MODEL DEPLOYMENT

▼ Model Persistance

```
1  import pandas as pd
```
✓ 4.4s

```
1  df = pd.read_csv('Advertising.csv')
```
✓ 0.5s

```
1  df
```
✓ 0.6s

	TV	radio	newspaper	sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9

```
1  X = df.drop("sales", axis=1)
2  y = df["sales"]
```
✓ 0.3s Python

```
1  from sklearn.model_selection import train_test_split
```
✓ 0.6s Python

```
1  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=101)
```
✓ 0.4s Python

```
1  # HOLD OUT SET
2
3  # Further split 30% of test into validation and
4  # ... hold-out (15% and 15% each)
5  X_validation, X_holdout_test, y_validation, y_holdout_test = train_test_split(X_test, y_test, test_size=0.5, random_state=101)
```

```
print(len(X_train))
print(len(X_test))
print(len(X_validation))
print(len(X_holdout_test))
```

✓ 0.8s

```
140
60
30
30
```

Model Training

```
from sklearn.ensemble import RandomForestRegressor
```

✓ 0.4s

+ Code + Markdown

```
model = RandomForestRegressor(n_estimators=3, random_state=101)
```

✓ 0.9s

```
model.fit(X_train, y_train)
```

✓ 0.6s

```
RandomForestRegressor(n_estimators=3, random_state=101)
```

```
from sklearn.metrics import mean_absolute_error, mean_squared_error
```

✓ 0.1s

```
val_pred = model.predict(X_validation)
```

✓ 0.8s

```
1 mean_absolute_error(y_validation,val_pred)
```
✓ 0.6s

```
0.853333333333333
```

+ Code + Markdown

```
1 mean_squared_error(y_validation,val_pred)**0.5
```
✓ 0.4s

```
1.1031268688998959
```

```
1 model = RandomForestRegressor(n_estimators=35,random_state=101)
2 model.fit(X_train,y_train)
```
✓ 0.1s

```
RandomForestRegressor(n_estimators=35, random_state=101)
```

```
1 val_pred = model.predict(X_validation)
```
✓ 0.6s

```
1 mean_absolute_error(y_validation,val_pred)
```
✓ 0.1s

```
0.6759047619047621
```

```
1 mean_squared_error(y_validation,val_pred)**0.5
```
✓ 0.8s

```
0.8585352183157281
```

Final Performance

```
1 holdout_preds = model.predict(X_holdout_test)
```
✓ 0.1s

```
1 mean_absolute_error(y_holdout_test, holdout_preds)
```
✓ 0.1s

```
0.5817142857142852
```

+ Code + Markdown

```
1 mean_squared_error(y_holdout_test, holdout_preds)**0.5
```
✓ 0.6s

```
0.730550812603694
```

```
1 final_model = RandomForestRegressor(n_estimators=35,random_state=101)
```
✓ 0.7s

```
1 final_model.fit(X,y)
```
✓ 0.1s

```
RandomForestRegressor(n_estimators=35, random_state=101)
```

```
1  import joblib
2  joblib.dump(final_model,'final_model.pkl')
```
✓ 0.9s

```
['final_model.pkl']
```

```
1  loaded_model = joblib.load('final_model.pkl')
```
✓ 0.1s

```
1  loaded_model.predict([[230.1,37.8,69.2]])
```
✓ 0.1s

```
array([21.98857143])
```

```
1
```

▼ Serving a Model as an API

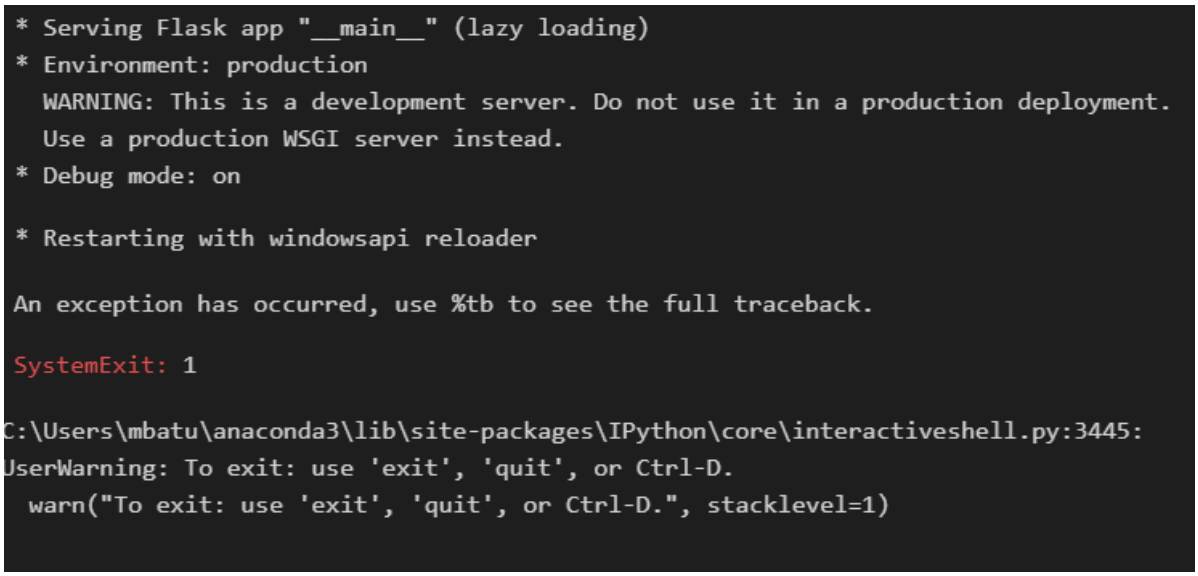
```
1  # Serving a Model as an API
```
Markdow

```
1  pip install flask
```
✓ 8.4s Pytho

```
Requirement already satisfied: flask in c:\users\mbatu\anaconda3\lib\site-packages
(1.1.2)Note: you may need to restart the kernel to use updated packages.
Requirement already satisfied: itsdangerous>=0.24 in
c:\users\mbatu\anaconda3\lib\site-packages (from flask) (1.1.0)
Requirement already satisfied: Jinja2>=2.10.1 in c:\users\mbatu\anaconda3\lib\site-
packages (from flask) (2.11.3)
Requirement already satisfied: Werkzeug>=0.15 in c:\users\mbatu\anaconda3\lib\site-
packages (from flask) (1.0.1)

Requirement already satisfied: click>=5.1 in c:\users\mbatu\anaconda3\lib\site-
packages (from flask) (7.1.2)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\mbatu\anaconda3\lib\site-
packages (from Jinja2>=2.10.1->flask) (1.1.1)
```

```python
from flask import Flask, request, jsonify
import joblib
import pandas as pd

# Create Flask App
app = Flask(__name__)


# Create API routing call
@app.route('/predict', methods=['POST'])
def predict():

    # Get JSON Request
    feat_data = request.json
    # Convert JSON request to Pandas DataFrame
    df = pd.DataFrame(feat_data)
    # Match Column Na,es
    df = df.reindex(columns=col_names)
    # Get prediction
    prediction = list(model.predict(df))
    # Return JSON version of Prediction
    return jsonify({'prediction': str(prediction)})



if __name__ == '__main__':

    # LOADS MODEL AND FEATURE COLUMNS
    model = joblib.load("final_model.pkl")
    col_names = joblib.load("column_names.pkl")

    app.run(debug=True)
```

⊗ 1.9s


* Serving Flask app "__main__" (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: on
* Restarting with windowsapi reloader
An exception has occurred, use %tb to see the full traceback.
SystemExit: 1
C:\Users\mbatu\anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3445:
UserWarning: To exit: use 'exit', 'quit', or Ctrl-D.
warn("To exit: use 'exit', 'quit', or Ctrl-D.", stacklevel=1)