Assignment-based Subjective Questions

1. From your analysis of the categorical variables from the dataset, what could you infer about their effect on the dependent variable? (3 marks)

Bike demand in the fall is the highest.

•Bike demand takes a dip in spring.

•Bike demand in year 2019 is higher as compared to 2018.

•Bike demand is high in the months from May to October.

•Bike demand is high if weather is clear or with mist cloudy while it is low when there is light rain or light snow.

•The demand of bike is almost similar throughout the weekdays.

•Bike demand doesn't change whether day is working day or not.

2. Why is it important to use drop_first=True during dummy variable creation? (2 mark)

Drop_first=True is important to use, as it helps in reducing the extra column created during dummy variable creation. Hence it reduces the correlations created among dummy variables. If you don't drop the first column then your dummy variables will be.

3. Looking at the pair-plot among the numerical variables, which one has the highest correlation with the target variable? (1 mark)

'temp' had the highest correlation coefficient of 0.63.

4. How did you validate the assumptions of Linear Regression after building the model on the training set? (3 marks)

The simple way to determine if this assumption is met or not is by creating a scatter plot x vs y. If the data points fall on a straight line in the graph, there is a linear relationship between the dependent and the independent variables, and the assumption holds.

5. Based on the final model, which are the top 3 features contributing significantly towards explaining the demand of the shared bikes? (2 marks)

- weathersit_Light_Snow(negative correlation).
- yr_2019(Positive correlation).
- temp(Positive correlation)

General Subjective Questions

1. Explain the linear regression algorithm in detail. (4 marks)

Linear regression is an algorithm that provides a linear relationship between an independent variable and a dependent variable to predict the outcome of future events. It is a statistical method used in data science and machine learning for predictive analysis.

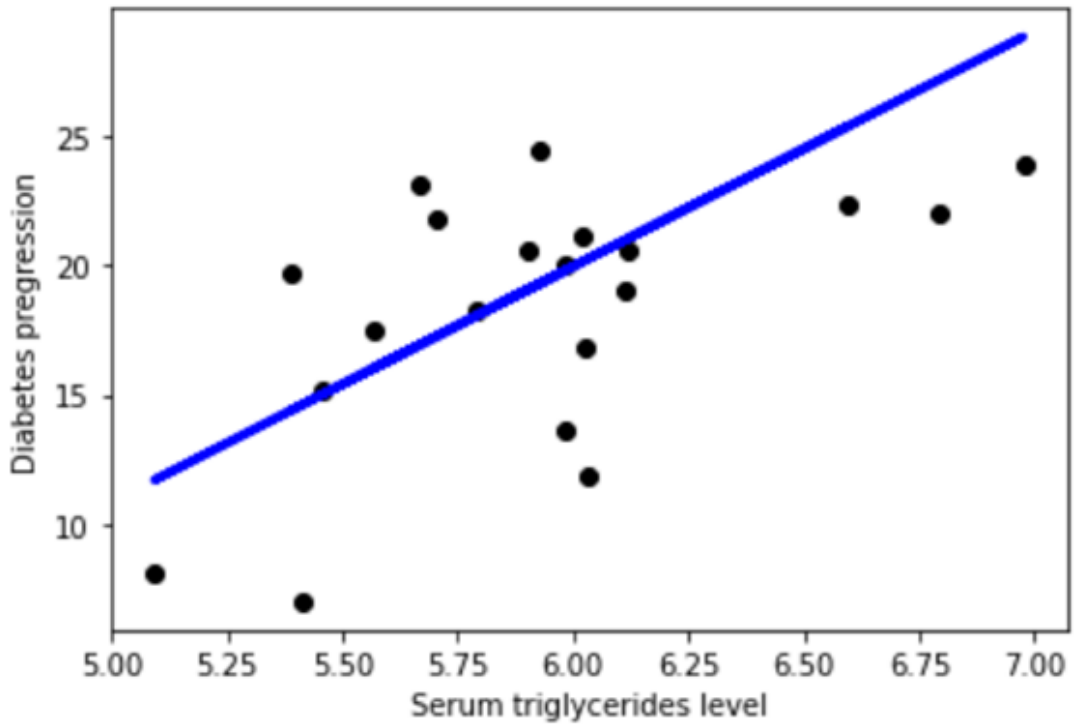
Linear regression is a supervised machine learning method that is used by the Train Using AutoML tool and finds a linear equation that best describes the correlation of the explanatory variables with the dependent variable. This is achieved by fitting a line to the data using least squares. The line tries to minimize the sum of the squares of the residuals. The residual is the distance between the line and the actual value of the explanatory variable. Finding the line of best fit is an iterative process.

The following is an example of a resulting linear regression equation:

$$y = b_0 + b_1x_1 + b_2x_2 + ...$$

In the example above, y is the dependent variable, and x1, x2, and so on, are the explanatory variables. The coefficients (b1, b2, and so on) explain the correlation of the explanatory variables with the dependent variable. The sign of the coefficients (+/-) designates whether the variable is positively or negatively correlated. b0 is the intercept that indicates the value of the dependent variable assuming all explanatory variables are 0.

In the following image, a linear regression model is described by the regression line y = 153.21 + 900.39x. The model describes the relationship between the dependent variable, Diabetes progression, and the explanatory variable, Serum triglycerides level. A positive correlation is shown. This example demonstrates a linear regression model with two variables. Although it is not possible to visualize models with more than three variables, practically, a model can have any number of variables.



A linear regression model helps in predicting the value of a dependent variable, and it can also help explain how accurate the prediction is. This is denoted by the R-squared and p-value values. The R-squared value indicates how much of the variation in the dependent variable can be explained by the

explanatory variable and the p-value explains how reliable that explanation is. The R-squared values range between 0 and 1. A value of 0.8 means that the explanatory variable can explain 80 percent of the variation in the observed values of the dependent variable. A value of 1 means that a perfect prediction can be made, which is rare in practice. A value of 0 means the explanatory variable doesn't help at all in predicting the dependent variable. Using a p-value, you can test whether the explanatory variable's effect on the dependent variable is significantly different from 0.

2. Explain the Anscombe's quartet in detail. (3 marks)

Anscombe's quartet comprises a set of four datasets, having identical descriptive statistical properties in terms of means, variance, R-squared, correlations, and linear regression lines but having different representations when we scatter plots on a graph. The datasets were created by the statistician Francis Anscombe in 1973 to demonstrate the importance of visualizing data and to show that summary statistics alone can be misleading.
The four datasets that make up Anscombe's quartet each include 11 x-y pairs of data. When plotted, each dataset seems to have a unique connection between x and y, with unique variability patterns and distinctive correlation strengths. Despite these variations, each dataset has the same summary statistics, such as the same x and y mean and variance, x and y correlation coefficient, and linear regression line.

Purpose of Anscombe's Quartet:
Anscombe's quartet is used to illustrate the importance of exploratory data analysis and the drawbacks of depending only on summary statistics. It also emphasizes the importance of using data visualization to spot trends, outliers, and other crucial details that might not be obvious from summary statistics alone.
Anscombe's Quartet Dataset
The four datasets of Anscombe's quartet.

I		II		III		IV	
x	y	x	y	x	y	x	y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89

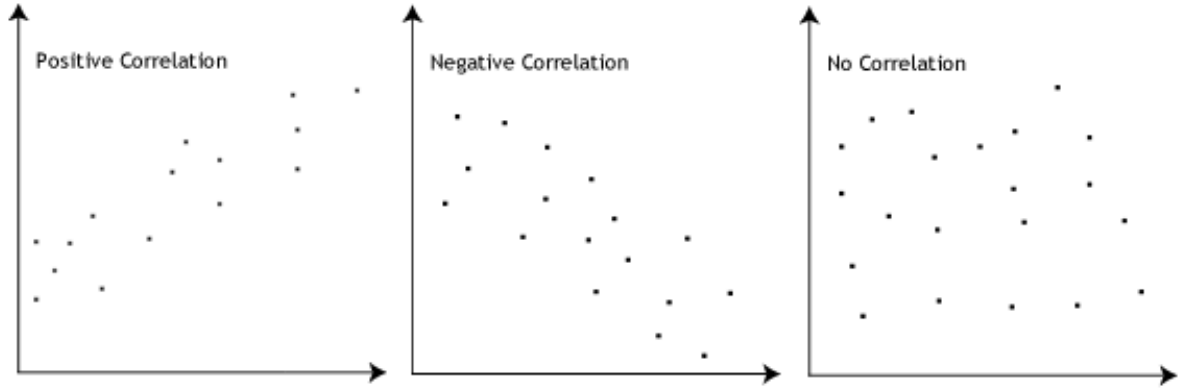
3. What is Pearson's R? (3 marks)

The Pearson product-moment correlation coefficient (or Pearson correlation coefficient, for short) is a measure of the strength of a linear association between two variables and is denoted by r. Basically, a

Pearson product-moment correlation attempts to draw a line of best fit through the data of two variables, and the Pearson correlation coefficient, r, indicates how far away all these data points are to this line of best fit (i.e., how well the data points fit this new model/line of best fit).

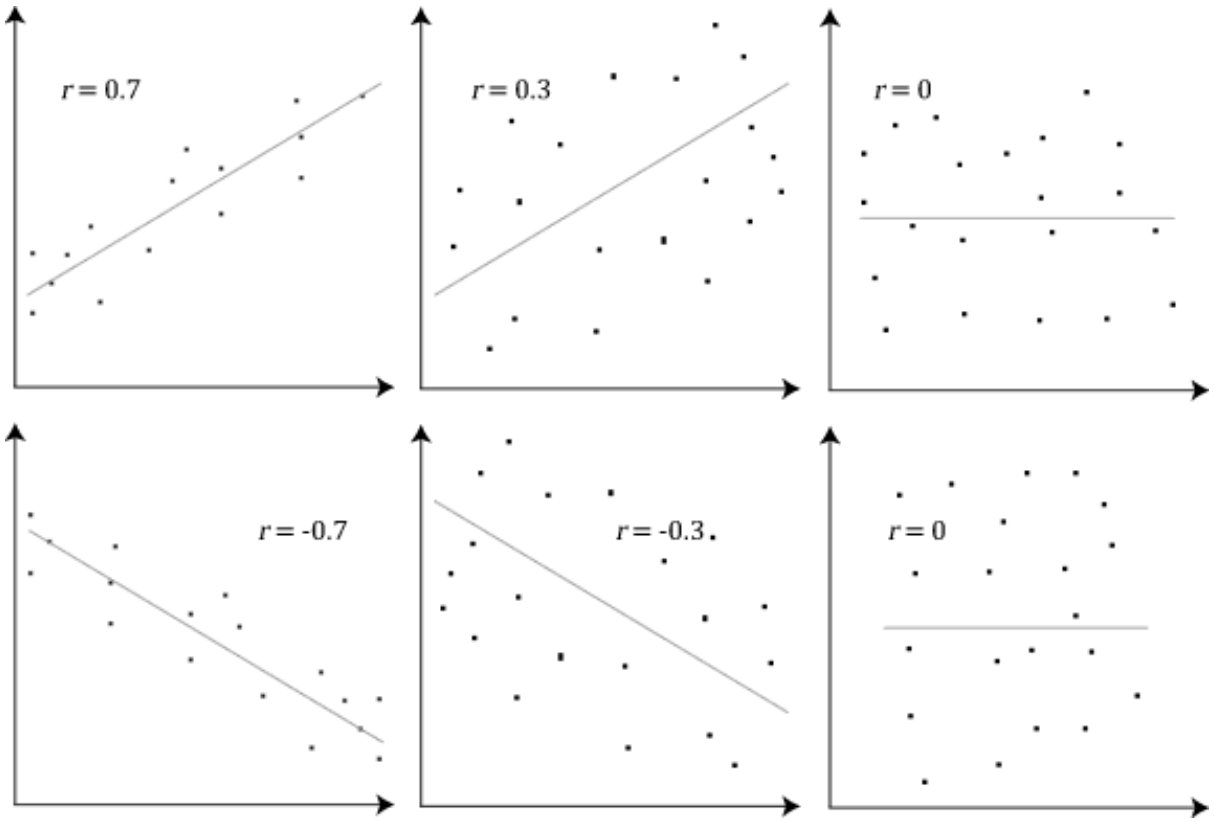
What values can the Pearson correlation coefficient take?

The Pearson correlation coefficient, r, can take a range of values from +1 to -1. A value of 0 indicates that there is no association between the two variables. A value greater than 0 indicates a positive association; that is, as the value of one variable increases, so does the value of the other variable. A value less than 0 indicates a negative association; that is, as the value of one variable increases, the value of the other variable decreases. This is shown in the diagram below:



How can we determine the strength of association based on the Pearson correlation coefficient?

The stronger the association of the two variables, the closer the Pearson correlation coefficient, r, will be to either +1 or -1 depending on whether the relationship is positive or negative, respectively. Achieving a value of +1 or -1 means that all your data points are included on the line of best fit – there are no data points that show any variation away from this line. Values for r between +1 and -1 (for example, r = 0.8 or -0.4) indicate that there is variation around the line of best fit. The closer the value of r to 0 the greater the variation around the line of best fit. Different relationships and their correlation coefficients are shown in the diagram below:



4. What is scaling? Why is scaling performed? What is the difference between normalized scaling and standardized scaling? (3 marks)

It is a step of data Pre-Processing which is applied to independent variables to normalize the data within a particular range. It also helps in speeding up the calculations in an algorithm.

Why?

Most of the times, collected data set contains features highly varying in magnitudes, units and range. If scaling is not done then algorithm only takes magnitude in account and not units hence incorrect modelling. To solve this issue, we have to do scaling to bring all the variables to the same level of magnitude. It is important to note that scaling just affects the coefficients and none of the other parameters like t-statistic, F-statistic, p-values, R-squared, etc.

Normalization/Min-Max Scaling:

- It brings all of the data in the range of 0 and 1. sklearn.preprocessing.MinMaxScaler helps to implement normalization in python.

$$\text{MinMax Scaling: } x = \frac{x - min(x)}{max(x) - min(x)}$$

Standardization Scaling:

- Standardization replaces the values by their Z scores. It brings all of the data into a standard normal distribution which has mean (μ) zero and standard deviation one (σ).

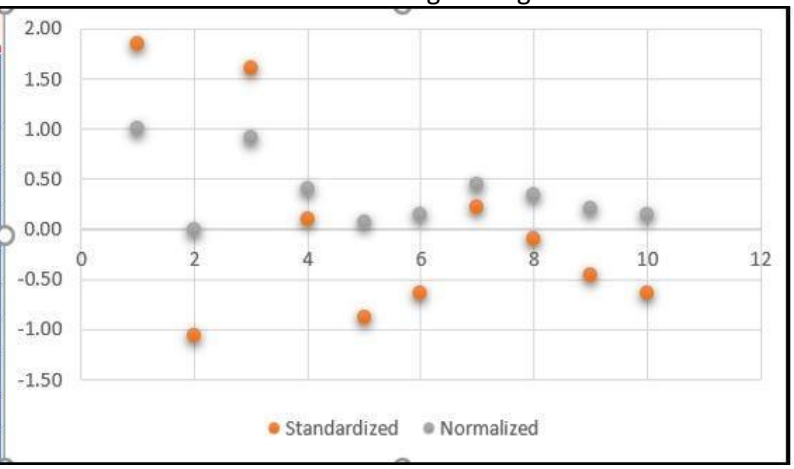
$$\text{Standardisation: } x = \frac{x - mean(x)}{sd(x)}$$

- sklearn.preprocessing.scale helps to implement standardization in python.

- One disadvantage of normalization over standardization is that it loses some information in the data, especially about outliers.

Example: Below shows example of Standardized and Normalized scaling on original values.

Original	Standardized	Normalized
59	1.85	1.000
11	-1.06	0.000
55	1.61	0.917
30	0.09	0.396
14	-0.88	0.063
18	-0.64	0.146
32	0.21	0.438
27	-0.09	0.333
21	-0.46	0.208
18	-0.64	0.146



4. You might have observed that sometimes the value of VIF is infinite. Why does this happen? (3 marks)

I know that VIF values have no upper limit, and that anything over 10 is usually bad news if you are trying to avoid multicollinearity especially for regression models such as multiple logistic regression.

However - most of what I have read suggests removing the offending infinity variable. Here in lies the problem.

I'm currently working on a dataset with nearly 2000 variables, and every single one has produced a VIF of infinity.

	VIF	Column
0	inf	Cog_Status
1269	inf	N-(7Z,10Z,13Z,16Z-docosatetraenoyl)-ethanolamide
1281	inf	C8 H4 O12 S2
1280	inf	C20 H37 N O4 + 11.435072
1279	inf	C20 H37 N O4
...	...	...
632	inf	C10 H9 N
631	inf	C6 H6 O4
630	inf	C5 H6 N2 O S
629	inf	C6 Cl2
1909	inf	C24:1-OH Sulfatide

1910 rows × 2 columns

I've been doing this on python:

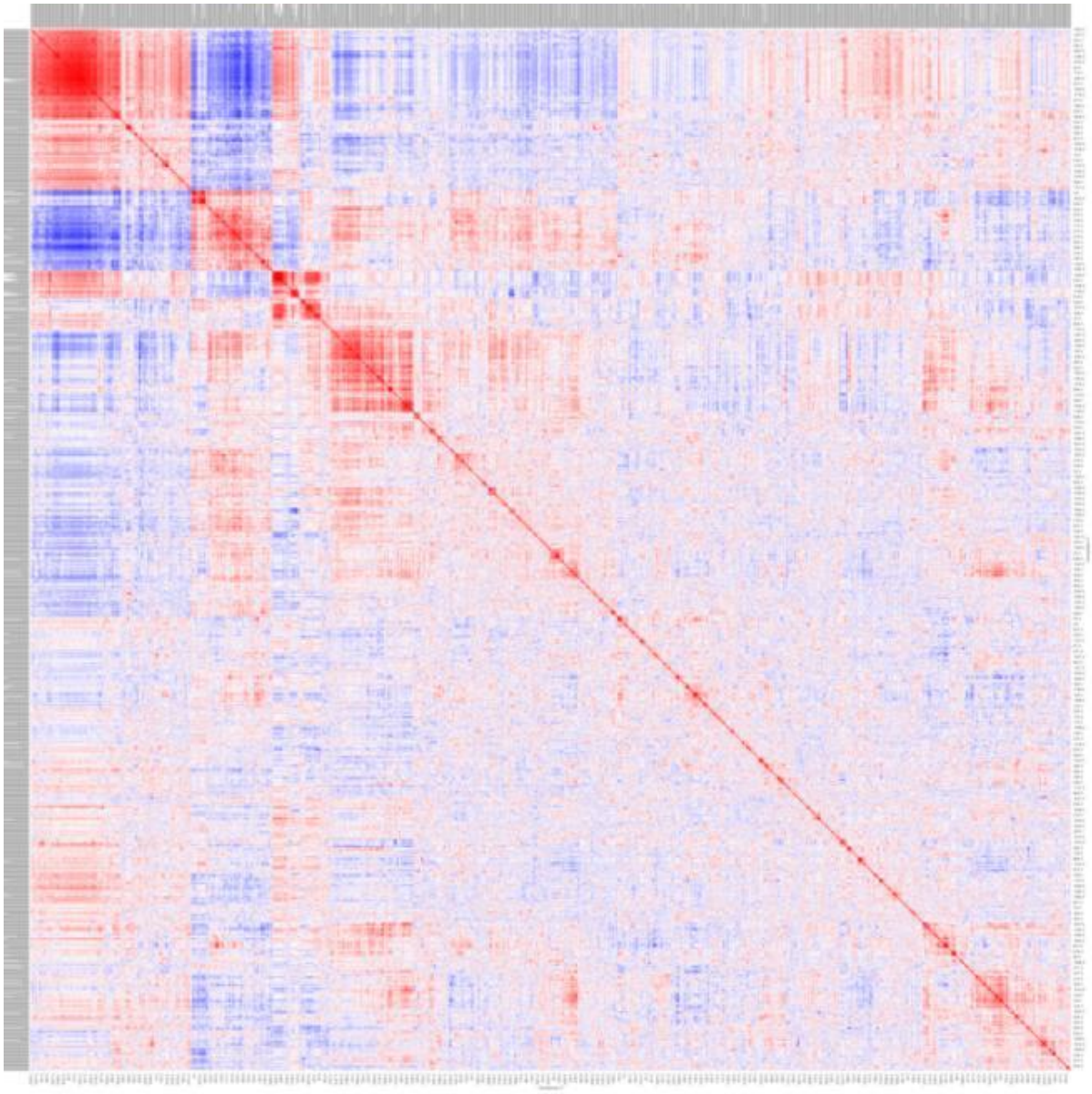
```
vif_info = pd.DataFrame()
vif_info['VIF'] = [variance_inflation_factor(df.values, i) for i in range(dif.shape[1])]
vif_info['Column'] = df.columns
vif_info.sort_values('VIF', ascending=False)
```

and I have tried various different methods, which have all produced the same results, so I'm relatively sure I haven't done something wrong there.

I have also tried log transforming the data and I still get the same result.

I'm not sure if it adds up though because when I produce something like a clustermap using a regular correlation matrix (see below) there seems to be clear specific variables that are highly correlated but not all variables seem to be this way.

In terms of statistical analysis I'm not sure how one would proceed with these results (or if theres something obvious that I've done wrong).



In [35]:
```
cor = ivar.corr()
cor
```

Out[35]:

Unnamed: 0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0	...	1900.0	1901.0	1902.0
1.0	1.000000	-0.139837	-0.089677	0.331120	-0.119784	0.297949	0.178494	-0.111110	-0.018429	0.179106	...	0.124847	-0.229134	0.026765
2.0	-0.139837	1.000000	0.129815	-0.597797	0.151429	-0.626286	-0.568345	0.546601	0.578889	-0.084949	...	-0.289782	0.543785	-0.392601
3.0	-0.089677	0.129815	1.000000	-0.279675	0.037574	-0.188329	-0.248785	-0.059310	-0.151189	0.150589	...	-0.070355	0.370002	-0.287399
4.0	0.331120	-0.597797	-0.279675	1.000000	-0.152801	0.860794	0.310430	-0.196935	-0.246762	0.043920	...	0.106049	-0.483341	0.256222
5.0	-0.119784	0.151429	0.037574	-0.152801	1.000000	-0.304050	-0.001173	0.077178	0.011016	-0.129274	...	-0.009628	0.103894	-0.000894
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1905.0	-0.100365	0.871629	0.190137	-0.542437	0.104284	-0.606138	-0.628903	0.551318	0.724621	-0.059022	...	-0.231635	0.618289	-0.266666
1906.0	-0.075118	-0.214525	-0.070217	-0.136798	0.075786	-0.204481	0.067481	-0.111291	-0.097138	0.131008	...	0.224309	0.038490	0.355991
1907.0	-0.145468	0.018287	-0.132273	-0.012518	-0.034888	-0.027710	0.034058	0.527983	-0.114257	0.233004	...	-0.044689	0.147953	-0.128202
1908.0	0.029896	0.040368	0.049334	0.011752	0.111803	0.009057	-0.121728	-0.137502	-0.016306	-0.063912	...	-0.191904	-0.211553	-0.132958
1909.0	-0.155793	0.849839	0.122780	-0.511310	0.087032	-0.566824	-0.610895	0.567448	0.735083	-0.028598	...	-0.210398	0.590943	-0.232542

1909 rows × 1909 columns

5. What is a Q-Q plot? Explain the use and importance of a Q-Q plot in linear regression. (3 marks)

Q-Q plots are also known as Quantile-Quantile plots. As the name suggests, they plot the quantiles of a sample distribution against quantiles of a theoretical distribution. Doing this helps us determine if a dataset follows any particular type of probability distribution like normal, uniform, exponential.

Before we dive into the Q-Q plot, let's discuss some of the probability distributions.

What are probability distributions?

In probability distributions, we represent data using charts where the x-axis represents the possible values of the sample and the y-axis represents the probability of occurrence.

There are various probability distribution types like Gaussian or Normal Distribution, Uniform distribution, Exponential distribution, Binomial distribution, etc.

In this blog, we will be looking into three types of distributions namely Normal, Uniform, and Exponential, and how we can identify them using a QQ plot.

- Normal distributions are the most popular ones. They are a probability distribution that peaks at the middle and decreases at the end of the axis. It is also known as a bell curve or Gaussian Distribution. As normal distributions are central to most algorithms, we will discuss this in detail below.

- Uniform distribution is a probability distribution type where the probability of occurrence of x is constant. For instance, if you throw a dice, the probability of any number is uniform.

- Exponential distributions are the ones in which an event occurs continuously and independently at a constant rate. It is commonly used to measure the expected time for an event to occur.

Why are probability distribution types important?

Probability distributions are essential in data analysis and decision-making. Some machine learning models work best under some distribution assumptions. Knowing which distribution we are working with can help us select the best model.

Hence understanding the type of distribution of feature variables is key to building robust machine learning algorithms.

Normal distributions

We regularly make the assumption of normality in our distribution as we perform statistical analysis and build predictive models. Machine learning algorithms like linear regression and logistic regression perform better where numerical features and targets follow a Gaussian or a uniform distribution.

It's an important assumption as normal distribution allows us to use the empirical rule of 68 – 95 – 99.7 and analysis where we can predict the percentage of values and how far they will fall from the mean.

In regression models, normality gains significance when it comes to error terms. You want the mean of the error terms to be zero. If the mean of error terms is significantly away from zero, it means that the features we have selected may not actually be having a significant impact on the outcome variable. It's time to review the feature selection for the model.

How Q-Q plots can help us identify the distribution types?

The power of Q-Q plots lies in their ability to summarize any distribution visually.

QQ plots is very useful to determine

- If two populations are of the same distribution
- If residuals follow a normal distribution. Having a normal error term is an assumption in regression and we can verify if it's met using this.
- Skewness of distribution

In Q-Q plots, we plot the theoretical Quantile values with the sample Quantile values. Quantiles are obtained by sorting the data. It determines how many values in a distribution are above or below a certain limit.

If the datasets we are comparing are of the same type of distribution type, we would get a roughly straight line. Here is an example of normal distribution.

Now let's look into how we can use the Q-Q plot in Python. To plot it, I have used the stats model library.

Below is the list of libraries that I have imported for this demonstration.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
import scipy.stats as stats
```

1. NumPy – to create data with normal, uniform, exponential distribution
2. Matplotlib & Seaborn to visualize various distributions
3. Statsmodels.api for Q-Q plot
4. SciPy.stats – for statistical functions

Now let’s use a Q-Q plot to analyze various sample distribution types

Normal Distribution

We can use the np.random.normal for this. In the below example, I am creating normally distributed

data with a mean 0 and a Standard deviation of 1.

```
np_normal = pd.Series(np.random.normal(0, 1, 200))
```

Let’s plot this using hist plot to see if it’s normally distributed.

```
sm.qqplot(np_normal,line='45',fit=True,dist=stats.norm)
```

Now let's plot the Q_Q plot for this dataset. If the datasets are distributed similarly, we would get a straight line.

We can see that since we are plotting the data with the theoretical quantiles of a normal distribution, we are getting almost a straight line

Uniform Distribution

Now let's try to plot uniformly distributed data and compare it with normal distribution.

Again, I am using the NumPy library to create sample uniformly distributed data.

```
np_uniform = pd.Series(np.random.uniform(-5, 5, 200))
```

Now let’s plot the Q-Q plot. Here we would plot the graph of uniform distribution against normal distribution.

```
sm.qqplot(np_uniform,line='45',fit=True,dist=stats.norm)
plt.show()
```

As you can see in the above Q-Q plot since our dataset has a uniform distribution, both the right and left tails are small and the extreme values in the above plot are falling close to the center. In a normal distribution, these theoretical extreme values will fall beyond 2 & -2 sigmas and hence the S shape of the Q-Q plot of a uniform distribution.

Exponential Distribution

If we plot a variable with exponential distribution with theoretical normal distribution, the graph would look like below. Code can be found in my git repository

Q-Q plots and skewness of data

Now let's see how we can determine skewness using a Q-Q plot

Q-Q plots can be used to determine skewness as well. If the see the left side of the plot deviating from the line, it is left-skewed. When the right side of the plot deviates, it's right-skewed.

Let's create a left-skewed distribution using skewnorm from the script library.

```
from scipy.stats import skewnorm
import matplotlib.pyplot as plt

skewness = -5 #Negative values are left skewed, positive values are right skewed.

random = skewnorm.rvs(a = skewness,loc=1000, size=50000)  #Skewnorm function

random = random - min(random)      #Shift the set so the minimum value is equal to zero.
random = random / max(random)      #Standadize all the vlues between 0 and 1.
random = random * 50000
random = pd.Series(random)
```

The distribution would look like below:

When we plot Q-Q plot, we should observe deviation on the left side

```
sm.qqplot(random,fit=True,line='45')
plt.show()
```

Similarly, a right-skewed distribution would look like below. We can observe deviation on the right side

Conclusion:

As you build your machine learning model, ensure you check the distribution of the error terms or prediction error using a Q-Q plot. If there is a significant deviation from the mean, you might want to check the distribution of your feature variable and consider transforming them into a normal shape.