PROJ302 INTERNSHIP POSTER

- Barışcan Köse / 00025413
- 11.06.2021 – 13.08.2021
- Machine Learning & Microprocessor Based System Design Internship at Ideaktif Danışmanlık & Yazılım Ltd. Şti.
- Supervisor : Ibrahim Tekin
- Sabancı University Electronics Engineering Program

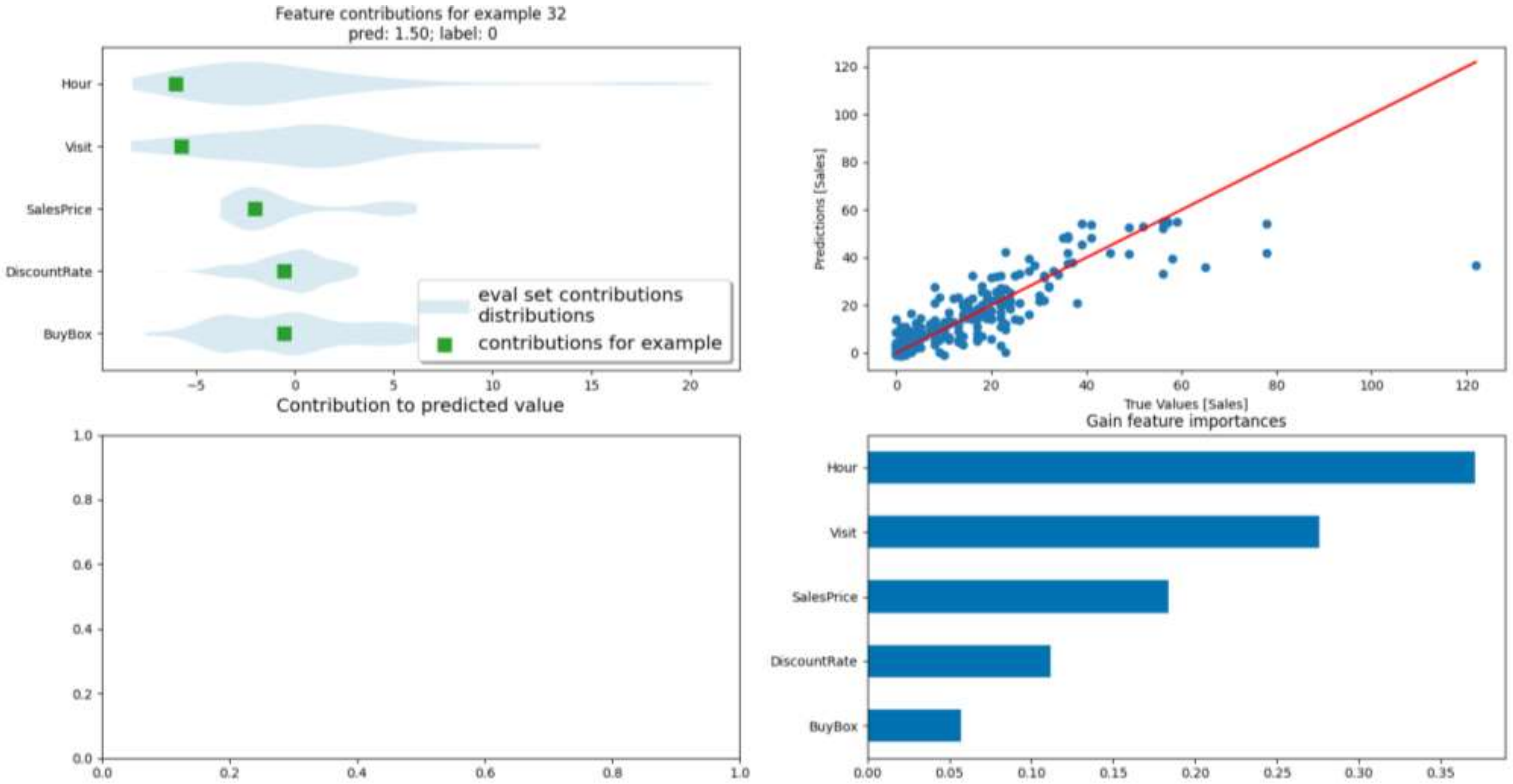
L’Oreal Distributor SALDOS’ Trendyol Sales Numbers Prediction & Sensitivity Analysis

- An e-commerce solutions provider company SALDOS, sells L’Oreal products at Trendyol in huge numbers.
- They need a way to understand what factors influence the sales numbers so that they can purchase and advertise products more strategically.
- The factors include :
  - Hour
  - Day
  - Month
  - Keyword Rank
  - Sales Price
  - Buybox status
  - Discount Rate
  - Product views

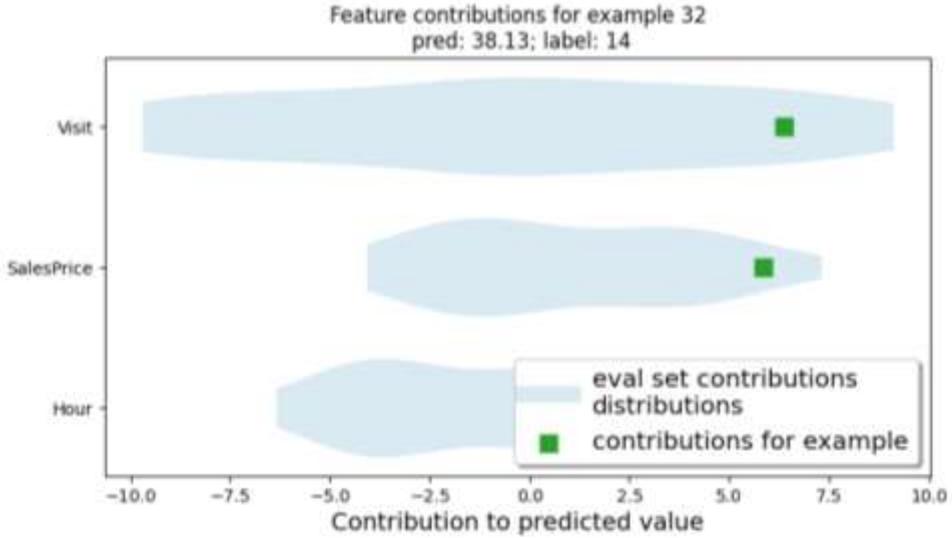
Methods & Tools

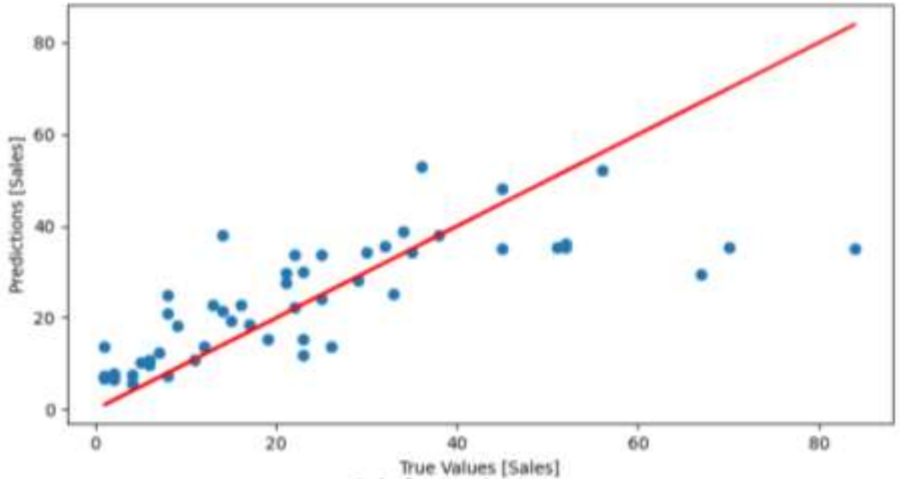
- PowerBI Analysis and exporting of the data to an excel file
- SQL and Python to keep the data in an SQL Server’s table
- Pandas dataframe to hold all the data during machine learning modelling
- Numpy array data format in order to feed the data to machine learning models
- Scikit learn to normalize and preprocess the data
- Tensorflow to create the deep neural network model, train & test the data
- Matplotlib and Seaborn to display the prediction and accuracy results from which SALDOS can predict how much a product would sell at any time with multiple varying criterias

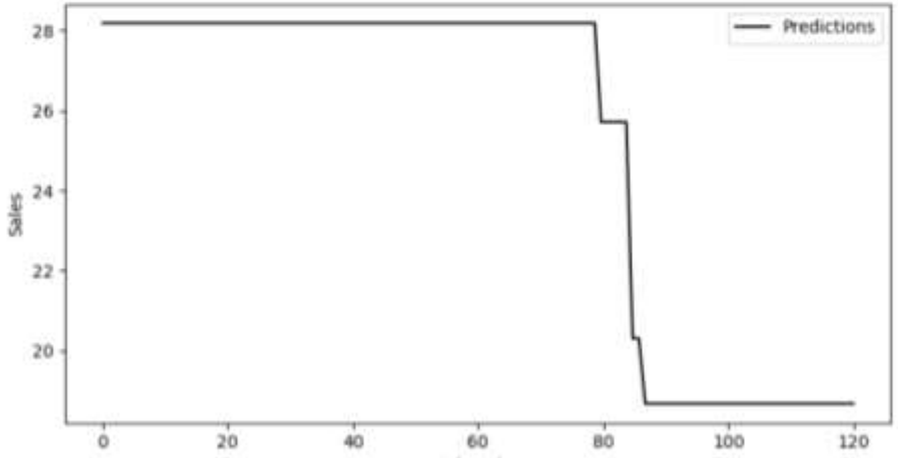
Training the Data & Sensitivity Analysis

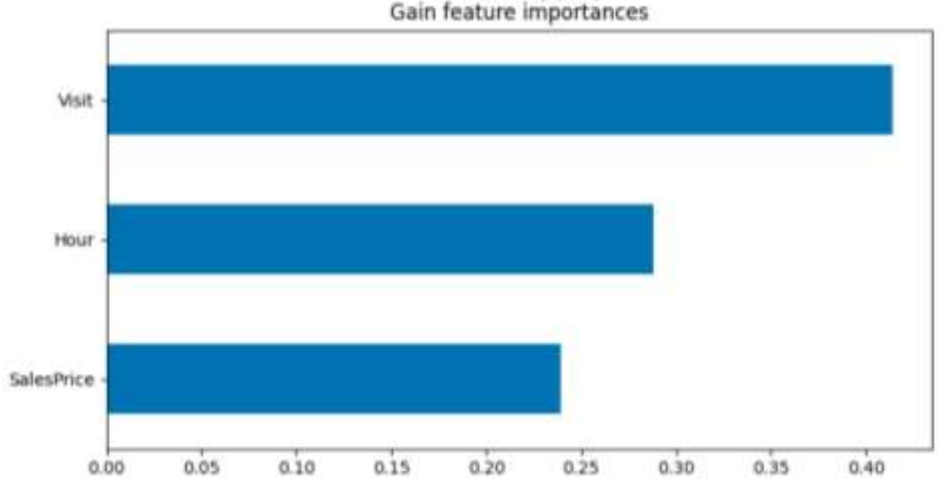


Prediction Results with only 3 Criterias









SOSCode Panic Button embedded in ID Cards for Emergency Situations in Hospitals

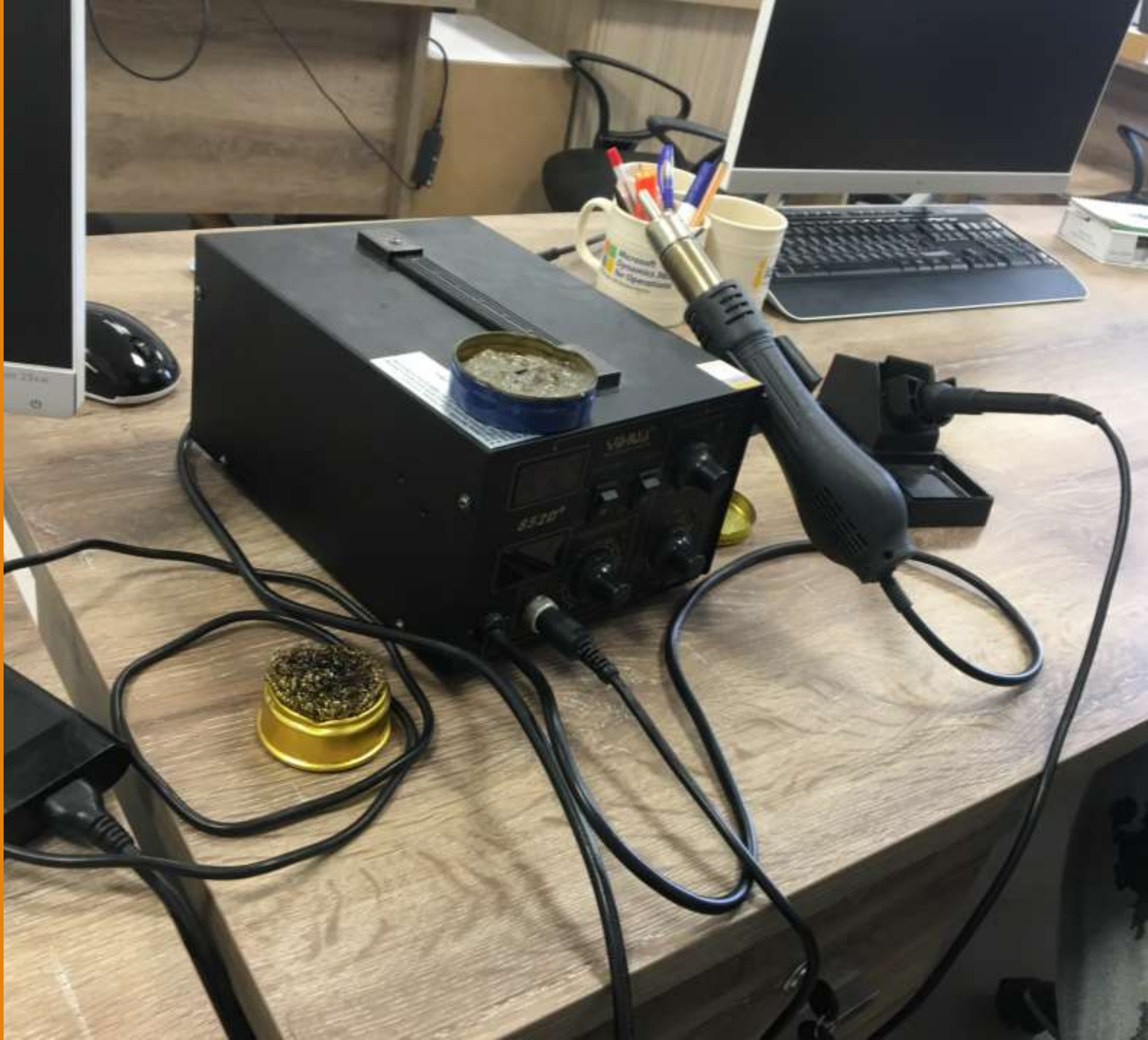
- Sometimes things get hectic in hospitals. Babies may get abducted or doctors might be assaulted. We need a way to stop or deter this from happening.
- We need a panic button which is quickly accessible and also quick to function that alerts security personnel to an emergency situation with the location data.
- The microprocessor system should also be small and be embedded in the ID cards .
- iBeacon beacons should be used for indoor positioning.
- Battery lifetime currently isn’t a priority as it is a prototype project.

Methods & Tools

- ESP32 Development board which has both Wifi and Bluetooth modules embedded saves space.
- PX402050 3.7 V battery is compatible with the ESP32 board’s operational voltage levels.
- FC-75 battery charger makes the battery rechargeable through a micro USB interface.
- iBeacon beacons periodically give out their major and minor numbers.
- 4 pin push button allows us to trigger the scan and alerting.
- Arduino IDE allows the coding of the ESP-Vroom-32 chip.
- Soldering station helps us connect all the components.
- Espressif Inc.’s bluetooth and wifi libraries for Arduino IDE allowed the ESP32 to connect to the wifi in the setup part and scan and alert in the loop part of microprocessor programming.

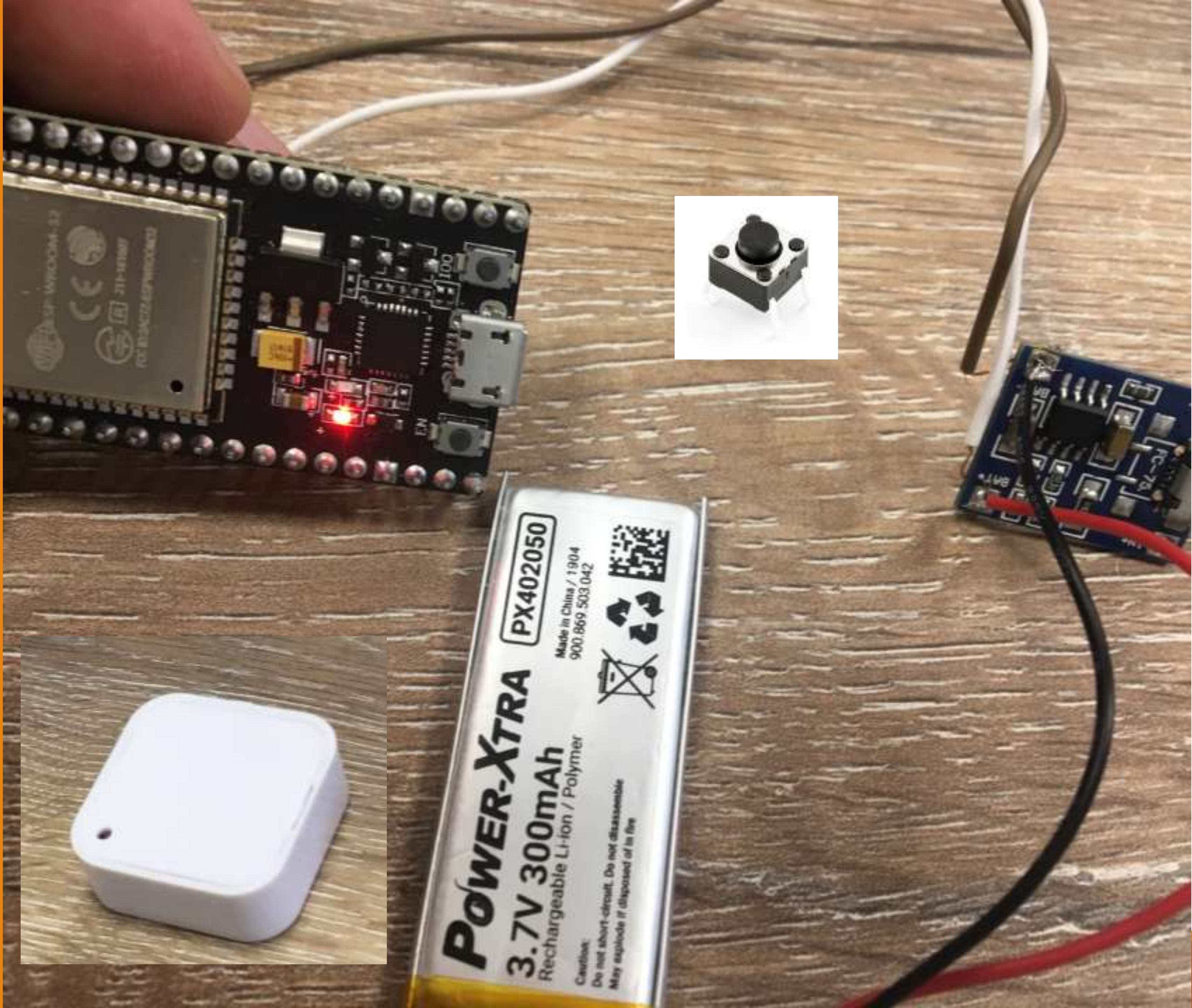
A soldering station consists of :

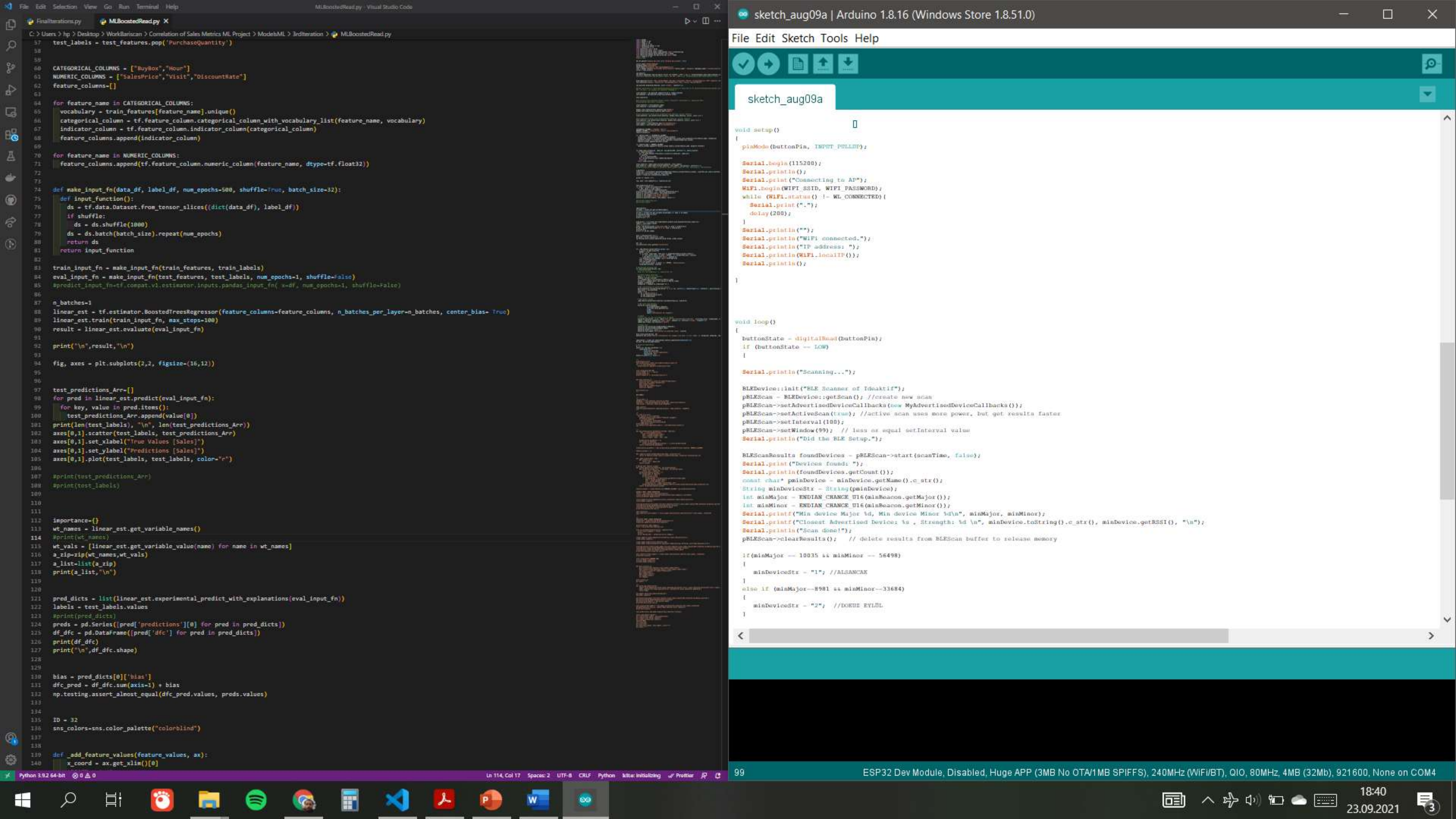
- Soldering iron
- Hot air gun
- Power station and controls
- Grease
- Soldering tip cleaner



The microprocessor system consisted of:

- 3.7V 300 mAh Power-Xtra PX402050 Rechargeable Li-Ion / Polymer Battery
- FC-75 Batter Charger
- ESP32S Dual Mode Development Board with ESP-Vroom-32 chip
- 4 pin push button
- iBeacon beacons


PX402050
Made in China / 1904
900.869.503.042
POWER-XTRA
3.7V 300mAh
Rechargeable Li-Ion / Polymer
Caution:
Do not short-circuit. Do not disassemble
May explode if disposed of in fire
FC-75


File Edit Selection View Go Run Terminal Help
MLBoostedRead.py - Visual Studio Code
FinalIterations.py
MLBoostedRead.py
C: > Users > hp > Desktop > WorkBariscan > Correlation of Sales Metrics ML Project > ModelsML > 3rdIteration > MLBoostedRead.py
Python 3.9.2 64-bit
Ln 114, Col 17
Spaces: 2
UTF-8
CRLF
Python
kite: initializing
Prettier
sketch_aug09a | Arduino 1.8.16 (Windows Store 1.8.51.0)
File Edit Sketch Tools Help
sketch_aug09a
18:40
23.09.2021

```python
test_labels = test_features.pop('PurchaseQuantity')


CATEGORICAL_COLUMNS = ["BuyBox","Hour"]
NUMERIC_COLUMNS = ["SalesPrice","Visit","DiscountRate"]
feature_columns=[]

for feature_name in CATEGORICAL_COLUMNS:
  vocabulary = train_features[feature_name].unique()
  categorical_column = tf.feature_column.categorical_column_with_vocabulary_list(feature_name, vocabulary)
  indicator_column = tf.feature_column.indicator_column(categorical_column)
  feature_columns.append(indicator_column)

for feature_name in NUMERIC_COLUMNS:
  feature_columns.append(tf.feature_column.numeric_column(feature_name, dtype=tf.float32))


def make_input_fn(data_df, label_df, num_epochs=500, shuffle=True, batch_size=32):
  def input_function():
    ds = tf.data.Dataset.from_tensor_slices((dict(data_df), label_df))
    if shuffle:
      ds = ds.shuffle(1000)
    ds = ds.batch(batch_size).repeat(num_epochs)
    return ds
  return input_function

train_input_fn = make_input_fn(train_features, train_labels)
eval_input_fn = make_input_fn(test_features, test_labels, num_epochs=1, shuffle=False)
#predict_input_fn=tf.compat.v1.estimator.inputs.pandas_input_fn( x=df, num_epochs=1, shuffle=False)

n_batches=1
linear_est = tf.estimator.BoostedTreesRegressor(feature_columns=feature_columns, n_batches_per_layer=n_batches, center_bias= True)
linear_est.train(train_input_fn, max_steps=100)
result = linear_est.evaluate(eval_input_fn)

print("\n",result,"\n")

fig, axes = plt.subplots(2,2, figsize=(16,12))


test_predictions_Arr=[]
for pred in linear_est.predict(eval_input_fn):
  for key, value in pred.items():
    test_predictions_Arr.append(value[0])
print(len(test_labels), "\n", len(test_predictions_Arr))
axes[0,1].scatter(test_labels, test_predictions_Arr)
axes[0,1].set_xlabel("True Values [Sales]")
axes[0,1].set_ylabel("Predictions [Sales]")
axes[0,1].plot(test_labels, test_labels, color="r")

#print(test_predictions_Arr)
#print(test_labels)



importance={}
wt_names = linear_est.get_variable_names()
#print(wt_names)
wt_vals = [linear_est.get_variable_value(name) for name in wt_names]
a_zip=zip(wt_names,wt_vals)
a_list=list(a_zip)
print(a_list,"\n")


pred_dicts = list(linear_est.experimental_predict_with_explanations(eval_input_fn))
labels = test_labels.values
#print(pred_dicts)
preds = pd.Series([pred['predictions'][0] for pred in pred_dicts])
df_dfc = pd.DataFrame([pred['dfc'] for pred in pred_dicts])
print(df_dfc)
print("\n",df_dfc.shape)


bias = pred_dicts[0]['bias']
dfc_pred = df_dfc.sum(axis=1) + bias
np.testing.assert_almost_equal(dfc_pred.values, preds.values)


ID = 32
sns_colors=sns.color_palette("colorblind")


def _add_feature_values(feature_values, ax):
    x_coord = ax.get_xlim()[0]
```

```cpp
void setup()
{
  pinMode(buttonPin, INPUT_PULLUP);

  Serial.begin(115200);
  Serial.println();
  Serial.print("Connecting to AP");
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED){
    Serial.print(".");
    delay(200);
  }
  Serial.println("");
  Serial.println("WiFi connected.");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  Serial.println();

}




void loop()
{
  buttonState = digitalRead(buttonPin);
  if (buttonState == LOW)
  {

  Serial.println("Scanning...");

  BLEDevice::init("BLE Scanner of Ideaktif");
  pBLEScan = BLEDevice::getScan(); //create new scan
  pBLEScan->setAdvertisedDeviceCallbacks(new MyAdvertisedDeviceCallbacks());
  pBLEScan->setActiveScan(true); //active scan uses more power, but get results faster
  pBLEScan->setInterval(100);
  pBLEScan->setWindow(99);  // less or equal setInterval value
  Serial.println("Did the BLE Setup.");

  BLEScanResults foundDevices = pBLEScan->start(scanTime, false);
  Serial.print("Devices found: ");
  Serial.println(foundDevices.getCount());
  const char* pminDevice = minDevice.getName().c_str();
  String minDeviceStr = String(pminDevice);
  int minMajor = ENDIAN_CHANGE_U16(minBeacon.getMajor());
  int minMinor = ENDIAN_CHANGE_U16(minBeacon.getMinor());
  Serial.printf("Min device Major %d, Min device Minor %d\n", minMajor, minMinor);
  Serial.printf("Closest Advertised Device: %s , Strength: %d \n", minDevice.toString().c_str(), minDevice.getRSSI(), "\n");
  Serial.println("Scan done!");
  pBLEScan->clearResults();   // delete results from BLEScan buffer to release memory

  if(minMajor == 10035 && minMinor == 56498)
  {
    minDeviceStr = "1"; //ALSANCAK
  }
  else if (minMajor==8981 && minMinor==33684)
  {
    minDeviceStr = "2";  //DOKUZ EYLÜL
  }
```

99 ESP32 Dev Module, Disabled, Huge APP (3MB No OTA/1MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600, None on COM4

PROJ302

Ideaktif Danışmanlık & Yazılım LTD. ŞTI.

Machine Learning & Microprocessor Based System Design Internship

11.06.2021-13.08.2021