Objective : You own an ice cream business and you would like to create a model that could predict the daily revenue in dollars based on the outside air temperature (degC). You decided to build a simple Artificial Neural Network (Perceptron) to solve this problem.

Data set:

Input (X): Outside Air Temperature Output (Y): Overall daily revenue generated in dollars

```
import pandas as pd
import tensorflow as tf
```

```
from google.colab import files
file_upload= files.upload()
```

Choose files SalesData (1).csv

- **SalesData (1).csv**(text/csv) - 12385 bytes, last modified: 29/04/2024 - 100% done

```
Saving SalesData (1).csv to SalesData (1) (1).csv
```

```
import pandas as pd
df=pd.read_csv('SalesData (1).csv')
df.head(5)
```

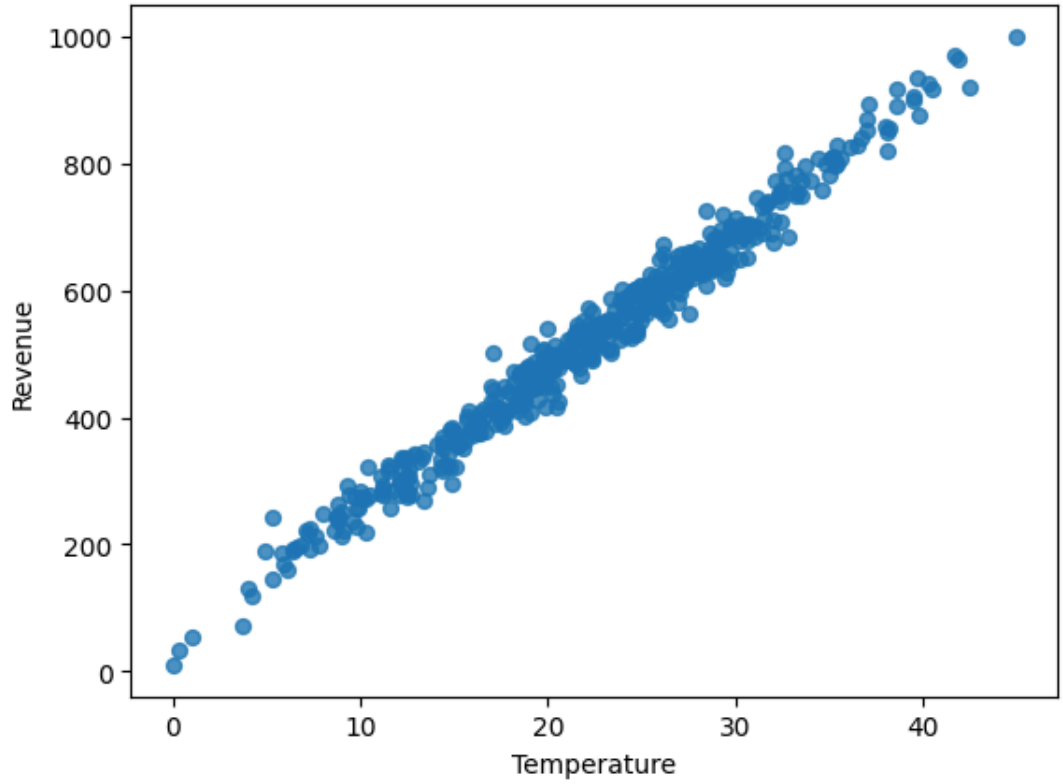
	Temperature	Revenue
0	24.566884	534.799028
1	26.005191	625.190122
2	27.790554	660.632289
3	20.595335	487.706960
4	11.503498	316.240194

Next steps: View recommended plots

Temperature vs Revenue

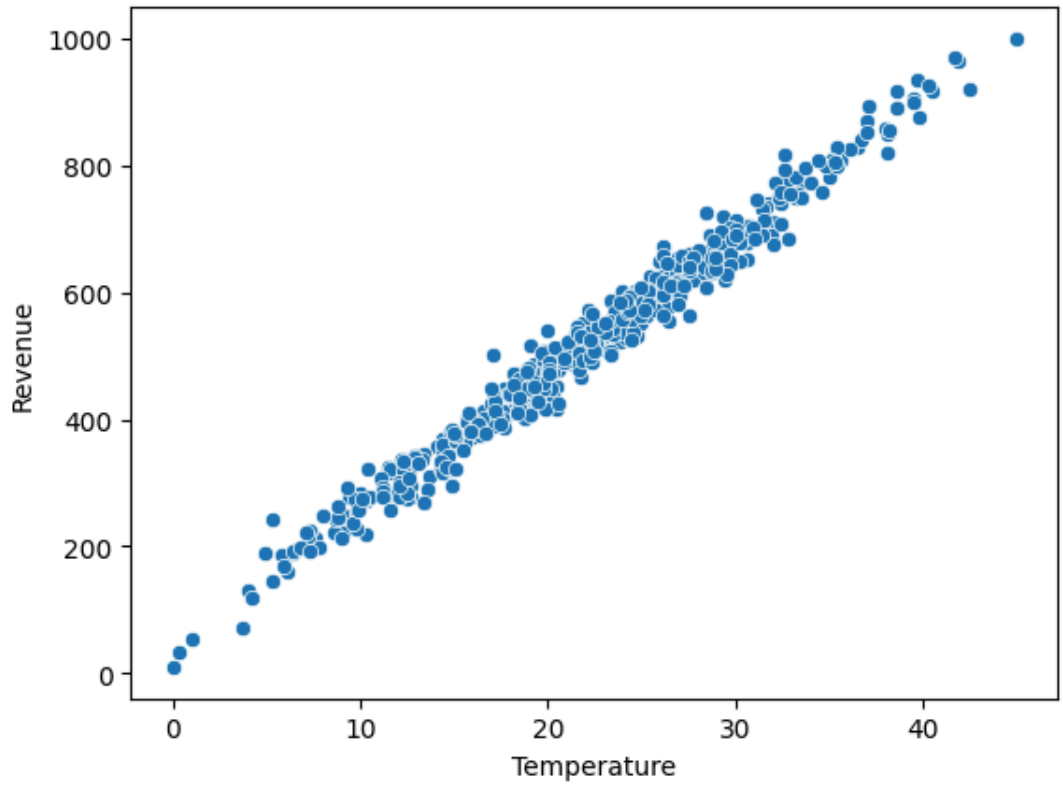
```
# @title Temperature vs Revenue

import matplotlib.pyplot as plt
df.plot(kind='scatter', x='Temperature', y='Revenue', s=32, alpha=.8)
```

```
<Axes: xlabel='Temperature', ylabel='Revenue'>
```




```
import seaborn as sns
sns.scatterplot(x=df['Temperature'],y=df['Revenue'])
```

```
<Axes: xlabel='Temperature', ylabel='Revenue'>
```




```
X_train=df['Temperature']
y_train=df['Revenue']
```

```
X_train.shape,y_train.shape
```

```
((500,), (500,))
```

```
model= tf.keras.Sequential()
model.add(tf.keras.layers.Dense(units=1,input_shape=[1]))
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 1)                 2

=================================================================
Total params: 2 (8.00 Byte)
Trainable params: 2 (8.00 Byte)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
model.compile(tf.keras.optimizers.Adam(learning_rate=0.5),loss='mean_squared_error')
```

```
epochs_hist = model.fit(X_train,y_train,epochs=1000)
```

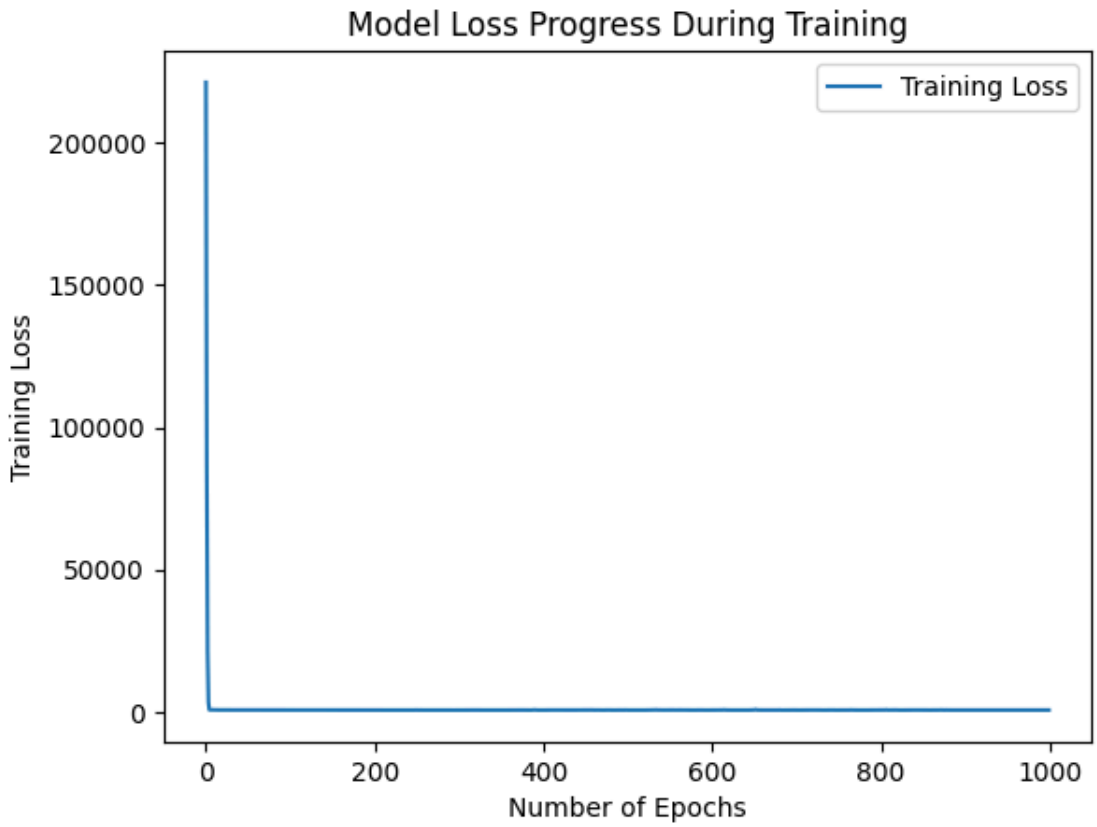
```
16/16 [==============================] - 0s 3ms/step - loss: 645.5348
Epoch 991/1000
16/16 [==============================] - 0s 2ms/step - loss: 628.7892
Epoch 992/1000
16/16 [==============================] - 0s 2ms/step - loss: 650.0618
Epoch 993/1000
16/16 [==============================] - 0s 2ms/step - loss: 658.6260
Epoch 994/1000
16/16 [==============================] - 0s 3ms/step - loss: 664.4907
Epoch 995/1000
16/16 [==============================] - 0s 3ms/step - loss: 635.5335
Epoch 996/1000
16/16 [==============================] - 0s 2ms/step - loss: 636.0042
Epoch 997/1000
16/16 [==============================] - 0s 2ms/step - loss: 634.9281
Epoch 998/1000
16/16 [==============================] - 0s 3ms/step - loss: 667.7681
Epoch 999/1000
16/16 [==============================] - 0s 2ms/step - loss: 634.8273
Epoch 1000/1000
16/16 [==============================] - 0s 2ms/step - loss: 638.4527
```

```
epochs_hist.history.keys()
```

```
dict_keys(['loss'])
```

```
import matplotlib.pyplot as plt
plt.plot(epochs_hist.history['loss'])
plt.title('Model Loss Progress During Training')
plt.xlabel('Number of Epochs')
plt.ylabel('Training Loss')
plt.legend(['Training Loss'])
```

```
<matplotlib.legend.Legend at 0x7a84140bffa0>
```



```
model.get_weights()
```

```
[array([[21.50733]], dtype=float32), array([44.847515], dtype=float32)]
```

```
temp_c=51
pred_revenue = model.predict([temp_c])
```

```
1/1 [==============================] - 0s 167ms/step
```

```
print(f'Revenue Prediction is:{pred_revenue}')
```

```
Revenue Prediction is:[[1141.7213]]
```

```
plt.scatter(X_train, y_train, color = 'gray')
plt.plot(X_train, model.predict(X_train), color = 'red')
plt.ylabel('Revenue [dollars]')
plt.xlabel('Temperature [degC]')
plt.title('Revenue Generated vs. Temperature @Ice Cream Stand')
```

```
16/16 [==============================] - 0s 2ms/step
Text(0.5, 1.0, 'Revenue Generated vs. Temperature @Ice Cream Stand')
```