**Python Programming**

Programming Questions:

**Data Analysis With Python**

**Data Analysis With MySQL**

**Data Analysis With Tableau**

**Statistics**

**Machine Learning**

1**. How does multicollinearity affect the linear regression?**

Ans: Multicollinearity causes a problem as it is against the basic assumption of linear regression. The presence of multicollinearity does not affect the predictive capability of the model. So, if you just want predictions, the presence of multicollinearity does not affect your output. However, if you want to draw some insights from the model and apply them in, let’s say, some business model, it may cause problems.

2. **Why use *Root Mean Squared Error (RMSE)* instead of *Mean Absolute Error (MAE)?***

Ans: This depends on your loss function. In many circumstances it makes sense to give more weight to points further away from the mean:

* If being off by 10 is more than *twice* as bad as being off by 5. In such cases, RMSE is a more appropriate measure of error
* If being off by 10 is just twice as bad as being off by 5, then MAE is more appropriate.

3. **Explain One-hot encoding and Label Encoding. How do they affect the dimensionality of the given dataset?**

Ans: One-hot encoding is the representation of categorical variables as binary vectors. Label Encoding is converting labels/words into numeric form. Using one-hot encoding increases the dimensionality of the data set. Label encoding doesn’t affect the dimensionality of the data set. One-hot encoding creates a new variable for each level in the variable whereas, in Label encoding, the levels of a variable get encoded as 1 and 0.

4. **Name some *Evaluation Metrics* for Regression Model and when you would use one?**

Ans: **Mean absolute error (MAE)**: calculates the absolute difference between actual and predicted values. It can be used when we want that our model be robust to outliers, but this metric has the disadvantage of not being differentiable so we can't use it if we want to apply optimizers like *Gradient descent*.

**Mean squared error (MSE)**: calculates the squared difference between actual and predicted value. We can use this metric if we want to give bigger penalization to *outliers* and apply optimizers who require differentiation. MSE is a differentiable function that makes it easy to perform mathematical operations in comparison to a non-differentiable function like MAE.

**Root mean squared error (RMSE)**: This is simply the square root of mean squared error. This metric is not so robust to outliers as the *mean absolute error* but it has the advantage to be differentiable so we can use it if we want to apply gradient descent to minimize losses.

When to use one depends on your loss function:

* **When to use MAE**: If being off by ten is just twice as bad as being off by 5. it is better to use the MAE if you don't want your performance metric to be overly sensitive to outliers.
* **When to use RMSE**: In many circumstances, it makes sense to give more weight to points further away from the mean - that is, being off by 10 is more than twice as bad as being off by 5. In such cases, RMSE is a more appropriate measure of error.

5. **What’s the difference between batch gradient descent and stochastic gradient descent?**

Ans: The main difference between batch gradient descent and stochastic gradient descent is that with batch gradient descent, the gradient is calculated using the entire dataset, while with stochastic gradient descent, the gradient is calculated using a single data point. This means that batch gradient descent is more computationally expensive, but it also means that the gradient is more accurate

6. **What are some common cases where gradient descent may fail to converge?**

Ans: There are a few reasons why gradient descent might fail to converge. One reason is if the function being optimized is not convex. If the function has multiple local minima, then gradient descent might get stuck in a local minimum that is not the global minimum. Another reason is if the step size is not chosen properly. If the step size is too large, then gradient descent might not converge. If the step size is too small, then gradient descent might converge slowly.

7. **Why is gradient descent needed when training a model?**

Ans: Gradient descent is an optimization algorithm used to find the values of parameters (weights) that minimize a cost function. When training a model, gradient descent is used to find the values of the weights that minimize the error between the predicted values and the actual values.

**8. Is it possible to apply gradient descent to solve non-convex optimization problems?**

Ans: Yes, it is possible to apply gradient descent to solve non-convex optimization problems, but it is important to keep in mind that doing so may lead to sub-optimal solutions. In general, gradient descent is more likely to find a local optimum when applied to non-convex optimization problems as opposed to the global optimum.

**Deep Learning**