In [ ]:
```
import pandas as pd
import numpy as np
```

In [ ]:
```
training_data = pd.read_csv("input_data_train.csv")
pred_data = pd.read_csv("input_data_pred.csv")
```

In [ ]:
```
from sklearn import preprocessing
```

In [ ]:

```
cols1 = ["temp_mean", "temp_max", "temp_min", "sunshine_quant", "price"]
cols2 = ["event", "location", "product"]
encoders = {}

df = df.fillna({"price": 0, "event": "N/A", "location": 0, "date": "N/A", "sa_quanti
                "temp_mean":  0, "temp_max": 0, "temp_min": 0, "sunshine_quant": 0})

df_Y = df[label_column]

for feature in cols1:
    cd = df[feature]
    cd_v = cd.values.reshape(-1, 1)

    scaler = preprocessing.MinMaxScaler()
    scaled = min_max_scaler.fit_transform(continuous_data_values)

    data_df[feature] = continuous_data_values_scaled

for feature in columns:
    le = preprocessing.LabelEncoder()
    le.fit(data_df[feature].values)
    data_df[feature] = le.transform(data_df[feature].values)
    encoders[feature] = le

df_columns = list(df.columns.values)
df_columns.remove(label_column)
df_X = df[df_columns]
```

In [ ]:
```
# Given a data frame and the columns containing numerical continuous features,
# output the data frame with normalized values per column.
def normalize_features(data_df, columns):
    for feature in columns:
        ...
```

In [ ]:
```
# Encodes columns using a LabelEncoder that exports for future transformation needs
def encode_categorical_features(data_df, columns):
    encoders = {}
    for feature in columns:
        ...
```

In [ ]:
```
# Performs preprocessing on the data for the training set.
def preprocess(df, continuous_feature_columns=[], categorical_feature_columns=[],
               label_column="sa_quantity", date_column="date"):
    df_Y = df[label_column]

    normalize_features(df, continuous_feature_columns)

    encoders = encode_categorical_features(df, categorical_feature_columns)

    ...
```

In [ ]:
```python
# Notebook exported as python module
from lewis_preprocess import preprocess

df = ...

preprocessed_df = preprocess(df)
```

In [ ]:
```python
# Do things with preprocessed_df in other cell(s)!
```

When working with other scientists and developers

In [ ]:
```
- Conda
- Colab
```

When working with everyone else...

In [1]:
```
# conda install -c conda-forge rise
# or
# pip install RISE

def uber_machin_lernin_model(x):
    return x > 0
```

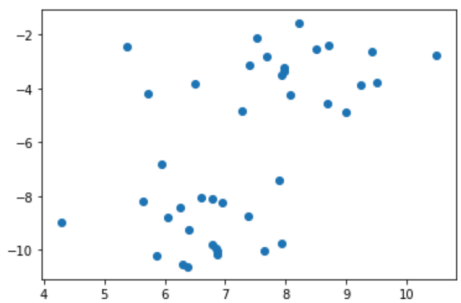
In [2]:
```
uber_machin_lernin_model(3)
```

Out[2]: True

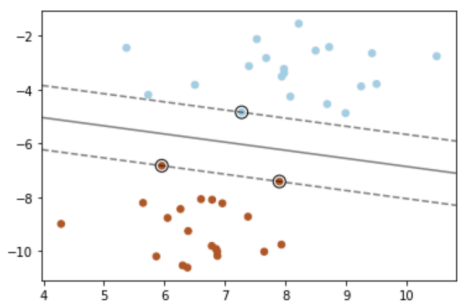
In [3]:
```
uber_machin_lernin_model(-5)
```

Out[3]: False

In [7]:
```
plot_raw_data()
```



In [8]:
```
plot_classified_data()
```



In [ ]:

```
Twitter: LewisEraik@
GitHub: LewisErick
Medium: luis061997@
```