Percetron - Train a model with first 100 rows of data for 47
epochs. Accuracy: 0.85
Model (89) - Adaline - Train a model with first 100 rows of data for 47 epochs.
Accuracy: 0.88
Model (90) - Linear Regression - Train a model with first 100 rows of data for
47 epochs. Accuracy: 0.87
Model (91) - Percetron - Train a model with first 100 rows of data for 52
epochs. Accuracy: 0.76
Model (92) - Adaline - Train a model with first 100 rows of data for 52 epochs.
Accuracy: 0.88
Model (93) - Linear Regression - Train a model with first 100 rows of data for
52 epochs. Accuracy: 0.87
Model (94) - Percetron - Train a model with first 100 rows of data for 57
epochs. Accuracy: 0.87
Model (95) - Adaline - Train a model with first 100 rows of data for 57 epochs.
Accuracy: 0.88
Model (96) - Linear Regression - Train a model with first 100 rows of data for
57 epochs. Accuracy: 0.87
Model (97) - Percetron - Train a model with first 100 rows of data for 62
epochs. Accuracy: 0.88
Model (98) - Adaline - Train a model with first 100 rows of data for 62 epochs.
Accuracy: 0.88
Model (99) - Linear Regression - Train a model with first 100 rows of data for
62 epochs. Accuracy: 0.88
Model (100) - Percetron - Train a model with first 100 rows of data for 67
epochs. Accuracy: 0.86
Model (101) - Adaline - Train a model with first 100 rows of data for 67 epochs.
Accuracy: 0.88
Model (102) - Linear Regression - Train a model with first 100 rows of data for
67 epochs. Accuracy: 0.88
Model (103) - Percetron - Train a model with first 100 rows of data for 72
epochs. Accuracy: 0.89
Model (104) - Adaline - Train a model with first 100 rows of data for 72 epochs.
Accuracy: 0.88
Model (105) - Linear Regression - Train a model with first 100 rows of data for
72 epochs. Accuracy: 0.88
Model (106) - Percetron - Train a model with first 100 rows of data for 77
epochs. Accuracy: 0.87
Model (107) - Adaline - Train a model with first 100 rows of data for 77 epochs.
Accuracy: 0.88
Model (108) - Linear Regression - Train a model with first 100 rows of data for
77 epochs. Accuracy: 0.88
Model (109) - Percetron - Train a model with first 100 rows of data for 82
epochs. Accuracy: 0.88
Model (110) - Adaline - Train a model with first 100 rows of data for 82 epochs.
Accuracy: 0.88
Model (111) - Linear Regression - Train a model with first 100 rows of data for
```

```
82 epochs. Accuracy: 0.88
Model (112) - Percetron - Train a model with first 100 rows of data for 87
epochs. Accuracy: 0.88
Model (113) - Adaline - Train a model with first 100 rows of data for 87 epochs.
Accuracy: 0.88
Model (114) - Linear Regression - Train a model with first 100 rows of data for
87 epochs. Accuracy: 0.88
Model (115) - Percetron - Train a model with first 100 rows of data for 92
epochs. Accuracy: 0.88
Model (116) - Adaline - Train a model with first 100 rows of data for 92 epochs.
Accuracy: 0.88
Model (117) - Linear Regression - Train a model with first 100 rows of data for
92 epochs. Accuracy: 0.88
Model (118) - Percetron - Train a model with first 100 rows of data for 97
epochs. Accuracy: 0.85
Model (119) - Adaline - Train a model with first 100 rows of data for 97 epochs.
Accuracy: 0.88
Model (120) - Linear Regression - Train a model with first 100 rows of data for
97 epochs. Accuracy: 0.88
Model (121) - Percetron - Train a model with first 150 rows of data for 2
epochs. Accuracy: 0.84
Model (122) - Adaline - Train a model with first 150 rows of data for 2 epochs.
Accuracy: 0.85
Model (123) - Linear Regression - Train a model with first 150 rows of data for
2 epochs. Accuracy: 0.81
Model (124) - Percetron - Train a model with first 150 rows of data for 7
epochs. Accuracy: 0.87
Model (125) - Adaline - Train a model with first 150 rows of data for 7 epochs.
Accuracy: 0.87
Model (126) - Linear Regression - Train a model with first 150 rows of data for
7 epochs. Accuracy: 0.86
Model (127) - Percetron - Train a model with first 150 rows of data for 12
epochs. Accuracy: 0.87
Model (128) - Adaline - Train a model with first 150 rows of data for 12 epochs.
Accuracy: 0.87
Model (129) - Linear Regression - Train a model with first 150 rows of data for
12 epochs. Accuracy: 0.87
Model (130) - Percetron - Train a model with first 150 rows of data for 17
epochs. Accuracy: 0.83
Model (131) - Adaline - Train a model with first 150 rows of data for 17 epochs.
Accuracy: 0.87
Model (132) - Linear Regression - Train a model with first 150 rows of data for
17 epochs. Accuracy: 0.87
Model (133) - Percetron - Train a model with first 150 rows of data for 22
epochs. Accuracy: 0.85
Model (134) - Adaline - Train a model with first 150 rows of data for 22 epochs.
Accuracy: 0.88
Model (135) - Linear Regression - Train a model with first 150 rows of data for
```

```
22 epochs. Accuracy: 0.87
Model (136) - Percetron - Train a model with first 150 rows of data for 27
epochs. Accuracy: 0.84
Model (137) - Adaline - Train a model with first 150 rows of data for 27 epochs.
Accuracy: 0.88
Model (138) - Linear Regression - Train a model with first 150 rows of data for
27 epochs. Accuracy: 0.87
Model (139) - Percetron - Train a model with first 150 rows of data for 32
epochs. Accuracy: 0.87
Model (140) - Adaline - Train a model with first 150 rows of data for 32 epochs.
Accuracy: 0.88
Model (141) - Linear Regression - Train a model with first 150 rows of data for
32 epochs. Accuracy: 0.87
Model (142) - Percetron - Train a model with first 150 rows of data for 37
epochs. Accuracy: 0.86
Model (143) - Adaline - Train a model with first 150 rows of data for 37 epochs.
Accuracy: 0.88
Model (144) - Linear Regression - Train a model with first 150 rows of data for
37 epochs. Accuracy: 0.87
Model (145) - Percetron - Train a model with first 150 rows of data for 42
epochs. Accuracy: 0.88
Model (146) - Adaline - Train a model with first 150 rows of data for 42 epochs.
Accuracy: 0.88
Model (147) - Linear Regression - Train a model with first 150 rows of data for
42 epochs. Accuracy: 0.88
Model (148) - Percetron - Train a model with first 150 rows of data for 47
epochs. Accuracy: 0.87
Model (149) - Adaline - Train a model with first 150 rows of data for 47 epochs.
Accuracy: 0.88
Model (150) - Linear Regression - Train a model with first 150 rows of data for
47 epochs. Accuracy: 0.87
Model (151) - Percetron - Train a model with first 150 rows of data for 52
epochs. Accuracy: 0.80
Model (152) - Adaline - Train a model with first 150 rows of data for 52 epochs.
Accuracy: 0.88
Model (153) - Linear Regression - Train a model with first 150 rows of data for
52 epochs. Accuracy: 0.88
Model (154) - Percetron - Train a model with first 150 rows of data for 57
epochs. Accuracy: 0.85
Model (155) - Adaline - Train a model with first 150 rows of data for 57 epochs.
Accuracy: 0.88
Model (156) - Linear Regression - Train a model with first 150 rows of data for
57 epochs. Accuracy: 0.88
Model (157) - Percetron - Train a model with first 150 rows of data for 62
epochs. Accuracy: 0.80
Model (158) - Adaline - Train a model with first 150 rows of data for 62 epochs.
Accuracy: 0.87
Model (159) - Linear Regression - Train a model with first 150 rows of data for
```

```
62 epochs. Accuracy: 0.88
Model (160) - Percetron - Train a model with first 150 rows of data for 67
epochs. Accuracy: 0.84
Model (161) - Adaline - Train a model with first 150 rows of data for 67 epochs.
Accuracy: 0.87
Model (162) - Linear Regression - Train a model with first 150 rows of data for
67 epochs. Accuracy: 0.88
Model (163) - Percetron - Train a model with first 150 rows of data for 72
epochs. Accuracy: 0.87
Model (164) - Adaline - Train a model with first 150 rows of data for 72 epochs.
Accuracy: 0.87
Model (165) - Linear Regression - Train a model with first 150 rows of data for
72 epochs. Accuracy: 0.88
Model (166) - Percetron - Train a model with first 150 rows of data for 77
epochs. Accuracy: 0.85
Model (167) - Adaline - Train a model with first 150 rows of data for 77 epochs.
Accuracy: 0.87
Model (168) - Linear Regression - Train a model with first 150 rows of data for
77 epochs. Accuracy: 0.88
Model (169) - Percetron - Train a model with first 150 rows of data for 82
epochs. Accuracy: 0.89
Model (170) - Adaline - Train a model with first 150 rows of data for 82 epochs.
Accuracy: 0.87
Model (171) - Linear Regression - Train a model with first 150 rows of data for
82 epochs. Accuracy: 0.88
Model (172) - Percetron - Train a model with first 150 rows of data for 87
epochs. Accuracy: 0.85
Model (173) - Adaline - Train a model with first 150 rows of data for 87 epochs.
Accuracy: 0.87
Model (174) - Linear Regression - Train a model with first 150 rows of data for
87 epochs. Accuracy: 0.88
Model (175) - Percetron - Train a model with first 150 rows of data for 92
epochs. Accuracy: 0.74
Model (176) - Adaline - Train a model with first 150 rows of data for 92 epochs.
Accuracy: 0.87
Model (177) - Linear Regression - Train a model with first 150 rows of data for
92 epochs. Accuracy: 0.88
Model (178) - Percetron - Train a model with first 150 rows of data for 97
epochs. Accuracy: 0.88
Model (179) - Adaline - Train a model with first 150 rows of data for 97 epochs.
Accuracy: 0.87
Model (180) - Linear Regression - Train a model with first 150 rows of data for
97 epochs. Accuracy: 0.88
Model (181) - Percetron - Train a model with first 200 rows of data for 2
epochs. Accuracy: 0.87
Model (182) - Adaline - Train a model with first 200 rows of data for 2 epochs.
Accuracy: 0.85
Model (183) - Linear Regression - Train a model with first 200 rows of data for
```

```
2 epochs. Accuracy: 0.81
Model (184) - Percetron - Train a model with first 200 rows of data for 7
epochs. Accuracy: 0.90
Model (185) - Adaline - Train a model with first 200 rows of data for 7 epochs.
Accuracy: 0.87
Model (186) - Linear Regression - Train a model with first 200 rows of data for
7 epochs. Accuracy: 0.86
Model (187) - Percetron - Train a model with first 200 rows of data for 12
epochs. Accuracy: 0.90
Model (188) - Adaline - Train a model with first 200 rows of data for 12 epochs.
Accuracy: 0.87
Model (189) - Linear Regression - Train a model with first 200 rows of data for
12 epochs. Accuracy: 0.87
Model (190) - Percetron - Train a model with first 200 rows of data for 17
epochs. Accuracy: 0.83
Model (191) - Adaline - Train a model with first 200 rows of data for 17 epochs.
Accuracy: 0.87
Model (192) - Linear Regression - Train a model with first 200 rows of data for
17 epochs. Accuracy: 0.87
Model (193) - Percetron - Train a model with first 200 rows of data for 22
epochs. Accuracy: 0.89
Model (194) - Adaline - Train a model with first 200 rows of data for 22 epochs.
Accuracy: 0.87
Model (195) - Linear Regression - Train a model with first 200 rows of data for
22 epochs. Accuracy: 0.87
Model (196) - Percetron - Train a model with first 200 rows of data for 27
epochs. Accuracy: 0.85
Model (197) - Adaline - Train a model with first 200 rows of data for 27 epochs.
Accuracy: 0.87
Model (198) - Linear Regression - Train a model with first 200 rows of data for
27 epochs. Accuracy: 0.87
Model (199) - Percetron - Train a model with first 200 rows of data for 32
epochs. Accuracy: 0.84
Model (200) - Adaline - Train a model with first 200 rows of data for 32 epochs.
Accuracy: 0.87
Model (201) - Linear Regression - Train a model with first 200 rows of data for
32 epochs. Accuracy: 0.87
Model (202) - Percetron - Train a model with first 200 rows of data for 37
epochs. Accuracy: 0.88
Model (203) - Adaline - Train a model with first 200 rows of data for 37 epochs.
Accuracy: 0.87
Model (204) - Linear Regression - Train a model with first 200 rows of data for
37 epochs. Accuracy: 0.87
Model (205) - Percetron - Train a model with first 200 rows of data for 42
epochs. Accuracy: 0.90
Model (206) - Adaline - Train a model with first 200 rows of data for 42 epochs.
Accuracy: 0.88
Model (207) - Linear Regression - Train a model with first 200 rows of data for
```

```
42 epochs. Accuracy: 0.87
Model (208) - Percetron - Train a model with first 200 rows of data for 47
epochs. Accuracy: 0.86
Model (209) - Adaline - Train a model with first 200 rows of data for 47 epochs.
Accuracy: 0.88
Model (210) - Linear Regression - Train a model with first 200 rows of data for
47 epochs. Accuracy: 0.87
Model (211) - Percetron - Train a model with first 200 rows of data for 52
epochs. Accuracy: 0.87
Model (212) - Adaline - Train a model with first 200 rows of data for 52 epochs.
Accuracy: 0.88
Model (213) - Linear Regression - Train a model with first 200 rows of data for
52 epochs. Accuracy: 0.87
Model (214) - Percetron - Train a model with first 200 rows of data for 57
epochs. Accuracy: 0.83
Model (215) - Adaline - Train a model with first 200 rows of data for 57 epochs.
Accuracy: 0.88
Model (216) - Linear Regression - Train a model with first 200 rows of data for
57 epochs. Accuracy: 0.87
Model (217) - Percetron - Train a model with first 200 rows of data for 62
epochs. Accuracy: 0.86
Model (218) - Adaline - Train a model with first 200 rows of data for 62 epochs.
Accuracy: 0.88
Model (219) - Linear Regression - Train a model with first 200 rows of data for
62 epochs. Accuracy: 0.87
Model (220) - Percetron - Train a model with first 200 rows of data for 67
epochs. Accuracy: 0.84
Model (221) - Adaline - Train a model with first 200 rows of data for 67 epochs.
Accuracy: 0.88
Model (222) - Linear Regression - Train a model with first 200 rows of data for
67 epochs. Accuracy: 0.87
Model (223) - Percetron - Train a model with first 200 rows of data for 72
epochs. Accuracy: 0.89
Model (224) - Adaline - Train a model with first 200 rows of data for 72 epochs.
Accuracy: 0.88
Model (225) - Linear Regression - Train a model with first 200 rows of data for
72 epochs. Accuracy: 0.87
Model (226) - Percetron - Train a model with first 200 rows of data for 77
epochs. Accuracy: 0.78
Model (227) - Adaline - Train a model with first 200 rows of data for 77 epochs.
Accuracy: 0.88
Model (228) - Linear Regression - Train a model with first 200 rows of data for
77 epochs. Accuracy: 0.87
Model (229) - Percetron - Train a model with first 200 rows of data for 82
epochs. Accuracy: 0.81
Model (230) - Adaline - Train a model with first 200 rows of data for 82 epochs.
Accuracy: 0.88
Model (231) - Linear Regression - Train a model with first 200 rows of data for
```

82 epochs. Accuracy: 0.87
Model (232) - Percetron - Train a model with first 200 rows of data for 87 epochs. Accuracy: 0.90
Model (233) - Adaline - Train a model with first 200 rows of data for 87 epochs. Accuracy: 0.87
Model (234) - Linear Regression - Train a model with first 200 rows of data for 87 epochs. Accuracy: 0.87
Model (235) - Percetron - Train a model with first 200 rows of data for 92 epochs. Accuracy: 0.87
Model (236) - Adaline - Train a model with first 200 rows of data for 92 epochs. Accuracy: 0.87
Model (237) - Linear Regression - Train a model with first 200 rows of data for 92 epochs. Accuracy: 0.87
Model (238) - Percetron - Train a model with first 200 rows of data for 97 epochs. Accuracy: 0.83
Model (239) - Adaline - Train a model with first 200 rows of data for 97 epochs. Accuracy: 0.87
Model (240) - Linear Regression - Train a model with first 200 rows of data for 97 epochs. Accuracy: 0.87
Model (241) - Percetron - Train a model with first 250 rows of data for 2 epochs. Accuracy: 0.87
Model (242) - Adaline - Train a model with first 250 rows of data for 2 epochs. Accuracy: 0.85
Model (243) - Linear Regression - Train a model with first 250 rows of data for 2 epochs. Accuracy: 0.82
Model (244) - Percetron - Train a model with first 250 rows of data for 7 epochs. Accuracy: 0.86
Model (245) - Adaline - Train a model with first 250 rows of data for 7 epochs. Accuracy: 0.86
Model (246) - Linear Regression - Train a model with first 250 rows of data for 7 epochs. Accuracy: 0.87
Model (247) - Percetron - Train a model with first 250 rows of data for 12 epochs. Accuracy: 0.87
Model (248) - Adaline - Train a model with first 250 rows of data for 12 epochs. Accuracy: 0.87
Model (249) - Linear Regression - Train a model with first 250 rows of data for 12 epochs. Accuracy: 0.86
Model (250) - Percetron - Train a model with first 250 rows of data for 17 epochs. Accuracy: 0.88
Model (251) - Adaline - Train a model with first 250 rows of data for 17 epochs. Accuracy: 0.87
Model (252) - Linear Regression - Train a model with first 250 rows of data for 17 epochs. Accuracy: 0.86
Model (253) - Percetron - Train a model with first 250 rows of data for 22 epochs. Accuracy: 0.88
Model (254) - Adaline - Train a model with first 250 rows of data for 22 epochs. Accuracy: 0.87
Model (255) - Linear Regression - Train a model with first 250 rows of data for

```
22 epochs. Accuracy: 0.87
Model (256) - Percetron - Train a model with first 250 rows of data for 27
epochs. Accuracy: 0.86
Model (257) - Adaline - Train a model with first 250 rows of data for 27 epochs.
Accuracy: 0.87
Model (258) - Linear Regression - Train a model with first 250 rows of data for
27 epochs. Accuracy: 0.87
Model (259) - Percetron - Train a model with first 250 rows of data for 32
epochs. Accuracy: 0.87
Model (260) - Adaline - Train a model with first 250 rows of data for 32 epochs.
Accuracy: 0.88
Model (261) - Linear Regression - Train a model with first 250 rows of data for
32 epochs. Accuracy: 0.87
Model (262) - Percetron - Train a model with first 250 rows of data for 37
epochs. Accuracy: 0.85
Model (263) - Adaline - Train a model with first 250 rows of data for 37 epochs.
Accuracy: 0.88
Model (264) - Linear Regression - Train a model with first 250 rows of data for
37 epochs. Accuracy: 0.87
Model (265) - Percetron - Train a model with first 250 rows of data for 42
epochs. Accuracy: 0.88
Model (266) - Adaline - Train a model with first 250 rows of data for 42 epochs.
Accuracy: 0.88
Model (267) - Linear Regression - Train a model with first 250 rows of data for
42 epochs. Accuracy: 0.87
Model (268) - Percetron - Train a model with first 250 rows of data for 47
epochs. Accuracy: 0.86
Model (269) - Adaline - Train a model with first 250 rows of data for 47 epochs.
Accuracy: 0.87
Model (270) - Linear Regression - Train a model with first 250 rows of data for
47 epochs. Accuracy: 0.87
Model (271) - Percetron - Train a model with first 250 rows of data for 52
epochs. Accuracy: 0.88
Model (272) - Adaline - Train a model with first 250 rows of data for 52 epochs.
Accuracy: 0.87
Model (273) - Linear Regression - Train a model with first 250 rows of data for
52 epochs. Accuracy: 0.87
Model (274) - Percetron - Train a model with first 250 rows of data for 57
epochs. Accuracy: 0.89
Model (275) - Adaline - Train a model with first 250 rows of data for 57 epochs.
Accuracy: 0.87
Model (276) - Linear Regression - Train a model with first 250 rows of data for
57 epochs. Accuracy: 0.87
Model (277) - Percetron - Train a model with first 250 rows of data for 62
epochs. Accuracy: 0.86
Model (278) - Adaline - Train a model with first 250 rows of data for 62 epochs.
Accuracy: 0.87
Model (279) - Linear Regression - Train a model with first 250 rows of data for
```

```
62 epochs. Accuracy: 0.87
Model (280) - Percetron - Train a model with first 250 rows of data for 67
epochs. Accuracy: 0.84
Model (281) - Adaline - Train a model with first 250 rows of data for 67 epochs.
Accuracy: 0.87
Model (282) - Linear Regression - Train a model with first 250 rows of data for
67 epochs. Accuracy: 0.86
Model (283) - Percetron - Train a model with first 250 rows of data for 72
epochs. Accuracy: 0.89
Model (284) - Adaline - Train a model with first 250 rows of data for 72 epochs.
Accuracy: 0.87
Model (285) - Linear Regression - Train a model with first 250 rows of data for
72 epochs. Accuracy: 0.86
Model (286) - Percetron - Train a model with first 250 rows of data for 77
epochs. Accuracy: 0.89
Model (287) - Adaline - Train a model with first 250 rows of data for 77 epochs.
Accuracy: 0.87
Model (288) - Linear Regression - Train a model with first 250 rows of data for
77 epochs. Accuracy: 0.87
Model (289) - Percetron - Train a model with first 250 rows of data for 82
epochs. Accuracy: 0.87
Model (290) - Adaline - Train a model with first 250 rows of data for 82 epochs.
Accuracy: 0.87
Model (291) - Linear Regression - Train a model with first 250 rows of data for
82 epochs. Accuracy: 0.87
Model (292) - Percetron - Train a model with first 250 rows of data for 87
epochs. Accuracy: 0.88
Model (293) - Adaline - Train a model with first 250 rows of data for 87 epochs.
Accuracy: 0.87
Model (294) - Linear Regression - Train a model with first 250 rows of data for
87 epochs. Accuracy: 0.87
Model (295) - Percetron - Train a model with first 250 rows of data for 92
epochs. Accuracy: 0.88
Model (296) - Adaline - Train a model with first 250 rows of data for 92 epochs.
Accuracy: 0.87
Model (297) - Linear Regression - Train a model with first 250 rows of data for
92 epochs. Accuracy: 0.87
Model (298) - Percetron - Train a model with first 250 rows of data for 97
epochs. Accuracy: 0.89
Model (299) - Adaline - Train a model with first 250 rows of data for 97 epochs.
Accuracy: 0.87
Model (300) - Linear Regression - Train a model with first 250 rows of data for
97 epochs. Accuracy: 0.87
Model (301) - Percetron - Train a model with first 300 rows of data for 2
epochs. Accuracy: 0.85
Model (302) - Adaline - Train a model with first 300 rows of data for 2 epochs.
Accuracy: 0.86
Model (303) - Linear Regression - Train a model with first 300 rows of data for
```

```
2 epochs. Accuracy: 0.85
Model (304) - Percetron - Train a model with first 300 rows of data for 7
epochs. Accuracy: 0.85
Model (305) - Adaline - Train a model with first 300 rows of data for 7 epochs.
Accuracy: 0.87
Model (306) - Linear Regression - Train a model with first 300 rows of data for
7 epochs. Accuracy: 0.86
Model (307) - Percetron - Train a model with first 300 rows of data for 12
epochs. Accuracy: 0.85
Model (308) - Adaline - Train a model with first 300 rows of data for 12 epochs.
Accuracy: 0.86
Model (309) - Linear Regression - Train a model with first 300 rows of data for
12 epochs. Accuracy: 0.87
Model (310) - Percetron - Train a model with first 300 rows of data for 17
epochs. Accuracy: 0.85
Model (311) - Adaline - Train a model with first 300 rows of data for 17 epochs.
Accuracy: 0.87
Model (312) - Linear Regression - Train a model with first 300 rows of data for
17 epochs. Accuracy: 0.87
Model (313) - Percetron - Train a model with first 300 rows of data for 22
epochs. Accuracy: 0.89
Model (314) - Adaline - Train a model with first 300 rows of data for 22 epochs.
Accuracy: 0.88
Model (315) - Linear Regression - Train a model with first 300 rows of data for
22 epochs. Accuracy: 0.87
Model (316) - Percetron - Train a model with first 300 rows of data for 27
epochs. Accuracy: 0.78
Model (317) - Adaline - Train a model with first 300 rows of data for 27 epochs.
Accuracy: 0.88
Model (318) - Linear Regression - Train a model with first 300 rows of data for
27 epochs. Accuracy: 0.87
Model (319) - Percetron - Train a model with first 300 rows of data for 32
epochs. Accuracy: 0.86
Model (320) - Adaline - Train a model with first 300 rows of data for 32 epochs.
Accuracy: 0.88
Model (321) - Linear Regression - Train a model with first 300 rows of data for
32 epochs. Accuracy: 0.87
Model (322) - Percetron - Train a model with first 300 rows of data for 37
epochs. Accuracy: 0.87
Model (323) - Adaline - Train a model with first 300 rows of data for 37 epochs.
Accuracy: 0.88
Model (324) - Linear Regression - Train a model with first 300 rows of data for
37 epochs. Accuracy: 0.87
Model (325) - Percetron - Train a model with first 300 rows of data for 42
epochs. Accuracy: 0.85
Model (326) - Adaline - Train a model with first 300 rows of data for 42 epochs.
Accuracy: 0.88
Model (327) - Linear Regression - Train a model with first 300 rows of data for
```

```
42 epochs. Accuracy: 0.87
Model (328) - Percetron - Train a model with first 300 rows of data for 47
epochs. Accuracy: 0.86
Model (329) - Adaline - Train a model with first 300 rows of data for 47 epochs.
Accuracy: 0.87
Model (330) - Linear Regression - Train a model with first 300 rows of data for
47 epochs. Accuracy: 0.87
Model (331) - Percetron - Train a model with first 300 rows of data for 52
epochs. Accuracy: 0.89
Model (332) - Adaline - Train a model with first 300 rows of data for 52 epochs.
Accuracy: 0.87
Model (333) - Linear Regression - Train a model with first 300 rows of data for
52 epochs. Accuracy: 0.87
Model (334) - Percetron - Train a model with first 300 rows of data for 57
epochs. Accuracy: 0.86
Model (335) - Adaline - Train a model with first 300 rows of data for 57 epochs.
Accuracy: 0.87
Model (336) - Linear Regression - Train a model with first 300 rows of data for
57 epochs. Accuracy: 0.87
Model (337) - Percetron - Train a model with first 300 rows of data for 62
epochs. Accuracy: 0.89
Model (338) - Adaline - Train a model with first 300 rows of data for 62 epochs.
Accuracy: 0.87
Model (339) - Linear Regression - Train a model with first 300 rows of data for
62 epochs. Accuracy: 0.87
Model (340) - Percetron - Train a model with first 300 rows of data for 67
epochs. Accuracy: 0.81
Model (341) - Adaline - Train a model with first 300 rows of data for 67 epochs.
Accuracy: 0.87
Model (342) - Linear Regression - Train a model with first 300 rows of data for
67 epochs. Accuracy: 0.87
Model (343) - Percetron - Train a model with first 300 rows of data for 72
epochs. Accuracy: 0.86
Model (344) - Adaline - Train a model with first 300 rows of data for 72 epochs.
Accuracy: 0.87
Model (345) - Linear Regression - Train a model with first 300 rows of data for
72 epochs. Accuracy: 0.87
Model (346) - Percetron - Train a model with first 300 rows of data for 77
epochs. Accuracy: 0.87
Model (347) - Adaline - Train a model with first 300 rows of data for 77 epochs.
Accuracy: 0.87
Model (348) - Linear Regression - Train a model with first 300 rows of data for
77 epochs. Accuracy: 0.87
Model (349) - Percetron - Train a model with first 300 rows of data for 82
epochs. Accuracy: 0.87
Model (350) - Adaline - Train a model with first 300 rows of data for 82 epochs.
Accuracy: 0.87
Model (351) - Linear Regression - Train a model with first 300 rows of data for
```

```
82 epochs. Accuracy: 0.87
Model (352) - Percetron - Train a model with first 300 rows of data for 87
epochs. Accuracy: 0.87
Model (353) - Adaline - Train a model with first 300 rows of data for 87 epochs.
Accuracy: 0.87
Model (354) - Linear Regression - Train a model with first 300 rows of data for
87 epochs. Accuracy: 0.88
Model (355) - Percetron - Train a model with first 300 rows of data for 92
epochs. Accuracy: 0.89
Model (356) - Adaline - Train a model with first 300 rows of data for 92 epochs.
Accuracy: 0.87
Model (357) - Linear Regression - Train a model with first 300 rows of data for
92 epochs. Accuracy: 0.88
Model (358) - Percetron - Train a model with first 300 rows of data for 97
epochs. Accuracy: 0.85
Model (359) - Adaline - Train a model with first 300 rows of data for 97 epochs.
Accuracy: 0.87
Model (360) - Linear Regression - Train a model with first 300 rows of data for
97 epochs. Accuracy: 0.88
Model (361) - Percetron - Train a model with first 350 rows of data for 2
epochs. Accuracy: 0.86
Model (362) - Adaline - Train a model with first 350 rows of data for 2 epochs.
Accuracy: 0.87
Model (363) - Linear Regression - Train a model with first 350 rows of data for
2 epochs. Accuracy: 0.85
Model (364) - Percetron - Train a model with first 350 rows of data for 7
epochs. Accuracy: 0.88
Model (365) - Adaline - Train a model with first 350 rows of data for 7 epochs.
Accuracy: 0.86
Model (366) - Linear Regression - Train a model with first 350 rows of data for
7 epochs. Accuracy: 0.86
Model (367) - Percetron - Train a model with first 350 rows of data for 12
epochs. Accuracy: 0.88
Model (368) - Adaline - Train a model with first 350 rows of data for 12 epochs.
Accuracy: 0.87
Model (369) - Linear Regression - Train a model with first 350 rows of data for
12 epochs. Accuracy: 0.87
Model (370) - Percetron - Train a model with first 350 rows of data for 17
epochs. Accuracy: 0.87
Model (371) - Adaline - Train a model with first 350 rows of data for 17 epochs.
Accuracy: 0.87
Model (372) - Linear Regression - Train a model with first 350 rows of data for
17 epochs. Accuracy: 0.86
Model (373) - Percetron - Train a model with first 350 rows of data for 22
epochs. Accuracy: 0.85
Model (374) - Adaline - Train a model with first 350 rows of data for 22 epochs.
Accuracy: 0.87
Model (375) - Linear Regression - Train a model with first 350 rows of data for
```

```
22 epochs. Accuracy: 0.87
Model (376) - Percetron - Train a model with first 350 rows of data for 27
epochs. Accuracy: 0.90
Model (377) - Adaline - Train a model with first 350 rows of data for 27 epochs.
Accuracy: 0.87
Model (378) - Linear Regression - Train a model with first 350 rows of data for
27 epochs. Accuracy: 0.87
Model (379) - Percetron - Train a model with first 350 rows of data for 32
epochs. Accuracy: 0.89
Model (380) - Adaline - Train a model with first 350 rows of data for 32 epochs.
Accuracy: 0.87
Model (381) - Linear Regression - Train a model with first 350 rows of data for
32 epochs. Accuracy: 0.87
Model (382) - Percetron - Train a model with first 350 rows of data for 37
epochs. Accuracy: 0.78
Model (383) - Adaline - Train a model with first 350 rows of data for 37 epochs.
Accuracy: 0.88
Model (384) - Linear Regression - Train a model with first 350 rows of data for
37 epochs. Accuracy: 0.87
Model (385) - Percetron - Train a model with first 350 rows of data for 42
epochs. Accuracy: 0.85
Model (386) - Adaline - Train a model with first 350 rows of data for 42 epochs.
Accuracy: 0.88
Model (387) - Linear Regression - Train a model with first 350 rows of data for
42 epochs. Accuracy: 0.87
Model (388) - Percetron - Train a model with first 350 rows of data for 47
epochs. Accuracy: 0.87
Model (389) - Adaline - Train a model with first 350 rows of data for 47 epochs.
Accuracy: 0.87
Model (390) - Linear Regression - Train a model with first 350 rows of data for
47 epochs. Accuracy: 0.87
Model (391) - Percetron - Train a model with first 350 rows of data for 52
epochs. Accuracy: 0.89
Model (392) - Adaline - Train a model with first 350 rows of data for 52 epochs.
Accuracy: 0.88
Model (393) - Linear Regression - Train a model with first 350 rows of data for
52 epochs. Accuracy: 0.87
Model (394) - Percetron - Train a model with first 350 rows of data for 57
epochs. Accuracy: 0.81
Model (395) - Adaline - Train a model with first 350 rows of data for 57 epochs.
Accuracy: 0.88
Model (396) - Linear Regression - Train a model with first 350 rows of data for
57 epochs. Accuracy: 0.87
Model (397) - Percetron - Train a model with first 350 rows of data for 62
epochs. Accuracy: 0.89
Model (398) - Adaline - Train a model with first 350 rows of data for 62 epochs.
Accuracy: 0.88
Model (399) - Linear Regression - Train a model with first 350 rows of data for
```

```
62 epochs. Accuracy: 0.87
Model (400) - Percetron - Train a model with first 350 rows of data for 67
epochs. Accuracy: 0.82
Model (401) - Adaline - Train a model with first 350 rows of data for 67 epochs.
Accuracy: 0.88
Model (402) - Linear Regression - Train a model with first 350 rows of data for
67 epochs. Accuracy: 0.87
Model (403) - Percetron - Train a model with first 350 rows of data for 72
epochs. Accuracy: 0.84
Model (404) - Adaline - Train a model with first 350 rows of data for 72 epochs.
Accuracy: 0.88
Model (405) - Linear Regression - Train a model with first 350 rows of data for
72 epochs. Accuracy: 0.87
Model (406) - Percetron - Train a model with first 350 rows of data for 77
epochs. Accuracy: 0.89
Model (407) - Adaline - Train a model with first 350 rows of data for 77 epochs.
Accuracy: 0.88
Model (408) - Linear Regression - Train a model with first 350 rows of data for
77 epochs. Accuracy: 0.87
Model (409) - Percetron - Train a model with first 350 rows of data for 82
epochs. Accuracy: 0.90
Model (410) - Adaline - Train a model with first 350 rows of data for 82 epochs.
Accuracy: 0.88
Model (411) - Linear Regression - Train a model with first 350 rows of data for
82 epochs. Accuracy: 0.87
Model (412) - Percetron - Train a model with first 350 rows of data for 87
epochs. Accuracy: 0.86
Model (413) - Adaline - Train a model with first 350 rows of data for 87 epochs.
Accuracy: 0.88
Model (414) - Linear Regression - Train a model with first 350 rows of data for
87 epochs. Accuracy: 0.87
Model (415) - Percetron - Train a model with first 350 rows of data for 92
epochs. Accuracy: 0.91
Model (416) - Adaline - Train a model with first 350 rows of data for 92 epochs.
Accuracy: 0.88
Model (417) - Linear Regression - Train a model with first 350 rows of data for
92 epochs. Accuracy: 0.87
Model (418) - Percetron - Train a model with first 350 rows of data for 97
epochs. Accuracy: 0.88
Model (419) - Adaline - Train a model with first 350 rows of data for 97 epochs.
Accuracy: 0.88
Model (420) - Linear Regression - Train a model with first 350 rows of data for
97 epochs. Accuracy: 0.87
Model (421) - Percetron - Train a model with first 400 rows of data for 2
epochs. Accuracy: 0.86
Model (422) - Adaline - Train a model with first 400 rows of data for 2 epochs.
Accuracy: 0.86
Model (423) - Linear Regression - Train a model with first 400 rows of data for
```

2 epochs. Accuracy: 0.86
Model (424) - Percetron - Train a model with first 400 rows of data for 7 epochs. Accuracy: 0.88
Model (425) - Adaline - Train a model with first 400 rows of data for 7 epochs. Accuracy: 0.87
Model (426) - Linear Regression - Train a model with first 400 rows of data for 7 epochs. Accuracy: 0.87
Model (427) - Percetron - Train a model with first 400 rows of data for 12 epochs. Accuracy: 0.90
Model (428) - Adaline - Train a model with first 400 rows of data for 12 epochs. Accuracy: 0.87
Model (429) - Linear Regression - Train a model with first 400 rows of data for 12 epochs. Accuracy: 0.87
Model (430) - Percetron - Train a model with first 400 rows of data for 17 epochs. Accuracy: 0.90
Model (431) - Adaline - Train a model with first 400 rows of data for 17 epochs. Accuracy: 0.87
Model (432) - Linear Regression - Train a model with first 400 rows of data for 17 epochs. Accuracy: 0.88
Model (433) - Percetron - Train a model with first 400 rows of data for 22 epochs. Accuracy: 0.87
Model (434) - Adaline - Train a model with first 400 rows of data for 22 epochs. Accuracy: 0.87
Model (435) - Linear Regression - Train a model with first 400 rows of data for 22 epochs. Accuracy: 0.88
Model (436) - Percetron - Train a model with first 400 rows of data for 27 epochs. Accuracy: 0.87
Model (437) - Adaline - Train a model with first 400 rows of data for 27 epochs. Accuracy: 0.88
Model (438) - Linear Regression - Train a model with first 400 rows of data for 27 epochs. Accuracy: 0.88
Model (439) - Percetron - Train a model with first 400 rows of data for 32 epochs. Accuracy: 0.88
Model (440) - Adaline - Train a model with first 400 rows of data for 32 epochs. Accuracy: 0.88
Model (441) - Linear Regression - Train a model with first 400 rows of data for 32 epochs. Accuracy: 0.88
Model (442) - Percetron - Train a model with first 400 rows of data for 37 epochs. Accuracy: 0.86
Model (443) - Adaline - Train a model with first 400 rows of data for 37 epochs. Accuracy: 0.88
Model (444) - Linear Regression - Train a model with first 400 rows of data for 37 epochs. Accuracy: 0.88
Model (445) - Percetron - Train a model with first 400 rows of data for 42 epochs. Accuracy: 0.88
Model (446) - Adaline - Train a model with first 400 rows of data for 42 epochs. Accuracy: 0.88
Model (447) - Linear Regression - Train a model with first 400 rows of data for

```
42 epochs. Accuracy: 0.87
Model (448) - Percetron - Train a model with first 400 rows of data for 47
epochs. Accuracy: 0.76
Model (449) - Adaline - Train a model with first 400 rows of data for 47 epochs.
Accuracy: 0.88
Model (450) - Linear Regression - Train a model with first 400 rows of data for
47 epochs. Accuracy: 0.87
Model (451) - Percetron - Train a model with first 400 rows of data for 52
epochs. Accuracy: 0.84
Model (452) - Adaline - Train a model with first 400 rows of data for 52 epochs.
Accuracy: 0.87
Model (453) - Linear Regression - Train a model with first 400 rows of data for
52 epochs. Accuracy: 0.87
Model (454) - Percetron - Train a model with first 400 rows of data for 57
epochs. Accuracy: 0.80
Model (455) - Adaline - Train a model with first 400 rows of data for 57 epochs.
Accuracy: 0.87
Model (456) - Linear Regression - Train a model with first 400 rows of data for
57 epochs. Accuracy: 0.87
Model (457) - Percetron - Train a model with first 400 rows of data for 62
epochs. Accuracy: 0.90
Model (458) - Adaline - Train a model with first 400 rows of data for 62 epochs.
Accuracy: 0.87
Model (459) - Linear Regression - Train a model with first 400 rows of data for
62 epochs. Accuracy: 0.87
Model (460) - Percetron - Train a model with first 400 rows of data for 67
epochs. Accuracy: 0.71
Model (461) - Adaline - Train a model with first 400 rows of data for 67 epochs.
Accuracy: 0.87
Model (462) - Linear Regression - Train a model with first 400 rows of data for
67 epochs. Accuracy: 0.87
Model (463) - Percetron - Train a model with first 400 rows of data for 72
epochs. Accuracy: 0.89
Model (464) - Adaline - Train a model with first 400 rows of data for 72 epochs.
Accuracy: 0.87
Model (465) - Linear Regression - Train a model with first 400 rows of data for
72 epochs. Accuracy: 0.87
Model (466) - Percetron - Train a model with first 400 rows of data for 77
epochs. Accuracy: 0.64
Model (467) - Adaline - Train a model with first 400 rows of data for 77 epochs.
Accuracy: 0.87
Model (468) - Linear Regression - Train a model with first 400 rows of data for
77 epochs. Accuracy: 0.87
Model (469) - Percetron - Train a model with first 400 rows of data for 82
epochs. Accuracy: 0.89
Model (470) - Adaline - Train a model with first 400 rows of data for 82 epochs.
Accuracy: 0.87
Model (471) - Linear Regression - Train a model with first 400 rows of data for
```

```
82 epochs. Accuracy: 0.87
Model (472) - Percetron - Train a model with first 400 rows of data for 87
epochs. Accuracy: 0.89
Model (473) - Adaline - Train a model with first 400 rows of data for 87 epochs.
Accuracy: 0.87
Model (474) - Linear Regression - Train a model with first 400 rows of data for
87 epochs. Accuracy: 0.87
Model (475) - Percetron - Train a model with first 400 rows of data for 92
epochs. Accuracy: 0.89
Model (476) - Adaline - Train a model with first 400 rows of data for 92 epochs.
Accuracy: 0.87
Model (477) - Linear Regression - Train a model with first 400 rows of data for
92 epochs. Accuracy: 0.87
Model (478) - Percetron - Train a model with first 400 rows of data for 97
epochs. Accuracy: 0.82
Model (479) - Adaline - Train a model with first 400 rows of data for 97 epochs.
Accuracy: 0.87
Model (480) - Linear Regression - Train a model with first 400 rows of data for
97 epochs. Accuracy: 0.87
Model (481) - Percetron - Train a model with first 450 rows of data for 2
epochs. Accuracy: 0.88
Model (482) - Adaline - Train a model with first 450 rows of data for 2 epochs.
Accuracy: 0.86
Model (483) - Linear Regression - Train a model with first 450 rows of data for
2 epochs. Accuracy: 0.87
Model (484) - Percetron - Train a model with first 450 rows of data for 7
epochs. Accuracy: 0.89
Model (485) - Adaline - Train a model with first 450 rows of data for 7 epochs.
Accuracy: 0.88
Model (486) - Linear Regression - Train a model with first 450 rows of data for
7 epochs. Accuracy: 0.86
Model (487) - Percetron - Train a model with first 450 rows of data for 12
epochs. Accuracy: 0.87
Model (488) - Adaline - Train a model with first 450 rows of data for 12 epochs.
Accuracy: 0.87
Model (489) - Linear Regression - Train a model with first 450 rows of data for
12 epochs. Accuracy: 0.88
Model (490) - Percetron - Train a model with first 450 rows of data for 17
epochs. Accuracy: 0.66
Model (491) - Adaline - Train a model with first 450 rows of data for 17 epochs.
Accuracy: 0.88
Model (492) - Linear Regression - Train a model with first 450 rows of data for
17 epochs. Accuracy: 0.88
Model (493) - Percetron - Train a model with first 450 rows of data for 22
epochs. Accuracy: 0.88
Model (494) - Adaline - Train a model with first 450 rows of data for 22 epochs.
Accuracy: 0.88
Model (495) - Linear Regression - Train a model with first 450 rows of data for
```

```
22 epochs. Accuracy: 0.88
Model (496) - Percetron - Train a model with first 450 rows of data for 27
epochs. Accuracy: 0.87
Model (497) - Adaline - Train a model with first 450 rows of data for 27 epochs.
Accuracy: 0.87
Model (498) - Linear Regression - Train a model with first 450 rows of data for
27 epochs. Accuracy: 0.87
Model (499) - Percetron - Train a model with first 450 rows of data for 32
epochs. Accuracy: 0.78
Model (500) - Adaline - Train a model with first 450 rows of data for 32 epochs.
Accuracy: 0.87
Model (501) - Linear Regression - Train a model with first 450 rows of data for
32 epochs. Accuracy: 0.87
Model (502) - Percetron - Train a model with first 450 rows of data for 37
epochs. Accuracy: 0.89
Model (503) - Adaline - Train a model with first 450 rows of data for 37 epochs.
Accuracy: 0.88
Model (504) - Linear Regression - Train a model with first 450 rows of data for
37 epochs. Accuracy: 0.87
Model (505) - Percetron - Train a model with first 450 rows of data for 42
epochs. Accuracy: 0.88
Model (506) - Adaline - Train a model with first 450 rows of data for 42 epochs.
Accuracy: 0.88
Model (507) - Linear Regression - Train a model with first 450 rows of data for
42 epochs. Accuracy: 0.87
Model (508) - Percetron - Train a model with first 450 rows of data for 47
epochs. Accuracy: 0.78
Model (509) - Adaline - Train a model with first 450 rows of data for 47 epochs.
Accuracy: 0.88
Model (510) - Linear Regression - Train a model with first 450 rows of data for
47 epochs. Accuracy: 0.87
Model (511) - Percetron - Train a model with first 450 rows of data for 52
epochs. Accuracy: 0.87
Model (512) - Adaline - Train a model with first 450 rows of data for 52 epochs.
Accuracy: 0.88
Model (513) - Linear Regression - Train a model with first 450 rows of data for
52 epochs. Accuracy: 0.87
Model (514) - Percetron - Train a model with first 450 rows of data for 57
epochs. Accuracy: 0.87
Model (515) - Adaline - Train a model with first 450 rows of data for 57 epochs.
Accuracy: 0.89
Model (516) - Linear Regression - Train a model with first 450 rows of data for
57 epochs. Accuracy: 0.87
Model (517) - Percetron - Train a model with first 450 rows of data for 62
epochs. Accuracy: 0.89
Model (518) - Adaline - Train a model with first 450 rows of data for 62 epochs.
Accuracy: 0.89
Model (519) - Linear Regression - Train a model with first 450 rows of data for
```

```
62 epochs. Accuracy: 0.87
Model (520) - Percetron - Train a model with first 450 rows of data for 67
epochs. Accuracy: 0.85
Model (521) - Adaline - Train a model with first 450 rows of data for 67 epochs.
Accuracy: 0.89
Model (522) - Linear Regression - Train a model with first 450 rows of data for
67 epochs. Accuracy: 0.87
Model (523) - Percetron - Train a model with first 450 rows of data for 72
epochs. Accuracy: 0.89
Model (524) - Adaline - Train a model with first 450 rows of data for 72 epochs.
Accuracy: 0.89
Model (525) - Linear Regression - Train a model with first 450 rows of data for
72 epochs. Accuracy: 0.87
Model (526) - Percetron - Train a model with first 450 rows of data for 77
epochs. Accuracy: 0.85
Model (527) - Adaline - Train a model with first 450 rows of data for 77 epochs.
Accuracy: 0.89
Model (528) - Linear Regression - Train a model with first 450 rows of data for
77 epochs. Accuracy: 0.87
Model (529) - Percetron - Train a model with first 450 rows of data for 82
epochs. Accuracy: 0.84
Model (530) - Adaline - Train a model with first 450 rows of data for 82 epochs.
Accuracy: 0.89
Model (531) - Linear Regression - Train a model with first 450 rows of data for
82 epochs. Accuracy: 0.88
Model (532) - Percetron - Train a model with first 450 rows of data for 87
epochs. Accuracy: 0.90
Model (533) - Adaline - Train a model with first 450 rows of data for 87 epochs.
Accuracy: 0.89
Model (534) - Linear Regression - Train a model with first 450 rows of data for
87 epochs. Accuracy: 0.88
Model (535) - Percetron - Train a model with first 450 rows of data for 92
epochs. Accuracy: 0.86
Model (536) - Adaline - Train a model with first 450 rows of data for 92 epochs.
Accuracy: 0.89
Model (537) - Linear Regression - Train a model with first 450 rows of data for
92 epochs. Accuracy: 0.88
Model (538) - Percetron - Train a model with first 450 rows of data for 97
epochs. Accuracy: 0.67
Model (539) - Adaline - Train a model with first 450 rows of data for 97 epochs.
Accuracy: 0.89
Model (540) - Linear Regression - Train a model with first 450 rows of data for
97 epochs. Accuracy: 0.88
Model (541) - Percetron - Train a model with first 500 rows of data for 2
epochs. Accuracy: 0.87
Model (542) - Adaline - Train a model with first 500 rows of data for 2 epochs.
Accuracy: 0.86
Model (543) - Linear Regression - Train a model with first 500 rows of data for
```

```
2 epochs. Accuracy: 0.86
Model (544) - Percetron - Train a model with first 500 rows of data for 7
epochs. Accuracy: 0.87
Model (545) - Adaline - Train a model with first 500 rows of data for 7 epochs.
Accuracy: 0.87
Model (546) - Linear Regression - Train a model with first 500 rows of data for
7 epochs. Accuracy: 0.86
Model (547) - Percetron - Train a model with first 500 rows of data for 12
epochs. Accuracy: 0.87
Model (548) - Adaline - Train a model with first 500 rows of data for 12 epochs.
Accuracy: 0.87
Model (549) - Linear Regression - Train a model with first 500 rows of data for
12 epochs. Accuracy: 0.87
Model (550) - Percetron - Train a model with first 500 rows of data for 17
epochs. Accuracy: 0.76
Model (551) - Adaline - Train a model with first 500 rows of data for 17 epochs.
Accuracy: 0.88
Model (552) - Linear Regression - Train a model with first 500 rows of data for
17 epochs. Accuracy: 0.87
Model (553) - Percetron - Train a model with first 500 rows of data for 22
epochs. Accuracy: 0.88
Model (554) - Adaline - Train a model with first 500 rows of data for 22 epochs.
Accuracy: 0.88
Model (555) - Linear Regression - Train a model with first 500 rows of data for
22 epochs. Accuracy: 0.87
Model (556) - Percetron - Train a model with first 500 rows of data for 27
epochs. Accuracy: 0.85
Model (557) - Adaline - Train a model with first 500 rows of data for 27 epochs.
Accuracy: 0.88
Model (558) - Linear Regression - Train a model with first 500 rows of data for
27 epochs. Accuracy: 0.87
Model (559) - Percetron - Train a model with first 500 rows of data for 32
epochs. Accuracy: 0.90
Model (560) - Adaline - Train a model with first 500 rows of data for 32 epochs.
Accuracy: 0.88
Model (561) - Linear Regression - Train a model with first 500 rows of data for
32 epochs. Accuracy: 0.87
Model (562) - Percetron - Train a model with first 500 rows of data for 37
epochs. Accuracy: 0.88
Model (563) - Adaline - Train a model with first 500 rows of data for 37 epochs.
Accuracy: 0.88
Model (564) - Linear Regression - Train a model with first 500 rows of data for
37 epochs. Accuracy: 0.87
Model (565) - Percetron - Train a model with first 500 rows of data for 42
epochs. Accuracy: 0.64
Model (566) - Adaline - Train a model with first 500 rows of data for 42 epochs.
Accuracy: 0.89
Model (567) - Linear Regression - Train a model with first 500 rows of data for
```

```
42 epochs. Accuracy: 0.87
Model (568) - Percetron - Train a model with first 500 rows of data for 47
epochs. Accuracy: 0.85
Model (569) - Adaline - Train a model with first 500 rows of data for 47 epochs.
Accuracy: 0.89
Model (570) - Linear Regression - Train a model with first 500 rows of data for
47 epochs. Accuracy: 0.87
Model (571) - Percetron - Train a model with first 500 rows of data for 52
epochs. Accuracy: 0.79
Model (572) - Adaline - Train a model with first 500 rows of data for 52 epochs.
Accuracy: 0.89
Model (573) - Linear Regression - Train a model with first 500 rows of data for
52 epochs. Accuracy: 0.87
Model (574) - Percetron - Train a model with first 500 rows of data for 57
epochs. Accuracy: 0.83
Model (575) - Adaline - Train a model with first 500 rows of data for 57 epochs.
Accuracy: 0.89
Model (576) - Linear Regression - Train a model with first 500 rows of data for
57 epochs. Accuracy: 0.87
Model (577) - Percetron - Train a model with first 500 rows of data for 62
epochs. Accuracy: 0.87
Model (578) - Adaline - Train a model with first 500 rows of data for 62 epochs.
Accuracy: 0.89
Model (579) - Linear Regression - Train a model with first 500 rows of data for
62 epochs. Accuracy: 0.87
Model (580) - Percetron - Train a model with first 500 rows of data for 67
epochs. Accuracy: 0.87
Model (581) - Adaline - Train a model with first 500 rows of data for 67 epochs.
Accuracy: 0.89
Model (582) - Linear Regression - Train a model with first 500 rows of data for
67 epochs. Accuracy: 0.87
Model (583) - Percetron - Train a model with first 500 rows of data for 72
epochs. Accuracy: 0.88
Model (584) - Adaline - Train a model with first 500 rows of data for 72 epochs.
Accuracy: 0.89
Model (585) - Linear Regression - Train a model with first 500 rows of data for
72 epochs. Accuracy: 0.87
Model (586) - Percetron - Train a model with first 500 rows of data for 77
epochs. Accuracy: 0.89
Model (587) - Adaline - Train a model with first 500 rows of data for 77 epochs.
Accuracy: 0.89
Model (588) - Linear Regression - Train a model with first 500 rows of data for
77 epochs. Accuracy: 0.88
Model (589) - Percetron - Train a model with first 500 rows of data for 82
epochs. Accuracy: 0.85
Model (590) - Adaline - Train a model with first 500 rows of data for 82 epochs.
Accuracy: 0.89
Model (591) - Linear Regression - Train a model with first 500 rows of data for
```

82 epochs. Accuracy: 0.88
Model (592) - Percetron - Train a model with first 500 rows of data for 87 epochs. Accuracy: 0.82
Model (593) - Adaline - Train a model with first 500 rows of data for 87 epochs. Accuracy: 0.89
Model (594) - Linear Regression - Train a model with first 500 rows of data for 87 epochs. Accuracy: 0.88
Model (595) - Percetron - Train a model with first 500 rows of data for 92 epochs. Accuracy: 0.85
Model (596) - Adaline - Train a model with first 500 rows of data for 92 epochs. Accuracy: 0.89
Model (597) - Linear Regression - Train a model with first 500 rows of data for 92 epochs. Accuracy: 0.88
Model (598) - Percetron - Train a model with first 500 rows of data for 97 epochs. Accuracy: 0.88
Model (599) - Adaline - Train a model with first 500 rows of data for 97 epochs. Accuracy: 0.89
Model (600) - Linear Regression - Train a model with first 500 rows of data for 97 epochs. Accuracy: 0.88
Model (601) - Percetron - Train a model with first 550 rows of data for 2 epochs. Accuracy: 0.84
Model (602) - Adaline - Train a model with first 550 rows of data for 2 epochs. Accuracy: 0.86
Model (603) - Linear Regression - Train a model with first 550 rows of data for 2 epochs. Accuracy: 0.86
Model (604) - Percetron - Train a model with first 550 rows of data for 7 epochs. Accuracy: 0.86
Model (605) - Adaline - Train a model with first 550 rows of data for 7 epochs. Accuracy: 0.87
Model (606) - Linear Regression - Train a model with first 550 rows of data for 7 epochs. Accuracy: 0.86
Model (607) - Percetron - Train a model with first 550 rows of data for 12 epochs. Accuracy: 0.91
Model (608) - Adaline - Train a model with first 550 rows of data for 12 epochs. Accuracy: 0.88
Model (609) - Linear Regression - Train a model with first 550 rows of data for 12 epochs. Accuracy: 0.88
Model (610) - Percetron - Train a model with first 550 rows of data for 17 epochs. Accuracy: 0.76
Model (611) - Adaline - Train a model with first 550 rows of data for 17 epochs. Accuracy: 0.88
Model (612) - Linear Regression - Train a model with first 550 rows of data for 17 epochs. Accuracy: 0.87
Model (613) - Percetron - Train a model with first 550 rows of data for 22 epochs. Accuracy: 0.81
Model (614) - Adaline - Train a model with first 550 rows of data for 22 epochs. Accuracy: 0.88
Model (615) - Linear Regression - Train a model with first 550 rows of data for

```
22 epochs. Accuracy: 0.87
Model (616) - Percetron - Train a model with first 550 rows of data for 27
epochs. Accuracy: 0.86
Model (617) - Adaline - Train a model with first 550 rows of data for 27 epochs.
Accuracy: 0.88
Model (618) - Linear Regression - Train a model with first 550 rows of data for
27 epochs. Accuracy: 0.87
Model (619) - Percetron - Train a model with first 550 rows of data for 32
epochs. Accuracy: 0.88
Model (620) - Adaline - Train a model with first 550 rows of data for 32 epochs.
Accuracy: 0.88
Model (621) - Linear Regression - Train a model with first 550 rows of data for
32 epochs. Accuracy: 0.87
Model (622) - Percetron - Train a model with first 550 rows of data for 37
epochs. Accuracy: 0.87
Model (623) - Adaline - Train a model with first 550 rows of data for 37 epochs.
Accuracy: 0.88
Model (624) - Linear Regression - Train a model with first 550 rows of data for
37 epochs. Accuracy: 0.87
Model (625) - Percetron - Train a model with first 550 rows of data for 42
epochs. Accuracy: 0.89
Model (626) - Adaline - Train a model with first 550 rows of data for 42 epochs.
Accuracy: 0.89
Model (627) - Linear Regression - Train a model with first 550 rows of data for
42 epochs. Accuracy: 0.87
Model (628) - Percetron - Train a model with first 550 rows of data for 47
epochs. Accuracy: 0.86
Model (629) - Adaline - Train a model with first 550 rows of data for 47 epochs.
Accuracy: 0.89
Model (630) - Linear Regression - Train a model with first 550 rows of data for
47 epochs. Accuracy: 0.87
Model (631) - Percetron - Train a model with first 550 rows of data for 52
epochs. Accuracy: 0.89
Model (632) - Adaline - Train a model with first 550 rows of data for 52 epochs.
Accuracy: 0.89
Model (633) - Linear Regression - Train a model with first 550 rows of data for
52 epochs. Accuracy: 0.87
Model (634) - Percetron - Train a model with first 550 rows of data for 57
epochs. Accuracy: 0.89
Model (635) - Adaline - Train a model with first 550 rows of data for 57 epochs.
Accuracy: 0.89
Model (636) - Linear Regression - Train a model with first 550 rows of data for
57 epochs. Accuracy: 0.88
Model (637) - Percetron - Train a model with first 550 rows of data for 62
epochs. Accuracy: 0.90
Model (638) - Adaline - Train a model with first 550 rows of data for 62 epochs.
Accuracy: 0.89
Model (639) - Linear Regression - Train a model with first 550 rows of data for
```

```
62 epochs. Accuracy: 0.88
Model (640) - Percetron - Train a model with first 550 rows of data for 67
epochs. Accuracy: 0.83
Model (641) - Adaline - Train a model with first 550 rows of data for 67 epochs.
Accuracy: 0.89
Model (642) - Linear Regression - Train a model with first 550 rows of data for
67 epochs. Accuracy: 0.88
Model (643) - Percetron - Train a model with first 550 rows of data for 72
epochs. Accuracy: 0.89
Model (644) - Adaline - Train a model with first 550 rows of data for 72 epochs.
Accuracy: 0.89
Model (645) - Linear Regression - Train a model with first 550 rows of data for
72 epochs. Accuracy: 0.88
Model (646) - Percetron - Train a model with first 550 rows of data for 77
epochs. Accuracy: 0.88
Model (647) - Adaline - Train a model with first 550 rows of data for 77 epochs.
Accuracy: 0.89
Model (648) - Linear Regression - Train a model with first 550 rows of data for
77 epochs. Accuracy: 0.88
Model (649) - Percetron - Train a model with first 550 rows of data for 82
epochs. Accuracy: 0.89
Model (650) - Adaline - Train a model with first 550 rows of data for 82 epochs.
Accuracy: 0.89
Model (651) - Linear Regression - Train a model with first 550 rows of data for
82 epochs. Accuracy: 0.88
Model (652) - Percetron - Train a model with first 550 rows of data for 87
epochs. Accuracy: 0.89
Model (653) - Adaline - Train a model with first 550 rows of data for 87 epochs.
Accuracy: 0.89
Model (654) - Linear Regression - Train a model with first 550 rows of data for
87 epochs. Accuracy: 0.88
Model (655) - Percetron - Train a model with first 550 rows of data for 92
epochs. Accuracy: 0.85
Model (656) - Adaline - Train a model with first 550 rows of data for 92 epochs.
Accuracy: 0.89
Model (657) - Linear Regression - Train a model with first 550 rows of data for
92 epochs. Accuracy: 0.88
Model (658) - Percetron - Train a model with first 550 rows of data for 97
epochs. Accuracy: 0.83
Model (659) - Adaline - Train a model with first 550 rows of data for 97 epochs.
Accuracy: 0.89
Model (660) - Linear Regression - Train a model with first 550 rows of data for
97 epochs. Accuracy: 0.87
Model (661) - Percetron - Train a model with first 600 rows of data for 2
epochs. Accuracy: 0.88
Model (662) - Adaline - Train a model with first 600 rows of data for 2 epochs.
Accuracy: 0.86
Model (663) - Linear Regression - Train a model with first 600 rows of data for
```

2 epochs. Accuracy: 0.87
Model (664) - Percetron - Train a model with first 600 rows of data for 7 epochs. Accuracy: 0.86
Model (665) - Adaline - Train a model with first 600 rows of data for 7 epochs. Accuracy: 0.87
Model (666) - Linear Regression - Train a model with first 600 rows of data for 7 epochs. Accuracy: 0.86
Model (667) - Percetron - Train a model with first 600 rows of data for 12 epochs. Accuracy: 0.91
Model (668) - Adaline - Train a model with first 600 rows of data for 12 epochs. Accuracy: 0.88
Model (669) - Linear Regression - Train a model with first 600 rows of data for 12 epochs. Accuracy: 0.88
Model (670) - Percetron - Train a model with first 600 rows of data for 17 epochs. Accuracy: 0.87
Model (671) - Adaline - Train a model with first 600 rows of data for 17 epochs. Accuracy: 0.88
Model (672) - Linear Regression - Train a model with first 600 rows of data for 17 epochs. Accuracy: 0.87
Model (673) - Percetron - Train a model with first 600 rows of data for 22 epochs. Accuracy: 0.85
Model (674) - Adaline - Train a model with first 600 rows of data for 22 epochs. Accuracy: 0.88
Model (675) - Linear Regression - Train a model with first 600 rows of data for 22 epochs. Accuracy: 0.87
Model (676) - Percetron - Train a model with first 600 rows of data for 27 epochs. Accuracy: 0.89
Model (677) - Adaline - Train a model with first 600 rows of data for 27 epochs. Accuracy: 0.88
Model (678) - Linear Regression - Train a model with first 600 rows of data for 27 epochs. Accuracy: 0.87
Model (679) - Percetron - Train a model with first 600 rows of data for 32 epochs. Accuracy: 0.85
Model (680) - Adaline - Train a model with first 600 rows of data for 32 epochs. Accuracy: 0.88
Model (681) - Linear Regression - Train a model with first 600 rows of data for 32 epochs. Accuracy: 0.87
Model (682) - Percetron - Train a model with first 600 rows of data for 37 epochs. Accuracy: 0.83
Model (683) - Adaline - Train a model with first 600 rows of data for 37 epochs. Accuracy: 0.88
Model (684) - Linear Regression - Train a model with first 600 rows of data for 37 epochs. Accuracy: 0.87
Model (685) - Percetron - Train a model with first 600 rows of data for 42 epochs. Accuracy: 0.81
Model (686) - Adaline - Train a model with first 600 rows of data for 42 epochs. Accuracy: 0.89
Model (687) - Linear Regression - Train a model with first 600 rows of data for

```
42 epochs. Accuracy: 0.87
Model (688) - Percetron - Train a model with first 600 rows of data for 47
epochs. Accuracy: 0.85
Model (689) - Adaline - Train a model with first 600 rows of data for 47 epochs.
Accuracy: 0.89
Model (690) - Linear Regression - Train a model with first 600 rows of data for
47 epochs. Accuracy: 0.87
Model (691) - Percetron - Train a model with first 600 rows of data for 52
epochs. Accuracy: 0.88
Model (692) - Adaline - Train a model with first 600 rows of data for 52 epochs.
Accuracy: 0.89
Model (693) - Linear Regression - Train a model with first 600 rows of data for
52 epochs. Accuracy: 0.87
Model (694) - Percetron - Train a model with first 600 rows of data for 57
epochs. Accuracy: 0.85
Model (695) - Adaline - Train a model with first 600 rows of data for 57 epochs.
Accuracy: 0.89
Model (696) - Linear Regression - Train a model with first 600 rows of data for
57 epochs. Accuracy: 0.87
Model (697) - Percetron - Train a model with first 600 rows of data for 62
epochs. Accuracy: 0.89
Model (698) - Adaline - Train a model with first 600 rows of data for 62 epochs.
Accuracy: 0.89
Model (699) - Linear Regression - Train a model with first 600 rows of data for
62 epochs. Accuracy: 0.88
Model (700) - Percetron - Train a model with first 600 rows of data for 67
epochs. Accuracy: 0.92
Model (701) - Adaline - Train a model with first 600 rows of data for 67 epochs.
Accuracy: 0.89
Model (702) - Linear Regression - Train a model with first 600 rows of data for
67 epochs. Accuracy: 0.88
Model (703) - Percetron - Train a model with first 600 rows of data for 72
epochs. Accuracy: 0.86
Model (704) - Adaline - Train a model with first 600 rows of data for 72 epochs.
Accuracy: 0.89
Model (705) - Linear Regression - Train a model with first 600 rows of data for
72 epochs. Accuracy: 0.88
Model (706) - Percetron - Train a model with first 600 rows of data for 77
epochs. Accuracy: 0.88
Model (707) - Adaline - Train a model with first 600 rows of data for 77 epochs.
Accuracy: 0.89
Model (708) - Linear Regression - Train a model with first 600 rows of data for
77 epochs. Accuracy: 0.88
Model (709) - Percetron - Train a model with first 600 rows of data for 82
epochs. Accuracy: 0.84
Model (710) - Adaline - Train a model with first 600 rows of data for 82 epochs.
Accuracy: 0.89
Model (711) - Linear Regression - Train a model with first 600 rows of data for
```

```
82 epochs. Accuracy: 0.88
Model (712) - Percetron - Train a model with first 600 rows of data for 87
epochs. Accuracy: 0.88
Model (713) - Adaline - Train a model with first 600 rows of data for 87 epochs.
Accuracy: 0.89
Model (714) - Linear Regression - Train a model with first 600 rows of data for
87 epochs. Accuracy: 0.88
Model (715) - Percetron - Train a model with first 600 rows of data for 92
epochs. Accuracy: 0.88
Model (716) - Adaline - Train a model with first 600 rows of data for 92 epochs.
Accuracy: 0.89
Model (717) - Linear Regression - Train a model with first 600 rows of data for
92 epochs. Accuracy: 0.88
Model (718) - Percetron - Train a model with first 600 rows of data for 97
epochs. Accuracy: 0.90
Model (719) - Adaline - Train a model with first 600 rows of data for 97 epochs.
Accuracy: 0.89
Model (720) - Linear Regression - Train a model with first 600 rows of data for
97 epochs. Accuracy: 0.88
Model (721) - Percetron - Train a model with first 650 rows of data for 2
epochs. Accuracy: 0.82
Model (722) - Adaline - Train a model with first 650 rows of data for 2 epochs.
Accuracy: 0.86
Model (723) - Linear Regression - Train a model with first 650 rows of data for
2 epochs. Accuracy: 0.86
Model (724) - Percetron - Train a model with first 650 rows of data for 7
epochs. Accuracy: 0.89
Model (725) - Adaline - Train a model with first 650 rows of data for 7 epochs.
Accuracy: 0.87
Model (726) - Linear Regression - Train a model with first 650 rows of data for
7 epochs. Accuracy: 0.87
Model (727) - Percetron - Train a model with first 650 rows of data for 12
epochs. Accuracy: 0.87
Model (728) - Adaline - Train a model with first 650 rows of data for 12 epochs.
Accuracy: 0.88
Model (729) - Linear Regression - Train a model with first 650 rows of data for
12 epochs. Accuracy: 0.87
Model (730) - Percetron - Train a model with first 650 rows of data for 17
epochs. Accuracy: 0.80
Model (731) - Adaline - Train a model with first 650 rows of data for 17 epochs.
Accuracy: 0.88
Model (732) - Linear Regression - Train a model with first 650 rows of data for
17 epochs. Accuracy: 0.87
Model (733) - Percetron - Train a model with first 650 rows of data for 22
epochs. Accuracy: 0.85
Model (734) - Adaline - Train a model with first 650 rows of data for 22 epochs.
Accuracy: 0.88
Model (735) - Linear Regression - Train a model with first 650 rows of data for
```

```
22 epochs. Accuracy: 0.87
Model (736) - Percetron - Train a model with first 650 rows of data for 27
epochs. Accuracy: 0.89
Model (737) - Adaline - Train a model with first 650 rows of data for 27 epochs.
Accuracy: 0.87
Model (738) - Linear Regression - Train a model with first 650 rows of data for
27 epochs. Accuracy: 0.87
Model (739) - Percetron - Train a model with first 650 rows of data for 32
epochs. Accuracy: 0.72
Model (740) - Adaline - Train a model with first 650 rows of data for 32 epochs.
Accuracy: 0.88
Model (741) - Linear Regression - Train a model with first 650 rows of data for
32 epochs. Accuracy: 0.87
Model (742) - Percetron - Train a model with first 650 rows of data for 37
epochs. Accuracy: 0.87
Model (743) - Adaline - Train a model with first 650 rows of data for 37 epochs.
Accuracy: 0.88
Model (744) - Linear Regression - Train a model with first 650 rows of data for
37 epochs. Accuracy: 0.87
Model (745) - Percetron - Train a model with first 650 rows of data for 42
epochs. Accuracy: 0.71
Model (746) - Adaline - Train a model with first 650 rows of data for 42 epochs.
Accuracy: 0.88
Model (747) - Linear Regression - Train a model with first 650 rows of data for
42 epochs. Accuracy: 0.87
Model (748) - Percetron - Train a model with first 650 rows of data for 47
epochs. Accuracy: 0.90
Model (749) - Adaline - Train a model with first 650 rows of data for 47 epochs.
Accuracy: 0.89
Model (750) - Linear Regression - Train a model with first 650 rows of data for
47 epochs. Accuracy: 0.87
Model (751) - Percetron - Train a model with first 650 rows of data for 52
epochs. Accuracy: 0.81
Model (752) - Adaline - Train a model with first 650 rows of data for 52 epochs.
Accuracy: 0.89
Model (753) - Linear Regression - Train a model with first 650 rows of data for
52 epochs. Accuracy: 0.87
Model (754) - Percetron - Train a model with first 650 rows of data for 57
epochs. Accuracy: 0.87
Model (755) - Adaline - Train a model with first 650 rows of data for 57 epochs.
Accuracy: 0.89
Model (756) - Linear Regression - Train a model with first 650 rows of data for
57 epochs. Accuracy: 0.87
Model (757) - Percetron - Train a model with first 650 rows of data for 62
epochs. Accuracy: 0.82
Model (758) - Adaline - Train a model with first 650 rows of data for 62 epochs.
Accuracy: 0.89
Model (759) - Linear Regression - Train a model with first 650 rows of data for
```

```
62 epochs. Accuracy: 0.88
Model (760) - Percetron - Train a model with first 650 rows of data for 67
epochs. Accuracy: 0.87
Model (761) - Adaline - Train a model with first 650 rows of data for 67 epochs.
Accuracy: 0.89
Model (762) - Linear Regression - Train a model with first 650 rows of data for
67 epochs. Accuracy: 0.88
Model (763) - Percetron - Train a model with first 650 rows of data for 72
epochs. Accuracy: 0.90
Model (764) - Adaline - Train a model with first 650 rows of data for 72 epochs.
Accuracy: 0.89
Model (765) - Linear Regression - Train a model with first 650 rows of data for
72 epochs. Accuracy: 0.88
Model (766) - Percetron - Train a model with first 650 rows of data for 77
epochs. Accuracy: 0.86
Model (767) - Adaline - Train a model with first 650 rows of data for 77 epochs.
Accuracy: 0.89
Model (768) - Linear Regression - Train a model with first 650 rows of data for
77 epochs. Accuracy: 0.88
Model (769) - Percetron - Train a model with first 650 rows of data for 82
epochs. Accuracy: 0.82
Model (770) - Adaline - Train a model with first 650 rows of data for 82 epochs.
Accuracy: 0.89
Model (771) - Linear Regression - Train a model with first 650 rows of data for
82 epochs. Accuracy: 0.88
Model (772) - Percetron - Train a model with first 650 rows of data for 87
epochs. Accuracy: 0.88
Model (773) - Adaline - Train a model with first 650 rows of data for 87 epochs.
Accuracy: 0.89
Model (774) - Linear Regression - Train a model with first 650 rows of data for
87 epochs. Accuracy: 0.88
Model (775) - Percetron - Train a model with first 650 rows of data for 92
epochs. Accuracy: 0.82
Model (776) - Adaline - Train a model with first 650 rows of data for 92 epochs.
Accuracy: 0.89
Model (777) - Linear Regression - Train a model with first 650 rows of data for
92 epochs. Accuracy: 0.88
Model (778) - Percetron - Train a model with first 650 rows of data for 97
epochs. Accuracy: 0.89
Model (779) - Adaline - Train a model with first 650 rows of data for 97 epochs.
Accuracy: 0.89
Model (780) - Linear Regression - Train a model with first 650 rows of data for
97 epochs. Accuracy: 0.88
Model (781) - Percetron - Train a model with first 700 rows of data for 2
epochs. Accuracy: 0.85
Model (782) - Adaline - Train a model with first 700 rows of data for 2 epochs.
Accuracy: 0.86
Model (783) - Linear Regression - Train a model with first 700 rows of data for
```

```
2 epochs. Accuracy: 0.86
Model (784) - Percetron - Train a model with first 700 rows of data for 7
epochs. Accuracy: 0.89
Model (785) - Adaline - Train a model with first 700 rows of data for 7 epochs.
Accuracy: 0.87
Model (786) - Linear Regression - Train a model with first 700 rows of data for
7 epochs. Accuracy: 0.86
Model (787) - Percetron - Train a model with first 700 rows of data for 12
epochs. Accuracy: 0.87
Model (788) - Adaline - Train a model with first 700 rows of data for 12 epochs.
Accuracy: 0.88
Model (789) - Linear Regression - Train a model with first 700 rows of data for
12 epochs. Accuracy: 0.87
Model (790) - Percetron - Train a model with first 700 rows of data for 17
epochs. Accuracy: 0.69
Model (791) - Adaline - Train a model with first 700 rows of data for 17 epochs.
Accuracy: 0.87
Model (792) - Linear Regression - Train a model with first 700 rows of data for
17 epochs. Accuracy: 0.87
Model (793) - Percetron - Train a model with first 700 rows of data for 22
epochs. Accuracy: 0.86
Model (794) - Adaline - Train a model with first 700 rows of data for 22 epochs.
Accuracy: 0.88
Model (795) - Linear Regression - Train a model with first 700 rows of data for
22 epochs. Accuracy: 0.87
Model (796) - Percetron - Train a model with first 700 rows of data for 27
epochs. Accuracy: 0.77
Model (797) - Adaline - Train a model with first 700 rows of data for 27 epochs.
Accuracy: 0.88
Model (798) - Linear Regression - Train a model with first 700 rows of data for
27 epochs. Accuracy: 0.87
Model (799) - Percetron - Train a model with first 700 rows of data for 32
epochs. Accuracy: 0.82
Model (800) - Adaline - Train a model with first 700 rows of data for 32 epochs.
Accuracy: 0.88
Model (801) - Linear Regression - Train a model with first 700 rows of data for
32 epochs. Accuracy: 0.87
Model (802) - Percetron - Train a model with first 700 rows of data for 37
epochs. Accuracy: 0.87
Model (803) - Adaline - Train a model with first 700 rows of data for 37 epochs.
Accuracy: 0.88
Model (804) - Linear Regression - Train a model with first 700 rows of data for
37 epochs. Accuracy: 0.87
Model (805) - Percetron - Train a model with first 700 rows of data for 42
epochs. Accuracy: 0.88
Model (806) - Adaline - Train a model with first 700 rows of data for 42 epochs.
Accuracy: 0.88
Model (807) - Linear Regression - Train a model with first 700 rows of data for
```

```
42 epochs. Accuracy: 0.87
Model (808) - Percetron - Train a model with first 700 rows of data for 47
epochs. Accuracy: 0.85
Model (809) - Adaline - Train a model with first 700 rows of data for 47 epochs.
Accuracy: 0.89
Model (810) - Linear Regression - Train a model with first 700 rows of data for
47 epochs. Accuracy: 0.87
Model (811) - Percetron - Train a model with first 700 rows of data for 52
epochs. Accuracy: 0.88
Model (812) - Adaline - Train a model with first 700 rows of data for 52 epochs.
Accuracy: 0.89
Model (813) - Linear Regression - Train a model with first 700 rows of data for
52 epochs. Accuracy: 0.87
Model (814) - Percetron - Train a model with first 700 rows of data for 57
epochs. Accuracy: 0.85
Model (815) - Adaline - Train a model with first 700 rows of data for 57 epochs.
Accuracy: 0.89
Model (816) - Linear Regression - Train a model with first 700 rows of data for
57 epochs. Accuracy: 0.88
Model (817) - Percetron - Train a model with first 700 rows of data for 62
epochs. Accuracy: 0.89
Model (818) - Adaline - Train a model with first 700 rows of data for 62 epochs.
Accuracy: 0.89
Model (819) - Linear Regression - Train a model with first 700 rows of data for
62 epochs. Accuracy: 0.88
Model (820) - Percetron - Train a model with first 700 rows of data for 67
epochs. Accuracy: 0.89
Model (821) - Adaline - Train a model with first 700 rows of data for 67 epochs.
Accuracy: 0.89
Model (822) - Linear Regression - Train a model with first 700 rows of data for
67 epochs. Accuracy: 0.88
Model (823) - Percetron - Train a model with first 700 rows of data for 72
epochs. Accuracy: 0.68
Model (824) - Adaline - Train a model with first 700 rows of data for 72 epochs.
Accuracy: 0.89
Model (825) - Linear Regression - Train a model with first 700 rows of data for
72 epochs. Accuracy: 0.88
Model (826) - Percetron - Train a model with first 700 rows of data for 77
epochs. Accuracy: 0.86
Model (827) - Adaline - Train a model with first 700 rows of data for 77 epochs.
Accuracy: 0.89
Model (828) - Linear Regression - Train a model with first 700 rows of data for
77 epochs. Accuracy: 0.88
Model (829) - Percetron - Train a model with first 700 rows of data for 82
epochs. Accuracy: 0.88
Model (830) - Adaline - Train a model with first 700 rows of data for 82 epochs.
Accuracy: 0.89
Model (831) - Linear Regression - Train a model with first 700 rows of data for
```

```
82 epochs. Accuracy: 0.88
Model (832) - Percetron - Train a model with first 700 rows of data for 87
epochs. Accuracy: 0.87
Model (833) - Adaline - Train a model with first 700 rows of data for 87 epochs.
Accuracy: 0.89
Model (834) - Linear Regression - Train a model with first 700 rows of data for
87 epochs. Accuracy: 0.88
Model (835) - Percetron - Train a model with first 700 rows of data for 92
epochs. Accuracy: 0.85
Model (836) - Adaline - Train a model with first 700 rows of data for 92 epochs.
Accuracy: 0.89
Model (837) - Linear Regression - Train a model with first 700 rows of data for
92 epochs. Accuracy: 0.88
Model (838) - Percetron - Train a model with first 700 rows of data for 97
epochs. Accuracy: 0.83
Model (839) - Adaline - Train a model with first 700 rows of data for 97 epochs.
Accuracy: 0.89
Model (840) - Linear Regression - Train a model with first 700 rows of data for
97 epochs. Accuracy: 0.88
```

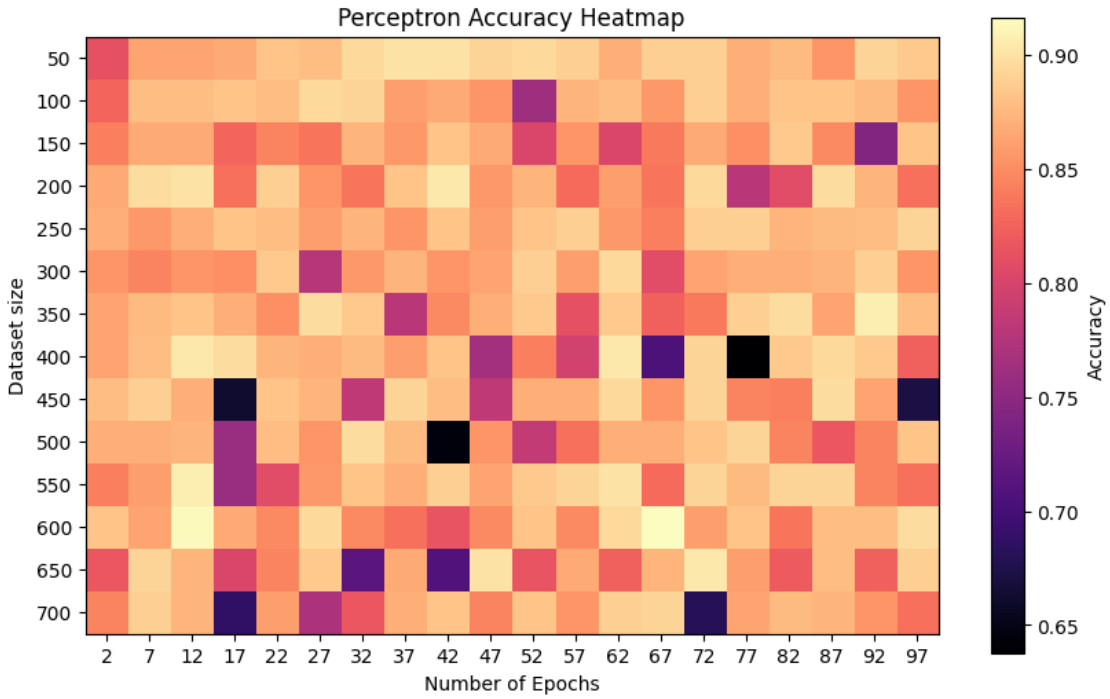
1.4.4 Performance visualization

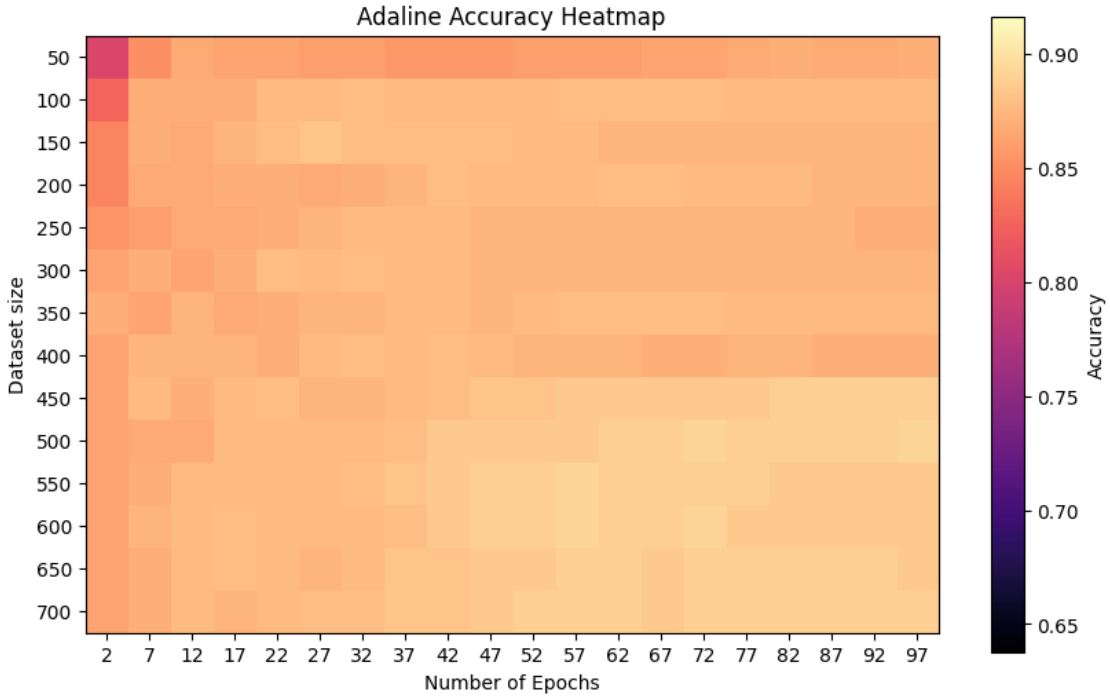
Plot the performance measure for all classifiers (accuracy on the test set; use the result array from above) of all the 280 variants for each classifier in a total of three heatmaps using, for example `seaborn` or `matplotlib` directly.

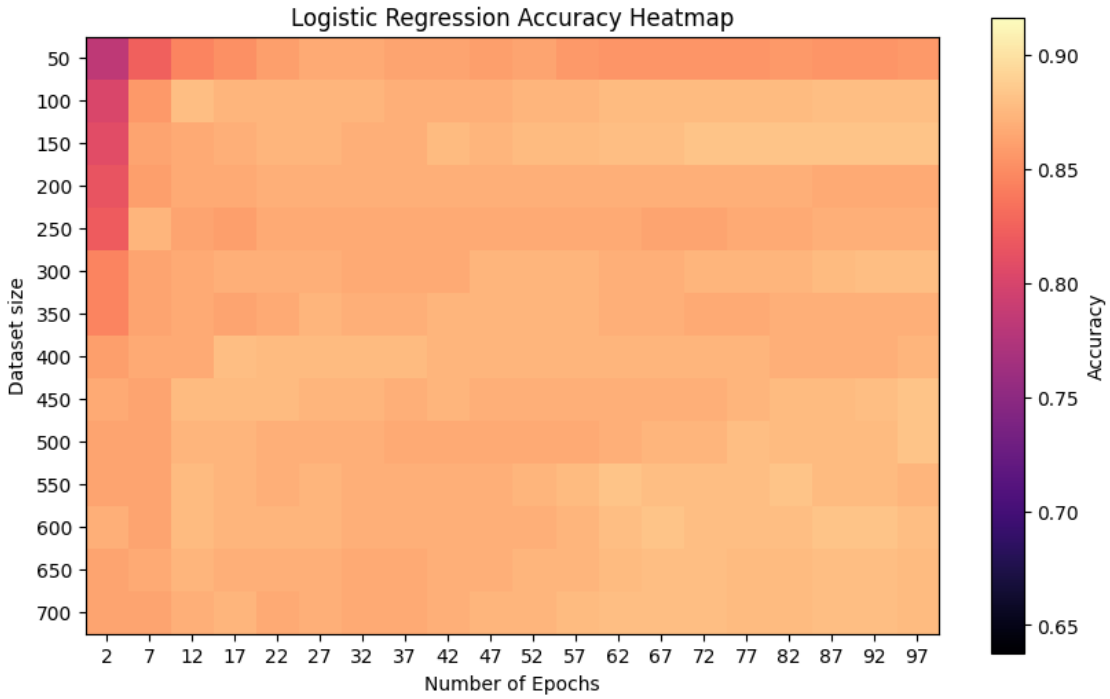
The color should represent the accuracy on the test set, and the x and y axes should represent the number of epochs and the dataset size, respectively. Which one is x and which one is y is up to you to decide. Look in the example output at the top of the assignment for inspiration for how the plot could look like and how it could be labeled nicely. (But use the correct numbers corresponding to your dataset sizes and number of epochs.)

[ ]:
```python
# compute the absolute min and max for the accurancies
min_accuracy = np.min([np.min(accuracy) for accuracy in all_accuracies])
max_accuracy = np.max([np.max(accuracy) for accuracy in all_accuracies])

classifiers = ['Perceptron', 'Adaline','Logistic Regression']
# Plot the heatmaps for each classifier
for idx in range(len(classifiers)):
    classifier_name = classifiers[idx]
    plt.figure(figsize=(10, 6))
    plt.imshow(all_accuracies[idx], cmap='magma')# change it to the color from␣
 ↪the example plot
    plt.colorbar(label='Accuracy', ax=plt.gca())  # Make sure the color bar␣
 ↪corresponds to the current axes
    plt.clim(min_accuracy, max_accuracy)  # Set the range of the color bar
    plt.title(f'{classifier_name} Accuracy Heatmap')
    plt.xlabel('Number of Epochs')
```

```
    plt.ylabel('Dataset size')
    plt.xticks(np.arange(len(num_epochs)), num_epochs)
    plt.yticks(np.arange(len(dataset_sizes)), dataset_sizes)
    plt.show()
```







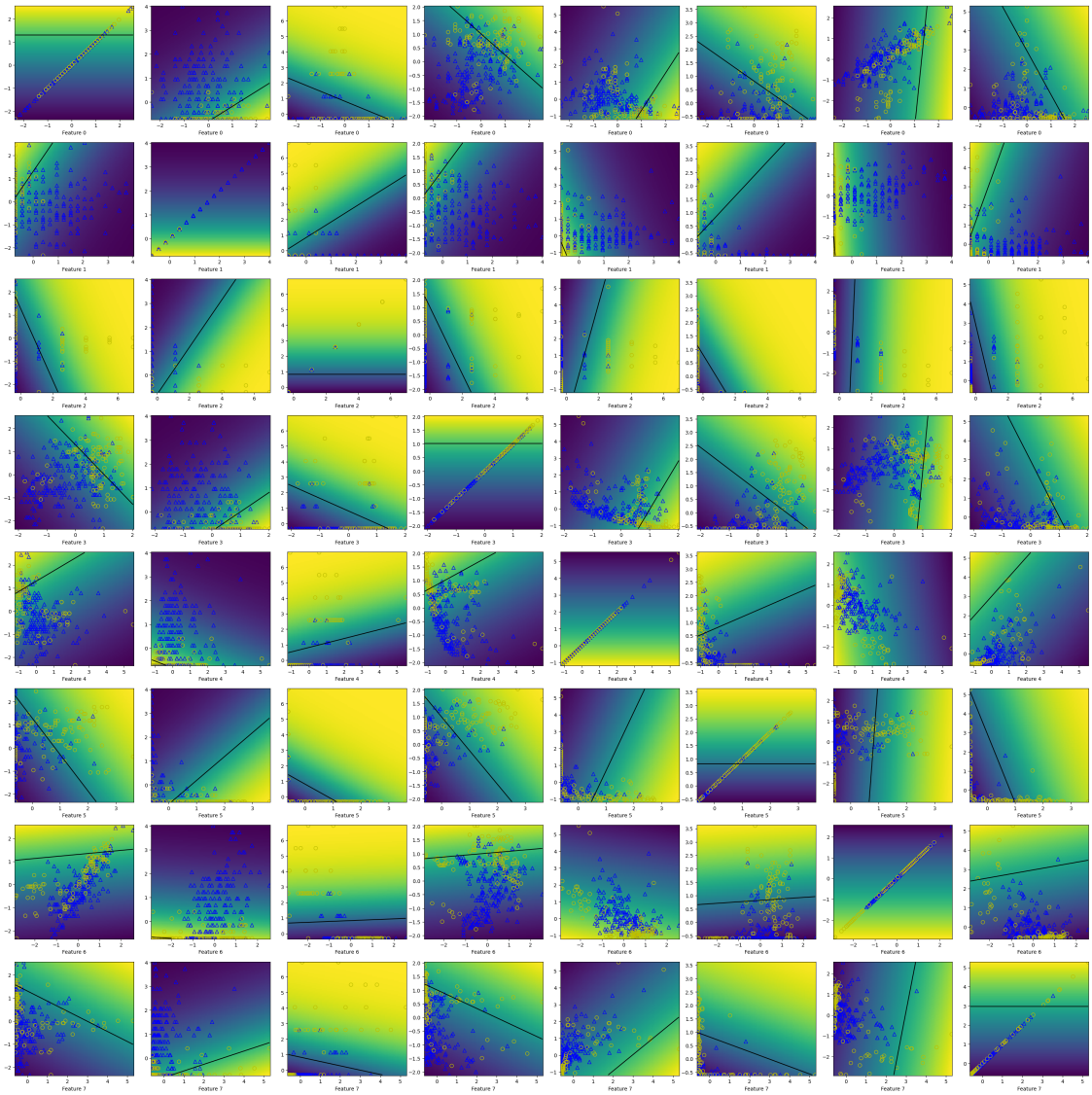
2 Part V: Some more plotting

For the following cell to execute you need to have the variable `X_test_scaled` with all samples of the test set and the variable `y_test` with the corresponding labels. Complete at least up until Part III. Executing the cell will plot something.

1. Add code comments explaining what the lines are doing.
2. What is the purpose of the plot?
   - vizualise the boundaries by logistic regression model for each pair of feature. The plots give an insight in how the classifier separated the two classes.
3. Describe all components of the subplot and then comment in general on the entire plot. What does it show? What does it not show?
   - colored mesh: predicted probabilities belong to each class. darker blue belongs indicates to belong to class 1.
   - blue triangles: reperesent class 0
   - yellow circles: reperesent class 1
   - lines: the boundary where the probabilty is 0.5

[ ]:
```
# Train and a logistic regression model with 300 epochs and learning rate 0.0001
clf = LogisticRegression(eta = 0.0001, epochs = 300, minibatches=1,␣
 ↪random_seed=42)
clf.fit(X_test_scaled, y_test)
```

```
fig, axes = plt.subplots(8, 8, figsize=(30, 30)) # initalise for the subplots
for i in range(0, 8):
    for j in range(0, 8):
        feature_1 = i
        feature_2 = j
        ax = axes[i, j] #

        #set labels for x-axis and y-axis
        ax.set_xlabel(f"Feature {feature_1}")

        # calculate min and max value
        mins = X_test_scaled.min(axis=0)
        maxs = X_test_scaled.max(axis=0)

        # makes the plot look nice by .....
        x0 = np.linspace(mins[feature_1], maxs[feature_1], 100)
        x1 = np.linspace(mins[feature_2], maxs[feature_2], 100)

        # creates a meshgrid
        X0, X1 = np.meshgrid(x0, x1) # makes a meshgrid
        X_two_features = np.c_[X0.ravel(), X1.ravel()]
        X_plot = np.zeros(shape=(X_two_features.shape[0], X_test_scaled.
 ↪shape[1])) # makes the plot with zeroes

        # give values to the plot
        X_plot[:, feature_1] = X_two_features[:, 0]
        X_plot[:, feature_2] = X_two_features[:, 1]

        # predict the probability for a point in array
        y_pred = clf.predict_proba(X_plot)
        Z = y_pred.reshape(X0.shape)

        # makes the plot
        ax.pcolor(X0, X1, Z) # colored mesh
        ax.contour(X0, X1, Z, levels=[0.5], colors='k') # countour lines where␣
 ↪the probability is 0.5
        ax.scatter(X_test_scaled[y_test == 0, feature_1], X_test_scaled[y_test␣
 ↪== 0, feature_2], color="b", marker="^", s=50, facecolors="none")
        ax.scatter(X_test_scaled[y_test == 1, feature_1], X_test_scaled[y_test␣
 ↪== 1, feature_2], color="y", marker="o", s=50, facecolors="none")

fig.tight_layout() # makes the layout look nice and that the subplots dont␣
 ↪overlap
plt.show() # display the plot
```



2.1 Part VI: Additional discussion

2.1.1 Part I:

1. What kind of plots did you use to visualize the raw data, and why did you choose these types of plots?
    - I used histograms, because it makes it easy to see if it is normal distributed and how the raw data is distributed.

2.1.2 Part II:

1. What happens if we don't shuffle the training data before training the classifiers like in Part IV?
    - if we don´t shuffle the training data, it could lead to biased training.

- All the examples from 0 and some from 1 would end up in training dataset and the test dataset would only be with 1.

2. How could you do the same train/test split (Point 1.-4.) using scikit-learn?
   - load the dataset with datasets.load
   - using train_test_split to split the dataset into separat training and testing sets
   - using standardScaler.fit from preprosessing to compjute the maean and sd for each festure and store this inside. the tranform to scale the other test and train, this to make sure data leakage doesnt happen.

2.1.3 Part IV:

1. How does increasing the dataset size affect the performance of the logistic regression model? Provide a summary of your findings.
   - increasing the datasize set usually leads to better preformance, because it has more information and gets more to learn from.
   - You can se this by looking at the heatmap for logistic regression bottom line, the accuracy is the best here, and it increase as the epoch size increase.
2. Describe the relationship between the number of epochs and model accuracy
   - training with more epochs, gives the model more data to lean from, however it may lead to overfitting beacuse it memorizes the data.
   - From the accurancy heatmaps you can se that for Adaline and Linear Regression, the accurancy increase with number of epochs.
3. Which classifier is much slower to train and why do you think that is?
   - from the accuracys it looks like its the Perceptron.
   - This is beacuse the percetion updates the weights after evaulating each training sample, and this takes a lot of time.
   - Adaline calculateions is based on the whole trainingset, and therefor takes less time.
4. One classifier shows strong fluctuations in accuracy for different dataset sizes and number of epochs. Which one is it and why do you think this happens?
   - from the accuracy heatmaps it looks like its the Adaline classifier, this may be because of the Adaline rule(Widroff-Hoff rule).
   - However, based on theory, it should have been linear Regression.