FlatIron Exercise

- Isabel Metzger

In [1]:
```
# importing packages
import pandas as pd
from statsmodels.distributions.empirical_distribution import ECDF
import warnings
warnings.filterwarnings('ignore')
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
import seaborn as sns
sns.set()
```

Question 1:
First, the clinic would like to know for which diseases they are seeing patients.
1a) Which types of cancer does the clinic see patients for?
1b) How many patients does the clinic see for each cancer type?

Question 2:
The clinic wants to know how long it takes for patients to start therapy after being diagnosed, which they consider to be helpful in understanding the quality of care for the patients.
2a) How long after being diagnosed do cancer patients start treatment for each cancer type?
2b) Are there any patients which are diagnosed but not treated at the practice?

Question 3:
After being treated with a first line of treatment (a drug or combination of drugs), what proportion of all cancer patients go on to be treated with a second line of treatment? (For more information on the concept of "first-line therapy", please reference https://www.cancer.net/navigating-cancer-care/how-cancer-treated/when-first-treatment-doesnt-work (https://www.cancer.net/navigating-cancer-care/how-cancer-treated/when-first-treatment-doesnt-work))

Question 4:
How does each drug used at the clinic compare in terms of its duration of therapy?

Load Data

In [2]:
```
# reading data provided

df_treatment = pd.read_csv('./TreatmentSample01.csv', )
print(df_treatment.shape)

df_diagnosis = pd.read_csv('./DiagnosisSample01.csv', )
print(df_diagnosis.shape)
```

```
(714, 3)
(44, 5)
```

In [3]:
```
# Let's view the format of each table
print('Treatment Table')
df_treatment.head()
```

```
Treatment Table
```

Out[3]:

	PatientID	TreatmentDate	DrugCode
0	2038	2010-01-24	A
1	2038	2010-01-27	A
2	2038	2010-01-30	A
3	2038	2010-02-02	A
4	2038	2010-02-06	A

In [4]:
```
print('Diagnosis Table')
df_diagnosis.head()
```

```
Diagnosis Table
```

Out[4]:

	PatientID	DiagnosisDate	DiagnosisCode	Diagnosis	IsCancerDiagnosis
0	2634	2011-02-19	285.8	Anemia	False
1	5657	2012-06-07	285.8	Anemia	False
2	7937	2013-01-06	285.8	Anemia	False
3	8615	2013-07-18	284.9	Anemia	False
4	4354	2012-02-04	284.9	Anemia	False

Question 1

To answer the first question, we count how many times a given diagnosis shows up in the Diagnosis table

In [5]:
```
#Let's first check whether the Diagnosis codes are normalized,
#and check for cases where there is missing information that could impac
t our analysis

# normally, would use a lib like pandas-profiling to run a quick analysi
s on the dataframe,
# but below I just do some manual checks

df_diagnosis.describe(include='all')
# we can confirm by looking at the 'count' row that there are no Nonetyp
e values in the dataframe
```

Out[5]:

	PatientID	DiagnosisDate	DiagnosisCode	Diagnosis	IsCancerDiagnosis
count	44.000000	44	44.000000	44	44
unique	NaN	38	NaN	5	2
top	NaN	2012-03-20	NaN	Breast Cancer	True
freq	NaN	3	NaN	22	33
mean	5228.181818	NaN	207.618182	NaN	NaN
std	2095.923976	NaN	76.155521	NaN	NaN
min	2038.000000	NaN	153.300000	NaN	NaN
25%	3449.000000	NaN	169.050000	NaN	NaN
50%	4533.000000	NaN	174.650000	NaN	NaN
75%	6922.000000	NaN	202.400000	NaN	NaN
max	9331.000000	NaN	401.900000	NaN	NaN

In [6]:
```
# We want to groupby on Diagnosis, and we confirmed above that each row
 has a value for Diagnosis
# now let's check that the Diagnosis terms are normalized (e.g. no obvio
us mispellings)


print('Listed Diagnoses:', ''.join([f'\n* {diag_name}' for diag_name in
df_diagnosis.Diagnosis.unique()]))
```

```
Listed Diagnoses:
* Anemia
* Breast Cancer
* Colon Cancer
* Hypertension
* Hypertension
```

In [7]:
```
# we see that there are two instances of Hypertension, and one clearly h
as a trailing space
# we'll strip the Diagnosis strings to fix this
df_diagnosis['Diagnosis'] = df_diagnosis.Diagnosis.str.strip()

# let's confirm that we no longer see the duplicate diagnosis string
print('Listed Diagnoses (cleaned):', ''.join([f'\n* {diag_name}' for dia
g_name in df_diagnosis.Diagnosis.unique()]))
```

```
Listed Diagnoses (cleaned):
* Anemia
* Breast Cancer
* Colon Cancer
* Hypertension
```

In [8]:
```
# now let's look at the counts for each Diagnosis
# we are concerned with just the cancer diagnoses,
# so we'll filter using the provided IsCancerDiagnosis flag
cancer_diagnosis_counts = (df_diagnosis
                           [df_diagnosis.IsCancerDiagnosis == True]
                           .groupby('Diagnosis')
                           .PatientID
                           .count()
                           .rename('Counts')
                           .to_frame()
                          )
cancer_diagnosis_counts
```

Out[8]:

Diagnosis	Counts
Breast Cancer	22
Colon Cancer	11

Q1 Answers:

1a) The types of cancers the clinic sees are : *Breast Cancer* and *Colon Cancer*

1b) The number fo patients for each cancer type are :
Breast Cancer : 22
Colon Cancer : 11

Question 2:
In order to look at the time it takes to treat a patient after diagnosis, we'll need to combine the two tables and compare the Diagnosis date to the Treatment date

In [9]:
```
# We want to look at the time between diagnosis and treatment,
# so let's first filter out our diagnosis table for just cancer diagnose
s
# Also recall that only anti-cancer drugs are being listed in the treatm
ent table
# so the only diagnoses being treated here will be cancer diagnoses
df_cancer_diagnosis = df_diagnosis[df_diagnosis.IsCancerDiagnosis==True]

# let's look to see how many cases we have where someone is diagnosed mu
ltiple times
df_cancer_diagnosis[df_cancer_diagnosis.duplicated(subset=['PatientID'],
keep=False)].sort_values('PatientID')
```

Out[9]:

	PatientID	DiagnosisDate	DiagnosisCode	Diagnosis	IsCancerDiagnosis
30	3095	2011-07-01	153.9	Colon Cancer	True
31	3095	2011-07-10	153.3	Colon Cancer	True
32	3449	2011-08-26	153.5	Colon Cancer	True
38	3449	2011-08-26	153.4	Colon Cancer	True
13	3757	2011-10-11	174.1	Breast Cancer	True
39	3757	2011-10-08	153.5	Colon Cancer	True
17	4374	2012-03-20	174.5	Breast Cancer	True
25	4374	2012-03-20	174.8	Breast Cancer	True
26	4374	2012-03-20	174.7	Breast Cancer	True
27	6877	2012-12-09	174.3	Breast Cancer	True
34	6877	2012-11-16	153.4	Colon Cancer	True

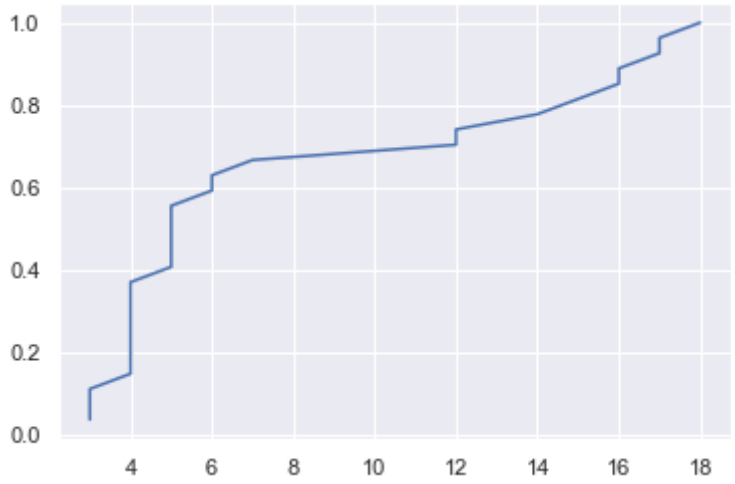
In [10]:
```
# let's also account for combination therapies, where a patient is given
multiple drugs together
# for each patient and each treatment day, we will combine rows where mu
ltiple drugs are listed
df_combo_treatment = df_treatment.groupby(['PatientID', 'TreatmentDate'
]).sum().reset_index()
```

In [11]:
```
# Unfortunately we don't have a way to tell which treatment corresponds
 to which diagnosis in the above cases
# For now let's assume the following:
#    - Each instance where a person has a change in treatment at some po
int is a case of second-line therapy
#    - For cases where a patient has multiple diagnoses, we consider all
diagnoses to be treated by all drugs used in treatment
# given this, we can effectively reduce the problem to finding the time
 between first diagnosis and first treatment

# for each patient get the first diagnosis date
df_first_cancer_diagnosis = (df_cancer_diagnosis
                             .sort_values(['PatientID', 'DiagnosisDate'
])
                             .groupby(['PatientID'])
                             .first()
                             .reset_index()
                             [['PatientID', 'DiagnosisDate', 'Diagnosis'
]]
                            )
# for each patient get the first date of any treatment
df_first_treatment = (df_combo_treatment
                      .sort_values(['PatientID', 'TreatmentDate'])
                      .groupby(['PatientID'])
                      .first()
                      .reset_index()
                      [['PatientID', 'TreatmentDate']]
)

df_diag_to_treat = df_first_cancer_diagnosis.merge(df_first_treatment, o
n='PatientID', how='outer')

# convert the date cols to datetime object, so we can more easily calcul
ate the difference
df_diag_to_treat['DiagnosisDate'] = pd.to_datetime(df_diag_to_treat.Diag
nosisDate, infer_datetime_format=True)
df_diag_to_treat['TreatmentDate'] = pd.to_datetime(df_diag_to_treat.Trea
tmentDate, infer_datetime_format=True)
df_diag_to_treat['DaysDiagToTreat'] = df_diag_to_treat.apply(lambda row:
(row['TreatmentDate'] - row['DiagnosisDate']).days, axis=1)

df_diag_to_treat.sort_values('DaysDiagToTreat')
```

Out[11]:

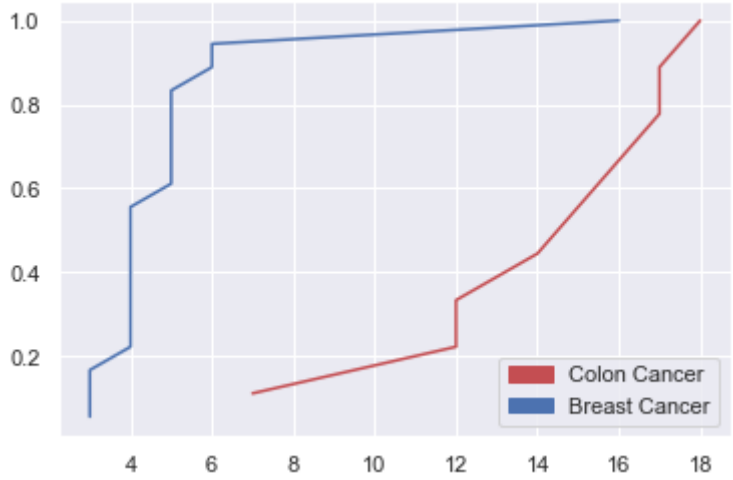
	PatientID	DiagnosisDate	Diagnosis	TreatmentDate	DaysDiagToTreat
0	2038	2010-01-21	Breast Cancer	2010-01-24	3
20	6889	2012-11-17	Breast Cancer	2012-11-20	3
14	4692	2012-04-27	Breast Cancer	2012-04-30	3
3	2425	2010-12-15	Breast Cancer	2010-12-19	4
4	2462	2011-01-07	Breast Cancer	2011-01-11	4
5	2763	2011-04-19	Breast Cancer	2011-04-23	4
17	6321	2012-09-06	Breast Cancer	2012-09-10	4
16	6281	2012-08-12	Breast Cancer	2012-08-16	4
15	5259	2012-05-13	Breast Cancer	2012-05-17	4
10	3948	2011-12-18	Breast Cancer	2011-12-22	4
24	7796	2013-01-16	Breast Cancer	2013-01-21	5
25	7976	2013-03-06	Breast Cancer	2013-03-11	5
13	4374	2012-03-20	Breast Cancer	2012-03-25	5
11	4256	2011-11-07	Breast Cancer	2011-11-12	5
12	4354	2012-02-04	Breast Cancer	2012-02-09	5
2	2407	2010-06-13	Breast Cancer	2010-06-19	6
26	9331	2013-08-23	Breast Cancer	2013-08-29	6
22	7230	2013-01-02	Colon Cancer	2013-01-09	7
7	3095	2011-07-01	Colon Cancer	2011-07-13	12
23	7242	2013-01-11	Colon Cancer	2013-01-23	12
9	3757	2011-10-08	Colon Cancer	2011-10-22	14
21	6922	2012-11-07	Colon Cancer	2012-11-22	15
6	2770	2011-04-06	Colon Cancer	2011-04-22	16
1	2120	2010-01-09	Breast Cancer	2010-01-25	16
18	6837	2012-10-08	Colon Cancer	2012-10-25	17
19	6877	2012-11-16	Colon Cancer	2012-12-03	17
8	3449	2011-08-26	Colon Cancer	2011-09-13	18

In [12]:
```
# let's look at the empirical CDF to see how the time between diagnosis
 and treatment is distributed
days_diag_to_treat = ECDF(df_diag_to_treat.DaysDiagToTreat)
sns.lineplot(days_diag_to_treat.x, days_diag_to_treat.y, estimator=None)
plt.show()
print(df_diag_to_treat.DaysDiagToTreat.describe())
# we notice there is a noticeable split in the times, where most treatme
nt occurs within 7 days, and the rest between 12-18 days
```




```
count    27.000000
mean      8.074074
std       5.363164
min       3.000000
25%       4.000000
50%       5.000000
75%      13.000000
max      18.000000
Name: DaysDiagToTreat, dtype: float64
```

In [13]:
```
# let's group this by type of cancer now. there is one particular case w
here a subject has diagnoses for both types of cancer,
# but we will assume the therapy is intially targeted for the first diag
nosis


colon_days_diag_to_treat = ECDF(df_diag_to_treat[df_diag_to_treat.Diagno
sis=='Colon Cancer'].DaysDiagToTreat)
breast_days_diag_to_treat = ECDF(df_diag_to_treat[df_diag_to_treat.Diagn
osis=='Breast Cancer'].DaysDiagToTreat)
sns.lineplot(colon_days_diag_to_treat.x, colon_days_diag_to_treat.y, est
imator=None, color='r')
sns.lineplot(breast_days_diag_to_treat.x, breast_days_diag_to_treat.y, e
stimator=None, color='b')
plt.legend(handles=[mpatches.Patch(color='r', label='Colon Cancer'), mpa
tches.Patch(color='b', label='Breast Cancer')])
plt.show()
# we've already included the first diagnosis in our table, so we can now
groupby the cancer type and recalc
df_diag_to_treat.groupby('Diagnosis').DaysDiagToTreat.describe()

# Ah-ha! we see that the two groupings in the ECDF above actually corres
pond to the cancer types,
# where Breast Cancer starts being treated soon after diagnosis, and Col
on Cancer typically starting treatment later
```



Out[13]:

Diagnosis	count	mean	std	min	25%	50%	75%	max
Breast Cancer	18.0	5.000000	2.890146	3.0	4.0	4.0	5.0	16.0
Colon Cancer	9.0	14.222222	3.456074	7.0	12.0	15.0	17.0	18.0

In [14]:
```
# in order to determine whether there are any patients who are diagnosed
but not treated,
# we can check if there are any patients in the diagnosis table with a c
ancer diagnosis, but does not show up in the treatment table
unique_cancer_diagnosis_patients = set(df_cancer_diagnosis.PatientID.uni
que())
# we can look at unique subjects in the treatment table since these are
 all; anti-cancer treatments
unique_treatment_patients = set(df_treatment.PatientID.unique())

patients_cancer_diagnosis_no_treatment = unique_cancer_diagnosis_patient
s - unique_treatment_patients
print('Count of patients diagnosed with cancer but not treated: {}'.form
at(len(patients_cancer_diagnosis_no_treatment)))
```

```
Count of patients diagnosed with cancer but not treated: 0
```

Q2 Answers:

2a) The time from diagnosis to treatment are:

Breast Cancer - between 3 to 16 days, with a mean of 5 and median of 4

Colon Cancer - between 7 and 18 days, with a mean of appx. 14.2 and median of 15

2b) No, there do not appear to be any patients that have been diagnosed with cancer but not treated

---

Question 3

We will determine how many patients go on to a second-line therapy

In [15]:
```
# to determine which patients go on to a second-line therapy we can how
 many unique therapies were given for each patient
# recall that we already identified combo treatment rows and grouped the
m together
df_therapy_counts = (df_combo_treatment
                     .drop_duplicates(subset=['PatientID', 'DrugCode'])
                     .groupby('PatientID')
                     .DrugCode
                     .count()
                     .rename('TherapyCounts')
                     .reset_index()
                    )
df_therapy_counts.sort_values('TherapyCounts', ascending=False)
```

Out[15]:

	PatientID	TherapyCounts
14	4692	2
23	7242	2
18	6837	2
10	3948	2
17	6321	2
16	6281	2
15	5259	2
0	2038	1
25	7976	1
24	7796	1
22	7230	1
21	6922	1
20	6889	1
19	6877	1
13	4374	1
1	2120	1
12	4354	1
11	4256	1
9	3757	1
8	3449	1
7	3095	1
6	2770	1
5	2763	1
4	2462	1
3	2425	1
2	2407	1
26	9331	1

In [16]:
```
# now we can take the proportion of patients with >1 counted therapies t
o the total count of patients
df_patients_second_line_therapy = df_therapy_counts[df_therapy_counts.Th
erapyCounts>1]
second_line_therapy_proportion = len(df_patients_second_line_therapy) /
len(df_therapy_counts)
print('Proportion of subjects who go on to 2nd Line Therapy: {} ({} of
{})'.format(round(second_line_therapy_proportion,3),

len(df_patients_second_line_therapy),

len(df_therapy_counts)

))
```

```
Proportion of subjects who go on to 2nd Line Therapy: 0.259 (7 of 27)
```

Q3 Answers:

The proportion of subjects who go on to 2nd Line Therapy is 0.259 (7 of 27)

Question 4

We will estimate the duration of each therapy

In [17]:
```
# to estimate the duration for each therapy we can take the avg time it
 is used during the treatment a patient
# the treatment dataframe includes a row for each time a therapy is give
n to a patient,
# so we can determine the duration by taking the difference between the
 first and last dates for a therapy for each patient

df_combo_treatment_duration = (df_combo_treatment
                               .sort_values(['PatientID', 'DrugCode', 'T
reatmentDate'])
                               .groupby(['PatientID', 'DrugCode'])
                               .TreatmentDate
                               .agg(['first','last'])
                               .reset_index()
                              )
# convert the first and last columns into datetimes to make it easier to
take the difference
df_combo_treatment_duration['first'] = pd.to_datetime(df_combo_treatment
_duration['first'], infer_datetime_format=True)
df_combo_treatment_duration['last'] = pd.to_datetime(df_combo_treatment_
duration['last'], infer_datetime_format=True)
df_combo_treatment_duration['TreatmentDuration'] = df_combo_treatment_du
ration.apply(lambda row: (row['last'] - row['first']).days, axis=1)
df_combo_treatment_duration.head()
```

Out[17]:

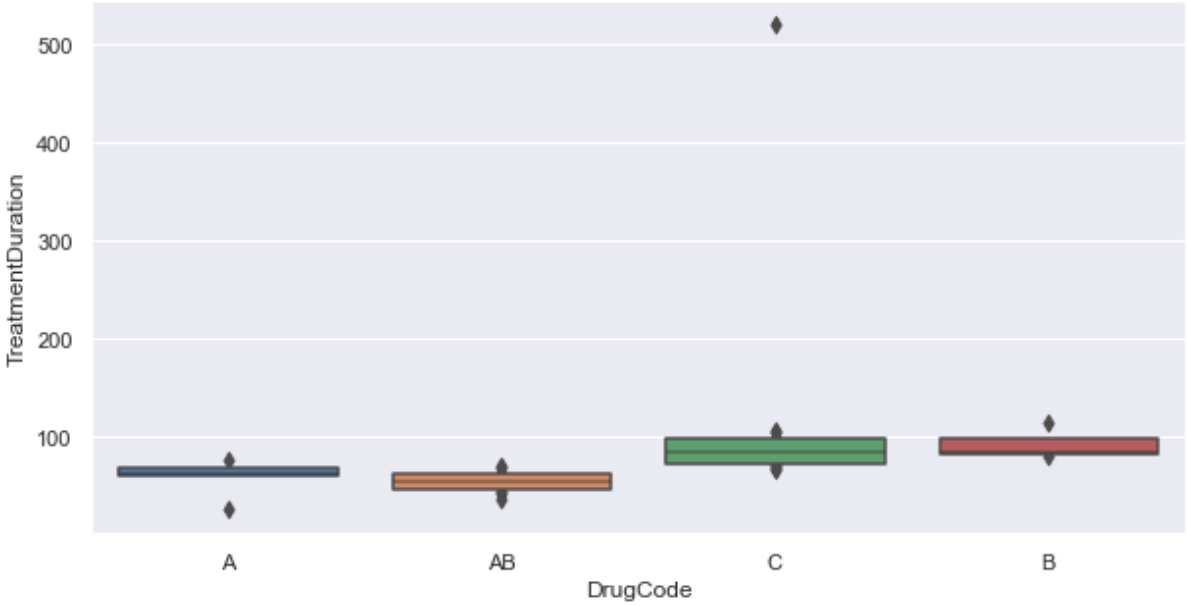
	PatientID	DrugCode	first	last	TreatmentDuration
0	2038	A	2010-01-24	2010-02-20	27
1	2120	AB	2010-01-25	2010-03-02	36
2	2407	AB	2010-06-19	2010-08-03	45
3	2425	AB	2010-12-19	2011-02-08	51
4	2462	AB	2011-01-11	2011-03-04	52

In [18]:
```
# now we can group by each therapy type and get some stats on their dura
tions
df_combo_treatment_duration.groupby('DrugCode').TreatmentDuration.descri
be()
```

Out[18]:

	count	mean	std	min	25%	50%	75%	max
DrugCode								
A	5.0	59.2000	19.110207	27.0	61.00	62.0	69.00	77.0
AB	10.0	54.6000	11.413442	36.0	46.50	55.0	62.25	70.0
B	3.0	93.0000	18.357560	80.0	82.50	85.0	99.50	114.0
C	16.0	112.0625	109.574005	66.0	73.75	85.5	98.25	520.0

In [19]:
```
plt.figure(figsize=(10,5))
sns.boxenplot(data=df_combo_treatment_duration, x='DrugCode', y='Treatme
ntDuration')
plt.show()
```

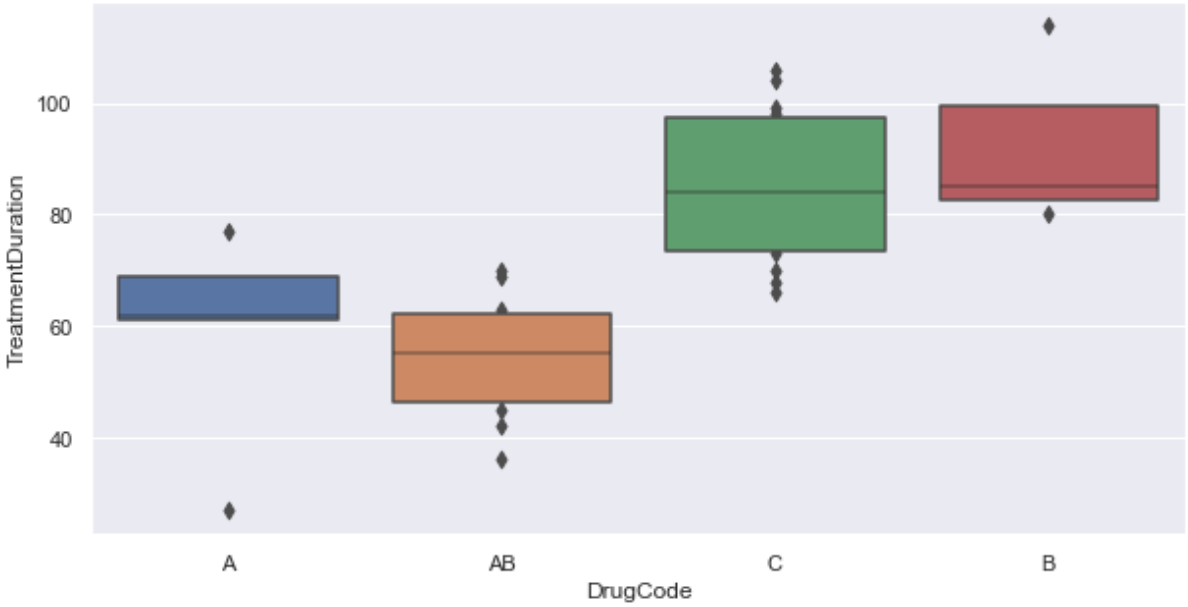


In [20]:
```
# hmm, clearly there is a large outlier for Therapy "C"
# if we look at the therapies sorted by duration we see a huge jump betw
een the longest and the second-longest
# 520 days, then 114 days
df_combo_treatment_duration.sort_values('TreatmentDuration', ascending=F
alse).head(10)
```

Out[20]:

	PatientID	DrugCode	first	last	TreatmentDuration
30	7242	C	2013-05-23	2014-10-25	520
29	7242	B	2013-01-23	2013-05-17	114
24	6837	C	2013-01-20	2013-05-06	106
22	6321	C	2012-11-24	2013-03-08	104
20	6281	C	2012-10-29	2013-02-05	99
18	5259	C	2012-07-24	2012-10-30	98
16	4692	C	2012-06-16	2012-09-21	97
33	9331	C	2013-08-29	2013-11-27	90
9	3757	C	2011-10-22	2012-01-17	87
11	3948	B	2012-03-03	2012-05-27	85

In [21]:
```
# for the sake of making it easier to compare the therapy durations in o
ur boxen plot, we'll plot a new figure but with this outlier removed
plt.figure(figsize=(10,5))
sns.boxenplot(data=df_combo_treatment_duration.sort_values('TreatmentDur
ation').iloc[:-1], x='DrugCode', y='TreatmentDuration')
plt.show()
# much easier to see the relative ranges for each therapy
```



Q3 Answers:

The duration of treatment for each therapy is:
A - range of 27-77 days, with a mean of appx 59.2 and a median of 62
B - range of 80-114 days, with a mean of 93 and median of 85
AB - range of 36-70 days, with a mean of appx. 54.6 and median of 55
C - range of 66-520 days, with a mean of appx. 112 and median of 85.5
(note: for therapy C, there is a large single outlier with a duration of 520 days. Ignoring this, the range would be 66-106 days)