Naive Bayes Classifier

1) Simulation with small training set and varying proportion used for imputation

sample size of training dataset = 20

total sample size = 3516

proportion used for imputation = 0, 0,001, 0,005, 0,01, 0,05, 0,1

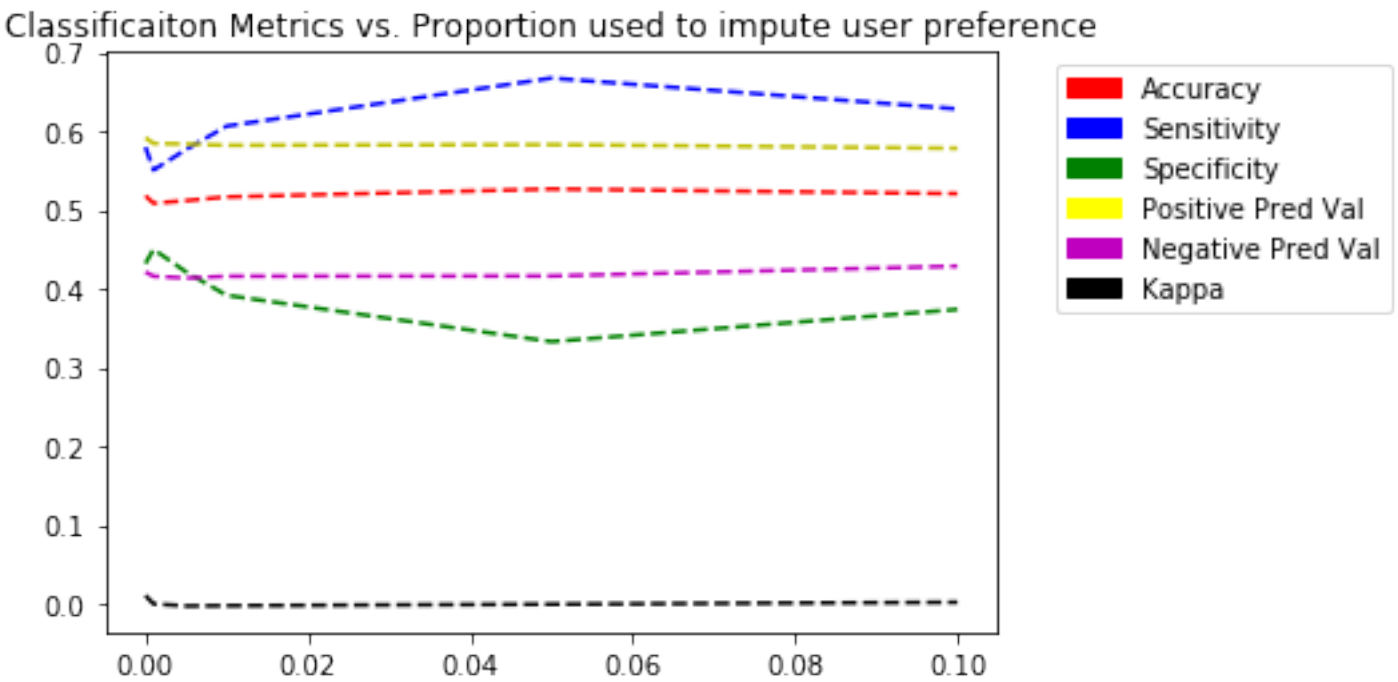
number of runs under each scenario: 100

statistic: mean

In [11]:

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
import pandas as pd
res = pd.read_csv("/Users/sibyl/recommender/res.csv")
x=res['prop']
y_1=res['acc']
y_2=res['sen']
y_3=res['spec']
y_4=res['pp']
y_5=res['np']
y_6=res['k']

plt.plot(x, y_1, 'r--', x, y_2, 'b--', x, y_3, 'g--', x, y_4, 'y--', x, y_5, 'm-
-', x, y_6, 'k--')
plt.title('Classificaiton Metrics vs. Proportion used to impute user preference'
)
red_patch = mpatches.Patch(color='red', label='Accuracy')
blue_patch = mpatches.Patch(color='blue', label='Sensitivity')
green_patch = mpatches.Patch(color='green', label='Specificity')
yellow_patch = mpatches.Patch(color='yellow', label='Positive Pred Val')
m_patch = mpatches.Patch(color='m', label='Negative Pred Val')
k_patch = mpatches.Patch(color='k', label='Kappa')
plt.legend(handles=[red_patch, blue_patch, green_patch, yellow_patch, m_patch, k
_patch], loc=2, bbox_to_anchor=(1.05, 1),)
plt.show()
```



2) RGB components and recipe rating

Here I explore the relationship between the RGB component and popularity of recipes scraped from Allrecipes.com. Rating scores are adjusted by overall ratings and number of reviews each recipe gets (see IMDb Bayesian estimator of movie ratings (http://www.imdb.com/help/show_leaf?votestopfaq)). Images are laoded from URL and processed to extract RGB components (average of all pixels). The RGB and ratings of a random sample of 200 recipes are displyed in the dotplot. The last plot displys the mean of RGB components across all recipes which have ratings <3, 3-4, and 4-5. Red component is the highest across the board, followed by green, and blue is the least represented in recipe images. There seems to be a slight trend where the highest rating recipes on average have even higher red component.

In [5]:

```
from PIL import Image
import pandas as pd
import requests
from io import BytesIO
from itertools import chain
import random
from random import randint
import numpy as np
recipes = pd.read_csv("/Users/sibyl/recommender/recipes_clean.csv")
```

In [6]:

```
def get_image(image):
    width, height = image.size
    pixel_values = list(image.getdata())
    if image.mode == 'RGB':
        channels = 3
    elif image.mode == 'L':
        channels = 1
    else:
        return None
    pixel_values = np.array(pixel_values).reshape((width, height, channels))
    return pixel_values

r_mean = []
g_mean = []
b_mean = []

rand_int=random.sample(list(range(1, 3516)), 200)
for path in recipes['photo_url'][rand_int]:
    response = requests.get(path)
    img = Image.open(BytesIO(response.content))
    arry = get_image(img)
    if img.mode == 'RGB':
        temp = list(chain.from_iterable(arry))
        r_mean.append(np.mean(temp, axis=0)[0])
        g_mean.append(np.mean(temp, axis=0)[1])
        b_mean.append(np.mean(temp, axis=0)[2])
    else:
        r_mean.append(0)
        g_mean.append(0)
        b_mean.append(0)

rgb_mean = pd.DataFrame({'red_mean': r_mean,
                         'green_mean': g_mean,
                         'blue_mean': b_mean})
```

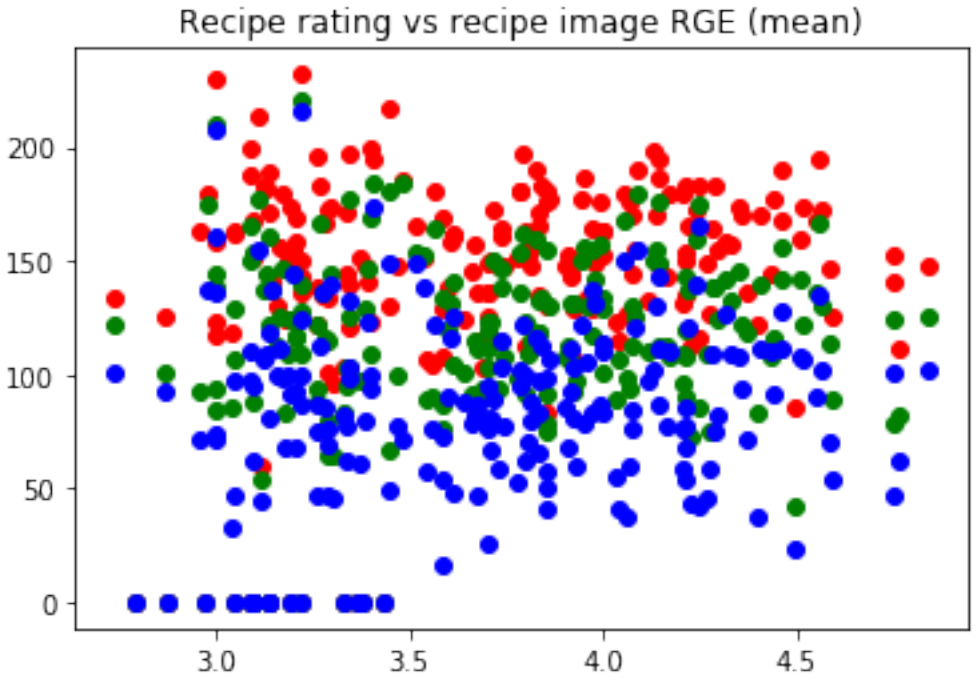
In [8]:

```
print(rgb_mean.head(5))
```

```
    blue_mean  green_mean    red_mean
0  109.353024  132.839968  157.716816
1   93.434779  109.305334  123.610068
2   99.247040  123.092752  162.712432
3   82.227789  115.058192  156.701451
4  102.190784  130.547136  172.473344
```

In [12]:

```
rating = recipes['rating_b'][rand_int]
#plt.plot(rating, rgb_mean['red_mean'], 'ro', rating, rgb_mean['green_mean'], 'g
o', rating, rgb_mean['blue_mean'], 'bo')
plt.plot(rating, rgb_mean['red_mean'], 'ro', rating, rgb_mean['green_mean'], 'go
', rating, rgb_mean['blue_mean'], 'bo')
plt.title('Recipe rating vs recipe image RGE (mean)')
plt.show()
```



In [9]:

```
colors = pd.concat([rating.reset_index(drop=True), rgb_mean], axis=1)
colors['rating_b']=round(colors['rating_b'])
colors_grp = colors.groupby(['rating_b']).mean()
print(colors_grp)
```

```
          blue_mean  green_mean    red_mean
rating_b
3.0       78.527736   99.968802  124.672779
4.0       88.268321  121.076838  152.260910
5.0       89.074737  119.025813  154.071114
```

In [13]:

```
plt.plot([3,4,5], colors_grp['red_mean'], 'r--', [3,4,5], colors_grp['green_mean
'], 'g--',
         [3,4,5], colors_grp['blue_mean'], 'b--')
plt.title('Recipe rating vs recipe image RGE (mean)')
plt.show()
```