Jr Data Scientist Evaluation [Solution]

Bonus Points Q&A

1. Write about any difficult problem that you solved. (According to us difficulty - is something which 90% of people would have only 10% probability in getting a similarly good solution).

- The problem difficulty was that we were unable to collect data for our minor project in college, which was on FastTag Fraud Detection. So, I manually built data as a prototype using (GEN AI) tools and then uploaded it on Kaggle, where I received 8K+ views and 1K+ downloads. why? I created this minor project. Because one day my guide told us that there is fraud taking place at toll gates using FastTag ID, where a mini car's FastTag ID is attached to a SUV car, and the SUV car pays the mini car's fee. So, to overcome this challenge, I proposed a strategy that involved creating synthetic data as a prototype. & Why did I create this data? Because officials stated that the data was confidential and weren't allowed to be given.

2. Explain back propagation and tell us how you handle a dataset if 4 out of 30 features have null values more than 40 percentage

- I) Back propagation is basically fundamental algorithm for artificial neural network. it is supervised learning technique that adjusts the weights of the network to minimize the Error between the predicted output and actual target output.
- ii) To handle the dataset if null values are more than 40 percent then we must drop the features if they are not important. if the features are important then (Fillna) the missing values like common imputation methods (mean, KNN imputation etc.) or else we can use data augmentation if the data is small (like generating synthetic data) to fill missing values or we can balance the data.

Problem Part 1:

Write a regex to extract all the numbers with orange color background from the text below in italics.

Steps:

- Import [re] Module (built-in library for working with regular expressions).
- 2. I defined Input Text (Which given in file). where (orders and errors are list of objects)
- 3. I used [re.findall] to extract numbers, and I give regular expression pattern [r'"id":(\d+)|"code":(\d+)], it will extract the orange color background numbers
- 4. Flatten the list of tuples and filter out empty strings.
- 5. Print the extracted numbers.

Part 1 Solution Link
: https://colab.research.google.com/drive/1AvP5k1_MnbxPkELWuaggw_ANQq6n053r?usp=sharing

Problem Part 2: The data set that contains the history of customer booking in a hotel.

[Predicts the customer who is going to be checked in.]

Part 2 Solution
Link: https://colab.research.google.com/drive/1v4AanxpoM4Fs4gbKZOU238Yq3X3S8ZCP?usp=sharing

Deployment Link: https://hotelcheckin.streamlit.app/

Documentation [Part_2_JrDataScientist]

Steps:

→ 1. Connected to Google Drive in a Colab Notebook

→ 2. Imported Necessary Libraries

→ 3. Train & Test Data has been Loaded

→ 4. Exploratory Data Analysis (EDA)

- head (), tail (), shape (), info (), dtypes () methods are used to examine the dataset structure and summary.
- Missing values are identified using. isnull (). sum ().
- Unique values for each column are determined with. nunique ().
- Describe () is used to compute statistical summaries for numerical columns.
- Pearson correlation between PersonsNights and RoomNights is calculated and visualized using a scatter plot.
- Boxplots are used to analyze the distribution and outliers of Age and PersonsNights.

→ Data Cleaning

- Missing Values: The Age column is imputed with the mean value.
- Invalid Data: Rows with invalid ages (Age < 0) are removed.
- Irrelevant Columns: A list of unnecessary columns (e.g., ID, Nationality, SR etc.) is dropped from both datasets.
- Visualizes the distribution of numerical features using histograms.

→ Feature Engineering

- Outlier Handling: Outliers are capped using the Interquartile Range (IQR) method for numerical columns.
- Feature Scaling: Numerical columns are scaled using MinMaxScaler to normalize values between 0 and 1.
- The scaler is saved using joblib for future reuse.
- Derived Features: New features (TotalBookings, CheckInRate, WillCheckIn) are created based on existing data. Features related to Bookings are dropped after calculating the derived features.

- Categorical Encoding: The DistributionChannel column is one-hot encoded using OneHotEncoder.

→ Train-Test Split

- The training dataset is split into features (X_train) and target (y_train).
- Similarly, the testing dataset is split into X_test and y_test.

→ Handling Class Imbalance

- Class weights are computed using compute_class_weight to handle imbalanced target classes (WillCheckIn).

→ Standardization

- Features are standardized using StandardScaler. The scaler is saved for later use.

→ Model Building

- A neural network is built using tensorflow.keras.Sequential. The architecture includes:
    - Input layer: Explicitly set to match the feature size.
    - Three hidden layers: Each with 64 or 32 units, ReLU activation, batch normalization, dropout, and L2 regularization.
    - Output layer: A single unit with a sigmoid activation for binary classification.
- Optimizer: Adam with a learning rate of 0.001.
- Loss function: binary_crossentropy.

→ Training the Model

- The model is trained with:
    - Early stopping to prevent overfitting.
    - A learning rate scheduler to reduce the learning rate after 10 epochs.
    - Class weights to account for the imbalanced dataset.
- Validation split: 20% of the training data is used for validation.

→ Evaluation

- **Results:**

Test MAE: 0.0100

Test RMSE: 0.1000

Test Accuracy: 0.9900

- Metrics: Accuracy, Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE) are calculated on the test dataset.
- Visualization: Training and validation accuracy and loss curves are plotted to analyze model performance.

→ Saving Artifacts

- The following artifacts are saved for later use:
  - scaler.pkl: Min-Max Scaler for scaling features.
  - standard_scaler.pkl: Standard Scaler for standardizing features.
  - encoder.pkl: One-hot encoder for encoding categorical features.
  - my_model.keras: Trained neural network model.

→ Deployment Streamlit

- App Title and Description
- Loading Pre-trained Models
- User Input Form
- Feature Processing
- Prediction

→ How to run?

- Download Folder from GitHub & Open VScode
- Pip install requirements.txt
- Streamlit run app.py