Classifying Muffins and Cupcakes with SVM

Step 1: Import Packages

In [1]:
```
# Packages for analysis
import pandas as pd
import numpy as np
from sklearn import svm

# Packages for visuals
import matplotlib.pyplot as plt
import seaborn as sns; sns.set(font_scale=1.2)

# Allows charts to appear in the notebook
%matplotlib inline

# Pickle package
import pickle
```

Step 2: Import Data

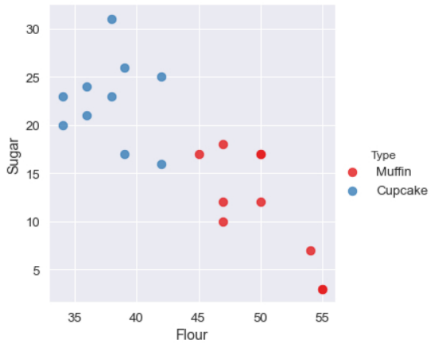
In [2]:
```
# Read in muffin and cupcake ingredient data
recipes = pd.read_csv('recipes_muffins_cupcakes.csv')
recipes
```

Out[2]:

	Type	Flour	Milk	Sugar	Butter	Egg	Baking Powder	Vanilla	Salt
0	Muffin	55	28	3	7	5	2	0	0
1	Muffin	47	24	12	6	9	1	0	0
2	Muffin	47	23	18	6	4	1	0	0
3	Muffin	45	11	17	17	8	1	0	0
4	Muffin	50	25	12	6	5	2	1	0
5	Muffin	55	27	3	7	5	2	1	0
6	Muffin	54	27	7	5	5	2	0	0
7	Muffin	47	26	10	10	4	1	0	0
8	Muffin	50	17	17	8	6	1	0	0
9	Muffin	50	17	17	11	4	1	0	0
10	Cupcake	39	0	26	19	14	1	1	0
11	Cupcake	42	21	16	10	8	3	0	0
12	Cupcake	34	17	20	20	5	2	1	0
13	Cupcake	39	13	17	19	10	1	1	0
14	Cupcake	38	15	23	15	8	0	1	0
15	Cupcake	42	18	25	9	5	1	0	0
16	Cupcake	36	14	21	14	11	2	1	0
17	Cupcake	38	15	31	8	6	1	1	0
18	Cupcake	36	16	24	12	9	1	1	0
19	Cupcake	34	17	23	11	13	0	1	0

Step 3: Prepare the Data

In [3]:
```
# Plot two ingredients
sns.lmplot('Flour', 'Sugar', data=recipes, hue='Type',
           palette='Set1', fit_reg=False, scatter_kws={"s": 70});
```



In [4]:
```
# Specify inputs for the model
# ingredients = recipes[['Flour', 'Milk', 'Sugar', 'Butter', 'Egg', 'Baking Powder', 'Vanilla', 'Salt']].as_matrix()
ingredients = recipes[['Flour','Sugar']]
type_label = np.where(recipes['Type']=='Muffin', 0, 1)

# Feature names
recipe_features = recipes.columns.values[1:].tolist()
recipe_features
```

Out[4]: `['Flour', 'Milk', 'Sugar', 'Butter', 'Egg', 'Baking Powder', 'Vanilla', 'Salt']`

Step 4: Fit the Model

In [5]:
```
# Fit the SVM model
model = svm.SVC(kernel='linear')
model.fit(ingredients, type_label)
```

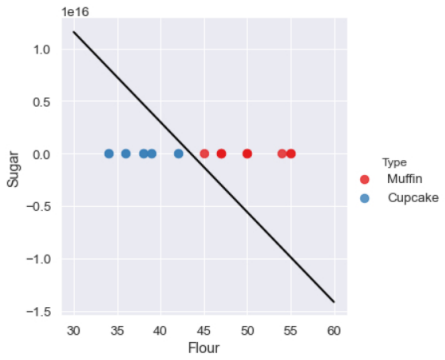
Out[5]: `SVC(kernel='linear')`

Step 5: Visualize Results

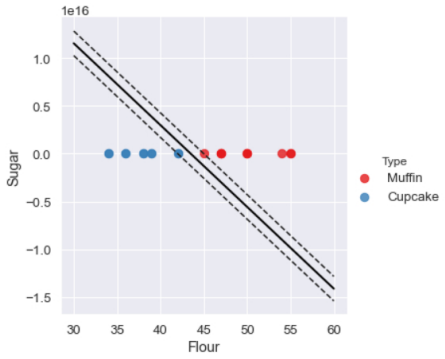
In [6]:
```
# Get the separating hyperplane
w = model.coef_[0]
a = -w[0] / w[1]
xx = np.linspace(30, 60)
yy = a * xx - (model.intercept_[0]) / w[1]

# Plot the parallels to the separating hyperplane that pass through the support vectors
b = model.support_vectors_[0]
yy_down = a * xx + (b[1] - a * b[0])
b = model.support_vectors_[-1]
yy_up = a * xx + (b[1] - a * b[0])
```

In [7]:
```
# Plot the hyperplane
sns.lmplot('Flour', 'Sugar', data=recipes, hue='Type', palette='Set1', fit_reg=False, scatter_kws={"s": 70})
plt.plot(xx, yy, linewidth=2, color='black');
```



In [8]:
```
# Look at the margins and support vectors
sns.lmplot('Flour', 'Sugar', data=recipes, hue='Type', palette='Set1', fit_reg=False, scatter_kws={"s": 70})
plt.plot(xx, yy, linewidth=2, color='black')
plt.plot(xx, yy_down, 'k--')
plt.plot(xx, yy_up, 'k--')
plt.scatter(model.support_vectors_[:, 0], model.support_vectors_[:, 1],
            s=80, facecolors='none');
```



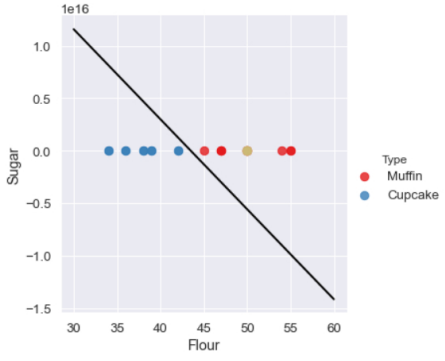
Step 6: Predict New Case

In [9]:
```
# Create a function to guess when a recipe is a muffin or a cupcake
def muffin_or_cupcake(flour, sugar):
    if(model.predict([[flour, sugar]]))==0:
        print('You\'re looking at a muffin recipe!')
    else:
        print('You\'re looking at a cupcake recipe!')
```

In [10]:
```
# Predict if 50 parts flour and 20 parts sugar
muffin_or_cupcake(50, 20)
```

```
You're looking at a muffin recipe!
```

In [11]:
```
# Plot the point to visually see where the point lies
sns.lmplot('Flour', 'Sugar', data=recipes, hue='Type', palette='Set1', fit_reg=False, scatter_kws={"s": 70})
plt.plot(xx, yy, linewidth=2, color='black')
plt.plot(50, 20, 'yo', markersize='9');
```



In [12]:
```
# Predict if 40 parts flour and 20 parts sugar
muffin_or_cupcake(40,20)
```

```
You're looking at a cupcake recipe!
```

In [13]:
```
muffin_cupcake_dict = {'muffin_cupcake_model': model, 'muffin_cupcake_features': ['Flour','Sugar'], 'all_features': recipe_featur
```

In [14]:
```
muffin_cupcake_dict
```

Out[14]:
```
{'muffin_cupcake_model': SVC(kernel='linear'),
 'muffin_cupcake_features': ['Flour', 'Sugar'],
 'all_features': ['Flour',
  'Milk',
  'Sugar',
  'Butter',
  'Egg',
  'Baking Powder',
  'Vanilla',
  'Salt']}
```

In [15]:
```
# Pickle
pickle.dump(muffin_cupcake_dict, open("muffin_cupcake_dict.p", "wb"))
```

In [16]:
```
# S = String
pickle.dumps(muffin_cupcake_dict)
```

Out[16]:
```
b'\x80\x04\x95f\x05\x00\x00\x00\x00\x00}\x94(\x8c\x14muffin_cupcake_model\x94\x8c\x14sklearn.svm._classes\x94\x8c\x03SVC\x9
4\x93\x94)\x81\x94}\x94(\x8c\x17decision_function_shape\x94\x8c\x03ovr\x94\x8c\nbreak_ties\x94\x89\x8c\x06kernel\x94\x8c\x06lin
ear\x94\x8c\x06degree\x94K\x03\x8c\x05gamma\x94\x8c\x05scale\x94\x8c\x05coef0\x94G\x00\x00\x00\x00\x00\x00\x00\x00\x8c\x03tol\x
94G?PbM\xd2\xf1\xa9\xfc\x8c\x01C\x94G?\xf0\x00\x00\x00\x00\x00\x00\x8c\x02nu\x94G\x00\x00\x00\x00\x00\x00\x00\x8c\x07epsilo
n\x94G\x00\x00\x00\x00\x00\x00\x00\x8c\tshrinking\x94\x88\x8c\x0bprobability\x94\x89\x8c\ncache_size\x94K\xc8\x8c\x0cclass_
weight\x94N\x8c\x07verbose\x94\x89\x8c\x08max_iter\x94J\xff\xff\xff\xff\x8c\x0crandom_state\x94N\x8c\x07_sparse\x94\x89\x8c\x0e
n_features_in_\x94K\x02\x8c\rclass_weight_\x94\x8c\x15numpy.core.multiarray\x94\x8c\x0c_reconstruct\x94\x93\x94\x8c\x05numpy\x9
4\x8c\x07ndarray\x94\x93\x94K\x00\x85\x94C\x01b\x94\x87\x94R\x94(K\x01K\x02\x85\x94h!\x8c\x05dtype\x94\x93\x94\x8c\x02f8\x94K\x
00K\x01\x87\x94R\x94(K\x03\x8c\x01<\x94NNNJ\xff\xff\xff\xffJ\xff\xff\xff\xffK\x00t\x94b\x89C\x10\x00\x00\x00\x00\x00\x00\xf0?\x
00\x00\x00\x00\x00\x00\xf0?\x94t\x94b\x8c\x08classes_\x94h h#K\x00\x85\x94h%\x87\x94R\x94(K\x01K\x02\x85\x94h*\x8c\x02i4\x94K\x
00K\x01\x87\x94R\x94(K\x03h.NNNJ\xff\xff\xff\xffJ\xff\xff\xff\xffK\x00t\x94b\x89C\x08\x00\x00\x00\x00\x01\x00\x00\x00\x94t\x94b
\x8c\x06_gamma\x94h\x1e\x8c\x06scalar\x94\x93\x94h-C\x08\x01\x1d\x01t\x04\xd0a?\x94\x86\x94R\x94\x8c\x08support_\x94h h#K\x00\x
85\x94h%\x87\x94R\x94(K\x01K\x03\x85\x94h9\x89C\x0c\x03\x00\x00\x00\x0b\x00\x00\x00\x0f\x00\x00\x00\x94t\x94b\x8c\x10support_ve
ctors_\x94h h#K\x00\x85\x94h%\x87\x94R\x94(K\x01K\x03K\x02\x86\x94h-\x89C0\x00\x00\x00\x00\x00\x80F@\x00\x00\x00\x00\x00\x001@
\x00\x00\x00\x00\x00\x00E@\x00\x00\x00\x00\x00\x000@\x00\x00\x00\x00\x00\x00E@\x00\x00\x00\x00\x00\x009@\x94t\x94b\x8c\n_n_supp
ort\x94h h#K\x00\x85\x94h%\x87\x94R\x94(K\x01K\x02\x85\x94h9\x89C\x08\x01\x00\x00\x00\x00\x02\x00\x00\x00\x94t\x94b\x8c\ndual_coef_
\x94h h#K\x00\x85\x94h%\x87\x94R\x94(K\x01K\x01K\x03\x86\x94h-\x89C\x18\x19;\x16\x81\xfdo\xcc\xbf5\xdf\xda9\x1aG\xc9? \xdf\xda9
\x1aG\x99?\x94t\x94b\x8c\nintercept_\x94h h#K\x00\x85\x94h%\x87\x94R\x94(K\x01K\x01\x85\x94h-\x89C\x08\xe9\xbcm\xd12\xfe<@\x94t
\x94b\x8c\x06_probA\x94h h#K\x00\x85\x94h%\x87\x94R\x94(K\x01K\x00\x85\x94h-\x89C\x00\x94t\x94b\x8c\x06_probB\x94h h#K\x00\x85
\x94h%\x87\x94R\x94(K\x01K\x00\x85\x94h-\x89hkt\x94b\x8c\x0bfit_status_\x94K\x00\x8c\nshape_fit_\x94K\x14K\x02\x86\x94\x8c\x0b_
intercept_\x94h h#K\x00\x85\x94h%\x87\x94R\x94(K\x01K\x01\x85\x94h-\x89C\x08\xe9\xbcm\xd12\xfe<\xc0\x94t\x94b\x8c\x0b_dual_coef
_\x94h h#K\x00\x85\x94h%\x87\x94R\x94(K\x01K\x01K\x03\x86\x94h-\x89C\x18\x19;\x16\x81\xfdo\xcc?5\xdf\xda9\x1aG\xc9\xbf \xdf\xda
9\x1aG\x99\xbf\x94t\x94b\x8c\x10_sklearn_version\x94\x8c\x060.23.1\x94ub\x8c\x17muffin_cupcake_features\x94]\x94(\x8c\x05Flour
\x94\x8c\x05Sugar\x94e\x8c\x0call_features\x94]\x94(\x8c\x05Flour\x94\x8c\x04Milk\x94\x8c\x05Sugar\x94\x8c\x06Butter\x94\x8c\x0
3Egg\x94\x8c\rBaking Powder\x94\x8c\x07Vanilla\x94\x8c\x04Salt\x94eu.'
```

In [ ]:

In [ ]: