Tutorial 5 Completion Document

What is machine learning?

Machine learning is a field of computer science concerned with programs that learn.

> The field of machine learning is concerned with the question of how to construct computer programs that automatically improve with experience. — Machine Learning, 1997.

From a **mathematical perspective** machine learning can be thought of as, function approximation.

There are approximately three different types of learning styles in machine learning algorithms

1. Supervised Learning
    a. Example problems are **classification** and **regression**.
    b. Input data has a known **label** or **result**
    c. Used when program needs to make predictions based off whether something is right or wrong.
2. Unsupervised Learning
3. Semi-Supervised Learning

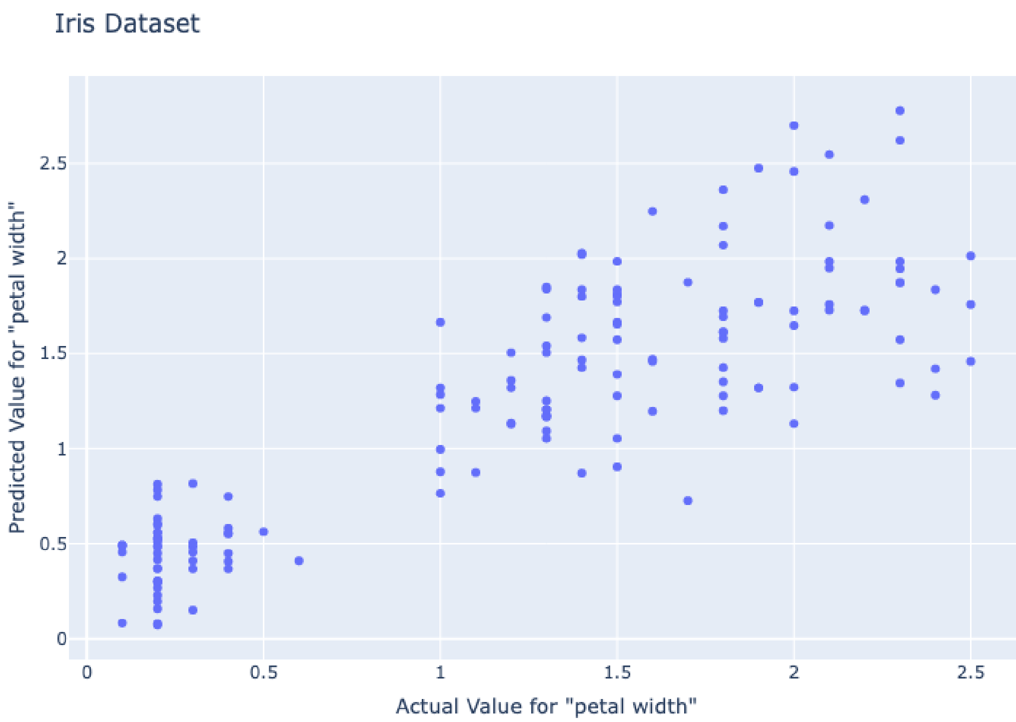
This tutorial, we will be focusing on Linear Regression.

Activity I

Fit/train the model with **sepal_width** and **sepal_length** and then predict the petal_length. Plot the actual and predicted values using Plotly.

```
import pandas as pd
import plotly.express as pd

from sktlearn.linear_model import LinearRegression

df = pdf.read_csv(fpath + '/iris.csv')

# Plotting the actual values
fig = px.bar(x=df["sepal_width"], y=df["sepal_length"])

fig.update_layout(
```

```
  xaxis_title = "Sepal Width",
  yaxis_title = "Sepal Length"
)
fig.show()

# Plotting Predicted Values

x = df[['sepal_width', 'sepal_length']]
y = df['petal_width']

model = LinearRegression()
model.fit(x, y)

prediction = model.predict(x)

plot = px.bar(x=y,
                 y=prediction,
                 labels={'x': 'Actual Value', 'y': 'Predicted Value'}
)

plot.update_layout(title="Iris Dataset',
                  autosize=False;
                  width=700,
                  height=500,
)

plot.show()
```



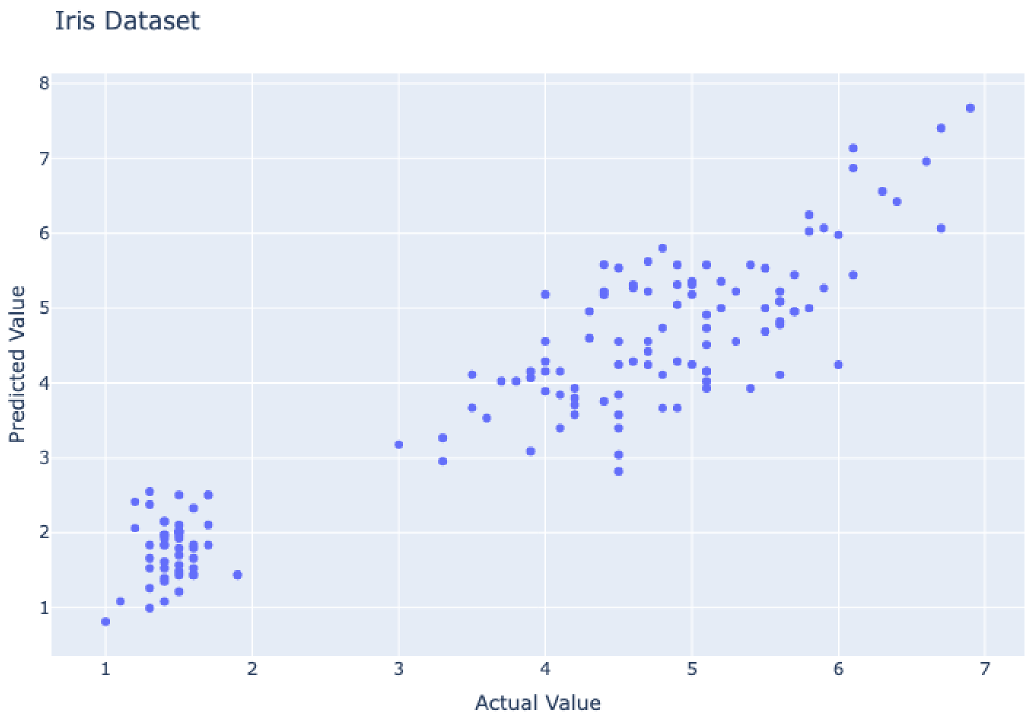
This is one of several scatter plots on my graph. Each data point contains two values: actual value and predicted value. In Juypter, you can however over the data points and it will show the actual and predicted values for each data point. Just by eyeballing this graph, I can see that almost every data point lines up vertical with another data point on the graph. The trend of the graph seems to trend upwards.

Task I

In the Tutorial Completion Document, describe your findings.

Fit/train the model with the number of **bedrooms** and **bathrooms.** And the, predict the **house price** if the number of **bedrooms is three** and the number of **bathrooms is two.** Plot the actual and predicted values using Plotly.

```
import pandas as pd
import plotly.express as px
from sklearn.linear_model import LinearRegression

fpath = "data/iris.csv"

df = pd.read_csv(fpath)

X = df[["sepal_width", "sepal_length"]]
y = df["petal_length"]

model = LinearRegression()
model.fit(X, y)

prediction = model.predict(X)

figure = px.scatter(
    x=y,
    y=prediction,
    labels={"x": "Actual Value", "y": "Predicted Value"}
)

figure.update_layout(
    title='Iris Dataset',
    width=700,
    height=500,
    margin=dict(
        l=0,
        r=0,
        b=0,
    ),
)

figure.show()
```



It seems that most the the data points on the graph are clustered together. The are no data points from 2-2.99 on the graph. Additionally, there seems to be no discernible outliers that I can observed just by taking an eye's glance. at this graph. It;s hard to see outside of a Juypter environment but the data points on the graph represent both the actual and predicted value. It seems the line of best fit trends upwards and is in close proximity with most of the points on the graph.

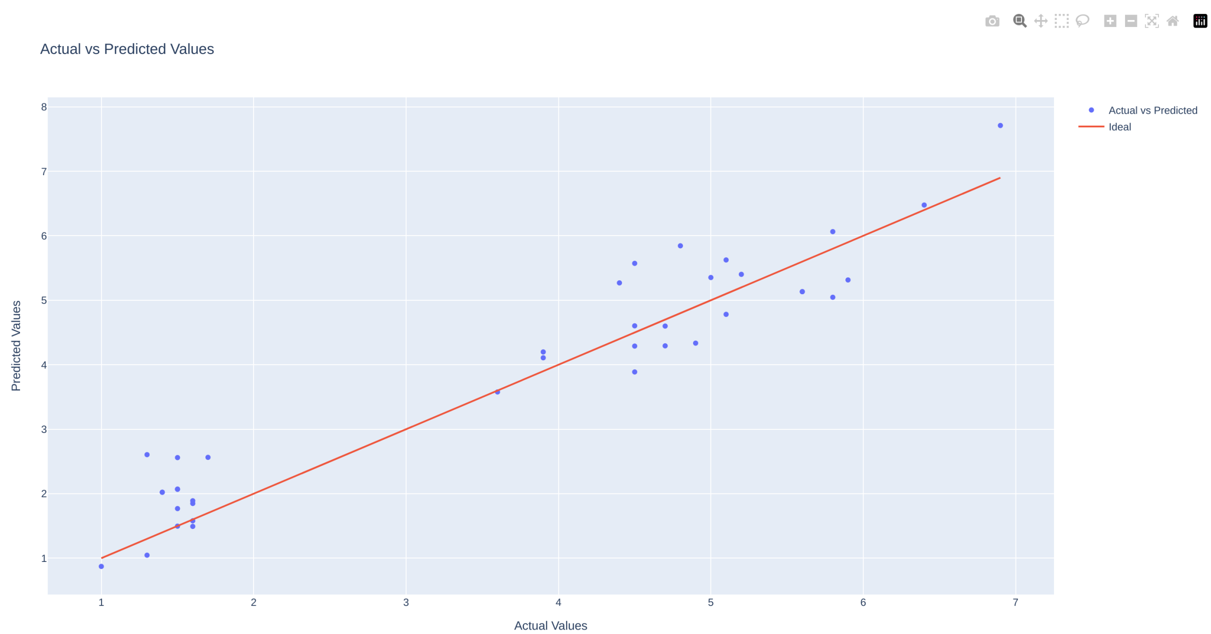
Activity II

In the Tutorial Completion Document, describe your findings.

Report and discuss the three regression evaluation metrics (MAR, MISE, and RMSE) while predicting the **petal_length** using **sepal_width** with 75% train and 25% test data. Plot the actual and predicted values.

```
import plotly.graph_objs as go
import plotly.offline as pyo
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error

fpath = "data/"

# load iris dataset
df = pd.read_csv(fpath+'iris.csv')

# extract features and target variable
X = df[['sepal_width', 'sepal_length']]
y = df['petal_length']
```

```
# split data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)

# fit linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# make predictions on test set
y_pred = model.predict(X_test)

# calculate evaluation metrics
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = mean_squared_error(y_test, y_pred, squared=False)

# print evaluation metrics
print(f"MAE: {mae:.3f}")
print(f"MSE: {mse:.3f}")
print(f"RMSE: {rmse:.3f}")

# plot actual vs predicted values
trace1 = go.Scatter(x=y_test, y=y_pred, mode='markers', name='Actual vs Predicted')
trace2 = go.Scatter(x=[y_test.min(), y_test.max()], y=[y_test.min(), y_test.max()], mode='lines', name='Ideal')
layout = go.Layout(title='Actual vs Predicted Values', xaxis=dict(title='Actual Values'), yaxis=dict(title='Predicted Values'))
fig = go.Figure(data=[trace1, trace2], layout=layout)
pyo.plot(fig)
```



This is my graph for this scatter plot. As you can observe there is a red line through the data points which shows the line of best fit. A line of best fit is a straight line that is drawn through a scatter plot to approximate the trend of the data. It seems that the data trends upwards. Similar to the graph above. It seems the the data from this graph contains only one outline. However, most of the points on the graph are not on the line of best fit.

Task II

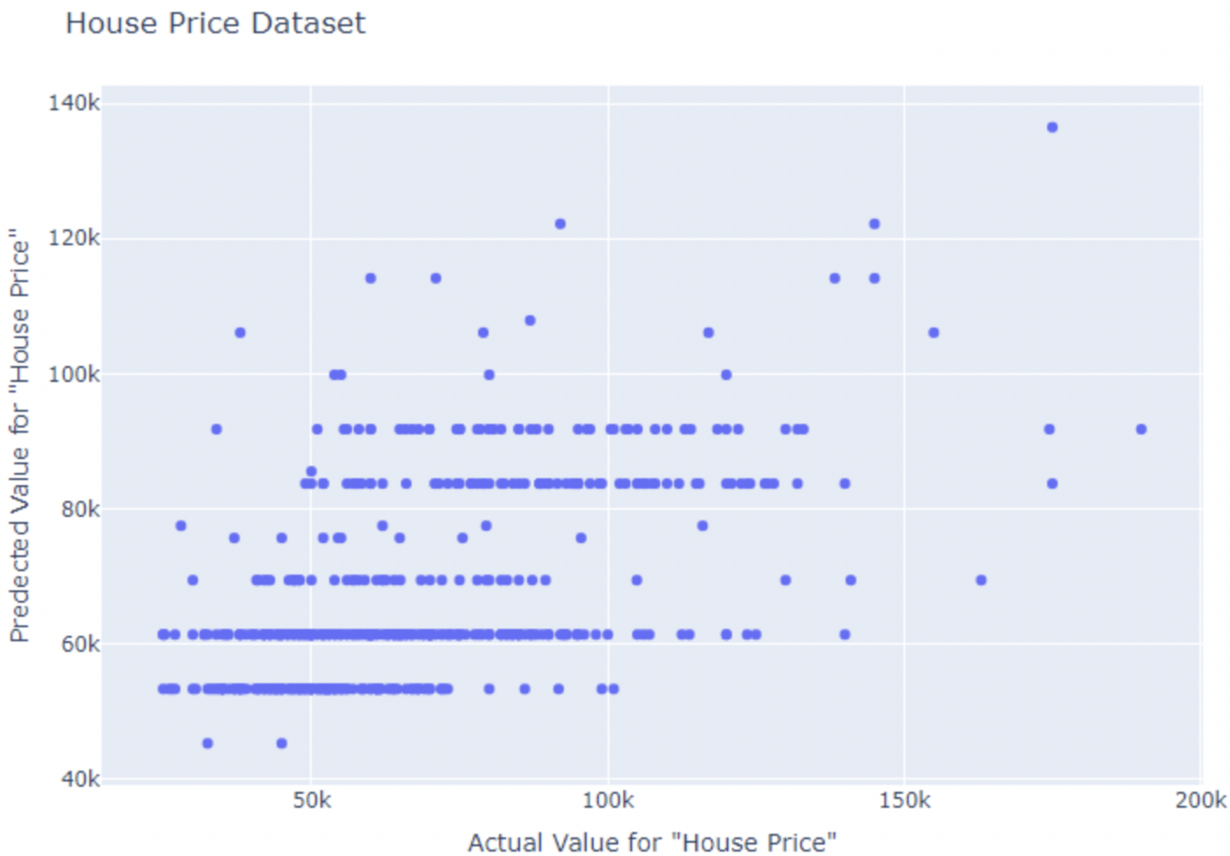
In the Tutorial Completion Document, describe your findings:

Report and discuss the three regression evaluation metrics (MAE, MISE, and RMSE) while predicting the ****house price**** using the number of **bedrooms** and the number of **bathrooms** with 80% train and 20% test data. Plot the actual and predicted values.

```
import pandas as pd
import plotly.express as px
from sklearn import metrics
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

# Load the dataset
data = pd.read_csv('data/HousePrices.csv')

# Split the data into training and testing sets
X = data[['bedrooms', 'bathrooms']]
y = data['price']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Fit a linear regression model to the training data
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on the testing data
y_pred = model.predict(X_test)

# Calculate the regression evaluation metrics
mae = metrics.mean_absolute_error(y_test, y_pred)
mse = metrics.mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)

# Print the regression evaluation metrics
print("MAE:", mae)
print("MSE:", mse)
print("RMSE:", rmse)

# Create a scatter plot of the actual versus predicted values
fig = px.scatter(x=y_test, y=y_pred)
fig.update_layout(xaxis_title="Actual Price", yaxis_title="Predicted Price")
fig.show()
```

Results:

```
To-do:

MAE: 16845.099393635086
MSE: 567245985.5854671
RMSE: 23816.92645127551
```



This is a code example in Python that trains a linear regression model on a dataset of house prices and evaluates its performance on a test set using mean absolute error (MAE), mean squared error (MSE), and root mean squared error (RMSE) metrics. Additionally, it creates a scatter plot to visualize the actual versus predicted values.

The `pandas` library is used to read the data from a CSV file and create a DataFrame. `sklearn` is used to split the data into training and testing sets, train a linear regression model, and evaluate its performance. Finally, `plotly.express` is used to create a scatter plot.

The graph generated by this code is a scatter plot. The line of best fit seems to trend upward. However, the data seems to be aggregate towards the lower left corner of the graph. In other words, most of the data points lie between `40K` and

`100K` on the actual value of the house price. The fact that are data is not evenly distributed may bias the models' prediction one way or another.