Lab # 8: Improving the Regression Model

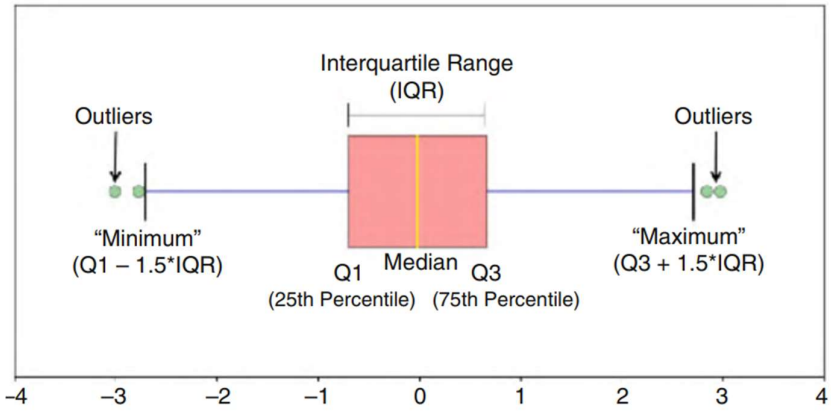
8.1 Objectives

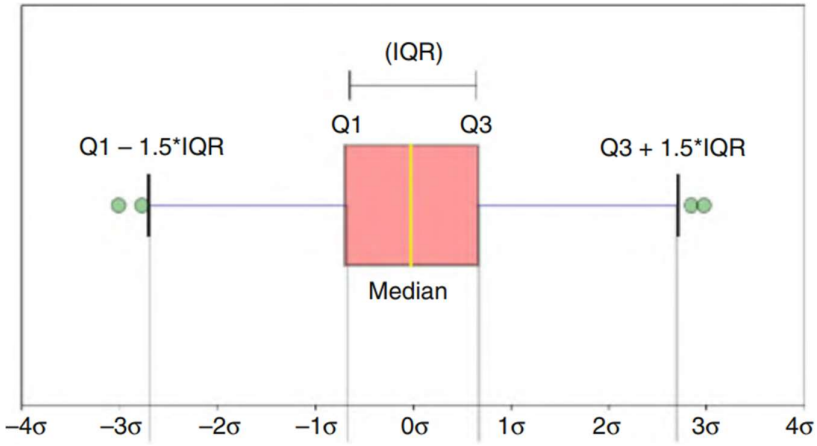
1. Understand the box plot, removing outliers and NA in the data.
2. Understand the feature Importance and Improving regression Model.

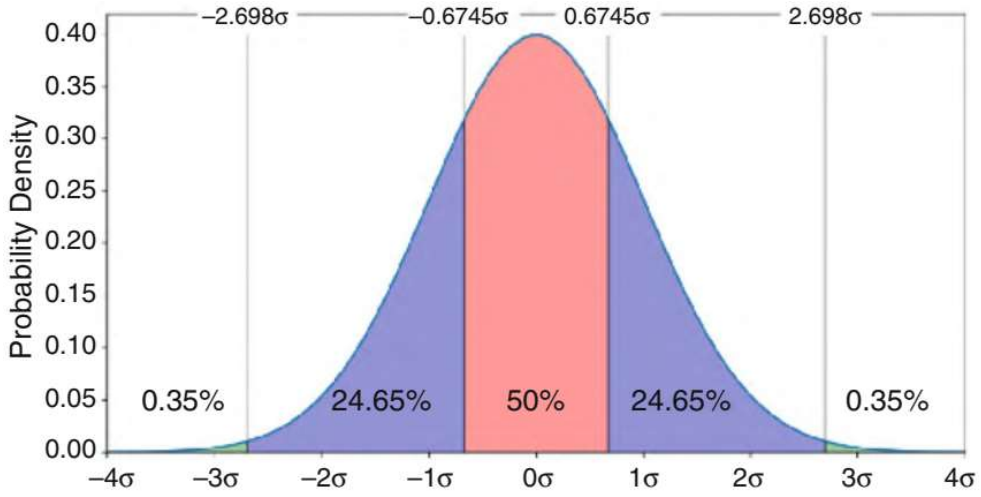
8.2 Pre-Lab

8.2.1 Box Plot

A boxplot is a standardized way of displaying the distribution of data based on a five number summaries ("minimum," first quartile (Q1), median, third quartile (Q3), and "maximum"). It tells you about your outliers and what their values are. It can also tell you if your data is symmetrical, how tightly your data is grouped, and if and how your data is skewed. Here is an image that shows normal distribution on a boxplot:







As seen, a boxplot is a great way to visualize your dataset.

8.3 In-Lab

Now, let us try to remove the outliers using our boxplot plot. This can be easily achieved with pandas dataframe. But do note that the dataset should be numerical to do this.

In-Lab Task 1

Write the following code to visualize data using box plot.

```
# Importing libraries needed
# Note that keras is generally used for deep learning as well
from keras.models import Sequential
from keras.layers import Dense, Dropout
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import numpy as np
from sklearn import linear_model
from sklearn import preprocessing
from sklearn import tree
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
import pandas as pd
import csv

import matplotlib.pyplot as plt

# ==============================================================================
# Read Data and fix seed
# ==============================================================================
# fix random seed for reproducibility
np.random.seed(7)
df = pd.read_csv("Alumni Giving Regression (Edited).csv", delimiter="," )
dd_df_1=df.head()
```

```
import seaborn as sns
import pandas as pd
boxplot = pd.DataFrame(df).boxplot()
```

Discuss the result.

8.3.1 Remove Outlier

Removing the outliers observed through the results in the boxplot can improve the model. The outlier can be removed through passing a quantile value.

In-Lab Task 2

Write the following code to remove outliers in the data. Discuss the difference between two pieces of code.

```
# %%
quantile99 = df.iloc[:,0].quantile(0.99)
df1 = df[df.iloc[:,0] < quantile99]
df1.boxplot()
```

```
# %%
quantile1 = df.iloc[:,0].quantile(0.01)
quantile99 = df.iloc[:,0].quantile(0.99)
df2 = df[(df.iloc[:,0] > quantile1) & (df.iloc[:,0] <quantile99)]
df2.boxplot()
```

Discuss the results obtained.

8.3.2 Remove NA

Drop the "NA" values in the data to improve the result. Remove "NA" values from data using the following code:

```
# %%
df.dropna()
```

8.3.3 Feature Importance

Apart from data cleaning, we can apply use variables that we deem to be important to us. One way of doing so is via feature importance of random forest trees. In many use cases it is equally important to not only have an accurate but also an interpretable model. Oftentimes, apart from wanting to know what our model's house price prediction is, we also wonder why it is these high/low and which features are most important in determining the forecast. Another example might be predicting customer churn—it is very nice to have a model that is successfully predicting which customers are prone to churn, but identifying which variables are important can help us in early detection and maybe even improving the product/service. Knowing feature importance indicated by machine learning models can benefit you in multiple ways, for example:

1. By getting a better understanding of the model's logic you can not only verify it being correct but also work on improving the model by focusing only on the important variables.
2. The above can be used for variable selection—you can remove x variables that are not that significant and have similar or better performance in much shorter training time.
3. In some business cases it makes sense to sacrifice some accuracy for the sake of interpretability.

For example, when a bank rejects a loan application, it must also have a reasoning behind the decision, which can also be presented to the customer.

```
# %%
df.dropna()

# %%
# ================================================================================
# Feature Ranking
# ================================================================================
RF = model3
importances = RF.feature_importances_
std = np.std([tree.feature_importances_ for tree in RF.estimators_],axis=0)
indices = np.argsort(importances)[::-1]
# Print the feature ranking
print("Feature ranking:")
for f in range(X.shape[1]):
    print("%d. feature (Column index) %s (%f)" % (f + 1,indices[f],
                                                  importances[indices[f]]))
```

List here the Feature Ranks obtained through this code:

Explain the feature ranks with relation to the correlation obtained in the previous labs.

In-Lab Task 3

Write the following code to improve the regression model using top three ranked feature and compare your results obtained in the previous labs.

```
# %%
indices_top3 = indices[:3]
print(indices_top3)
dataset=df
df = pd.DataFrame(df)
Y_position = 5
TOP_N_FEATURE = 3
X = dataset.iloc[:,indices_top3]
Y = dataset.iloc[:,Y_position]
# create model
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.20,
                                                    random_state=2020)
#Model 1 : linear regression
model1 = linear_model.LinearRegression()
model1.fit(X_train, y_train)
y_pred_train1 = model1.predict(X_train)
print("Regression")
print("================================")
RMSE_train1 = mean_squared_error(y_train,y_pred_train1)
print("Regression TrainSet: RMSE {}".format(RMSE_train1))
print("================================")
y_pred1 = model1.predict(X_test)
RMSE_test1 = mean_squared_error(y_test,y_pred1)
print("Regression Testset: RMSE {}".format(RMSE_test1))
print("================================")
```

Discuss Results and Compare with previous labs.

Rubric for Lab Assessment

The student performance for the assigned task during the lab session was:			
Excellent	The student completed assigned tasks without any help from the instructor and showed the results appropriately.	4	
Good	The student completed assigned tasks with minimal help from the instructor and showed the results appropriately.	3	
Average	The student could not complete all assigned tasks and showed partial results.	2	
Worst	The student did not complete assigned tasks.	1	

Instructor Signature: _____ Date: _____