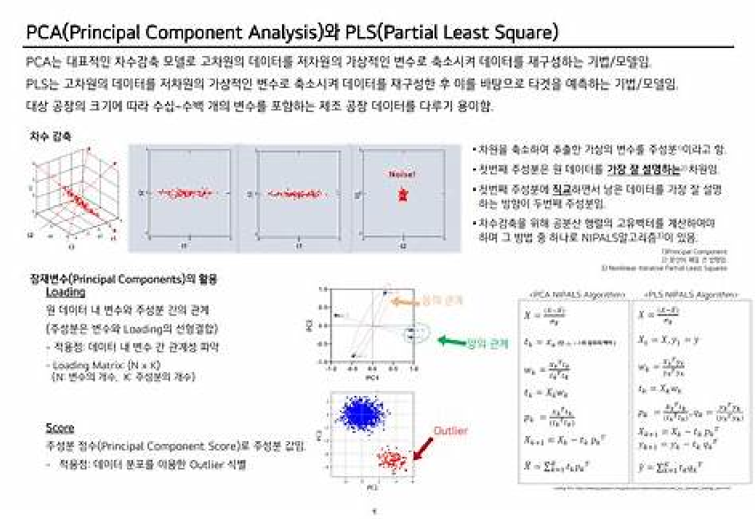
높은통과율PCA최신버전시험공부자료인증시험공부



PCA(Principal Component Analysis)와 PLS(Partial Least Square)

PCA는 대표적인 차수감축 모델로 고차원의 데이터를 저차원의 가상적인 변수로 축소시켜 데이터를 재구성하는 기법/모델임.

PLS는 고차원의 데이터를 저차원의 가상적인 변수로 축소시켜 데이터를 재구성한 후 이를 바탕으로 타겟을 예측하는 기법/모델임.

대상 공장의 크기에 따라 수십-수백 개의 변수를 포함하는 제조 공장 데이터를 다루기 용이함.

차수 감축

- 차원을 축소하여 추출한 가상의 변수를 주성분[1]이라고 함.
- 첫번째 주성분은 원 데이터를 **가장 잘 설명하는**[2] 차원임.
- 첫번째 주성분에 **직교**하면서 남은 데이터를 가장 잘 설명하는 방향이 두번째 주성분임.
- 차수감축을 위해 공분산 행렬의 고유벡터를 계산하여야 하며 그 방법 중 하나로 NIPALS알고리즘[3]이 있음.

[1] Principal Component
[2] 분산이 제일 큰 방향임
[3] Nonlinear Iterative Partial Least Squares

잠재변수(Principal Components)의 활용

Loading

원 데이터 내 변수와 주성분 간의 관계
(주성분은 변수와 Loading의 선형결합)
- 적용점: 데이터 내 변수 간 관계성 파악
- Loading Matrix: (N x K)
(N: 변수의 개수, K: 주성분의 개수)

Score

주성분 점수(Principal Component Score)로 주성분 값임.
- 적용점: 데이터 분포를 이용한 Outlier 식별

BONUS!!! PassTIP PCA 시험 문제집 전체 버전을 무료로 다운로드하세요: https://drive.google.com/open?id=1iETjv8HrPxtGo8Ikop-deBQxMFnn3L83

PassTIP의Linux Foundation PCA인증시험의 자료 메뉴에는Linux Foundation PCA인증시험실기와Linux Foundation PCA인증시험 문제집으로 나누어져 있습니다.우리 사이트에서 관련된 학습가이드를 만나보실 수 있습니다. 우리 PassTIP의Linux Foundation PCA인증시험자료를 자세히 보시면 제일 알맞고 보장도가 높으며 또한 제일 전면적인 것을 느끼게 될 것입니다.

Linux Foundation PCA 시험요강:

주제	소개
주제 1	• Instrumentation and Exporters: This domain evaluates the abilities of Software Engineers and addresses the methods for integrating Prometheus into applications. It includes the use of client libraries, the process of instrumenting code, and the proper structuring and naming of metrics. The section also introduces exporters that allow Prometheus to collect metrics from various systems, ensuring efficient and standardized monitoring implementation.
주제 2	• PromQL: This section of the exam measures the skills of Monitoring Specialists and focuses on Prometheus Query Language (PromQL) concepts. It covers data selection, calculating rates and derivatives, and performing aggregations across time and dimensions. Candidates also study the use of binary operators, histograms, and timestamp metrics to analyze monitoring data effectively, ensuring accurate interpretation of system performance and trends.
주제 3	• Observability Concepts: This section of the exam measures the skills of Site Reliability Engineers and covers the essential principles of observability used in modern systems. It focuses on understanding metrics, logs, and tracing mechanisms such as spans, as well as the difference between push and pull data collection methods. Candidates also learn about service discovery processes and the fundamentals of defining and maintaining SLOs, SLAs, and SLIs to monitor performance and reliability.
주제 4	• Alerting and Dashboarding: This section of the exam assesses the competencies of Cloud Operations Engineers and focuses on monitoring visualization and alert management. It covers dashboarding basics, alerting rules configuration, and the use of Alertmanager to handle notifications. Candidates also learn the core principles of when, what, and why to trigger alerts, ensuring they can create reliable monitoring dashboards and proactive alerting systems to maintain system stability.

주제 5	- Prometheus Fundamentals: This domain evaluates the knowledge of DevOps Engineers and emphasizes the core architecture and components of Prometheus. It includes topics such as configuration and scraping techniques, limitations of the Prometheus system, data models and labels, and the exposition format used for data collection. The section ensures a solid grasp of how Prometheus functions as a monitoring and alerting toolkit within distributed environments.

>> PCA최신버전 시험공부자료 <<

PCA완벽한 덤프공부자료 & PCA퍼펙트 최신버전 덤프샘플

이 글을 보시게 된다면Linux Foundation인증 PCA시험패스를 꿈꾸고 있는 분이라고 믿습니다. Linux Foundation인증 PCA시험공부를 아직 시작하지 않으셨다면 망설이지 마시고PassTIP의Linux Foundation인증 PCA덤프를 마련하여 공부를 시작해 보세요. 이렇게 착한 가격에 이정도 품질의 덤프자료는 찾기 힘들것입니다. PassTIP의Linux Foundation인증 PCA덤프는 고객님께서 Linux Foundation인증 PCA시험을 패스하는 필수품입니다.

최신 Cloud & Containers PCA 무료샘플문제 (Q16-Q21):

질문 # 16

Which PromQL expression computes how many requests in total are currently in-flight for the following time series data?
apiserver_current_inflight_requests{instance="1"} 5
apiserver_current_inflight_requests{instance="2"} 7

- A. min(apiserver_current_inflight_requests)
- B. max(apiserver_current_inflight_requests)
- C. sum_over_time(apiserver_current_inflight_requests[10m])
- D. sum(apiserver_current_inflight_requests)

정답： D

설명：

In Prometheus, when you have multiple time series that represent the same type of measurement across different instances, the sum() aggregation operator is used to compute their total value.
Here, each instance (1 and 2) exposes the metric apiserver_current_inflight_requests, indicating the number of active API requests currently being processed.
To find the total number of in-flight requests across all instances, the correct expression is:
sum(apiserver_current_inflight_requests)
This returns 5 + 7 = 12.
min() would return the lowest value (5).
max() would return the highest value (7).
sum_over_time() calculates the cumulative sum over a range vector, not the current value, so it's incorrect here.
Reference:
Verified from Prometheus documentation - Aggregation Operators and Summing Across Dimensions sections.

질문 # 17

Which exporter would be best suited for basic HTTP probing?

- A. SNMP exporter
- B. Blackbox exporter
- C. JMX exporter
- D. Apache exporter

정답： B

설명：

The Blackbox Exporter is the Prometheus component designed specifically for probing endpoints over various network protocols, including HTTP, HTTPS, TCP, ICMP, and DNS. It acts as a generic probe service, allowing Prometheus to test endpoints' availability, latency, and correctness without requiring instrumentation in the target application itself.

For basic HTTP probing, the Blackbox Exporter performs HTTP GET or POST requests to defined URLs and exposes metrics like probe success, latency, response code, and SSL certificate validity. This makes it ideal for uptime and availability monitoring. By contrast, the JMX exporter is used for collecting metrics from Java applications, the Apache exporter for Apache HTTP Server metrics, and the SNMP exporter for network devices. Thus, only the Blackbox Exporter serves the purpose of HTTP probing.
Reference:
Verified from Prometheus documentation - Blackbox Exporter Overview and Exporter Usage Guidelines.

질문 # 18
What is api_http_requests_total in the following metric?
api_http_requests_total{method="POST", handler="/messages"}

- A. "api_http_requests_total" is a metric type.
- B. "api_http_requests_total" is a metric label name.
- C. "api_http_requests_total" is a metric name.
- D. "api_http_requests_total" is a metric field.

정답： C

설명：
In Prometheus, the part before the curly braces {} represents the metric name. Therefore, in the metric api_http_requests_total{method="POST", handler="/messages"}, the term api_http_requests_total is the metric name. Metric names describe the specific quantity being measured - in this example, the total number of HTTP requests received by an API.
The portion within the braces defines labels, which provide additional dimensions to the metric. Here, method="POST" and handler="/messages" are labels describing request attributes. The metric name should follow Prometheus conventions: lowercase letters, numbers, and underscores only, and ending in _total for counters.
This naming scheme ensures clarity and standardization across instrumented applications. The metric type (e.g., counter, gauge) is declared separately in the exposition format, not within the metric name itself.
Reference:
Verified from Prometheus documentation - Metric and Label Naming, Data Model, and Instrumentation Best Practices sections.

질문 # 19
How can you use Prometheus Node Exporter?

- A. You can use it to collect metrics for hardware and OS metrics.
- B. You can use it to instrument applications with metrics.
- C. You can use it to probe endpoints over HTTP, HTTPS.
- D. You can use it to collect resource metrics from the application HTTP server.

정답： A

설명：
The Prometheus Node Exporter is a core system-level exporter that exposes hardware and operating system metrics from *nix-based hosts. It collects metrics such as CPU usage, memory, disk I/O, filesystem space, network statistics, and load averages.
It runs as a lightweight daemon on each host and exposes metrics via an HTTP endpoint (default: :9100/metrics), which Prometheus scrapes periodically.
Key clarification:
It does not instrument applications (A).
It does not collect metrics directly from application HTTP endpoints (B).
It is unrelated to HTTP probing tasks - those are handled by the Blackbox Exporter (D).
Thus, the correct use of the Node Exporter is to collect and expose hardware and OS-level metrics for Prometheus monitoring.
Reference:
Extracted and verified from Prometheus documentation - Node Exporter Overview, Host-Level Monitoring, and Exporter Usage Best Practices sections.

질문 # 20
Given the metric prometheus_tsdb_lowest_timestamp_seconds, how do you know in which month the lowest timestamp of your Prometheus TSDB belongs?

- A. month(prometheus_tsdb_lowest_timestamp_seconds)
- B. format_date(prometheus_tsdb_lowest_timestamp_seconds,"%M")
- C. (time() - prometheus_tsdb_lowest_timestamp_seconds) / 86400
- D. prometheus_tsdb_lowest_timestamp_seconds % month

정답：C

설명:
The metric prometheus_tsdb_lowest_timestamp_seconds provides the oldest stored sample timestamp in Prometheus's local TSDB (in Unix epoch seconds). To determine the age or approximate date of this timestamp, you compare it with the current time (using time() in PromQL).
The expression:
(time() - prometheus_tsdb_lowest_timestamp_seconds) / 86400
converts the difference between the current time and the oldest timestamp from seconds into days (1 day = 86,400 seconds). This gives the number of days since the earliest sample was stored, allowing you to infer the time range and approximate month manually.
The other options are invalid because PromQL does not support direct date formatting (format_date) or month() extraction functions.
Reference:
Extracted and verified from Prometheus documentation - TSDB Internal Metrics, Time Functions in PromQL, and Using time() for Relative Calculations.

질문 # 21
......