BIOM5405/SYSC5405

Assignment 2

Contents

BIOM5405/SYSC5405

Assignment 2

BIOM5405/SYSC5405

Assignment 2

List of Figures

BIOM5405/SYSC5405

Assignment 2

Q1

Let's focus on our healthy patients for this question. Create a categorical version of the temperature feature data using the rule:

if T_healthy <= 36.8, T_cat = t-normal; else, T_cat = t-fever.

Now create an ordinal version of the respiration rate feature data using the rule:

if RR_healthy < 19.0, RR_ord = RR-low;
else if RR_healthy < 23.0, RR_ord = RR-med;
else RR_ord = RR-high.

Create a contingency table for your new data and use a χ2 test to check if t_cat is significantly correlated with RR_ord. Report your null hypothesis H0 (~15 words), your alternate hypothesis H1, your χ2 value, your degrees of freedom, your p-value, and your conclusion (~50 words).

Solution

Part 1: Building Contingency Table

Table 1 and 2 is built using the information presented in (Green, 2023). They provide the information on the contingency tables needed for the χ2 test.

T_cat	Outcome=RR-high	Outcome=RR-low	Outcome=RR-med	Totals
t-fever	30	33	31	94
t-normal	35	26	45	106
Totals	65	59	76	200

Table 1: Contingency Table (f_o): Joint distribution of Outcomes (Respiration Rate) and Temperature

Extra 1: Calculating the Row Marginals

T_cat	Outcome=RR-high	Outcome=RR-low	Outcome=RR-med	Totals
t-fever	31.91 %	35.11 %	32.98 %	94
t-normal	33.02 %	24.53 %	42.45 %	106
Totals	65	59	76	200

Table 2: Row Marginal Counts: Distribution in Table 1 expressed as percentages.

Part 2: χ2 test

T_cat	Outcome=RR-high	Outcome=RR-low	Outcome=RR-med	Totals
t-fever	30.55	27.73	35.72	94
t-normal	34.45	31.27	40.28	106
Totals	65	59	76	200

Table 3: Expected frequencies (f_e) of Table 1

***H_0*: The variables T_cat and RR_ord are independent of each other.**

***H_1*: The variables T_cat and RR_ord are dependent on each other.**

$\chi^2 = 3.0851$

***p-value* = 0.21**

***degrees of freedom* = 2**

***Conclusion*: Failed to reject null hypothesis (H_0) for significance level (alpha) 0.05**

BIOM5405/SYSC5405

Assignment 2

Extra 2: χ^2 formula

$$\chi^2 = \sum \frac{(f_o - f_e)^2}{f_e} \quad (1)$$

χ^2 value is calculate using values in Table 1 (f_o) and Table 3 (f_e) in equation (1).

Extra 3: Degrees of freedom

Degrees of freedom = (number of rows – 1) x (number of columns – 1)

= (2-1) x (3-1)

= 1 x 2

= 2

Chi-Square (χ^2) Distribution

Degrees of Freedom	Area to the Right of Critical Value									
	0.995	0.99	0.975	0.95	0.90	0.10	0.05	0.025	0.01	0.005
1	—	—	0.001	0.004	0.016	2.706	3.841	5.024	6.635	7.879
2	0.010	0.020	0.051	0.103	0.211	4.605	5.991	7.378	9.210	10.597
3	0.072	0.115	0.216	0.352	0.584	6.251	7.815	9.348	11.345	12.838
4	0.207	0.297	0.484	0.711	1.064	7.779	9.488	11.143	13.277	14.860
5	0.412	0.554	0.831	1.145	1.610	9.236	11.071	12.833	15.086	16.750

Figure 1: Chi-Square Distribution Table

Extra 4: χ^2 Table

We know the degrees of freedom is 2. The significance level (alpha) is not specified in the question. Based on the p-value calculated from χ^2 (0.21) we will fail to reject the null hypothesis (H_0) as long as alpha < p-value. A good reference to understand what is going on is (Tutor, 2019).

Extra 5: Cell Proportion Chart

From figure 2 we can see that there no change in the proportion of values in a category for different values of RR_ord and T_cat. This means that there very little or no correlation among the two.

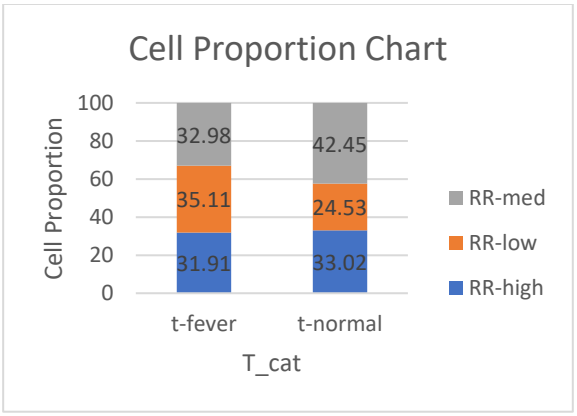

Figure 2: Cell proportion for T_cat and RR_ord

BIOM5405/SYSC5405

Assignment 2

Q2

Compute the inter-quartile range and the “10% trimmed mean” of T_healthy. (10% means dropping the top and bottom 5% of samples)

Solution

True Inter Quartile Range: 0.3629

True Mean: 36.7869

10% trimmed Inter Quartile range: 0.3158

10% trimmed mean: 36.7863

Extra 1: Box Plots

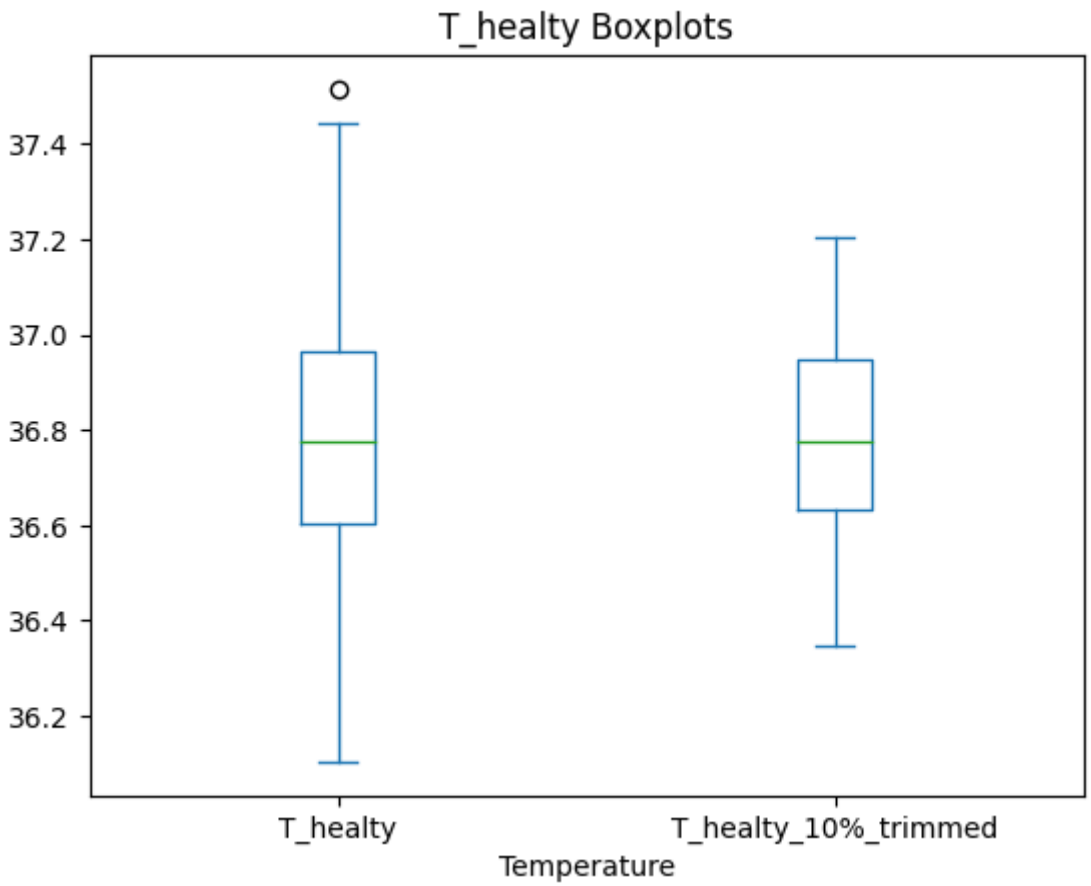

Figure 3: Box plot for T_healthy and 10% trimmed T_healthy. Outliers are depicted with black circles.

Extra 2: Conclusion

It can be assumed that, since the mean did not change, after the 10% trim, that there were in fact no outliers in the data. This assumption would be wrong, as we see from the Figure 3 that there was 1 outlier in the data shown in the T_healty box plot.

BIOM5405/SYSC5405

Assignment 2

Extra 3: Quartile Ranges

Table 4 lists the quartile ranges. For 10% dropped mean.

Quartile	Lowest value	Highest Value	Quartile range
1	36.104046	36.60181	0.497764
2	36.602745	36.775505	0.17276
3	36.778291	36.964375	0.186084
4	36.968679	37.516024	0.547345

Table 4: Quartile ranges

BIOM5405/SYSC5405

Assignment 2

Q3

Using bootstrapping, compute the 90% confidence interval of the “10% trimmed mean” of T_healthy. Follow Procedure 5.6 from Cohen’s text:

1) Construct a distribution from K bootstrap samples for a statistic u; *

2) Sort the values in the distribution

3) The lower bound of the 90% confidence interval is the (K*0.05)th value, the upper bound is the (K*0.95)th value in the sorted distribution.

*Here, u is the observed trimmed mean and a bootstrap sample will consist of 200 samples drawn with replacement from T_healthy.

Solution

Lower confidence interval: 36.75491145
Upper confidence interval: 36.820047422222224

Extra 1: Real Mean

Real mean: 36.78693378

Extra 2: Visual Representation

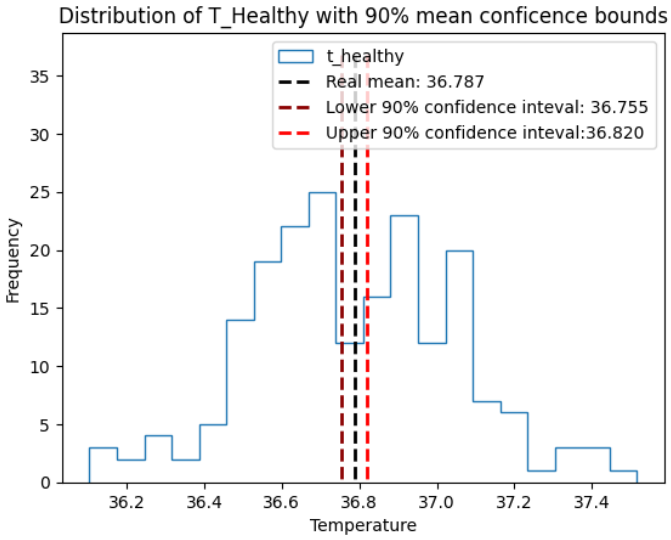

Figure 4: Histogram of Temperature of Healthy people along with means

BIOM5405/SYSC5405

Assignment 2

Q4

Examine the RR feature. Combine the RR feature data for both classes to create RR_combined. Do the RR_combined feature data contain outliers? Describe how you tested this and what conclusions you drew. How did the mean and median of RR_combined change with the outliers (if any) removed? (50 words + calculations)

Solution:

***Ans*: no outliers**

***Technique*: Box plots (Figure 6)**

***How it works*: First IRQ (Inter Quartile Range) is calculated which is upper value of Quartile 3 – lower value of Quartile 1. Then the upper and lower whisker limits are calculated which are 1.5 times the IQR. Any values that don't fall in the whisker limits are considered as outliers. More details in** (Yi)

Extra 1: Validation

Looking at the histogram in Figure 7 and 8. We see that ther are no sudden peaks at both left and right ends. Additionally, the data is also not spread too wide.

Extra 2: Box plot:

Figure blow from (Wikipedia) depicts how the box plots are create.

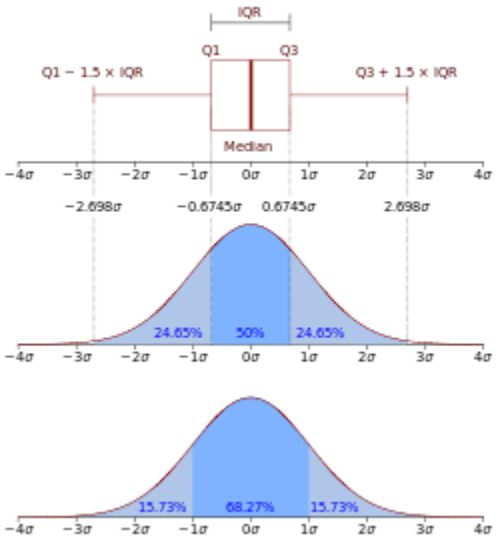

Figure 5: Box plot creation as illustrated by (Wikipedia)

BIOM5405/SYSC5405

Assignment 2

Extra 3: Plots

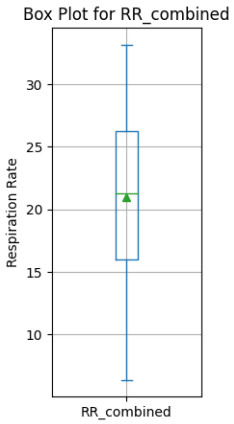

Figure 6: Box Plot for RR_combined

Figure 6 illustrates the box plot for RR_combined (RR_healthy and RR_covid). The green triangle in the box denotes the mean while the green line denotes the median We see that the mean and median are similar.

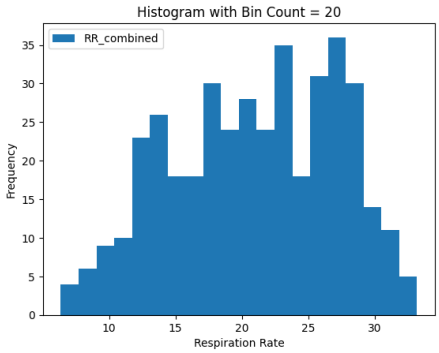

Figure 7: Histogram of RR_combined with bin count = 20

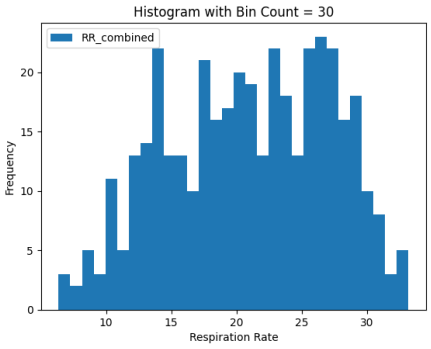

Figure 8: Histogram of RR_combined with bin count = 30

The data for RR_combined in tri-modal. More details in Q5 - Extra 3.

BIOM5405/SYSC5405

Assignment 2

Q5

Using randomization (or permutation), test whether RR_covid has significantly greater mean than RR_healthy. Briefly describe how you did this. What p-value did you obtain? What conclusion do you draw? (50 words)

Solution:

***P-value*: 0.55565**

***Conclusion*: Significant evidence DOES NOT exists that there is a greater difference in mean for RR_covid and RR_healthy**

Descreption: Following the information presented in "Slide 65" in "SYSC5405-Slides-04-ClassificationAccuracy" the test was conducted. A further detailed explanation is found in the code.

Extra 1: observed deference in means

Observed mean difference: -0.08615704999999707 (RR_covid - RR_healthy)

Extra 2: Hypothesis

H_0: There is no significant difference in means between RR_covid and RR_healthy

H_1: There is significant evidence that RR_covid has a greater mean than RR_healthy.

Extra 3: Deep Dive

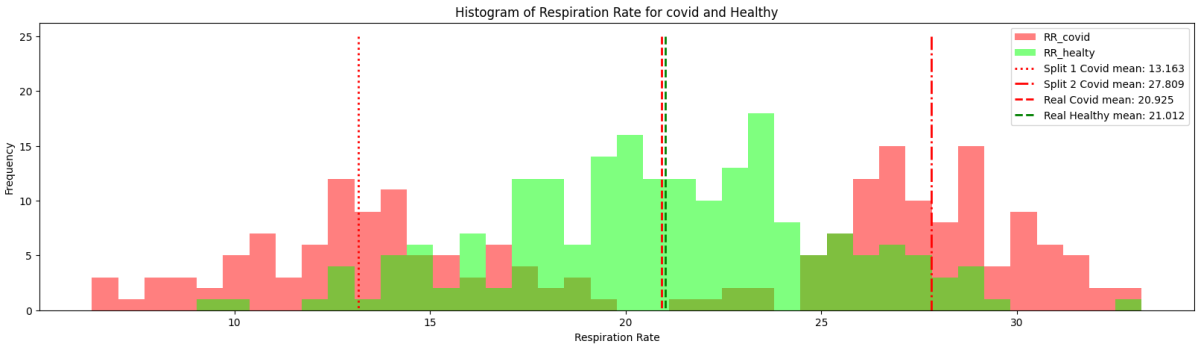

Figure 9: Histogram for Respiration Rate for covid and healthy patients with means marked

The data for Covid is bi-modal. When the two modes are separated, there is a significant difference in mean, but when the two covid mode are combined the mean is similar to healthy patients.

BIOM5405/SYSC5405

Assignment 2

Q6

Let's use temperature alone to create a simple classifier. Plot an ROC curve for temperature Assume that T_covid samples actually have class = +1 and T_healthy samples actually have class = 0. Our classifier will apply a tunable threshold to determine whether each sample should be predicted to have class 0 (healthy) or class 1 (covid). Report the AUC value in the title of the plot.

Solution:

Figure 10 illustrates the ROC and AUC values of the simple classifier (Normalized Temp).
Some additional work is also done for the question reported in the extra sections.

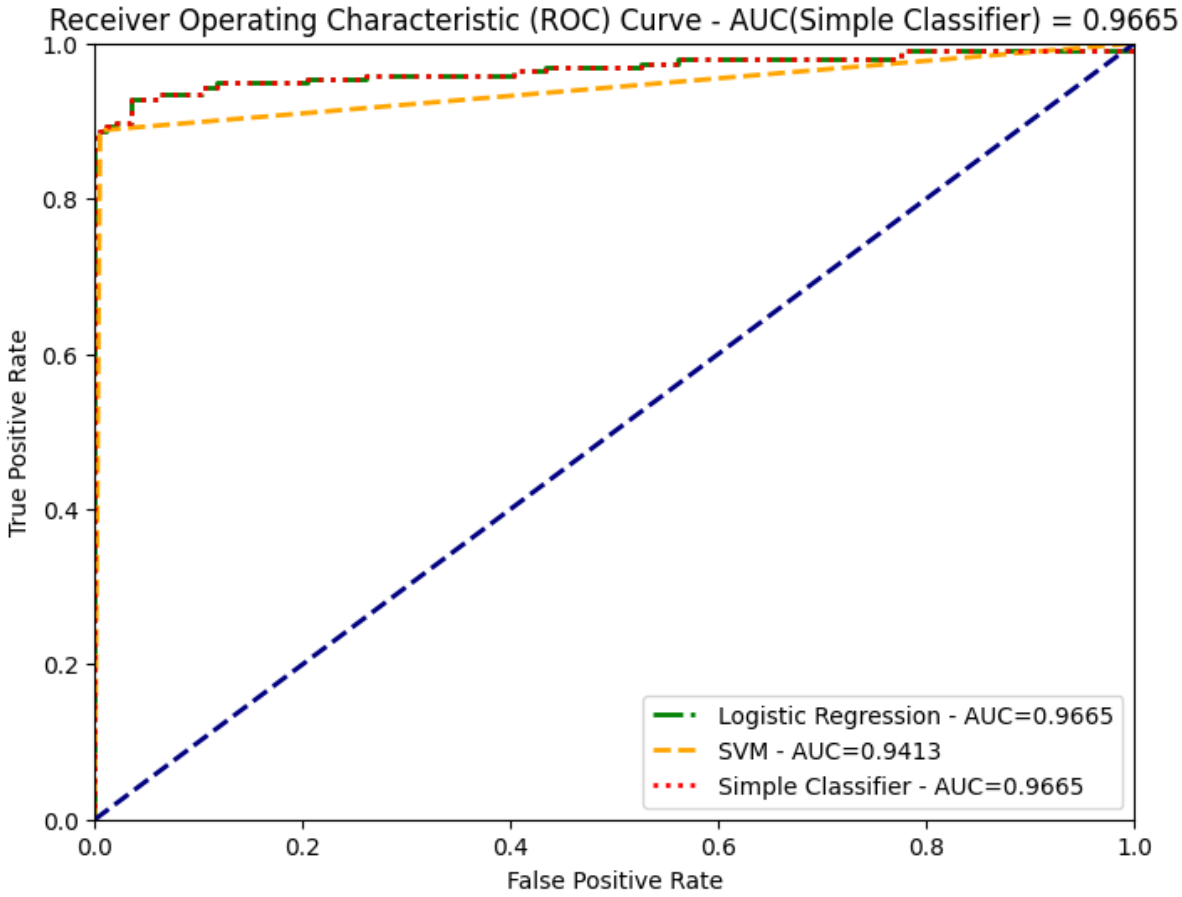

Figure 10: ROC for different classifiers

Extra 1: Comparison with additional classifiers

It can be noted that simple classifier had the highest AUC score along with Logistic regression of 0.9665.

Note: Figure 10 reports the AUC value for normalized data in the title. Additionally two more classifiers Logistic Regression and SVM have been trained to compare the classification power of the simple classifier. The respective AUC values are presented in the legend on the bottom right.

BIOM5405/SYSC5405

Assignment 2

Extra 2: Reasoning for Simple classifier perming high

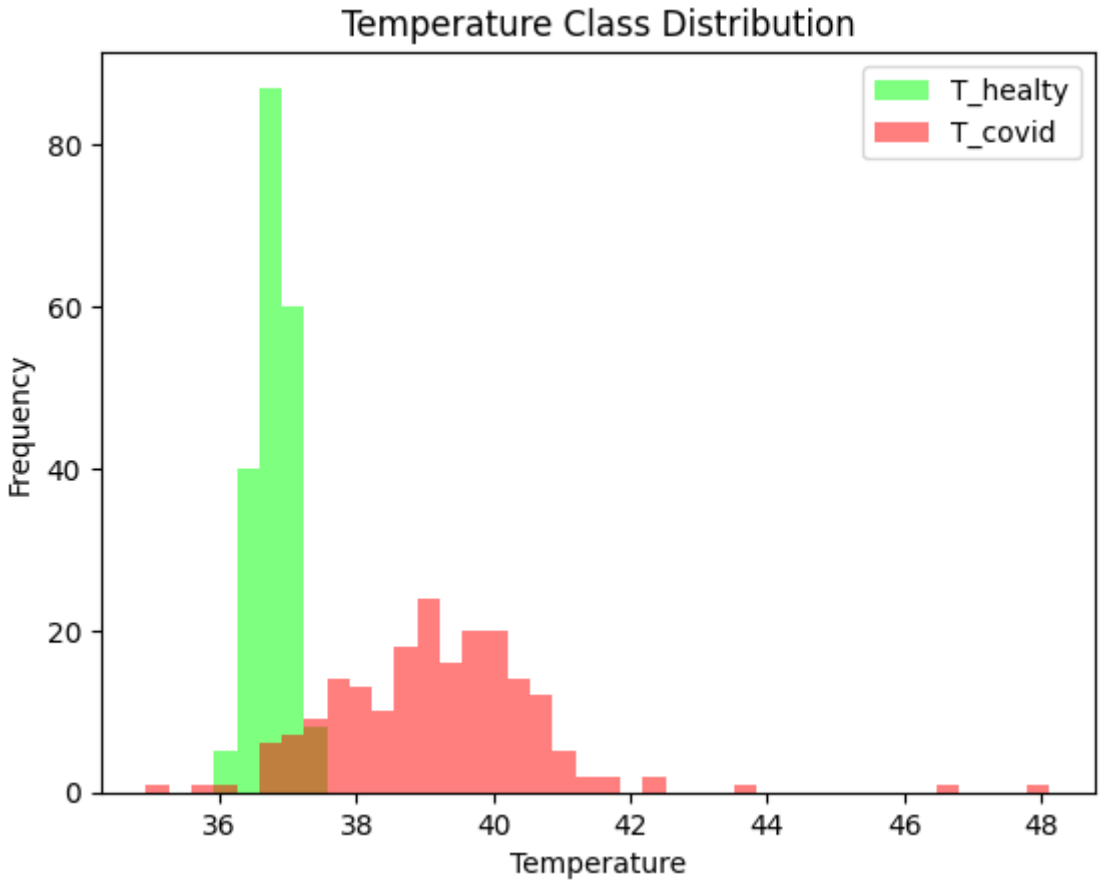

Figure 11: Histogram of Temperature of Covid an Healthy people

Since the variance of the “healthy” patients is small and at the near start of data a simple “normalization” classifier worked to separate the data.

Note: In most (complex) cases this should never happen.

Extra 3: Simple Classifier Prediction Plot

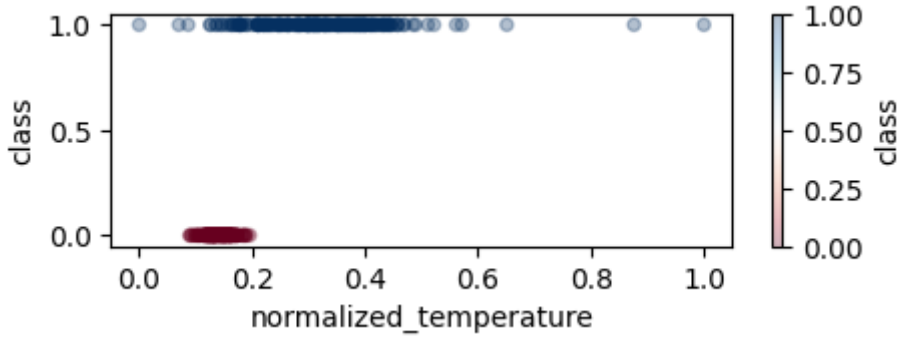

Figure 12: Simple Classifier predictions

BIOM5405/SYSC5405

Assignment 2

Code:

```
# %%
import pandas as pd
import numpy as np
import math
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_curve, roc_auc_score, auc
from scipy.stats import chi2_contingency

data_path = "../data/A2Q2.csv"
columns = ['T_healty', 'T_covid', 'RR_healty', 'RR_covid']

dataset = pd.read_csv(data_path, header=None)
dataset.columns = columns # set the headers
dataset.tail()

# # Q1
healthy_patients_df = pd.DataFrame()

# create a Temp Category
healthy_patients_df['T_cat'] = dataset['T_healty'].apply(
    lambda temp: "t-normal" if temp <= 36.8 else "t-fever",
    )

# Create a Resperation Category
healthy_patients_df['RR_ord'] = dataset['RR_healty'].apply(
    lambda rr: "RR-low" if rr < 19.0 else "RR-med" if rr < 23.0 else "RR-high"
    )

healthy_patients_df.head()

# %%
# Create a Contengency Table
contangency_table_df = healthy_patients_df[['T_cat',
'RR_ord']].value_counts().reset_index().pivot(index='T_cat', columns='RR_ord',
values='count')
contangency_table_df

# %%
# Create the Expected Frequency Table:
temp_totals = dict(contangency_table_df.sum(axis=1))
rr_totals = dict(contangency_table_df.sum(axis=0))
```

BIOM5405/SYSC5405

Assignment 2

```
total = sum(temp_totals.values())

assert total == sum(rr_totals.values()), f"Totals are not correct"

rr_exp = {}
for rr in rr_totals:
    t_exp = {}
    for t in temp_totals:
        freq_exp = temp_totals[t] * rr_totals[rr] / total
        t_exp[t] = freq_exp
    rr_exp[rr] = t_exp

exp_frequencies_df = pd.DataFrame(rr_exp)
exp_frequencies_df

# %%
# Chi Square test
chi_square = (((contangency_table_df - exp_frequencies_df)**2) /
exp_frequencies_df).sum().sum()
chi_square

# %%
# validation
statistic, pvalue, dof, exp_f = chi2_contingency(contangency_table_df.values)

print(f"Chi-Square Value = {statistic}")
print(f"p-value = {pvalue}")
print(f"Degree of Freedom = {dof}")
print(f"Expected Frequencies = {exp_f}")

# %% [markdown]
# do not reject null hypothess for significance level 0.05

# %% [markdown]
# # Q2

# %%
iqr = dataset["T_healty"].describe()["75%"] -
dataset["T_healty"].describe()["25%"]
trimed_t_heathy = dataset["T_healty"].sort_values().reset_index().iloc[10:190]
trimmed_iqr = trimed_t_heathy["T_healty"].describe()["75%"] -
trimed_t_heathy["T_healty"].describe()["25%"]

mean_trimed = trimed_t_heathy["T_healty"].mean()
real_mean = dataset["T_healty"].describe()["mean"]
```

BIOM5405/SYSC5405

Assignment 2

```
print(f"Inter Quyartile Range: {iqr}")
print(f"Real mean: {real_mean}")
print(f"Trimmed 10% Inter Quyartile Range: {trimmed_iqr}")
print(f"Trimmed 10% mean: {mean_trimed}")


# %%
t_cat_df =
pd.concat([dataset["T_healty"],trimed_t_heathy[["T_healty"]]],axis=1)
t_cat_df.columns = ["T_healty", "T_healty_10%_trimmed"]
t_cat_df.plot.box(
    xlabel="Temperature",
    title="T_healty Boxplots"
)

# %%
t_healthy_list = dataset["T_healty"].values.tolist() # geta list of data
t_healthy_list.sort() # sort the data

len_t_health = len(t_healthy_list) # get the length of data
len_of_quartile = int(math.floor(len_t_health/4)) # get quartile size
quartiles = [
    t_healthy_list[len_of_quartile * quartile: len_of_quartile * (quartile +
1)]
    for quartile in range(4)
]

# print Quartile Ranges
for index, quartile in enumerate(quartiles):
    print(f"Quartile {index+1}: Start({quartile[0]}) End({quartile[-1]})")

# %%
# verification
dataset["T_healty"].describe()

# %%
dataset["T_healty"].plot.hist(bins=20)

# %%
min_cutoff_index = int(math.floor(len_t_health * 0.05))
max_cutoff_index = len_t_health - min_cutoff_index
t_healthy_list_10_trimmed = t_healthy_list[min_cutoff_index:max_cutoff_index]
sum(t_healthy_list_10_trimmed)/len(t_healthy_list_10_trimmed)

# %% [markdown]
# # Q3
```

BIOM5405/SYSC5405

Assignment 2

```
# %%
# init
K = 200000
bootstrap_means = []
t_healthy = dataset['T_healty'].values
real_mean = t_healthy.mean()
len_t_health = len(t_healthy)
lower_10_bound  = int(math.floor(len_t_health*0.05))
upper_10_bound = len_t_health - lower_10_bound
K_lower_confidence_bound = int(math.floor(K*0.05))
K_upper_confidence_bound = K - K_lower_confidence_bound

# Run Bootstrapping
for _ in range(K):
    # get bootstrap samples with replacement
    bootstrap_sample = np.random.choice(
        t_healthy,
        size=len_t_health,
        replace= True)

    # get 10% trimmed mean
    trimmed_mean = bootstrap_sample[lower_10_bound:upper_10_bound].mean()

    # add sampled mean
    bootstrap_means.append(trimmed_mean)


# sort the data
bootstrap_means.sort()

# calculate the confidence intervals
lower_confidence_interval = bootstrap_means[K_lower_confidence_bound]
upper_confidence_interval = bootstrap_means[K_upper_confidence_bound]



print(f"Lower confidence interval: {lower_confidence_interval}")
print(f"Upper confidence interval: {upper_confidence_interval}")
print(f"-----------------------------------------------------")
print(f"actual mean: {real_mean}")

# %%
ax = dataset['T_healty'].plot.hist(
    bins=20,
    histtype=u'step',
```

BIOM5405/SYSC5405

Assignment 2

```
    label="t_healthy",
    title="Distribution of T_Healthy with 90% mean conficence bounds",
    xlabel="Temperature"
)

plt.plot([real_mean, real_mean], [real_mean, 0], color='black', lw=2,
linestyle='--', label=f'Real mean: {real_mean:.3f}')
plt.plot([lower_confidence_interval, lower_confidence_interval],
[lower_confidence_interval, 0], color='darkred', lw=2, linestyle='--',
label=f'Lower 90% confidence inteval: {lower_confidence_interval:.3f}')
plt.plot([upper_confidence_interval, upper_confidence_interval],
[upper_confidence_interval, 0], color='red', lw=2, linestyle='--',
label=f'Upper 90% confidence inteval:{upper_confidence_interval:.3f}')
plt.legend(loc='upper right')

# %% [markdown]
# # Q4

# %%
# build combined features
RR_combined = pd.DataFrame()
RR_combined["RR_combined"] = pd.concat(
    [dataset["RR_covid"], dataset["RR_healty"]],
    axis=0)

# get dataset stats
stats = RR_combined.describe()
min_val = stats.loc['min'].values[0]
max_val = stats.loc['max'].values[0]
q3 = stats.loc['75%'].values[0]
q1 = stats.loc['25%'].values[0]
iqr = q3 - q1
upper_wisker_limit = q3 + (iqr * 1.5)
lower_wisker_limit = q1 - (iqr * 1.5)

print(f"""min: {min_val:0.4f} --- lower_limit: {lower_wisker_limit:.4f} ==>
{"no outliers" if min_val > lower_wisker_limit else "outliers exists"}""" )
print(f"""max: {max_val:0.4f} --- upper_limit: {upper_wisker_limit:.4f} ==>
{"no outliers" if max_val < upper_wisker_limit else "outliers exists"}""" )

# %%
RR_combined.plot.box(
    ylabel="Respiration Rate",
    title="Box Plot for RR_combined",
    figsize=(2,5),
    showmeans=True,
```

BIOM5405/SYSC5405

Assignment 2

```
    grid=True,
    legend=True
)

# %%
RR_combined.plot.hist(
    bins=20,
    xlabel="Respiration Rate",
    title="Histogram with Bin Count = 20"
)

RR_combined.plot.hist(
    bins=30,
    xlabel="Respiration Rate",
    title="Histogram with Bin Count = 30"
)

# %% [markdown]
# # Q5

# %%
# init
significance_level_p_val = 0.05
num_permutations = 100000
permuation_mean_diff_samples = []
rr_covid = dataset["RR_covid"].values
rr_healthy = dataset["RR_healty"].values
rr_covid_count = len(rr_covid)
rr_healthy_count = len(rr_healthy)


# calculate observed mean
rr_covid_mean = rr_covid.mean()
rr_healthy_mean = rr_healthy.mean()
real_mean_diff = rr_covid_mean - rr_healthy_mean


# Concatenate data
rr_conbined = np.concatenate([rr_covid, rr_healthy])

# Run Permuation
for _ in range(num_permutations):
    # shuffle data
    np.random.shuffle(rr_conbined)

    # sample the data without replacements
```

BIOM5405/SYSC5405

Assignment 2

```
    sampled_rr_covid = rr_conbined[:rr_covid_count]
    sampled_rr_healthy = rr_conbined[rr_covid_count:]

    # Calcuate test statstatics
    sampled_rr_covid_mean = sampled_rr_covid.mean()
    sampled_rr_healthy_mean = sampled_rr_healthy.mean()
    sampled_mean_diffrence = sampled_rr_covid_mean - rr_healthy_mean

    # add the observation
    permuation_mean_diff_samples.append(sampled_mean_diffrence)

# calculate p-value
p_val = sum([sample >= real_mean_diff for sample in
permuation_mean_diff_samples])/num_permutations

print(f"Observed mean diffrenc: {real_mean_diff}")
print(f"P-value: {p_val}")

# Conclusion
if p_val < significance_level_p_val:
    print("Significant evidence exists that there is a greater deiffrence in
mean for RR_covid and RR_healthy")
else:
    print("Significant evidence DOES NOT exists that there is a greater
deiffrence in mean for RR_covid and RR_healthy")


# %%
dataset[["RR_covid","RR_healty"]].plot.hist(
    bins=40,
    color=[(1,0,0,0.5),(0,1,0,0.5)],
    title="Histogram of Respiration Rate for covid and Healthy",
    xlabel="Respiration Rate"
)

# %% [markdown]
# # Q6

# %%
# Plot distributions
dataset[['T_healty', 'T_covid']].plot.hist(
    bins=40,
    color=[(0,1,0,0.5),(1,0,0,0.5)],
    xlabel="Temperature",
    title="Temperature Class Distribution"
)
```

BIOM5405/SYSC5405

Assignment 2

```
# %%
# lable covid class
temp_1_df = pd.DataFrame()
temp_1_df["temp"] = dataset['T_covid']
temp_1_df["class"] = 1
# lable healthy class
temp_0_df = pd.DataFrame()
temp_0_df["temp"] = dataset['T_healty']
temp_0_df["class"] = 0
# get dataset
temp_df = pd.concat([temp_1_df, temp_0_df])
temp_df

# %%
# split the data for test train
X_train, X_test, y_train, y_test = train_test_split(temp_df["temp"].values,
temp_df["class"].values, test_size=0.02, random_state=42)
X_train = X_train.reshape(-1, 1)
X_test = X_test.reshape(-1, 1)

# %%
# train model
lrc = LogisticRegression(random_state=0).fit(X_train, y_train)

svm = SVC(random_state=0).fit(X_train, y_train)

# %%
y_stand = ((X_train - X_train.mean()) / (X_train.std()))
y_norm = ((X_train - X_train.min()) / (X_train.max() - X_train.min()))

# %%
y_lrc = lrc.predict_proba(X_train)[:, 1]
y_svm = svm.predict(X_train)

fpr_lrc, tpr_lrc, _ = roc_curve(y_train, y_lrc)
roc_auc_lrc = auc(fpr_lrc, tpr_lrc)

fpr_svm, tpr_svm, _ = roc_curve(y_train, y_svm )
roc_auc_svm = auc(fpr_svm, tpr_svm)

fpr_norm, tpr_norm, _ = roc_curve(y_train, y_norm )
roc_auc = auc(fpr_norm, tpr_norm)

plt.figure(figsize=(8, 6))
```

BIOM5405/SYSC5405

Assignment 2

```
plt.plot(fpr_lrc, tpr_lrc, color='green', lw=2, linestyle='-.',
label=f'Logistic Regression - AUC={roc_auc_lrc:.4f}')
plt.plot(fpr_svm, tpr_svm, color='orange',  lw=2, linestyle='--', label=f'SVM
- AUC={roc_auc_svm:.4f}')
plt.plot(fpr_norm, tpr_norm, color='red',  lw=2, linestyle=':', label=f'Simple
Classifier - AUC={roc_auc:.4f}')
plt.plot([0, 1], [0, 1], color='black', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title(f'Receiver Operating Characteristic (ROC) Curve - AUC(Simple
Classifier) = {roc_auc:.4f}')
plt.legend(loc='lower right')
plt.show()

# %%
```

BIOM5405/SYSC5405

Assignment 2

References

Green, J. (2023). Part 3: Experimental design. *BrightSpace*, 20-22. Retrieved from https://brightspace.carleton.ca/d2l/le/content/212097/viewContent/3302629/View

Tutor, T. O. (2019). Chi Square Test. YouTube. Retrieved from https://www.youtube.com/watch?v=HKDqlYSLt68

Wikipedia. (n.d.). *Boxplot.* Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Interquartile_range

Yi, M. (n.d.). A Complete Guide to Box Plots. CartIO. Retrieved from https://chartio.com/learn/charts/box-plot-complete-guide/