Snowflake
DSA-C03
SnowPro Advanced: Data Scientist Certification Exam

For More Information – Visit link below:
https://www.examsempire.com/

Product Version

1. **Up to Date products, reliable and verified.**
2. **Questions and Answers in PDF Format.**

Latest Version: 6.0

Question: 1

A data scientist is building a model in Snowflake to predict customer churn. They have a dataset with features like 'age', 'monthly_spend', 'contract_length', and 'complaints'. The target variable is 'churned' (0 or 1). They decide to use a Logistic Regression model. However, initial performance is poor. Which of the following actions could MOST effectively improve the model's performance, considering best practices for Supervised Learning in a Snowflake environment focused on scalable and robust deployment?

A. Increase the learning rate significantly to speed up convergence during training.
B. Implement feature scaling (e.g., StandardScaler or MinMaxScaler) on numerical features within Snowflake, before training the model. Leverage Snowflake's user-defined functions (UDFs) for transformation and then train the model.
C. Reduce the number of features by randomly removing some columns, as this always prevents overfitting.
D. Fit a deep neural network with numerous layers directly within Snowflake without any data preparation, as this will automatically extract complex patterns.
E. Ignore missing values in the dataset as the Logistic Regression model will handle it automatically without skewing the results.

Answer: B

Explanation:
Feature scaling is crucial for Logistic Regression. Features with different scales can disproportionately influence the model's coefficients. Snowflake UDFs allow for scalable data transformation within the platform. Increasing the learning rate excessively can lead to instability. Randomly removing features can remove important information. Deep neural networks require substantial tuning and resources and aren't always the best starting point and can have issues deploying inside of Snowflake. Ignoring missing values will negatively impact performance.

Question: 2

A data scientist is tasked with predicting house prices using Snowflake. They have a dataset stored in a Snowflake table called 'HOUSE PRICES' with columns such as 'SQUARE FOOTAGE, 'NUM BEDROOMS, 'LOCATION_ID, and 'PRICE. They choose a Random Forest Regressor model. Which of the following steps is MOST important to prevent overfitting and ensure good generalization performance on unseen data, and how can this be effectively implemented within a Snowflake-centric workflow?

A. Increase the number of estimators (trees) in the Random Forest to the maximum possible value to capture all potential patterns, without cross validation.
B. Tune the hyperparameters of the Random Forest model (e.g., 'max_deptm, 'n_estimators') using cross-validation. You can achieve this by splitting the 'HOUSE PRICES table into training and validation

sets using Snowflake's 'QUALIFY clause or temporary tables, then train and evaluate the model within a loop or stored procedure.
C. Train the Random Forest model on the entire 'HOUSE PRICES table without splitting into training and validation sets, as this will provide the model with the most data.
D. Randomly select a small subset of the features (e.g., only use 'SQUARE FOOTAGE and 'NUM BEDROOMS) to simplify the model and prevent overfitting.
E. Eliminate outliers without understanding the data properly to reduce noise.

Answer: B

Explanation:
Hyperparameter tuning with cross-validation is crucial to prevent overfitting. By splitting the data into training and validation sets, we can evaluate the model's performance on unseen data and adjust the hyperparameters accordingly. Snowflake's 'QUALIFY' clause and temporary tables can be used to efficiently manage these splits. Using a maximum number of estimators without validation is prone to overfitting. Training on the entire dataset without validation provides no indication of generalization performance. Randomly selecting a subset of features may remove important predictors and eliminating outliers without proper investigation can skew your data and reduce the efficacy of the model.

Question: 3

A financial institution aims to detect fraudulent transactions using a Supervised Learning model deployed in Snowflake. They have a dataset with transaction details, including amount, timestamp, merchant category, and customer ID. The target variable is 'is_fraudulent' (0 or 1). They are considering different Supervised Learning algorithms. Which of the following algorithms would be MOST suitable for this fraud detection task, considering the need for interpretability, scalability, and the potential for imbalanced classes, and what specific strategies can be employed within Snowflake to handle the class imbalance?

A. K-Nearest Neighbors (KNN), because it is simple to implement and doesn't require extensive training.
B. Support Vector Machine (SVM) with a radial basis function (RBF) kernel, as it can capture complex non-linear relationships without concern for interpretability.
C. Decision Tree or Random Forest, combined with techniques like oversampling the minority class (fraudulent transactions) within Snowflake using SQL or UDFs to balance the dataset before training. These models provide reasonable interpretability and can handle non-linear relationships effectively.
D. Linear Regression, because it's computationally efficient and easy to understand, even though fraud detection is a classification problem.
E. Naive Bayes, because it requires no hyperparameter tuning and works well on numerical data.

Answer: C

Explanation:
Decision Trees and Random Forests are well-suited for fraud detection due to their ability to handle non-linear relationships and provide interpretability. The class imbalance problem (where fraudulent transactions are much rarer than legitimate ones) is a common challenge in fraud detection. Oversampling the minority class or using techniques like SMOTE within Snowflake before training can

significantly improve the model's performance. KNN is not well-suited for high-dimensional data or imbalanced datasets. SVM can be computationally expensive and lacks interpretability. Linear Regression is inappropriate for a classification problem. Naive Bayes makes strong independence assumptions that may not hold in fraud detection scenarios.

Question: 4

A data scientist is tasked with identifying customer segments for a new marketing campaign using transaction data stored in Snowflake. The transaction data includes features like transaction amount, frequency, recency, and product category. Which unsupervised learning algorithm would be MOST appropriate for this task, considering scalability and Snowflake's data processing capabilities, and what preprocessing steps are crucial before applying the algorithm?

A. K-Means clustering, after standardizing numerical features (transaction amount, frequency, recency) and using one-hot encoding for product category. This is highly scalable within Snowflake using UDFs and SQL.
B. Hierarchical clustering, using the complete linkage method and Euclidean distance. No preprocessing is necessary, as hierarchical clustering can handle raw data.
C. DBSCAN, using raw data without any scaling or encoding. The algorithm's density-based nature will automatically handle the varying scales of the features.
D. Principal Component Analysis (PCA) followed by K-Means. This reduces dimensionality and then clusters, improving the visualization of the cluster.
E. K-Means clustering, after applying min-max scaling to numerical features and converting categorical features to numerical representation. The optimal 'k' (number of clusters) should be determined using the elbow method or silhouette analysis.

Answer: E

Explanation:
K-Means clustering is a suitable algorithm for customer segmentation due to its scalability and efficiency. Min-max scaling is important to ensure that features with larger ranges don't dominate the distance calculations. Converting categorical features to numerical representation (e.g., one-hot encoding) is also essential for K-Means. The elbow method or silhouette analysis helps determine the optimal number of clusters. Options A, B, C, and D have flaws related to scaling requirements, algorithm suitability for large datasets, or lack of pre-processing.

Question: 5

You are analyzing website clickstream data stored in Snowflake to identify user behavior patterns. The data includes user ID, timestamp, URL visited, and session ID. Which of the following unsupervised learning techniques, combined with appropriate data transformations in Snowflake SQL, would be most effective in discovering common navigation paths followed by users? (Choose two)

A. Association rule mining (e.g., Apriori) applied directly to the raw URL data to find frequent itemsets of URLs visited together within the same session. No SQL transformations are required.

B. Sequence clustering using time-series analysis techniques (e.g., Hidden Markov Models), after transforming the data into a sequence of URLs for each session using Snowflake's LISTAGG function ordered by timestamp.
C. K-Means clustering on features extracted from the URL data, such as the frequency of visiting specific domains or the number of pages visited per session. This requires feature engineering using SQL.
D. DBSCAN clustering on the raw URL data, treating each URL as a separate dimension. This will identify URLs that are frequently visited by many users.
E. Principal Component Analysis (PCA) to reduce the dimensionality of the URL data, followed by hierarchical clustering. This will group similar URLs together.

Answer: B,C

Explanation:
Sequence clustering is appropriate for identifying navigation paths because it considers the order of URLs visited within a session. Using Snowflake's LISTAGG function allows for creating the required sequential data. K-Means clustering can also be effective if relevant features are engineered from the URL data (e.g., frequency of visiting specific domains). Association rule mining is less suitable for identifying navigation paths as it focuses on co-occurrence rather than sequence. PCA followed by hierarchical clustering and DBSCAN are not well-suited for identifying sequential navigation paths from clickstream data. Option 'A' is incorrect because association rule mining directly on raw URL data is unlikely to be effective without prior sequence extraction. Option 'D' and 'E' are not suitable for this type of analysis.

Question: 6

A data scientist is using Snowflake to perform anomaly detection on sensor data from industrial equipment. The data includes timestamp, sensor ID, and sensor readings. Which of the following approaches, leveraging unsupervised learning and Snowflake features, would be the MOST efficient and scalable for detecting anomalies, assuming anomalies are rare events?

A. Use a Support Vector Machine (SVM) with a radial basis function (RBF) kernel trained on the entire dataset to classify data points as normal or anomalous. Implement the SVM model as a Snowflake UDF.
B. Implement an Isolation Forest model. Train the Isolation Forest model on a representative sample of the sensor data and create UDF to score each row in snowflake.
C. Use K-Means clustering to group sensor readings into clusters and identify data points that are far from the cluster centroids as anomalies. No model training necessary.
D. Calculate the moving average of sensor readings over a fixed time window using Snowflake SQL and flag data points that deviate significantly from the moving average as anomalies. No ML model needed.
E. Apply Autoencoders to the sensor data using a Snowflake external function. Data points are considered anomalous if the reconstruction error from the autoencoder exceeds a certain threshold.

Answer: B

Explanation:
Isolation Forest is specifically designed for anomaly detection and performs well with high-dimensional data. Because anomalies are defined as ‘few and different,’ Isolation Forest builds an ensemble of trees

and isolates observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature. Anomalies require fewer splits to be isolated and consequently have a shorter path length in the tree, where this path length is the measurement of 'solation'. It is scalable and well-suited for large datasets within Snowflake, especially when integrated via a UDF.SVM is computationally intensive. K-Means only effective when anomalies are caused by shifted data, no individual outliers. Calculationg the moving average is quick to compute, and has a faster throughput, but is extremely sensitive to outliers. Option A is computationally expensive and may not scale well. Options C is suitable for a high level initial assessment, and not for accuracy. Option E, Autoencoders would have difficulty training and might not perform well.

Question: 7

A data scientist is building a linear regression model in Snowflake to predict customer churn based on structured data stored in a table named 'CUSTOMER DATA'. The table includes features like 'CUSTOMER D', 'AGE, 'TENURE MONTHS', 'NUM PRODUCTS', and 'AVG MONTHLY SPEND'. The target variable is 'CHURNED' (1 for churned, 0 for active). After building the model, the data scientist wants to evaluate its performance using Mean Squared Error (MSE) on a held-out test set. Which of the following SQL queries, executed within Snowflake's stored procedure framework, is the MOST efficient and accurate way to calculate the MSE for the linear regression model predictions against the actual 'CHURNED values in the 'CUSTOMER DATA TEST table, assuming the linear regression model is named 'churn _ model' and the predicted values are generated by the MODEL APPLY() function?

A.

○ ```sql CREATE OR REPLACE PROCEDURE CALCULATE_MSE() RETURNS FLOAT LANGUAGE SQL AS $$ DECLARE total_squared_error FLOAT := 0.0; n INTEGER := 0; cursor1 CURSOR FOR SELECT CHURNED, MODEL_APPLY(churn_model, (AGE, TENURE_MONTHS, NUM_PRODUCTS, AVG_MONTHLY_SPEND)) AS predicted FROM CUSTOMER_DATA_TEST; BEGIN FOR record IN cursor1 DO total_squared_error := total_squared_error + POWER((record.CHURNED - record.predicted), 2); n := n + 1; END FOR; RETURN total_squared_error / n; END; $$; ```

B.

○ ```sql CREATE OR REPLACE PROCEDURE CALCULATE_MSE() RETURNS FLOAT LANGUAGE SQL AS $$ BEGIN RETURN (SELECT AVG(POWER((CHURNED - MODEL_APPLY(churn_model, (AGE, TENURE_MONTHS, NUM_PRODUCTS, AVG_MONTHLY_SPEND))), 2)) FROM CUSTOMER_DATA_TEST); END; $$; ```

C.

○ ```sql CREATE OR REPLACE PROCEDURE CALCULATE_MSE() RETURNS FLOAT LANGUAGE JAVASCRIPT AS $$ var rs = snowflake.execute({ sqlText: `SELECT CHURNED, MODEL_APPLY(churn_model, (AGE, TENURE_MONTHS, NUM_PRODUCTS, AVG_MONTHLY_SPEND)) AS predicted FROM CUSTOMER_DATA_TEST` }); var total_squared_error = 0.0; var n = 0; while (rs.next()) { total_squared_error += Math.pow(rs.getColumnValue('CHURNED') - rs.getColumnValue('predicted'), 2); n++; } return total_squared_error / n; $$; ```

D.

○ ```sql CREATE OR REPLACE PROCEDURE CALCULATE_MSE() RETURNS FLOAT LANGUAGE SQL AS $$ BEGIN RETURN (SELECT SUM(POWER((CHURNED - MODEL_APPLY(churn_model, (AGE, TENURE_MONTHS, NUM_PRODUCTS, AVG_MONTHLY_SPEND))), 2)) / COUNT( ) FROM CUSTOMER_DATA_TEST); END; $$; ```

E.

○ ```sql CREATE OR REPLACE PROCEDURE CALCULATE_MSE() RETURNS FLOAT LANGUAGE PYTHON AS $$ import snowflake.connector ctx = snowflake.connector.connect() cs = ctx.cursor() cs.execute("SELECT CHURNED, MODEL_APPLY(churn_model, (AGE, TENURE_MONTHS, NUM_PRODUCTS, AVG_MONTHLY_SPEND)) AS predicted FROM CUSTOMER_DATA_TEST") results = cs.fetchall() total_squared_error = sum([(row[0] - row[1]) 2 for row in results]) n = len(results) mse = total_squared_error / n cs.close() ctx.close() return mse $$; ```

Answer: D

Explanation:
Option D is the most efficient and accurate because it uses a single SQL query to calculate the MSE directly. It avoids using cursors or procedural logic, which are less performant in Snowflake. It uses SUM to calculate the sum of squared errors and COUNT( ) to get the total number of records, then divides to

obtain the average (MSE). Option B calculates the average of power, that is wrong mathematical operation, Option A is correct from mathematical point but slow because of cursor and not following Snowflake best practices, option C is using JavaScript which is also valid, but Snowflake recommends to use SQL when possible for performance, and option E is using external python for model calculation, that not best for this scenarios.

Question: 8

You are tasked with predicting sales (SALES AMOUNT') for a retail company using linear regression in Snowflake. The dataset includes features like 'ADVERTISING SPEND', 'PROMOTIONS', 'SEASONALITY INDEX', and 'COMPETITOR PRICE'. After training a linear regression model named 'sales model', you observe that the model performs poorly on new data, indicating potential issues with multicollinearity or overfitting. Which of the following strategies, applied directly within Snowflake, would be MOST effective in addressing these issues and improving the model's generalization performance? Choose ALL that apply.

A. Apply Ridge Regression by adding an L2 regularization term during model training. This can be achieved by setting the 'REGULARIZATION' parameter of the 'CREATE MODEL' statement to 'L2'.
B. Manually remove highly correlated features (e.g., if 'ADVERTISING SPEND and 'PROMOTIONS' have a correlation coefficient above 0.8) based on a correlation matrix calculated using 'CORR function and feature selection techniques.
C. Increase the size of the training dataset significantly by querying data from external sources.
D. Perform feature scaling (e.g., standardization or min-max scaling) on the input features before training the model, using Snowflake's built-in functions or user-defined functions (UDFs) for scaling.
E. Decrease the 'MAX_ITERATIONS' parameter in the 'CREATE MODEL' statement to prevent the model from overfitting to the training data.

Answer: A,B,D

Explanation:
Options A, B, and D are the most effective strategies for addressing multicollinearity and overfitting in this scenario. Ridge Regression (A) adds an L2 regularization term, which penalizes large coefficients and reduces overfitting. Manually removing highly correlated features (B) addresses multicollinearity directly. Performing feature scaling (D) ensures that features with different scales do not disproportionately influence the model. Increasing training data (C) is generally helpful, but doesn't directly solve multicollinearity. Decreasing MAX ITERATIONS (E) might prevent the model from fully converging, but is a less targeted approach than regularization or feature selection.

Question: 9

You are developing a model to predict house prices based on structured data including size, number of bedrooms, location, and age. You have built a linear regression model within Snowflake. During the evaluation, you observe that the residuals exhibit heteroscedasticity. Which of the following actions is the LEAST appropriate to address heteroscedasticity in this scenario, considering you want to implement the solution primarily using Snowflake's built-in features and capabilities?

A. Apply a logarithmic transformation to the target variable ('SALES_PRICE) using the 'LOG' function within Snowflake before training the linear regression model.
B. Implement Weighted Least Squares (WLS) regression by calculating weights inversely proportional to the variance of the residuals for each data point. This involves creating a UDF to calculate weights and modifying the linear regression model fitting process. (Assume direct modification of the fitting process is possible within Snowflake).
C. Use robust standard errors in the linear regression analysis, even though Snowflake doesn't directly support calculating them. You decide to export model coefficients to an external statistics package (e.g., Python with Statsmodels) to compute robust standard errors and then bring insights back to Snowflake.
D. Include interaction terms between the independent variables in your linear regression model.
E. Transform independent variables using Box-Cox transformation and include in Snowflake Linear Regression Model Training

Answer: C

Explanation:
Option C is the least appropriate because it requires exporting model coefficients to an external tool to calculate robust standard errors. While robust standard errors are a valid way to address heteroscedasticity's impact on inference (hypothesis testing), the question explicitly prioritizes using Snowflake's built-in capabilities. Options A, B, D and E involve transformations/modifications within Snowflake itself. Applying a logarithmic transformation (A) can stabilize variance. Implementing WLS (B) directly addresses the unequal variances. Including interaction terms (D) can capture non-linear relationships and address some heteroscedasticity. Box-Cox Transformation (E) is a general way to transform non-normal independent variables.

Question: 10

A financial institution wants to predict fraudulent transactions on credit card data stored in Snowflake. The dataset includes features like transaction amount, merchant ID, location, time of day, and user profile information. The target variable is 'is_fraudulent' (0 or 1). You have trained several binary classification models (Logistic Regression, Random Forest, and Gradient Boosting) using scikit-learn and persisted them using a Snowflake external function for inference. To optimize for both performance (inference speed) and accuracy, which of the following steps should you consider before deploying your model for real-time scoring using the external function? SELECT ALL THAT APPLY.

A. Implement feature selection techniques (e.g., using feature importance scores from Random Forest or Gradient Boosting) to reduce the number of features passed to the external function, improving inference speed.
B. Increase the batch size of requests sent to the external function to amortize the overhead of invoking the external function itself, even if it increases latency for individual transactions.
C. Normalize or standardize the input features in Snowflake using SQL before passing them to the external function to ensure consistent scaling and potentially improve model performance.
D. Evaluate the models on a representative held-out dataset within Snowflake using SQL queries (e.g., calculating AUC, precision, recall) to choose the model with the best balance of performance and accuracy before deploying it.

E. Replace the trained models with a simple rule-based system based solely on transaction amount. If the amount is greater than a threshold, flag it as fraudulent, as this will be faster than calling the external function.

Answer: A,C,D

Explanation:
A, C, and D are the correct answers. A addresses optimizing inference speed by reducing input complexity. C addresses ensuring data quality and model performance consistency. D covers rigorous model validation and selection. B could potentially improve throughput, but could also hurt latency, so it is not always an optimal choice without careful consideration of requirements. E is incorrect as its a huge oversimplification and will likely have low accuracy.

Thank You for Trying Our Product

Special 16 USD Discount Coupon: NSZUBG3X

Email: support@examsempire.com

Check our Customer Testimonials and ratings available on every product page.

Visit our website.

https://examsempire.com/