Lab Assignment #5: Developing a Web App with AI capabilities

Due Date: Week 13.

Purpose: The purpose of this assignment is to:

- Develop an Express App with AI capabilities
- Use ML and NLP in your project
- Study the impact of AI for sustainability and environment protection.

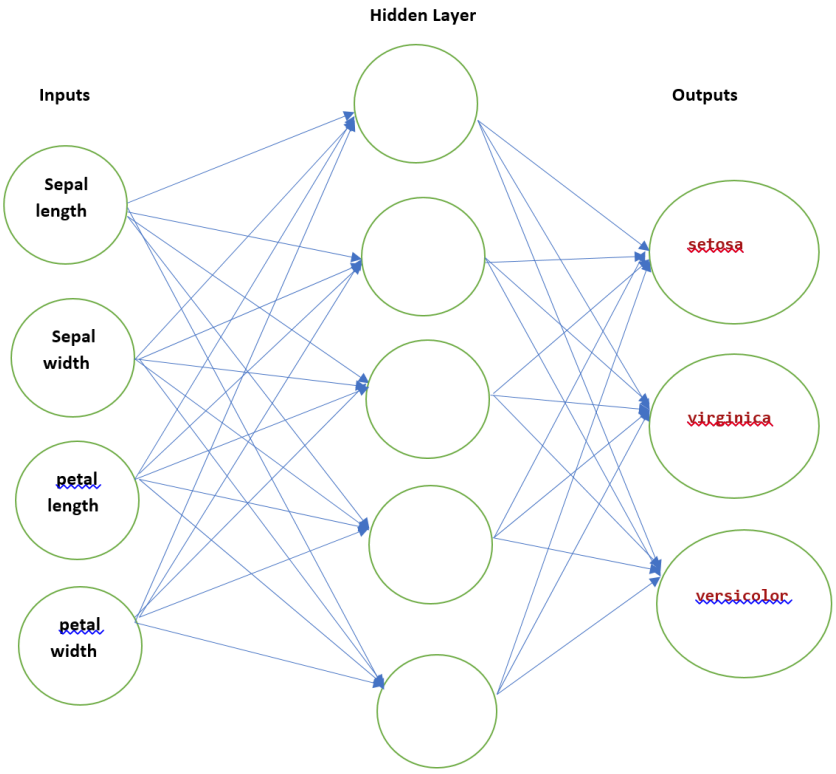
References: Read the textbook, lecture slides, and class examples. This material provides the necessary information that you need to complete the exercises.

Be sure to read the following general instructions carefully:

- This assignment must be completed using the **pair programming technique** (https://en.wikipedia.org/wiki/Pair_programming).
- You will have to demonstrate your solution in a scheduled lab session, and submit the project using the assignment link on Dropbox. **You VS project name should be named "YourFullNameLab4" and should be zipped in a file YourFullNameLab5.zip**.

Exercise 1

Write an Express app that builds and tests a three-layer neural network based on **iris** data. The shape of the network is given below:

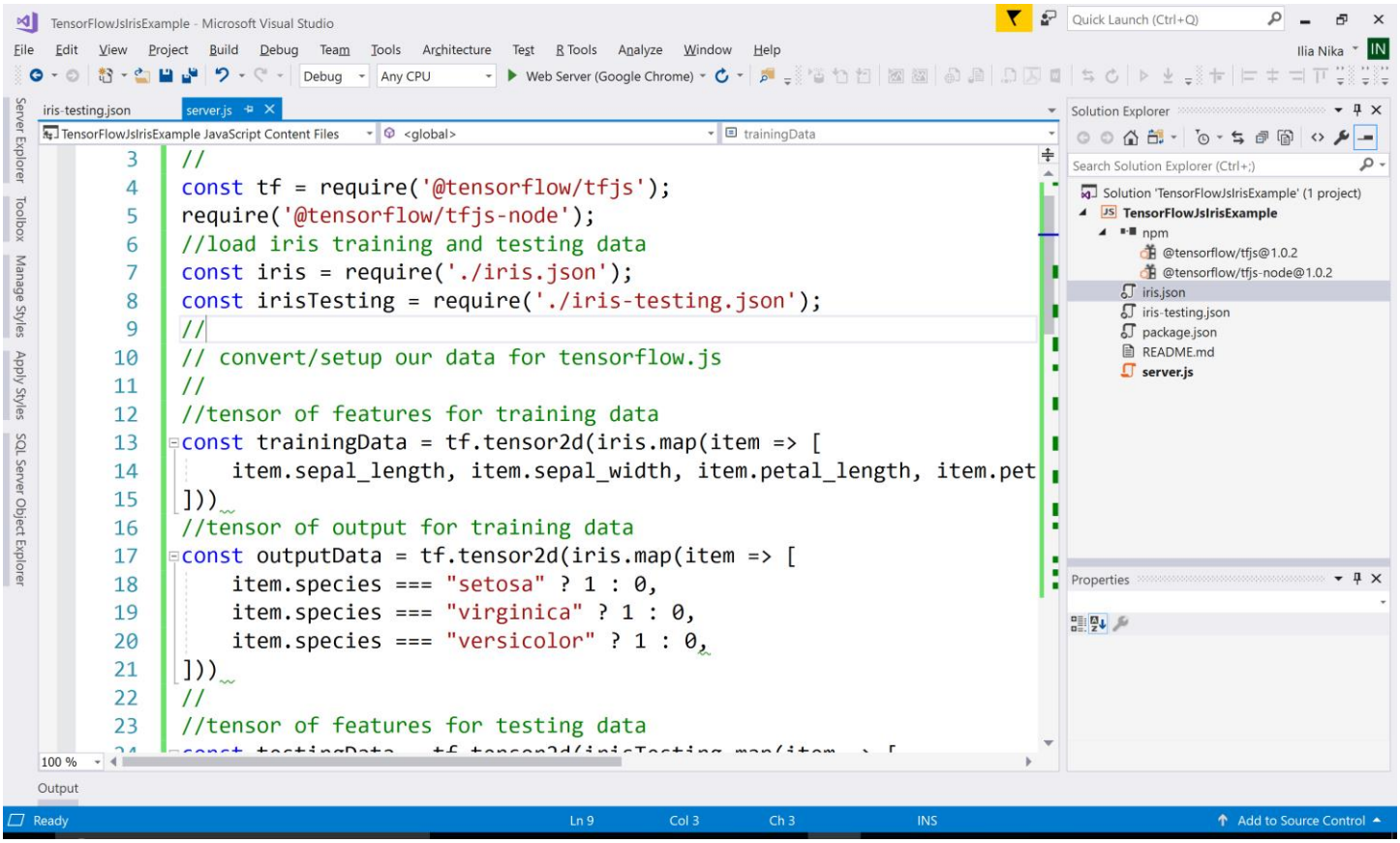


The training and testing data are provided as json files on eCentennial.

Your application should load the data from json files, convert the data into tensor format for TensorFlow.js, build the three-layer neural network using a sequential model, train the model, and predict the new entries using testing data.

Allow the user to enter **new data** to be tested (**sepal length, sepal width, petal length, petal width**), the number of **epochs and learning rate** in an **ejs** view, and display the prediction results in another **ejs** view.

The following is an initial view of the project without ejs views:




```
3   //
4   const tf = require('@tensorflow/tfjs');
5   require('@tensorflow/tfjs-node');
6   //load iris training and testing data
7   const iris = require('./iris.json');
8   const irisTesting = require('./iris-testing.json');
9   //
10  // convert/setup our data for tensorflow.js
11  //
12  //tensor of features for training data
13  const trainingData = tf.tensor2d(iris.map(item => [
14      item.sepal_length, item.sepal_width, item.petal_length, item.pet
15  ]))
16  //tensor of output for training data
17  const outputData = tf.tensor2d(iris.map(item => [
18      item.species === "setosa" ? 1 : 0,
19      item.species === "virginica" ? 1 : 0,
20      item.species === "versicolor" ? 1 : 0,
21  ]))
22  //
23  //tensor of features for testing data
```

Here is the code for your controller (without user input):

```
//
//https://github.com/PacktPublishing/Hands-on-Machine-Learning-with-
TensorFlow.js/tree/master/Section5_4
//
const tf = require('@tensorflow/tfjs');
require('@tensorflow/tfjs-node');
//load iris training and testing data
const iris = require('./iris.json');
const irisTesting = require('./iris-testing.json');
//
// convert/setup our data for tensorflow.js
//
```

```
//tensor of features for training data
const trainingData = tf.tensor2d(iris.map(item => [
    item.sepal_length, item.sepal_width, item.petal_length, item.petal_width,
]))
//tensor of output for training data
const outputData = tf.tensor2d(iris.map(item => [
    item.species === "setosa" ? 1 : 0,
    item.species === "virginica" ? 1 : 0,
    item.species === "versicolor" ? 1 : 0,
]))
//
//tensor of features for testing data
const testingData = tf.tensor2d(irisTesting.map(item => [
    item.sepal_length, item.sepal_width, item.petal_length, item.petal_width,
]))

// build neural network using a sequential model
const model = tf.sequential()
//add the first layer
model.add(tf.layers.dense({
    inputShape: [4], // four input neurons
    activation: "sigmoid",
    units: 5, //dimension of output space (first hidden layer)
}))
//add the hidden layer
model.add(tf.layers.dense({
    inputShape: [5], //dimension of hidden layer
    activation: "sigmoid",
    units: 3, //dimension of final output
}))
//add output layer
model.add(tf.layers.dense({
    activation: "sigmoid",
    units: 3, //dimension of final output
}))
//compile the model with an MSE loss function and Adam algorithm
model.compile({
    loss: "meanSquaredError",
    optimizer: tf.train.adam(.06),
})
// train/fit the model for the fixed number of epochs
const startTime = Date.now()
model.fit(trainingData, outputData, { epochs: 100 })
    .then((history) => {
        //console.log(history)
        //display prediction results for the inpud samples
        model.predict(testingData).print()
        elapsedTime = Date.now() - startTime;
        console.log(elapsedTime)

    })
```

Run this code first, and then build the views and controller. Here is a console output of running code:

(6 marks)

Exercise 2

Write an Express app for Article Summarization. The app summarizes these articles:

1. https://www.recode.net/ad/18027288/ai-sustainability-environment
2. https://www.washingtonpost.com/opinions/2019/01/23/can-we-make-artificial-intelligence-ethical/?noredirect=on&utm_term=.6a309f51e7b5

Use the example from Week 12. Display the summaries in ejs views.

(4 marks)

Evaluation:

Correct neural network model, parameters	30%
Correct prediction of new data	20%
Correct summaries of articles	25%
Correct UI	15
Friendliness (using css to align the HTML elements, etc.)	10%
Total:	100%