Assignment-based Subjective Questions:

Q1 . From your analysis of the categorical variables from the dataset, what could you infer about their effect on the dependent variable?

Ans: The final model came up with the following summary:

	coef	std err	t	P>|t|	[0.025	0.975]
const	1992.3997	240.464	8.286	0.000	1519.952	2464.847
atemp	4921.8397	207.083	23.767	0.000	4514.977	5328.702
hum	-1593.8583	334.943	-4.759	0.000	-2251.931	-935.786
windspeed	-1456.3517	229.798	-6.338	0.000	-1907.842	-1004.862
'summer	897.2008	97.415	9.210	0.000	705.806	1088.596
winter	1219.8319	94.956	12.846	0.000	1033.269	1406.395
Light Rain	-1929.3430	234.998	-8.210	0.000	-2391.050	-1467.636
Mist	-447.2098	93.656	-4.775	0.000	-631.218	-263.202
Aug	615.8566	143.828	4.282	0.000	333.274	898.439
sep	1080.6249	144.226	7.493	0.000	797.259	1363.990
year_of_operation	1995.4127	72.479	27.531	0.000	1853.012	2137.814

By analysing the co-efficient it is clear if all other variables are kept constant , then :

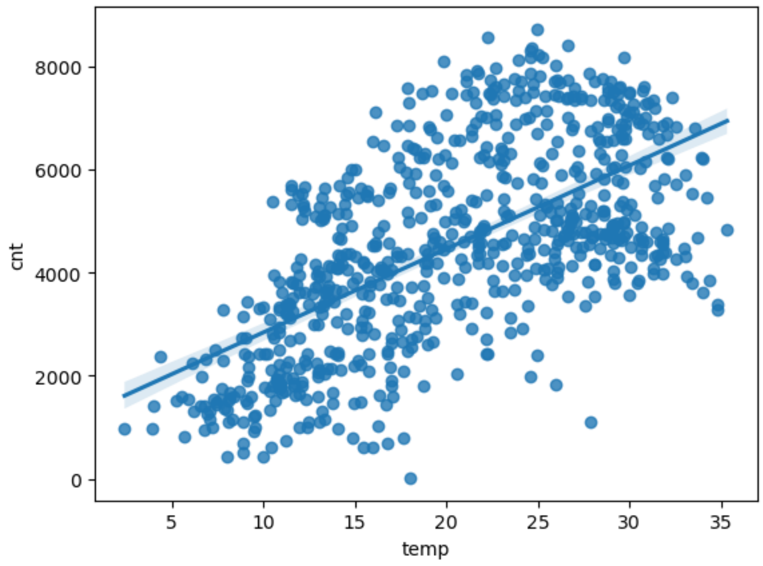
- ‘**const’:** Each day is going to by default attract 1992 approx. number of bike rentals despite every other contributing variables**.**
- **‘atemp’ :** Every unit rise of apparent temperature leads to an increase of approx. 4922 bike rentals counts.
- **‘hum’ :** Every unit increase in humidity decreases count of bike rentals by 1594 approx.
- **‘windspeed’ :** Every unit rise of windspeed there is a decrease of 1456 bike rentals.
- **‘summer’ :** Every summer there is an approx. increase of 897 count of bike rentals.
- **‘winter’ :** Every summer there is an increase of approx. 897 count of bike rentals.
- **‘Light Rain’ :** Every time there is light rain there is approx. 1929 decrease in count of bike rentals.
- **‘Mist’:** Mist reduces the bike rentals by approx. 447 counts.
- **‘Aug’:** Every August month the count of bike rentals increases by a count of approx. 616.
- **‘sep’:** Every September month bike rentals increases by a count of 1081 approx.
- **‘year_of_operation’:** As the year of operation increases there is an increase of approx. 1996 count of bike rentals.

Q2. Why is it important to use drop_first=True during dummy variable creation?

Ans: The pandas.get_dummies() method creates N dummy variables for every categorical variables having N levels . Out of the N new variables one variable is redundant which can be dropped for simplicity as the absence of all other dummy variables automatically ensures the presence of the dropped variable.

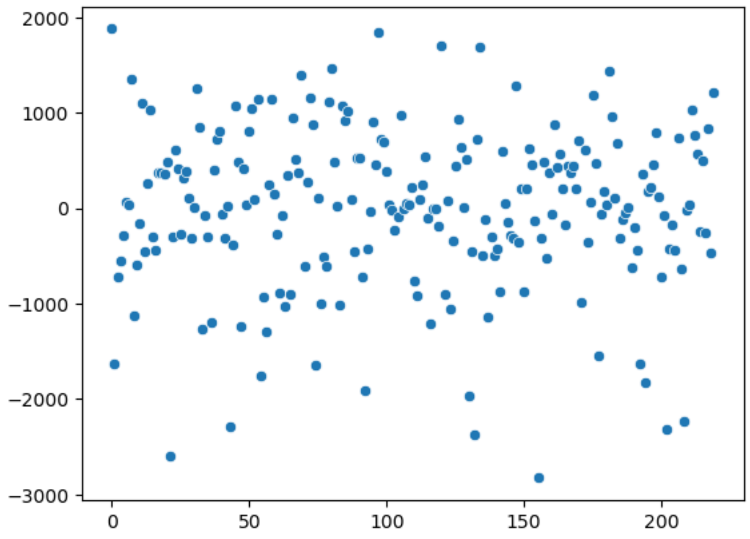
Q3. Looking at the pair-plot among the numerical variables, which one has the highest correlation with the target variable?

Ans: As per the scatter plot rental counts has the most linear relation with temperature.



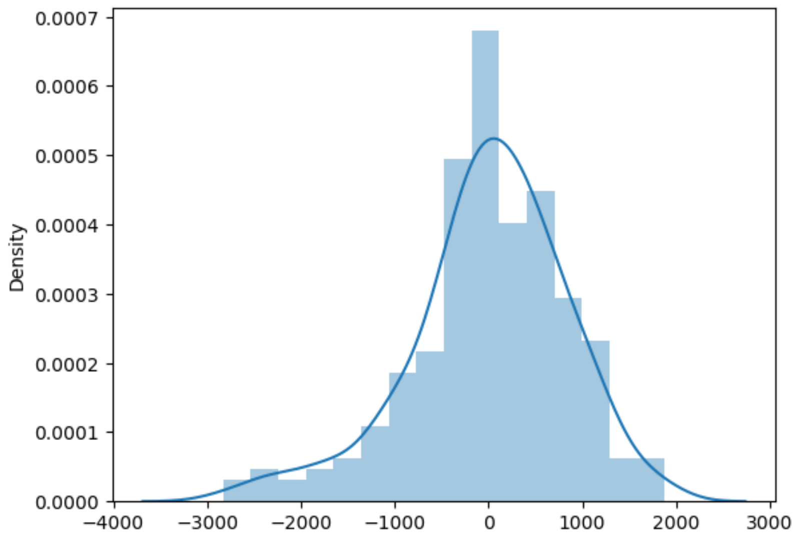
Q4. How did you validate the assumptions of Linear Regression after building the model on the training set?

Ans: <u>**Homoscedacity and no Multicollinearity:**</u> When the residuals(difference between predicted and actual target values) were plotted in a scatter plot ,it gets clear that there is no pattern to the errors and the errors are randomly distributed and there is constant variance. The graph below justifies that.



Normalized Errors:

When the residuals were plotted as a histogram we get a near normal distribution . It is evident from the graph below.



Both these indicate that the assumptions of Linear Regression are maintained in our model and Linear Regression is good fit to our data.

Q5. Based on the final model, which are the top 3 features contributing significantly towards explaining the demand of the shared bikes?

Ans: Based on the co-efficient obtained from the model it is evident that the top three contributing features are as follows:

a) **Apparent Temperature**: Every unit rise of apparent temperature leads to an increase of approx. 4922 bike rentals counts
b) **Humidity**: Every unit increase in humidity decreases count of bike rentals by 1594 approx.
c) **Year of operation**: As the year of operation increases there is an increase of approx. 1996 count of bike rentals.

General Subjective Questions:

Q1. Explain the linear regression algorithm in detail.

Ans: Below are the simplified steps for a linear regression algorithm:

1. **Data Collection**: Gather the data you want to analyse.
2. **Data Pre-processing**: Clean the data, handle missing values, and remove any outliers.
3. **Choose the Model**: Select the linear regression model that best fits your data.
4. **Split the Data**: Divide the data into training and testing sets.
5. **Train the Model**: Use the training data to fit the model to the data.
6. **Make Predictions**: Use the trained model to make predictions on the testing data.
7. **Evaluate the Model**: Assess the model's performance using metrics like mean squared error or R-squared.
8. **Use the Model**: Once the model is trained and evaluated, it can be used to make predictions on new data.
9. **Check the validity of assumptions of linear regression**: Once the predicted data is obtained, calculate the residuals and check whether the residuals are homoscedastic and follow normal distribution.

These are the basic steps involved in a simplified linear regression algorithm.

Own Implementation of gradient descent:

For multiple linear regression, the cost function looks like the following:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

In the equation above, x^i is the column vector of all the feature inputs of the i^{th} training example, **m** is the number of training examples, h_θ is the prediction from our regression model, and y^i is the column vector of the dependent variable.

Cost function:

```
def compute_cost(X, y, theta):
        return np.sum(np.square(np.matmul(X, theta) - y)) / (2 * len(y))
```

In the cost function above, the hypothesis or the predicted value is given by the following linear model

$$h_\theta(x^i) = \theta^T x^i$$

x^i here is the matrix consisting of the 1st row as the coefficient on intercept, i.e, 1 and the rest values are the values of the different features.

We need to minimise the cost function J(θ). One way to do this is to use the batch gradient decent algorithm. In batch gradient decent, the values are updated in each iteration:

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_0^{(i)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_1^{(i)}$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_2^{(i)}$$

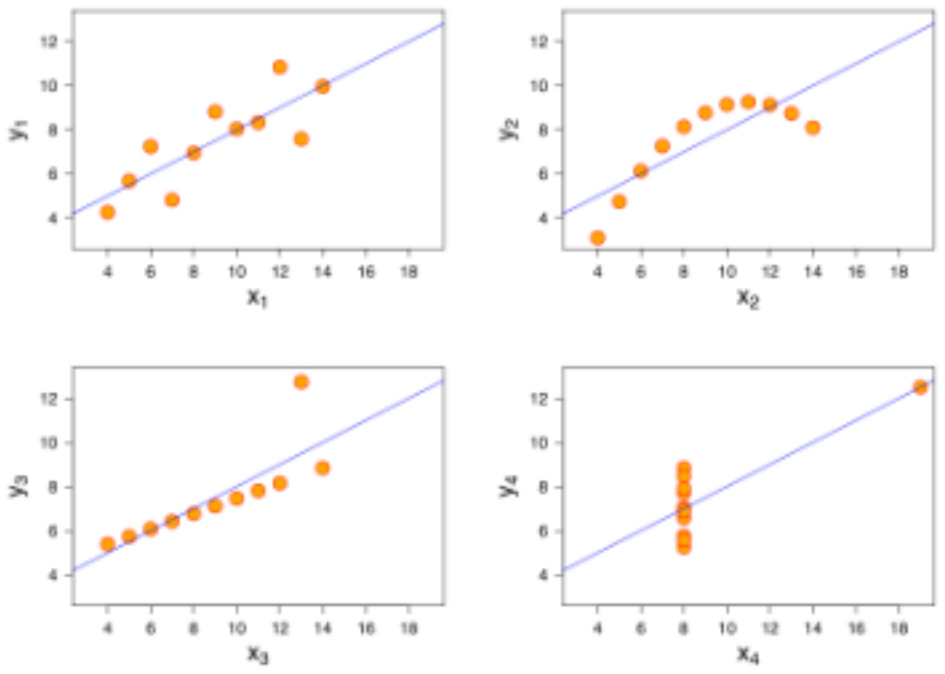
With each iteration, the parameter comes θ closer to the optimal value that will achieve the lowest cost J (θ) .

Gradient descent function:

```
def gradient_descent_multi(X, y, theta, alpha, iterations):
        theta = np.zeros(X.shape[1])
        m = len(X)
        gdm_df = pd.DataFrame( columns = ['Beta','cost'])
        for i in range(iterations):
                gradient = (1/m) * np.matmul(X.T, np.matmul(X, theta) - y))
                theta = theta - alpha * gradient
                cost = compute_cost(X, y, theta)
                gdm_df.loc[i] = [theta,cost]
        return gdm_df
```

The code above will iterate and update the cost function J (θ) .

Q2. Explain the Anscombe's quartet in detail.
Ans:



<u>Anscombe's Quartet</u>

The above quartet is called as Anscombe's Quartet. Its is evident from the graph that all four data types have the exact same best fit line as given by linear regression algorithm. But while the first graph seems to do a decent job at explaining the data, the rest have obviously fooled the linear regression.
In the second graph it is clear that linear regression cant handle data of polynomial types or ones that don't possess a linear relationship.
The next two graphs clearly show the sensitivity of linear regression to outliers. Had the outliers been absent we would have got a very good fitting line through the data.
Hence we should always have a good look at the data before applying linear regression. Anscombe's quartet is a good reference to watch out for.

Q.3. What is Pearson's R?

Ans: Pearson's R is a co-efficient which is used to check the corelation between two variables.
If two variables are corelated it is very much possible that it is not a linear one. Pearson's R very much helps in verifying the same. If there is a linear relationship then Pearson's R gives a value close to 1. For polynomial relationship between variables, the value of Pearson's R drops as the degree of polynomial increases.

For $y = x^3$,
Pearson's R $\approx$ 0.91

While for $y = x^{10}$,
Pearson's R $\approx$ 0.66

Q.4. What is scaling? Why is scaling performed? What is the difference between normalized scaling and standardized scaling?

Ans: Scaling is a step in data pre-processing which helps to normalize the data and aids in faster convergence of the algorithm.

Why: The data often obtained has wide variety of values ,scales and ranges. In order to bring all the variables into a same level of magnitude and account for their varying units scaling is done so that modelling is correct and proper inferences can be made from the model.

Note: Scaling doesn’t affect the projection capability of a model but does affect the prediction capacity.

Normalized Scaling: It brings all the data n the range of 0 to 1. It is also called as MinMaxScaler in sklearn.

$$\text{MinMaxScaling (x)} = \frac{\text{x}-x_{min}}{x_{max}-x_{min}}$$

Standardized Scaling: It brings all the data into a standard normal distribution with mean(□) =0 and std_dev(σ) =1. It is also called as the StandardScaler in sklearn.

$$\text{Standardization (x)} = \frac{\text{x}-\square}{\sigma}$$

Q.5. You might have observed that sometimes the value of VIF is infinite. Why does this happen?

Ans: VIF is measure of how well a predictor variable is a linear regression of all other predictor variables. Higher the value of VIF higher is the correlation and chances of multicollinearity. In such cases the coefficients prescribed by the model become unreliable.

A variable having VIF value of infinite means that there exists a perfect correlation between the variable and other predictor variables.

The best remedy would be drop the variable and analyse the model again in such a scenario.

Q.6. What is a Q-Q plot? Explain the use and importance of a Q-Q plot in linear regression.

Ans: A Q-Q plot, short for quantile-quantile plot, is a scatterplot that compares the quantiles of two distributions. One distribution is usually the observed data, and the other is a theoretical or reference distribution, such as the normal distribution. The idea is to see how well the data fit the expected distribution by checking if the points lie on or near a straight line.

Use and importance of Q-Q Plot:

A Q-Q plot can be used in regression models to check some of the assumptions that are required for valid inference. For example, you can use a Q-Q plot to check if the residuals of the model are normally distributed, which is an assumption for many parametric tests and confidence intervals. You can also use a Q-Q plot to check if the residuals have a constant variance, which is an assumption for the homoscedasticity of the model. To do this, you need to create a Q-Q plot for the residuals of the model and compare them with the normal distribution.