

Name: Armin Khayati,

Batch code: LISUM07,

Submission date: 3/26/2022,

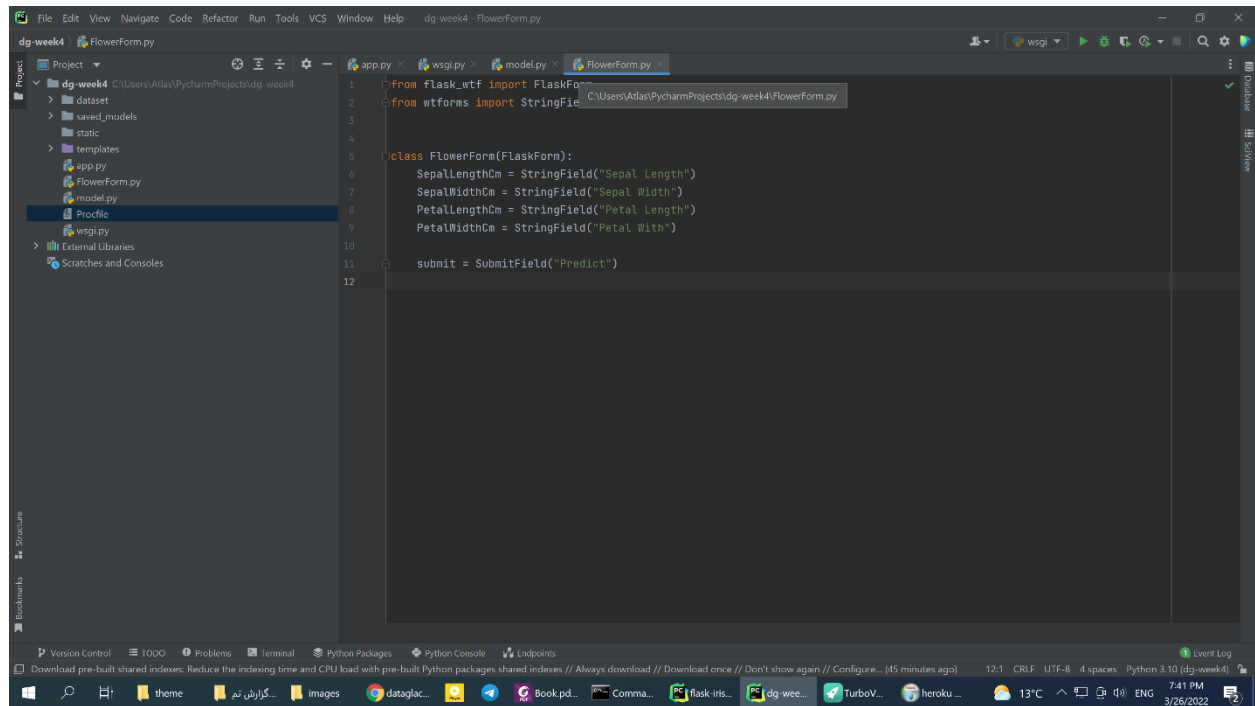
Submitted to: DataGlacier

Creating HTML Form

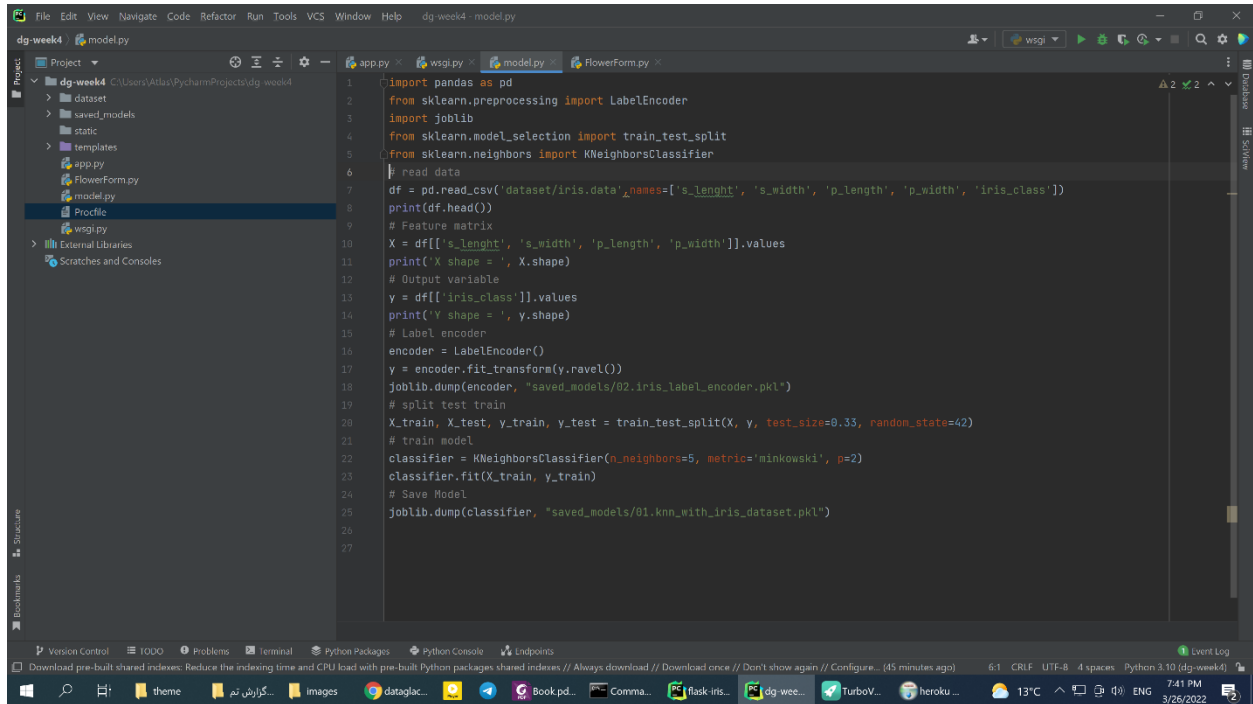
```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <title>Prediction</title>
6 <meta charset="UTF-8">
7 <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
8 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-gg0yR...
```

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <title>Prediction</title>
6 <meta charset="UTF-8">
7 <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
8 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-gg0yR...
```

Creating Class related to HTML Form Inputs



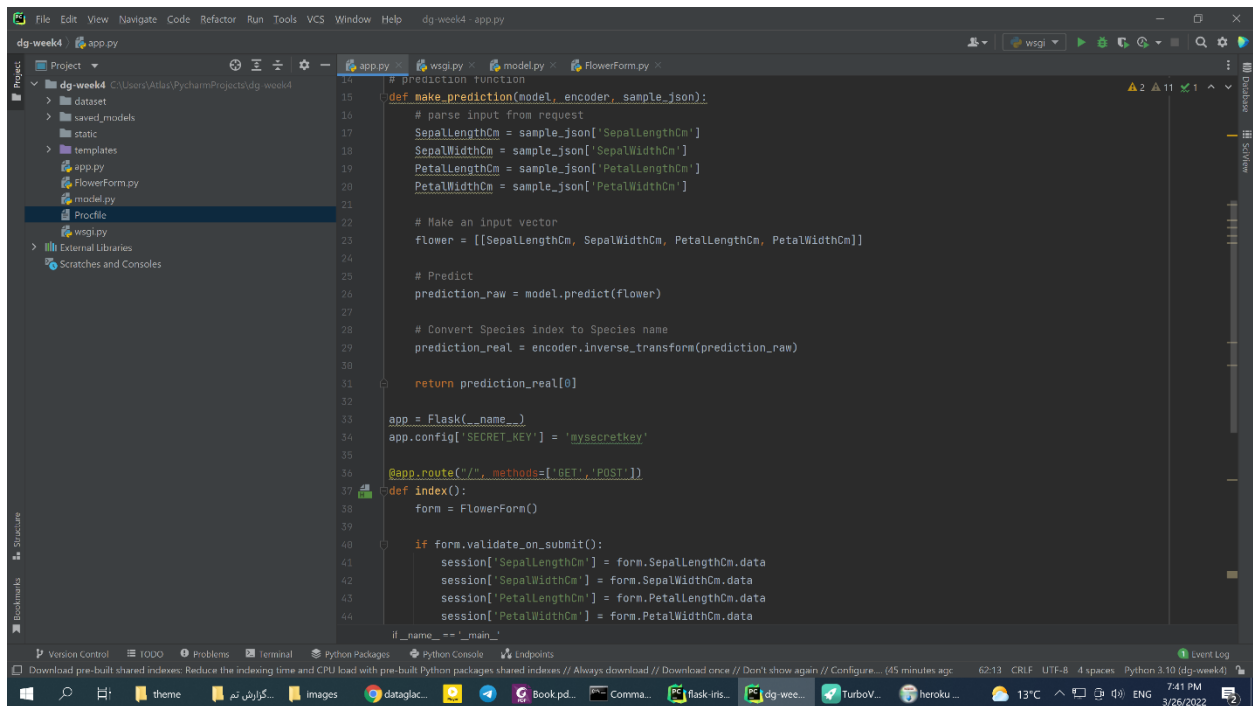
Training model



The screenshot shows the PyCharm IDE with a Python file named `model.py` open. The code imports necessary libraries: `pandas`, `sklearn.preprocessing.LabelEncoder`, `sklearn.model_selection.train_test_split`, `sklearn.neighbors.KNeighborsClassifier`, and `joblib`. It reads data from `dataset/iris.data` into a DataFrame `df`. The features are extracted into a matrix `X` and the target variable into `y`. A `LabelEncoder` is used to encode the target variable. The data is split into training and testing sets using `train_test_split`. A `KNeighborsClassifier` is trained on the training data. The trained model is saved using `joblib.dump` to a file named `01.knn_with_iris_dataset.pkl`.

```
1 import pandas as pd
2 from sklearn.preprocessing import LabelEncoder
3 import joblib
4 from sklearn.model_selection import train_test_split
5 from sklearn.neighbors import KNeighborsClassifier
6
7 # read data
8 df = pd.read_csv('dataset/iris.data', names=['s_length', 's_width', 'p_length', 'p_width', 'iris_class'])
9 print(df.head())
10 # Feature matrix
11 X = df[['s_length', 's_width', 'p_length', 'p_width']].values
12 print('X shape = ', X.shape)
13 # Output variable
14 y = df[['iris_class']].values
15 print('Y shape = ', y.shape)
16 # Label encoder
17 encoder = LabelEncoder()
18 y = encoder.fit_transform(y.ravel())
19 joblib.dump(encoder, "saved_models/02.iris_label_encoder.pkl")
20 # split test train
21 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
22 # train model
23 classifier = KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2)
24 classifier.fit(X_train, y_train)
25 # Save Model
26 joblib.dump(classifier, "saved_models/01.knn_with_iris_dataset.pkl")
27
```

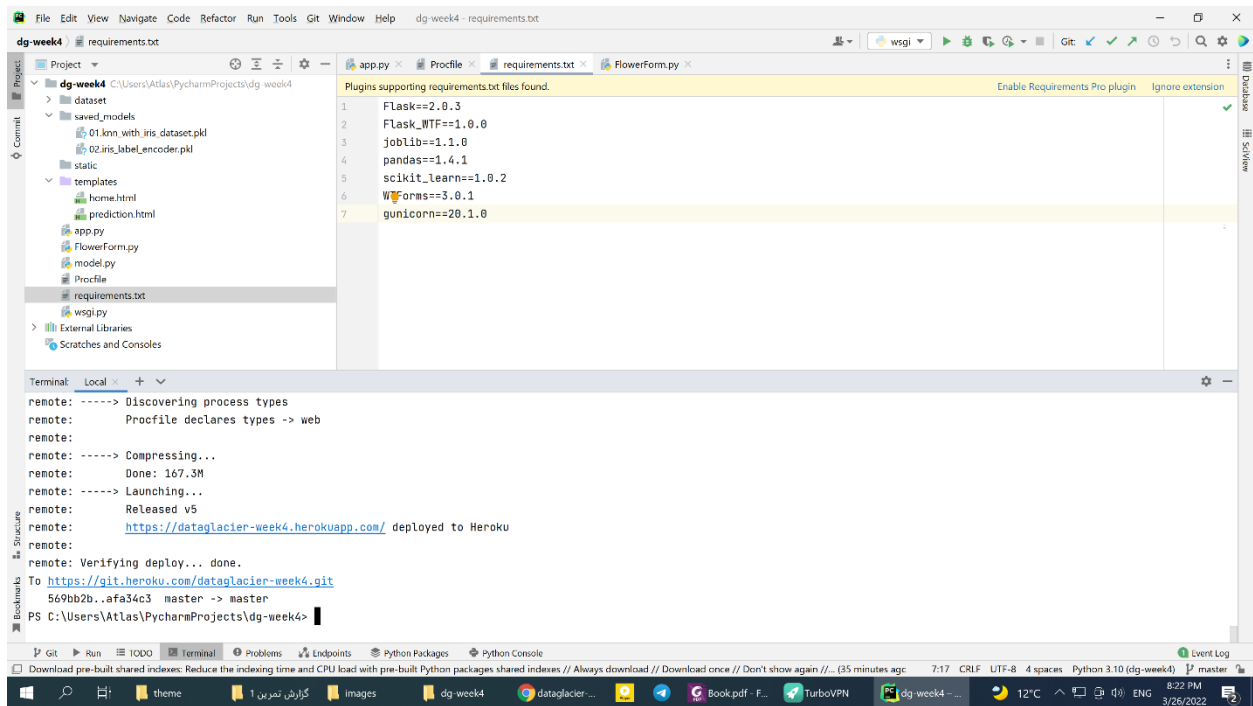
Create web app using flask



The screenshot shows the PyCharm IDE with a Python file named `app.py` open. The code defines a `make_prediction` function that takes a model, an encoder, and a sample JSON as input. It parses the input JSON to extract sepal and petal measurements, creates an input vector, and uses the trained model to make a prediction. The prediction is then converted back to a species name using the inverse transform of the encoder. The code also sets up a Flask application with a secret key and a route for the prediction function. A web form is created using `FlowerForm` and the prediction function is called when the form is submitted.

```
14 # Prediction function
15 def make_prediction(model, encoder, sample_json):
16     # parse input from request
17     SepalLengthCm = sample_json['SepalLengthCm']
18     SepalWidthCm = sample_json['SepalWidthCm']
19     PetalLengthCm = sample_json['PetalLengthCm']
20     PetalWidthCm = sample_json['PetalWidthCm']
21
22     # Make an input vector
23     flower = [[SepalLengthCm, SepalWidthCm, PetalLengthCm, PetalWidthCm]]
24
25     # Predict
26     prediction_raw = model.predict(flower)
27
28     # Convert Species index to Species name
29     prediction_real = encoder.inverse_transform(prediction_raw)
30
31     return prediction_real[0]
32
33 app = Flask(__name__)
34 app.config['SECRET_KEY'] = 'mysecretkey'
35
36 @app.route('/', methods=['GET', 'POST'])
37 def index():
38     form = FlowerForm()
39
40     if form.validate_on_submit():
41         session['SepalLengthCm'] = form.SepalLengthCm.data
42         session['SepalWidthCm'] = form.SepalWidthCm.data
43         session['PetalLengthCm'] = form.PetalLengthCm.data
44         session['PetalWidthCm'] = form.PetalWidthCm.data
45
46 if __name__ == '__main__':
47     app.run()
```

Create Requirements file



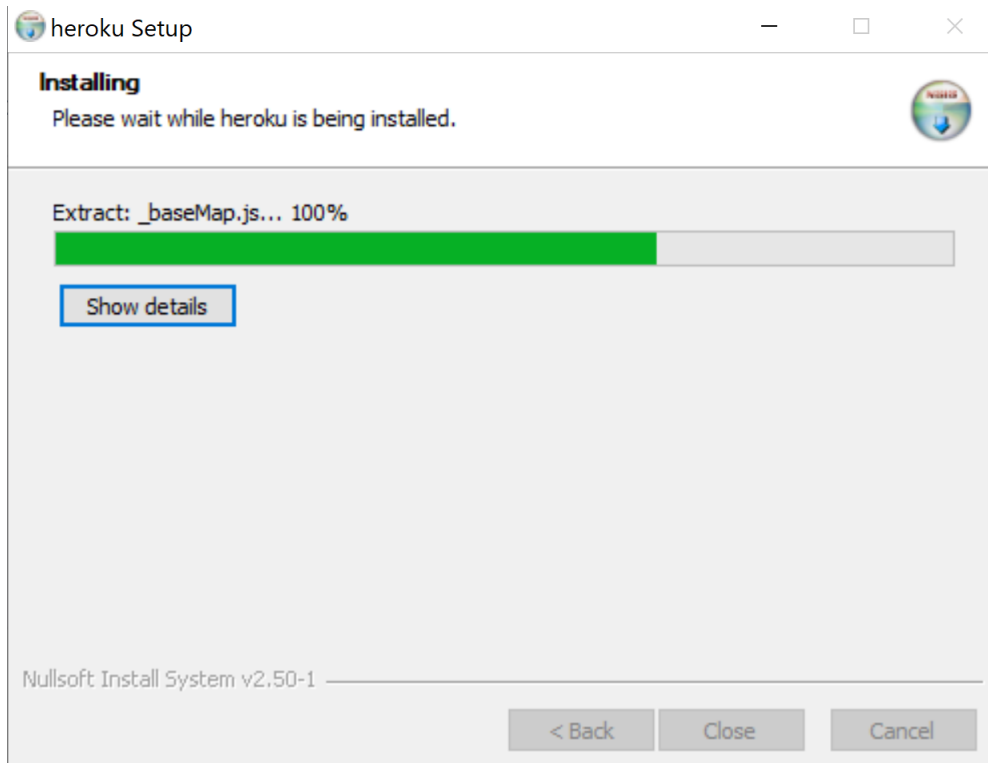
Testing Web App

Thank you. Here is the information you have provided.

Sepal Length: 10
Sepal Width: 10
Petal Length: 5
Petal Width: 5

Prediction Result: **Iris-virginica**

Installing Heroku CLI



Login to Heroku CLI

Windows PowerShell

```
PS C:\Users\Atlas\PycharmProjects\dg-week4> heroku --version
PS C:\Users\Atlas\PycharmProjects\dg-week4> heroku login
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/e73cb0da-944b-4500-ac42-b0c62121LRqXkW4\_IFnu5ccdrQjin6b8RZU
Logged in as northatlas@gmail.com
PS C:\Users\Atlas\PycharmProjects\dg-week4> git init
Initialized empty Git repository in C:/Users/Atlas/PycharmProjects/dg-week4/.git/
PS C:\Users\Atlas\PycharmProjects\dg-week4> heroku git:remote -a dataglacier-week4
» Warning: heroku update available from 7.53.0 to 7.60.0.
```

Commit to Heroku git and deploy web app

```
Initialized empty Git repository in C:/Users/Atlas/PycharmProjects/dg-week4/.git/
PS C:\Users\Atlas\PycharmProjects\dg-week4> heroku git:remote -a dataglacier-week4
» Warning: heroku update available from 7.53.0 to 7.60.0.
set git remote heroku to https://git.heroku.com/dataglacier-week4.git
PS C:\Users\Atlas\PycharmProjects\dg-week4> git add .
warning: LF will be replaced by CRLF in .idea/inspectionProfiles/profiles_settings.xml.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in dataset/iris.data.
The file will have its original line endings in your working directory
PS C:\Users\Atlas\PycharmProjects\dg-week4> git commit -am "DataGlacier Week 4"
[master (root-commit) 2b30eb9] DataGlacier Week 4
18 files changed, 403 insertions(+)
create mode 100644 .idea/.gitignore
```

The screenshot shows the Heroku dashboard for the application 'dataglacier-week4'. The top navigation bar includes the Heroku logo, a search bar, and buttons for 'Open app' and 'More'. Below the navigation bar, there are tabs for 'Overview', 'Resources', 'Deploy', 'Metrics', 'Activity', 'Access', and 'Settings'. The main content area is divided into two sections. The left section, titled 'Add this app to a pipeline', provides instructions on creating a new pipeline or choosing an existing one. The right section, titled 'Add this app to a stage in a pipeline to enable additional features', explains how pipelines can connect multiple apps and promote code, and includes a 'Choose a pipeline' dropdown menu. At the bottom, the 'Deployment method' section offers three options: 'Heroku Git: Use Heroku CLI', 'GitHub: Connect to GitHub', and 'Container Registry: Use Heroku CLI'.