Pymaceuticals Inc Analysis

Conclusions of treatments:

1. Among the four treatments for tumor volume (Capomulin, Infubinol, Ketapril, and Placebo), only Capomulin had a real positive effect decreasing it from 45.000000 mm^3 on day 0 to 36.236114 mm^3 on day 45. For Ketapril, Infubinol, and Placebo, we found they had similar results among them during the procedure, with Ketapril reaching out a bit higher volume on day 45, followed by Placebo and Infubinol. So, considering that Placebo is not a real medicine for any treatment, we could say that Ketapril and Infubinol did not have any positive effect on this aspect of treatment. Additionally, although Ketrapril ended up the treatment with the higher volume, the distance between Ketapril and Placebo is not enough to say that Ketrapril contributes to increase the tumor volume;
2. During the treatment of metastatic (cancer spreading) sites, Placebo led the numbers but was overcome by Ketapril nearby day 45;
3. Capomulin had the most positive effect among the four treatments for metastatic sites followed by Infubinol;
4. For survival rate, again, Capomulin obtained the most satisfactory result among those four treatments, leading it with 84%.

In [1]:
```
# Load Dependencies
%matplotlib inline
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

In [2]:
```
# Hide warning messages
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
# Set variables for Files
clinical_trial_file = "./datasource/clinicaltrial_data.csv"
mouse_drug_file = "./datasource/mouse_drug_data.csv"
```

In [4]:
```
# Read clinical_trial file
clinical_trial = pd.read_csv(clinical_trial_file)
clinical_trial.head()
```

Out[4]:

	Mouse ID	Timepoint	Tumor Volume (mm3)	Metastatic Sites
0	b128	0	45.0	0
1	f932	0	45.0	0
2	g107	0	45.0	0
3	a457	0	45.0	0
4	c819	0	45.0	0

In [5]:
```
# Read mouse_drug file
mouse_drug = pd.read_csv(mouse_drug_file)
mouse_drug.head()
```

Out[5]:

	Mouse ID	Drug
0	f234	Stelasyn
1	x402	Stelasyn
2	a492	Stelasyn
3	w540	Stelasyn
4	v764	Stelasyn

In [6]:
```
# Merge dfs
main_data = pd.merge(mouse_drug , clinical_trial, on="Mouse ID", how="outer")
main_data.info() # No blank in columns.
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1906 entries, 0 to 1905
Data columns (total 5 columns):
Mouse ID              1906 non-null object
Drug                  1906 non-null object
Timepoint             1906 non-null int64
Tumor Volume (mm3)    1906 non-null float64
Metastatic Sites      1906 non-null int64
dtypes: float64(1), int64(2), object(2)
memory usage: 89.3+ KB
```

In [7]:
```
main_data.head(5)
```

Out[7]:

	Mouse ID	Drug	Timepoint	Tumor Volume (mm3)	Metastatic Sites
0	f234	Stelasyn	0	45.000000	0
1	f234	Stelasyn	5	47.313491	0
2	f234	Stelasyn	10	47.904324	0

3	f234	Stelasyn	15	48.735197	1
4	f234	Stelasyn	20	51.112713	2

Tumor Response to Treatment

In [8]:
```
# Store the Mean Tumor Volume Data Grouped by Drug and Timepoint
tumor_vol_mean = main_data.groupby(["Drug","Timepoint"],axis=0).agg({"Tumor Volume (mm3)":"mean"})
tumor_vol_mean.reset_index(inplace=True)
# Preview DataFrame
tumor_vol_mean
```

Out[8]:

	Drug	Timepoint	Tumor Volume (mm3)
0	Capomulin	0	45.000000
1	Capomulin	5	44.266086
2	Capomulin	10	43.084291
3	Capomulin	15	42.064317
4	Capomulin	20	40.716325
5	Capomulin	25	39.939528
6	Capomulin	30	38.769339
7	Capomulin	35	37.816839
8	Capomulin	40	36.958001
9	Capomulin	45	36.236114
10	Ceftamin	0	45.000000
11	Ceftamin	5	46.503051
12	Ceftamin	10	48.285125
13	Ceftamin	15	50.094055
14	Ceftamin	20	52.157049
15	Ceftamin	25	54.287674
16	Ceftamin	30	56.769517
17	Ceftamin	35	58.827548
18	Ceftamin	40	61.467895
19	Ceftamin	45	64.132421
20	Infubinol	0	45.000000
21	Infubinol	5	47.062001
22	Infubinol	10	49.403909
23	Infubinol	15	51.296397
24	Infubinol	20	53.197691
25	Infubinol	25	55.715252
26	Infubinol	30	58.299397
27	Infubinol	35	60.742461
28	Infubinol	40	63.162824
29	Infubinol	45	65.755562
...	...	...	...
70	Ramicane	0	45.000000
71	Ramicane	5	43.944859
72	Ramicane	10	42.531957
73	Ramicane	15	41.495061
74	Ramicane	20	40.238325
75	Ramicane	25	38.974300
76	Ramicane	30	38.703137
77	Ramicane	35	37.451996
78	Ramicane	40	36.574081
79	Ramicane	45	34.955595
80	Stelasyn	0	45.000000
81	Stelasyn	5	47.527452
82	Stelasyn	10	49.463844
83	Stelasyn	15	51.529409
84	Stelasyn	20	54.067395
85	Stelasyn	25	56.166123
86	Stelasyn	30	59.826738
87	Stelasyn	35	62.440699
88	Stelasyn	40	65.356386
89	Stelasyn	45	68.438310
90	Zoniferol	0	45.000000
91	Zoniferol	5	46.851818
92	Zoniferol	10	48.689881
93	Zoniferol	15	50.779059
94	Zoniferol	20	53.170334
95	Zoniferol	25	55.432935
96	Zoniferol	30	57.713531
97	Zoniferol	35	60.089372

98	Zoniferol	40	62.916692
99	Zoniferol	45	65.960888

100 rows × 3 columns

In [9]:
```
# Store the Standard Error of Tumor Volumes Grouped by Drug and Timepoint
tumor_vol_sem = main_data.groupby(["Drug","Timepoint"],axis=0).agg({"Tumor Volume (mm3)":"sem"})
tumor_vol_sem.reset_index(inplace=True)
# Preview DataFrame
tumor_vol_sem.head()
```

Out[9]:

	Drug	Timepoint	Tumor Volume (mm3)
0	Capomulin	0	0.000000
1	Capomulin	5	0.448593
2	Capomulin	10	0.702684
3	Capomulin	15	0.838617
4	Capomulin	20	0.909731

In [10]:
```
# Minor Data Munging to Re-Format the Data Frames (sem)
tumor_vol_grp_sem = tumor_vol_sem.pivot_table(index="Timepoint",columns="Drug", values='Tumor Volume (mm3)')
# Preview that Reformatting worked
tumor_vol_grp_sem
```

Out[10]:

Drug	Capomulin	Ceftamin	Infubinol	Ketapril	Naftisol	Placebo	Propriva	Ramicane	Stelasyn	Zoniferol
Timepoint										
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
5	0.448593	0.164505	0.235102	0.264819	0.202385	0.218091	0.231708	0.482955	0.239862	0.188950
10	0.702684	0.236144	0.282346	0.357421	0.319415	0.402064	0.376195	0.720225	0.433678	0.263949
15	0.838617	0.332053	0.357705	0.580268	0.444378	0.614461	0.466109	0.770432	0.493261	0.370544
20	0.909731	0.359482	0.476210	0.726484	0.595260	0.839609	0.555181	0.786199	0.621889	0.533182
25	0.881642	0.439356	0.550315	0.755413	0.813706	1.034872	0.577401	0.746991	0.741922	0.602513
30	0.934460	0.490620	0.631061	0.934121	0.975496	1.218231	0.746045	0.864906	0.899548	0.800043
35	1.052241	0.692248	0.984155	1.127867	1.013769	1.287481	1.084929	0.967433	1.003186	0.881426
40	1.223608	0.708505	1.055220	1.158449	1.118567	1.370634	1.564779	1.128445	1.410435	0.998515
45	1.223977	0.902358	1.144427	1.453186	1.416363	1.351726	1.888586	1.226805	1.576556	1.003576

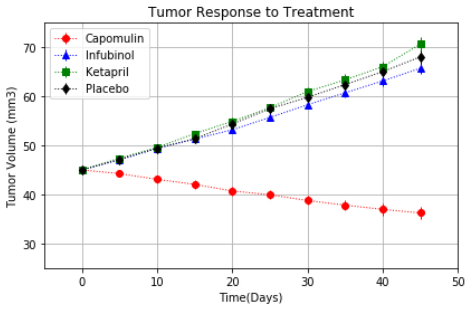
In [11]:
```
# Minor Data Munging to Re-Format the Data Frames (mean)
tumor_vol_grp_mean = tumor_vol_mean.pivot_table(index=["Timepoint"], columns='Drug', values='Tumor Volume (mm3)')
# Preview that Reformatting worked
tumor_vol_grp_mean
```

Out[11]:

Drug	Capomulin	Ceftamin	Infubinol	Ketapril	Naftisol	Placebo	Propriva	Ramicane	Stelasyn	Zoniferol
Timepoint										
0	45.000000	45.000000	45.000000	45.000000	45.000000	45.000000	45.000000	45.000000	45.000000	45.000000
5	44.266086	46.503051	47.062001	47.389175	46.796098	47.125589	47.248967	43.944859	47.527452	46.851818
10	43.084291	48.285125	49.403909	49.582269	48.694210	49.423329	49.101541	42.531957	49.463844	48.689881
15	42.064317	50.094055	51.296397	52.399974	50.933018	51.359742	51.067318	41.495061	51.529409	50.779059
20	40.716325	52.157049	53.197691	54.920935	53.644087	54.364417	53.346737	40.238325	54.067395	53.170334
25	39.939528	54.287674	55.715252	57.678982	56.731968	57.482574	55.504138	38.974300	56.166123	55.432935
30	38.769339	56.769517	58.299397	60.994507	59.559509	59.809063	58.196374	38.703137	59.826738	57.713531
35	37.816839	58.827548	60.742461	63.371686	62.685087	62.420615	60.350199	37.451996	62.440699	60.089372
40	36.958001	61.467895	63.162824	66.068580	65.600754	65.052675	63.045537	36.574081	65.356386	62.916692
45	36.236114	64.132421	65.755562	70.662958	69.265506	68.084082	66.258529	34.955595	68.438310	65.960888

In [12]:
```
# Generate the Plot (with Error Bars)
# Get errors
Capomulin_sem = tumor_vol_grp_sem['Capomulin']
Infubinol_sem = tumor_vol_grp_sem['Infubinol']
Ketapril_sem = tumor_vol_grp_sem['Ketapril']
Placebo_sem = tumor_vol_grp_sem['Placebo']

#Get the mean
Capomulin_mean = tumor_vol_grp_mean['Capomulin']
Infubinol_mean = tumor_vol_grp_mean['Infubinol']
Ketapril_mean = tumor_vol_grp_mean['Ketapril']
Placebo_mean = tumor_vol_grp_mean['Placebo']

# Get the x axis
x_axis = tumor_vol_grp_mean.index

# Plot our lines
plt.errorbar(x_axis, Capomulin_mean, yerr= Capomulin_sem, color="red", marker='o', \
             linestyle=":",linewidth=1, label="Capomulin")
plt.errorbar(x_axis, Infubinol_mean, yerr= Infubinol_sem, color="blue", marker='^', \
             linestyle=":",linewidth=1, label="Infubinol")
plt.errorbar(x_axis, Ketapril_mean,  yerr= Ketapril_sem, color="green", marker='s', \
             linestyle=":",linewidth=1, label="Ketapril")
plt.errorbar(x_axis, Placebo_mean,   yerr= Placebo_sem, color="black", marker='d', \
             linestyle=":",linewidth=1, label="Placebo")

# Place a legend on "best" location
plt.legend(loc="best")

# Set x and y limits
plt.xlim(-5, 50)
```

```
plt.ylim(25, 75)

# Set Title and Labels
plt.title("Tumor Response to Treatment")
plt.xlabel("Time(Days)")
plt.ylabel("Tumor Volume (mm3)")
plt.grid()

# Save the Figure
plt.savefig("tumor_response_to_treatment.png")

# Show the Figure
plt.tight_layout()
plt.show()
```



Metastatic Response to Treatment

In [13]:
```
# Store the Mean Met. Site Data Grouped by Drug and Timepoint
metastatic_mean = main_data.groupby(["Drug","Timepoint"],axis=0).agg({"Metastatic Sites":"mean"})
metastatic_mean.reset_index(inplace=True)

# Preview DataFrame
metastatic_mean.head()
```

Out[13]:

	Drug	Timepoint	Metastatic Sites
0	Capomulin	0	0.000000
1	Capomulin	5	0.160000
2	Capomulin	10	0.320000
3	Capomulin	15	0.375000
4	Capomulin	20	0.652174

In [14]:
```
# Minor Data Munging to Re-Format the Data Frames
metastatic_grp_mean = metastatic_mean.pivot_table(index=["Timepoint"], columns='Drug', values='Metastatic Sites')
# Preview that Reformatting worked
metastatic_grp_mean.head()
```

Out[14]:

Drug	Capomulin	Ceftamin	Infubinol	Ketapril	Naftisol	Placebo	Propriva	Ramicane	Stelasyn	Zoniferol
Timepoint										
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
5	0.160000	0.380952	0.280000	0.304348	0.260870	0.375000	0.320000	0.120000	0.240000	0.166667
10	0.320000	0.600000	0.666667	0.590909	0.523810	0.833333	0.565217	0.250000	0.478261	0.500000
15	0.375000	0.789474	0.904762	0.842105	0.857143	1.250000	0.764706	0.333333	0.782609	0.809524
20	0.652174	1.111111	1.050000	1.210526	1.150000	1.526316	1.000000	0.347826	0.952381	1.294118

In [15]:
```
# Store the Standard Error associated with Met. Sites Grouped by Drug and Timepoint
metastatic_sem = main_data.groupby(["Drug","Timepoint"],axis=0).agg({"Metastatic Sites":"sem"})
metastatic_sem.reset_index(inplace=True)
# Preview DataFrame
metastatic_sem.head()
```

Out[15]:

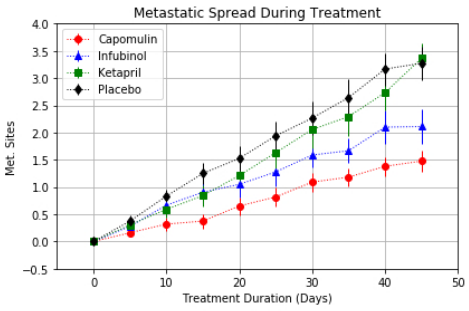
	Drug	Timepoint	Metastatic Sites
0	Capomulin	0	0.000000
1	Capomulin	5	0.074833
2	Capomulin	10	0.125433
3	Capomulin	15	0.132048
4	Capomulin	20	0.161621

In [16]:
```
# Minor Data Munging to Re-Format the Data Frames
metastatic_grp_sem = metastatic_sem.pivot_table(index='Timepoint', columns='Drug',values='Metastatic Sites')
# Preview that Reformatting worked
metastatic_grp_sem.head()
```

Out[16]:

Drug	Capomulin	Ceftamin	Infubinol	Ketapril	Naftisol	Placebo	Propriva	Ramicane	Stelasyn	Zoniferol
Timepoint										
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
5	0.074833	0.108588	0.091652	0.098100	0.093618	0.100947	0.095219	0.066332	0.087178	0.077709
10	0.125433	0.152177	0.159364	0.142018	0.163577	0.115261	0.105690	0.090289	0.123672	0.109109
15	0.132048	0.180625	0.194015	0.191381	0.158651	0.190221	0.136377	0.115261	0.153439	0.111677
20	0.161621	0.241034	0.234801	0.236680	0.181731	0.234064	0.171499	0.119430	0.200905	0.166378

In [17]:
```
# Generate the Plot (with Error Bars)

# Get the errors
Capomulin_sem = metastatic_grp_sem['Capomulin']
Infubinol_sem = metastatic_grp_sem['Infubinol']
Ketapril_sem = metastatic_grp_sem['Ketapril']
Placebo_sem = metastatic_grp_sem['Placebo']

# Get the means
Capomulin_mean = metastatic_grp_mean['Capomulin']
Infubinol_mean = metastatic_grp_mean['Infubinol']
Ketapril_mean = metastatic_grp_mean['Ketapril']
Placebo_mean = metastatic_grp_mean['Placebo']

# Get the x axis
x_axis = metastatic_grp_mean.index

# Plot our lines
plt.errorbar(x_axis, Capomulin_mean, yerr= Capomulin_sem, color="red", marker='o', linestyle=":", \
             linewidth=1, label="Capomulin")
plt.errorbar(x_axis, Infubinol_mean, yerr= Infubinol_sem, color="blue", marker='^', linestyle=":", \
             linewidth=1, label="Infubinol")
plt.errorbar(x_axis, Ketapril_mean, yerr= Ketapril_sem, color="green", marker='s', linestyle=":", \
             linewidth=1, label="Ketapril")
plt.errorbar(x_axis, Placebo_mean, yerr= Placebo_sem,  color="black", marker='d', linestyle=":", \
             linewidth=1, label="Placebo")

# Place a legend on the "best" location
plt.legend(loc="best")

# Set x and y limits
plt.xlim(-5, 50)
plt.ylim(-0.5, 4.0)

# Set Title and labels
plt.title("Metastatic Spread During Treatment")
plt.xlabel("Treatment Duration (Days)")
plt.ylabel("Met. Sites")
plt.grid()

# Save the Figure
plt.savefig("metastatic_spread_during_treatment.png")

# Show the Figure
plt.tight_layout()
plt.show()
```



Survival Rates

In [18]:
```
# Store the Count of Mice Grouped by Drug and Timepoint (W can pass any metric)
# Convert to DataFrame
# Preview DataFrame

mice_count = main_data.groupby(["Drug","Timepoint"],axis=0).agg({"Mouse ID":"count"})
mice_count.reset_index(inplace=True)
mice_count.rename(columns={"Mouse ID":"Mouse Count"}, inplace=True)
mice_count.head(10)
```

Out[18]:

	Drug	Timepoint	Mouse Count
0	Capomulin	0	25
1	Capomulin	5	25
2	Capomulin	10	25
3	Capomulin	15	24
4	Capomulin	20	23
5	Capomulin	25	22
6	Capomulin	30	22
7	Capomulin	35	22
8	Capomulin	40	21
9	Capomulin	45	21

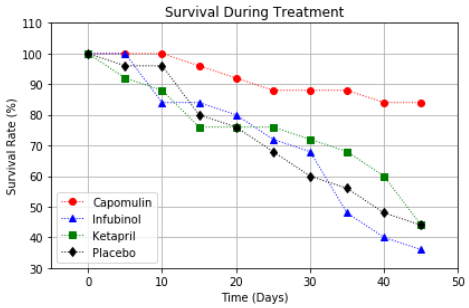
In [19]:
```
# Minor Data Munging to Re-Format the Data Frames
# Preview the Data Frame
mice_count_grp = mice_count.pivot_table(index=["Timepoint"], columns='Drug', values='Mouse Count')
mice_count_grp
```

Out[19]:

Drug	Capomulin	Ceftamin	Infubinol	Ketapril	Naftisol	Placebo	Propriva	Ramicane	Stelasyn	Zoniferol
Timepoint										

0	25	25	25	25	25	25	26	25	26	25
5	25	21	25	23	23	24	25	25	25	24
10	25	20	21	22	21	24	23	24	23	22
15	24	19	21	19	21	20	17	24	23	21
20	23	18	20	19	20	19	17	23	21	17
25	22	18	18	19	18	17	14	23	19	16
30	22	16	17	18	15	15	13	23	18	15
35	22	14	12	17	15	14	10	21	16	14
40	21	14	10	15	15	12	9	20	12	14
45	21	13	9	11	13	11	7	20	11	14

In [20]:
```
# Generate the Plot (Accounting for percentages)

# Compute the percentages
Capomulin = [ x*100/25 for x in  mice_count_grp['Capomulin'] ]
Infubinol = [ x*100/25 for x in  mice_count_grp['Infubinol'] ]
Ketapril = [ x*100/25 for x in  mice_count_grp['Ketapril'] ]
Placebo = [ x*100/25 for x in  mice_count_grp['Placebo'] ]

# Get the x axis
x_axis = mice_count_grp.index

# Plot lines
plt.plot(x_axis, Capomulin, color="red", marker='o', linestyle=":",linewidth=1, label="Capomulin")
plt.plot(x_axis, Infubinol, color="blue", marker='^', linestyle=":",linewidth=1, label="Infubinol")
plt.plot(x_axis, Ketapril, color="green", marker='s', linestyle=":",linewidth=1,label="Ketapril")
plt.plot(x_axis, Placebo, color="black", marker='d', linestyle=":",linewidth=1, label="Placebo")

# Get "best" position for Legend
plt.legend(loc="best")

# Set x and y limits
plt.ylim(30, 110)
plt.xlim(-5, 50)

# Set Title and labels
plt.title("Survival During Treatment")
plt.xlabel("Time (Days)")
plt.ylabel("Survival Rate (%)")
plt.grid()

# Save the Figure
plt.savefig("survival_during_treatment.png")

# Show the Figure
plt.tight_layout()
plt.show()
```



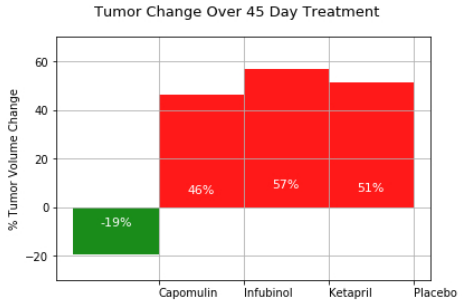
Summary Bar Graph

In [21]:
```
# Calculate the percent changes for each drug (Tumor Volume (mm3)) and store into a Tuple
drugs  = ['Capomulin','Ceftamin','Infubinol','Ketapril','Naftisol','Placebo','Propriva','Ramicane','Stelasyn','Zoniferol']
drugs_vol_pct = [  (x, ((tumor_vol_grp_mean.loc[45,x] * 100) / tumor_vol_grp_mean.loc[0,x] ) - 100 )  for x in drugs]
drugs_vol_pct
```

Out[21]:
```
[('Capomulin', -19.47530266789417),
 ('Ceftamin', 42.51649185589744),
 ('Infubinol', 46.123471727851864),
 ('Ketapril', 57.02879468660606),
 ('Naftisol', 53.923347134769244),
 ('Placebo', 51.29796048315151),
 ('Propriva', 47.24117486320637),
 ('Ramicane', -22.320900462766673),
 ('Stelasyn', 52.085134287899024),
 ('Zoniferol', 46.579750865095264)]
```

In [22]:
```
# Classify Drugs between passing and failing drugs based on Tumor Volume decreasing.

# Set the list of Drugs to be presented on the Graph
drugs_sel = ['Capomulin','Infubinol','Ketapril', 'Placebo']
# List for passing drugs
passing_y = []
# List for failing drugs
failing_y = []

# Start the graph
fig, ax = plt.subplots()

# List x labels for passing and failing drugs
x_labels = [[],[]]
# Function to splice the data between passing and failing drugs and get the x labels
```

```
# Function to split the data between passing and failing drugs and get the X labels
def classify_drugs(drug):
    global count
    # Filter the Drugs to be presented.
    if (drug[0] in drugs_sel):
        if (float(drug[1]) < 0):
            # Get the percent of passing drug
            passing_y.append(drug[1])
            # Get the x label for passing drug.
            x_labels[0].append(drug[0])
        else:
            # Get the percent of failing drug
            failing_y.append(drug[1])
            # Get the x label for failing drug.
            x_labels[1].append(drug[0])

# Call function to set the x labels and classify drugs between passing and failing
for drug in drugs_vol_pct:
    classify_drugs(drug)

# Set x for passing drugs
x_axis_pass = [ y for y in range(0, len(passing_y))  ]
# Set x for failing drugs
x_axis_fail = [ y for y in range( len(passing_y) , len(failing_y)+1 )]
# Set x ticks
xticks = [ y for y in range( 0, len(passing_y) + len(failing_y) )]
ax.set_xticks(xticks)

# Set the x tick labels
ax.set_xticklabels(x_labels[0] + x_labels[1], horizontalalignment='left')
# Set y tick
ax.set_yticks([-20, 0, 20, 40, 60])

# Set Bar chart for passing drugs
pass_bar = ax.bar(x_axis_pass, passing_y, color='g', alpha=0.9, align="edge", width = -1)
# Set Bar chart for failing drugs
fail_bar = ax.bar(x_axis_fail, failing_y, color='r', alpha=0.9, align="edge", width = -1)

# Function to label the percentages
def pct_label(bars):
    for bar in bars:
        height = bar.get_height()
        ax.text(bar.get_x() + bar.get_width()/2, .2 * height, \
        "{:+0.0f}%".format(int(height)).replace('+',''), \
        fontsize=11, fontweight="light", ha="center", va="top", color="white")

# Call functions to set percentages
pct_label(pass_bar)
pct_label(fail_bar)
plt.grid()

# Sets the y limit
plt.ylim(-30, 70)

# Set Title
fig.suptitle("Tumor Change Over 45 Day Treatment", fontsize=13, fontweight="light")
# Set y Label
ax.set_ylabel("% Tumor Volume Change")

# Save the Figure
plt.savefig("change_over_45day_treatment.png")
# Show the Figure
fig.show()
```



In [ ]: