Model Deployment Documentation

1. **Step 1: Create the Dataset**
    - # Dataset Creation -> same one from week 4
    - # Converts list of lists into DataFrame where x contains features while y contains labels

Code: python

```python
import pandas as pd
data = [[1,1,1], [2,1,1], [3,2,0], [4,2,0], [5,3,1], [6,3,1]]
df = pd.DataFrame(data, columns=['feature1', 'feature2', 'label'])
x = df[['feature1', 'feature2']]
y = df['label']
```

2. **Step 2: Train and Save the Model**
    - # Train and save model
    - # Trains logistic regression model using dataset
    - # creates instance of logistic regression model
    - # trains model with features 'x' and labels 'y'
    - # Save the trained model into file
    - # saves trained model to file

Code: python

```python
from sklearn.linear_model import LogisticRegression
import joblib
model = LogisticRegression()
model.fit(x, y)
joblib.dump(model, 'simple_model.pkl')
```

3. **Step 3: Create and Deploy the Flask App**
    - # initializes flask application
    - # loads trained model from file
    - # defines endpoint that accepts POST requests
    - # retrieves JSON data sent to the '/predict' endpoint
    - # converts the features from JSON into a NumPy array and reshapes to match model's input
    - # makes prediction using trained model
    - # returns prediction in JSON format
    - # Make sure the app is accessible externally

Code: python

```
from flask import Flask, request, jsonify
import joblib
import numpy as np

app = Flask(__name__)
model = joblib.load('simple_model.pkl')

@app.route('/predict', methods=['POST'])
def predict():
    data = request.get_json(force=True)
    features = np.array(data['features']).reshape(1, -1)
    prediction = model.predict(features)
    return jsonify({'prediction': int(prediction[0])})

if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0', port=5000)
```

4. **Testing**
    - Testing code using cURL command and getting expected output.

cURL Command:

sh

```
curl -X POST http://127.0.0.1:5000/predict -H "Content-Type:
application/json" -d '{"features": [3, 2]}'
```

Output:

```
curl -X POST http://127.0.0.1:5000/predict -H "Content-Type:
application/json" -d '{"features": [3, 2]}'

{

  "prediction": 1

}
```

name: Build and Deploy Python app to Azure Web App

```yaml
on:
  push:
    branches:
      - main

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
    - name: Checkout code
      uses: actions/checkout@v2

    - name: Set up Python
      uses: actions/setup-python@v2
      with:
        python-version: '3.12'

    - name: Install dependencies
      run: |
        python -m pip install --upgrade pip
        pip install -r requirements.txt

    - name: Deploy to Azure Web App
      uses: azure/webapps-deploy@v2
      with:
        app-name: 'logistic-model-app'
        publish-profile: ${{ secrets.AZURE_WEBAPP_PUBLISH_PROFILE }}
        package: '.'
```