Gino Parages

Libraries & Packages

Data Exploration & Prep — pandas, NumPy

Machine Learning Predictions — scikit learn

Data Visualizations — Seaborn, matplotlib

Describe how you would compute the length of treatment for each patient.

The length of treatment is obtained by aggregating by each Patient_ID then take the resulting data structure and subtract the min date (start of treatment) from the max date (current treatment date).

The data must be prepared however as the date column is actually a string (object) type and we cannot do operations on these. Below are methods of processing this data as we need.

```
# converts the string object type of the Drug_admin_date column and converts it into
# a datetime object with a placeholder value for the year of 1900 across the dataset

admin_date['Date'] =  pd.to_datetime(admin_date['Drug_admin_date'], format='%d-%b')

# Calculating Treatment Length
"""
The data is grouped by the Patient_Id and then the max and min are subtracted to leave
a series object which we have to reset the index on and then do a merge with the
original dataframe using the Patient_ID column as the 'primary key'
"""
calc_dates = (admin_date.groupby('Patient_ID')['Date'].max() -
admin_date.groupby('Patient_ID')['Date'].min()).reset_index()

# merge this new group by on the index to the main dataframe

admin_date = admin_date.merge(calc_dates, left_on='Patient_ID',right_on='Patient_ID',
suffixes=('','_Goupby'))
```

Describe or show how you might go about comparing the length of treatment by drug 390 vs. the generic.

Figure A

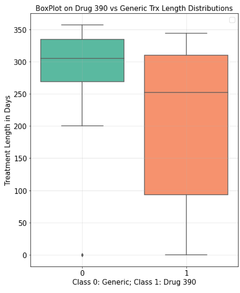


Figure A Upon first glance you notice there is an outlier in the 0 label, 'not on drug 390', for column drug_390_admin_flag. Looking at the boxplot we see we have zeros as well in marker 1, 'on drug 390', and these are affecting the data. These must also be corrected for.

When running a minimum statistic for our data it shows 0 as a value which is actually an outlier and can be dropped if the dataset was larger. We could say this patient never received treatment and is not valid in the sample. However, since samples are limited on our dataset we can fill this specific instance with the median. The median is used as there is quite a variance in our values.

Simulations were run which ranged from 10K samples to 1M samples based on our sample means and standard deviations for both markers.

Using t-scores due to low number of samples we can be 95% confident our sample means are representative of the population means for each drug marker group, on 390 or not. To verify three simulations were run using our specific sample means and standard deviations. Each subsequent simulation was also confidence tested and provided confirmation the mean was representative of the true population with 95% confidence.

From this we can continue working with our values for the 'Trx_Length' columns as they are close representations of the true values observed if given to larger groups.

Figure B

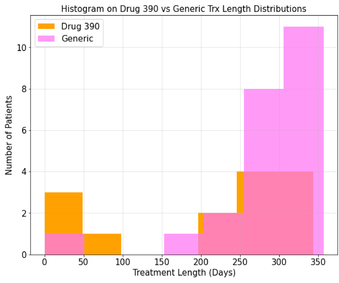


Figure B The histogram shows us that the data is not normally distrubuted for both drug datasets.
This shows an assymetric pattern in the data which would make it difficult to calculate confidence intervals and see how representative these numbers are of an actual population's results with our two drugs.
From the sample mean and standard deviation we collect we can create random sampling simulations with different sample sizes to see how accurate our estimation can be.

Drug 390
Sample Mean: 242.142857142857142
Sample Std: 95.35751121466492
Sample Size: 14
Degrees of Freedom: 13
95% Confidence Less Than 30 Samples Using t-score:
(187.09444649737307, 297.1912495883412)

Simulation of 10000 mean: 241.9391750848481
95% Confidence Interval of (240.06379042714826, 243.81455974254794)
Sample Size Simulation of 100000 mean: 242.36334485033526
95% Confidence Interval of (241.77229878705393, 242.95439091365128)
Sample Size Simulation of 1000000 mean: 241.97902447586867
95% Confidence Interval of (241.79197009690426, 242.16607885477308)

Generic Drug
Sample Mean: 300.3478260869565
Sample Std: 43.98409701828417
Sample Size: 23
Degrees of Freedom: 22
95% Confidence Less Than 30 Samples Using t-score:
(281.21645628504905, 319.47919588886396)

Simulation of 10000 mean: 300.2538767672502
95% Confidence Interval of (299.3884868270467, 301.1189067117957)
Sample Size Simulation of 100000 mean: 300.4495270709093
95% Confidence Interval of (300.1769043161912, 300.7221498256274)
Sample Size Simulation of 1000000 size mean: 300.2722575048093
95% Confidence Interval of (300.1859777996804, 300.3585372099382)

Figure C

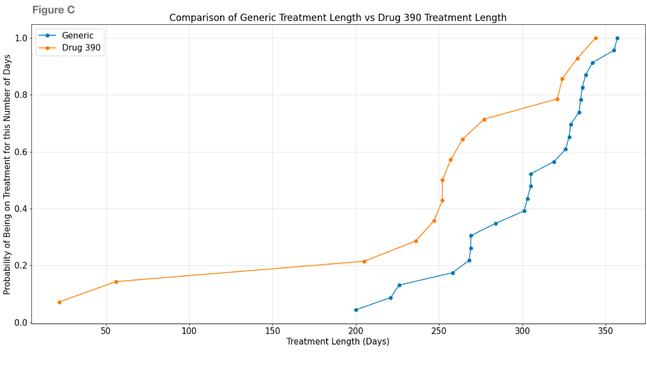


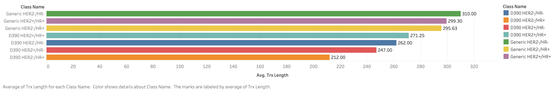
Figure C The data shows that a patient **on the generic,** has a higher probability, based on the ECDF scale, to be on treatment for longer than if they were on our **'Drug 390'**. Using this chart, one could estimate a new patient with similar health conditions as these used in the sample, would demonstrate similar behaviors. This can be backed up by our confidence interval calculations further up in the document.

Due to the length of the code required to generate these charts, simulations, and further analysis please review the associate Jupyter Notebook file.

```
def stats_comparison_95_confident(dataframe):
    # p_values for t test with a 95% confidence interval
    ninetyfive_confidence_pvals = {1:12.71, 2:4.303, 3:3.182, 4:2.776, 5:2.571,
6:2.447, 7:2.365, 8:2.306, 9:2.262, 10:2.228, 11:2.201, 12:2.179, 13:2.16, 14:2.145,
15:2.131,
                                   16:2.12, 17:2.11, 18:2.101, 19:2.093, 20:2.086,
21:2.086, 22:2.086, 23:2.086, 24:2.086, 25:2.086, 26:2.086, 27:2.086, 28:2.086,
29:2.086}
```

Describe or show how you might go about comparing the length of treatment by drug 390 vs. the generic.

Figure D



Average of Trx Length for each Class Name. Color shows details about Class Name. The marks are labeled by average of Trx Length.

Figure D This is an excerpt from a Tableau Dashboard created to visualize by the different patient cancer categories and the subsequent Drug Assignment (Generic vs Drug 390). The data is significantly showing Drug 390 offering shorter time periods for treatment, some up to one month or more in difference.

Figure E

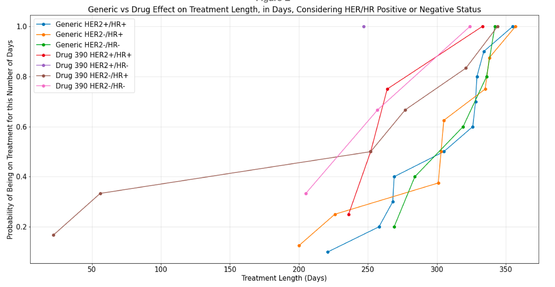


Figure E The graph shows a plotted probability distribution for all possible categories. There is one category missing due to no patients having this tumor status, Generic HER2+/HR-.

Figure F

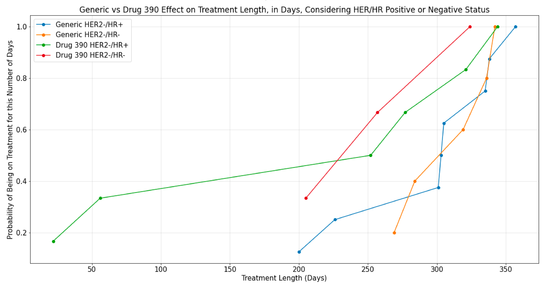


Figure F When comparing on just HER2-/HR+ and HER2-/HR- we see that Drug 390 provided a significantly shorter treatment length over the Generic in similar patient groups.

Figure G

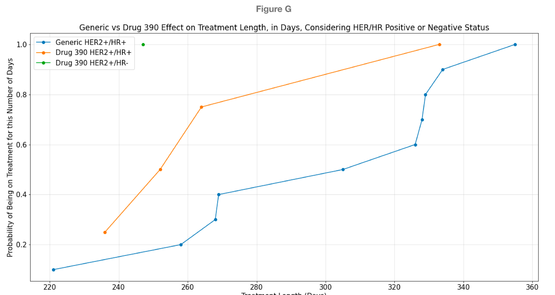


Figure G When comparing on just HER2+/HR+ and HER2+/HR- we see that Drug 390 provided a significantly shorter treatment length over the Generic in similar patient groups. Notice how for Drug 390 HER2+/HR- there is not a line rather a single dot, representative of just one patient and this category's complement on Drug 390 is not present. This would lead one to believe this cancer type is either rare or just not present in this sample patient group.

Experimentation with Machine Learning Algorithms & Results

As a method of experimentation with this dataset it was attempted to 'predict' whether a patient was on Drug 390 or on the Generic working backwards with Treatment Length, ER Positive, HER2 Positive, and the engineered HR Status.

The models used were: **K-Nearest-Neighbors Classifier, Logistic Regression Classifier, Linear SVC, Gaussian Naive Bayes, Bernoulli Naive Bayes**

These models were chosen as this is a supervised learning type dataset with labels provided for us before-hand. Also because of the dual class nature, on Drug 390 or Generic, these work best with this type of dataset. Also the heavy use of categorical data provided for use-cases only relevant to models which were not heavily based on the numerical properties of the features (LinearRegression, Lasso, Ridge, etc.)

The best performing models were the KNN and LogisticRegression models although they produced just between 70-75% accuracy scores. These non-stellar scores would be improved if there was more data to train and test with. Below are each model's results based on accuracy scores. Although mentioned in the introduction of this section, the Linear SVC model performed poorly and the results were not included as they offered no value to this analysis.

Due to the size of the code required for these models the associated logic is included in the aforementioned Jupyter Notebook file.

Figure H

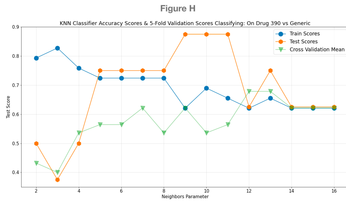


Figure H The KNN algorithm classifies based on the Euclidean distance principle and minimizes the distance between a point and its 'neighbors' to be as close to 0 as possible. These are then grouped as appropriately. When the model class is instantiated one can set the parameters for the neighbors as:

```
knn = KNeighborsClassifier(n_neighbors=5)
```

In this chart we plotted the differences in accuracy score and cross-validation fold mean as the neighbors setting changed. Our model was most accurate and consistent at 4-8 neighbors.

Figure I

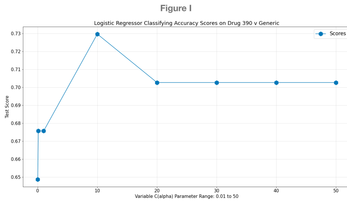


Figure I For the Logistic Regressor model the classification is based on the sigmoid function classifying by numbers between 0 and 1. This is a method of binary classification which fits our needs for this project. The model allows for a simple tuning using the alpha(C) parameter in the class instance as per below. ne can set the parameters for the neighbors as:

```
lreg = LogisticRegression(C=10)
```

In this chart we plotted the differences in accuracy score as this alpha(C) setting changed. Our model was most accurate, consistent with the KNN model, at alpha=10 with ~73% accuracy.

Figure J

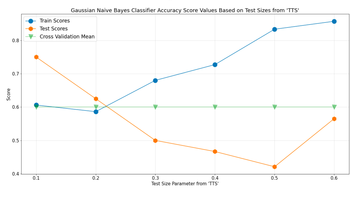


Figure J The Gaussian NB model allows for binary classification leveraging the probability of the conditional features involved.

To tune the model as best possible given the state of the data we are experimenting with we used the test_size parameter of train_test_split as the variable per below.

```
sizer = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6]
    for sz in sizer:
        X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size = sz,
random_state=42, stratify=y)
```

As you can see on the chart the model did not generalize very and underfoot the data, but still performed better than the LinearSVC model.

Figure K

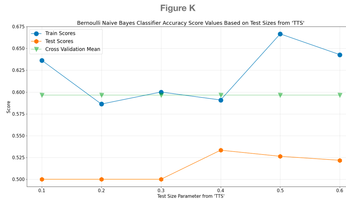


Figure K The Bernoulli NB model was setup similarly to the Gaussian NB model following similar principles and it performed very poorly in generalizing to the test data.

Although a poor performance, it performed better and more comprehensibly than the LinearSVC model.