Data Science Skills Evaluation Framework

CodeSignal Skills Evaluation Lab

Abstract—This research paper presents a comprehensive framework, the CodeSignal Skills Taxonomy, for quantifying and evaluating skills associated with IT roles. Then it applies this framework to the role of Data Scientist. The proposed framework identifies relevant skills across diverse job tasks and classifies them into foundational, essential, and specialized areas for precision and accuracy. The distinctive contribution of this research resides in the development of a mathematical representation of job roles, incorporating consideration of hard skills proficiency through a robust scoring system - the Job Fit Score. Assuring content validity, the framework effectively simulates real-world job activities and aligns evaluation mechanisms to job-specific tasks, ensuring a direct correlation between job-fitness and assessment score. The research illustrates applications of the framework using the roles of a Pastry Chef and a Data Scientist, demonstrating broad applicability across different industry sectors. The proposed framework revolutionizes the skill evaluation process in the technical hiring space, aiding in the empirically accurate evaluation of candidate qualifications.

In an era brimming with digital knowledge, measuring skills, particularly in the Information Technology sector, proves a daunting task. This research paper introduces a universal framework, the CodeSignal Skills Taxonomy, that meticulously captures, classifies, and evaluates job-relevant skills across numerous IT roles, such as a Data Scientist, with precision and rigor. By capturing signals about essential work activities and mapping them to corresponding cognitive and psychological attributes, the framework successfully pinpoints a candidate's competency in hard skills, tool skills, and people skills. Through a substantial mathematical model underscoring each job's diverse spectrum, the framework further provides a robust scoring system - the Job Fit Score. This scoring system measures an individual's compatibility for a job based on their hard skills, thus offering an empirical evaluation of their qualification for a specific role. This pioneering research revolutionizes technical hiring by furnishing a comprehensive, well-documented assessment standard, effectively bridging the gap between job-relevant skill recognition and skill proficiency evaluation.

1 Universal Skills Evaluation Framework

One of the most effective methods of measuring skills is via simulation-based evaluations, which capture signals about skills by asking candidates to engage in behaviors that reflect job-relevant tasks. To build simulation-based evaluations for a variety of job roles at scale, we have developed the CodeSignal Skills Taxonomy, a model for organizing the relationships between job-relevant tasks and the skill areas required to complete those tasks. This model is the basis for designing effective content for both evaluating job-relevant skills in a pre-hire context and teaching job-relevant skills in a learning context.

Skill Areas are the underlying cognitive and psychological attributes that support the execution of Job Tasks. Skill Areas consist of three types of skills: Hard Skills (i.e., tool-independent technical skills), Tool Skills (i.e., tool proficiency-related skills), and Soft Skills (i.e., people skills). Skill Areas and Job Tasks can be further categorized into three high-level categories: Foundational, Essential, and Specialized. Skill Sets are sub-groups of Skill Areas that are relevant to the corresponding Job Tasks. The conceptualizations of Job Tasks, Skill Sets, Skill Areas, and Tools within the CodeSignal Skills Taxonomy allow clear mappings between jobs and skills, thereby facilitating the development of effective content for pre-hire evaluations and learning.

1.1 CodeSignal Skills Taxonomy

CodeSignal Skills Taxonomy is a formal classification system that organizes a diverse range of job tasks and associated skills into concrete entities. The Taxonomy offers a comprehensive framework for understanding and mapping the various psychological attributes that allow individuals to accomplish job tasks in different fields and professions. The Taxonomy is divided into Domains, Job Categories, Jobs, Job Tasks, Skill Sets, Skill Areas, and Tools.

Together, these elements of CodeSignal Skills Taxonomy form Skills Maps, which provide a holistic understanding of the skills needed for various roles in diverse domains. Skills Maps can help in structuring the identification, evaluation, and development of skills, making it easier for individuals and organizations to assess their strengths, areas for improvement, and requirements effectively.

Jobs

At the core of this Taxonomy are jobs. Jobs constitute the particular roles or positions within organizations that allow individuals to create value within organizations. A job captures the particular responsibilities, tasks, and skills necessary to succeed in a role. For instance, within the category of Data, jobs might include roles like Data Analyst, Data Scientist, and Data Engineer.

Inter-Job Hierarchy

First, we discuss the concepts of Domains and Job Categories, which can be used to organize and differentiate across jobs.

Domains

Domains represent the broad sectors or industries in which a variety of jobs and roles exist. They encompass the entire spectrum of professional fields and sectors such as Information Technology, Finance, Healthcare, and more. Each domain is distinct and characterized by a specific set of requirements, principles, and practices. For instance, the Healthcare domain would encompass all roles related to medical and health services, from physicians to healthcare analytics specialists.

Job Categories

Job categories are more specific subgroups within a domain, emphasizing the fundamental roles and responsibilities within the larger context. Job categories offer a more precise classification, providing a lens into the various pathways one can take within a particular domain. For instance, within the domain of Information Technology Engineering, we may find job categories such as Software and Application Development, Data, Security, and more.

Intra-Job Hierarchy

Within a particular job, individuals must perform a collection of related or complementary duties or job tasks to create value and accomplish organizational goals. Moreover, individuals must possess the necessary skills to perform their job duties successfully. Next, we discuss Job Tasks, Skill Sets, Skill Areas, and Tools, which are important concepts for measuring job-relevant skills.

Job Tasks

Job Tasks define the essential tasks and activities inherent to a specific job, representing what people do on the job. They represent the responsibilities required of a role, regardless of the tools employed. For instance, in data-focused roles, Data Cleaning is indispensable. In application development, Performance Optimization is pivotal, while Model Development is central to machine learning or artificial intelligence. Job Tasks may be relevant to more than one job, but their actual execution may vary in different job contexts.

Skill Sets

Skill Sets are the grouping of Skill Areas relevant for proficiently executing a job task. Skill Sets are uniquely mapped to each Job Task, but may consist of different combinations of individual Skill Areas with varying degrees of relevance.

Skill Areas

Skill Areas encompass the underlying cognitive and psychological attributes that support the execution of Job Tasks. Notably, a single Skill Area within a Skill Set may be relevant to multiple Job Tasks. For example, the Skill Area of "Computer Programming" is relevant to both the "Analyze Data" Job Task and the "Develop a Model" Job Task for Data Science jobs.

Tools

Tools refer to the platforms, software, applications, or other technological resources that are used to effectively accomplish job tasks. Tools are the means by which a skill area is expressed when completing a job task. Tools are usually job-specific or field-specific. For instance, React may be a preferred tool for Front-end Engineers, while Data Scientists might utilize tools like Pytorch or Salesforce. A tool may be relevant for more than one Skill Area and thus multiple Job Tasks.

Supplemental Information for Intra-Job Categorizations

Skill Area and Job Task Categories

While identifying and defining all possible skill areas and job tasks is beneficial, the extensive variety within a job can make the raw list impractical. Therefore, we have streamlined the identification of important skill areas and job tasks by grouping them into three distinct categories for better utility.

Figure 1: Skills Taxonomy

Foundational

The foundational category represents skill sets that are the basic, underlying skill areas that everyone in a particular role or field must possess to participate effectively. These skill areas form the bedrock of proficiency. Foundational skill areas are necessary for entry-level competence and are often developed early in training or education. These skill areas are often shared across a variety of related jobs. Examples might include basic "Ingredient Measurement and Scaling" for a Pastry Chef or "Mathematics and Statistics" for a Data Scientist.

Essential

The essential category represents the tasks and activities that professionals in a particular field use regularly in their work. They are specific to the role and are often tied to the core responsibilities of that role. Typically, essential skill set proficiency is developed after foundational skill sets and is central to effective job performance. For example, the essential skill set for a Pastry Chef could include dough/batter preparation, fermentation/proofing, and oven management skills. For a Data Scientist, it might involve exploratory data analysis and data modeling skills.

Specialized

The specialized category represents skill sets that are less generalizable and typically either more advanced or are not common to every role within a field, but offer specific advantages in certain contexts or positions. Specialized skill sets are tied to unique job tasks that not everyone in the same role performs day-to-day. They usually build upon essential skill sets and require deeper knowledge or expertise. Specialized skill sets can differentiate a professional in the field and often align with specific types of work or industries. For example, a Pastry Chef may develop sugar craft or chocolate sculpting skills, while a Data Scientist might specialize in deep learning or big data processing.

1.2 Four Degrees of Competency

Most skills have a degree of mastery or competency, and the true mathematical representation of competency would be a continuous function with natural ability and time invested in deliberate practice as the primary inputs. However, it is convenient for practical measures to define degrees of competency and to split the entire spectrum into a handful of categories. In his widely adopted Taxonomy of Educational Objectives, Benjamin Bloom (1956) [1] described six degrees of competency, which were later revised in a paper titled "A Taxonomy for Teaching, Learning, and Assessment" [2]. To simplify this system further, we define four degrees of competency while mapping them to the Revised Bloom's Taxonomy.

Developing

At this stage, individuals are in the preliminary stages of acquiring the skill set. They demonstrate a foundational understanding or familiarity with the skill areas within the skill set but will need significant learning and practice to increase proficiency. Individuals at this level of proficiency are still in the process of understanding and remembering distinct aspects of the skill set, mirroring the ‘Remember’ and ‘Understand’ cognitive domains in Bloom’s Taxonomy.

Intermediate

Those at the Intermediate level exhibit a moderate level of proficiency in the skill set, showing an understanding and application of the key concepts or techniques related to the skill areas within the skill set. There is room for further learning and practice in handling more advanced aspects of the skill set, which aligns with ’Apply’ in Bloom’s Taxonomy, where individuals can apply knowledge to new situations.

Advanced

Individuals at the Advanced proficiency level demonstrate a high level of expertise, showing an ability to handle most tasks and situations associated with the skill set. They may still require additional learning or support to navigate more complex situations. This level aligns with ‘Analyze’ and ‘Evaluate’ in Bloom’s Taxonomy, denoting that individuals can break down information into parts and make judgments about the information.

Expert

Those at the Expert level of proficiency showcase mastery over the skill set. They possess a comprehensive understanding and ability to apply the skill set accurately in all evaluated scenarios. At this level, individuals have reached the ‘Create’ cognitive domain in Bloom’s Taxonomy, suggesting they can create new work or make new insights.

1.3 Mathematical Representation

Before describing the process for identifying Job Tasks, Skill Sets, and Skill Areas, it’s helpful to take a step back and understand the mathematical representation of a job.

We aim to model each job as a composite function derived from a series of sub-functions (or “job tasks”), each contributing to the job’s overall completion and each defined in terms of the underlying skills that drive the outcome of the function/task.

We start by defining the following concepts:

1) Input Space - I: Performing a particular job is a process that starts from an initial point, which we’ll call the input space I.
2) Output Space - O: Successfully getting the job done results in a set of outputs, which we’ll call the output space O.
3) Job Tasks - J_i: These are individual sub-processes required to perform the job. Each J_i is a function that contributes to transforming the input space I into the output space O.
4) Skills Space - S: All possible skills related to the completion of a particular job can be grouped into a skills space, denoted as S. The Skills Space consists of three disjoint sets, Hard Skills - S^h (tool-independent technical skill sets), Tools Skills - S^t (tool-specific skill sets), and Soft Skills - S^s (non-technical people skill sets).

Let J denote the job function. Then, by definition, we get the following equations that help us mathematically describe the above-defined concepts:

$$J = J_1 \circ J_2 \circ J_3 \circ \cdots \circ J_n \text{ where } J : I \to O$$

This means that J is a composite function of several Job Tasks consecutively applied to the input space to produce the output space. Also

$$S = S^h \cup S^t \cup S^s$$

And

$$J_i = f(m_i, S^h, S^t, S^s)$$

[1]

This last equation represents that each Job Task is a function of motivation at the time of completing the task and Hard, Tool, and Soft skills. As we are primarily interested in measuring an individual’s qualification to be effective in a particular job modeled through measurements of their different skill areas, let’s assume that $M(S)$ represents a numeric measurement of an individual’s skill set in a skill area S ranging from 0 to 1. Then we can model an individual’s degree of job/role fit, $F(J)$, for a particular job J, as follows:

1. Motivation Factors - m_i: This is the level of motivation at the moment of completing each job task J_i. If all of the job tasks are happening within a short time span, it could be argued that there is one shared motivation factor, however, many job tasks take place across days or weeks which means motivation levels can vary depending on the task.

$$F(J) = \sum_{i=0}^{|S^h|} h_i \cdot M(S_i^h) + \sum_{j=0}^{|S^t|} t_j \cdot M(S_j^t) + \sum_{k=0}^{|S^s|} s_k \cdot M(S_k^s) + B$$

Where h_i, t_j, s_k are the linear combination coefficients representing the relative importance of a skill area for the specific job J and B is the theoretical baseline qualification level for someone whose skill areas are at an absolute 0 (mostly impossible in practice). Note that motivation factors m_i are not present in this equation since this is an evaluation of fit at a specific time. This means, in particular, that an individual highly qualified for a certain job might still fail to effectively carry out certain job tasks if their motivation is low during the completion of the specific task.

If we group the h_i, t_j, s_k coefficients into Hard Skills, Tools Skills, and Soft Skills vectors, you can further simplify this equation to the following:

$$F(J) = \overline{H} \cdot \overline{M(S^h)} + \overline{T} \cdot \overline{M(S^t)} + \overline{S} \cdot \overline{M(S^s)} + B$$

While it could be argued that the relationship between these different skill areas and their impact on performance is non-linear, we postulate here that job fit has a linear dependency on the three skill areas, for simplicity. This postulate can be proven by analyzing different jobs and individuals with varied degrees of measured competencies and performance levels.

Such a study requires a large data set of individual performances and detailed measurements of their competencies across Hard, Tool, and Soft skills. Assuming this postulate is accurate, a standard job analysis can reveal the Skills Vectors for a given job helping determine which skills have the most impact on performance and should be prioritized during the evaluation process, and which are not very significant and should be ignored. For the remainder of this paper, we'll focus our attention on ways to calculate $M(S^h)$, i.e., ways to measure Hard Skills. However, future papers will also address measurements of the other skill areas, calculation of the Skills Vectors for specific jobs, and empirical validation of the above postulate.

1.4 Identifying Job Tasks, Skill Sets, and Skill Areas

To identify Skill Areas for each job, the process revolves around a five-step process: identifying relevant job tasks, ascertaining the skill areas, interlinking these skill areas and tasks to create a Skill Set, identifying relevant tools, and validating. The detailed breakdown is as follows:

Identification Process

1) Identify Job Tasks
   a) Begin with the most prevalent job titles within the given job category.
   b) Extract 500-1000 current job postings from public job boards.
   c) Use GPT-4 (or another advanced LLM available) to discern the job tasks from each job description.
   d) Identify the prevalence or frequency in which the job tasks appear across the job descriptions to create a rank-ordered list of relevant job tasks for the role.
2) Identify Skill Areas Required for Job Tasks
   a) Use the LLM to discern the relevant skills and knowledge from each job description.
   b) Identify the prevalence or frequency rate in which the skill areas appear across the job descriptions to create a rank-ordered list of the requested skill areas for the role.
3) Link Skill Areas and Job Tasks to Create the Skill Set
   a) For each identified job task, use the LLM to map the associated skill areas in Step 2. This creates a matrix where each task is mapped to one or more related skill areas.
   b) Examine overlaps between skill areas and job tasks. Those with significant overlap can be grouped into broader dimensions. This helps in understanding which skill areas are multifaceted and applicable across multiple tasks.
   c) The collection of Skill Areas linked to a Job Task represents the Skill Set, or collection of skill areas that encapsulates the range of skills and knowledge relevant to completing the job task.
   d) Create concise, representative labels and definitions for each Job Task.
   e) Create concise, representative labels and definitions for each Skill Area within the Skill Set.
4) Tools
   a) Use the LLM to discern the relevant tools from each job description.
   b) Allocate popularity ratings (High, Mid, Low) to the tools identified based on their frequency in the job descriptions. This offers insights into the most relevant tools for the skill areas associated with each job task.
5) Validation
   a) Engage a sample of two to five subject matter experts to review the links between the Skill Areas within the Skill Sets and Job Tasks, ensuring that the mapping accurately reflects real-world job de-

mands.

Throughout this process, we ensure that jobs are selected from various industries and seniority levels to maximize the generalizability of the identified skill areas and job tasks chosen to mirror what an individual does in the role. Additionally, note that we perform this analysis every six months to ensure that our Skill Sets, Tools, and chosen Job Tasks meet industry norms.

1.5 Skill Set Examples and Ordering

For each job category, the Foundational Skill Areas will typically be organized in the order that beginners learn them and represent the shared underlying skill areas that are the foundation the Essential skill sets are built upon. On the other hand, the Essential Skill Set consists of relevant and generalizable skill areas that are essential to the role and are organized in the order they are likely to be performed on the job. The Specialized Skill Set contains the remaining skill areas that may be more complex or specialized in nature, organized in the order of importance or popularity. For instance, in a given bake, pastry chefs are likely to demonstrate recipe interpretation and planning, followed by ingredient selection and preparation, and then some mixing and dough preparation, eventually leading to baking, cooling, finishing, and decoration. Similarly, a Data Scientist, whose job it is to perform data analysis and provide insights and recommendations, is more likely to start by running some exploratory data analysis and visualizations to understand the data, then clean and process the data based on their findings, then model construction on the cleaned and processed data, and finally evaluation, validation, and selection of the best approach.

Here are examples of two very different jobs unified by this framework.

Pastry Chef Example

```
Pastry Chef:
  - Basic Baking Techniques
      category: foundational
  - Knowledge of Ingredients
      category: foundational
  - Food Safety and Sanitation
      category: foundational
  - Measurement and Scaling
      category: foundational
  - Recipe Interpretation and Planning
      category: essential
  - Ingredient Selection and Preparation
      category: essential
  - Mixing and Dough/Batter Preparation
      category: essential
  - Fermentation and Proofing
      category: essential
  - Shaping and Molding
      category: essential
  - Baking and Oven Management
      category: essential
  - Cooling and Storage
      category: essential
  - Finishing and Decoration
      category: essential
  - Presentation and Plating
      category: essential
  - Quality Control and Evaluation
      category: essential
  - Chocolate Work
      category: specialized
  - Specialized Dietary Baking
      category: specialized
  - Advanced Decorative Techniques
      category: specialized
```

Data Scientist Example

```
Data Scientist:
  - Mathematics and Statistics:
      category: foundation
  - Programming and Algorithms for Data Manipulation:
      category: foundation
  - Data Querying and Retrieval:
      category: foundation
  - Exploratory Data Analysis and Visualization:
      category: essential
  - Data Cleaning and Preprocessing:
      category: essential
  - Machine Learning and Predictive Modeling:
      category: essential
  - Model Evaluation, Validation, and Selection:
      category: essential
  - Deep Learning and Neural Networks:
      category: specialized
  - Big Data Processing:
      category: specialized
  - Advanced Data Visualization and Storytelling:
      category: specialized
  - Ethical Practices, Data Privacy and Data Governance:
      category: specialized
```

1.6 Patterns of Evaluation Design

The essence of effective skills evaluation lies in the use of simulations, where individuals are placed in realistic scenarios to demonstrate their proficiency in one or more targeted skill or knowledge domains. For example, evaluations for software engineering jobs would require individuals to write code and create software, whereas evaluations for data analytics would require individuals to extract insights from data sets. Through the use of simulations, we can recreate a variety of job tasks in a story-like fashion that replicates the behaviors required to complete day-to-day responsibilities within a given job, allowing for a comprehensive view of an individual's relevant Skill Areas for that job.

Replicating Job Tasks Through Simulations

1.6.1 Role-Specific Evaluations

As mentioned, simulations can be designed to mirror actual tasks and responsibilities of a given role. The holistic evaluation of competence for a role typically involves a project-based assessment where skills and knowledge are evaluated using realistic scenarios that simulate the actual duties of the role, requiring the use of a wide range of skill areas. For example, a pastry chef's abilities can be holistically evaluated in a practical session where they rotate between various techniques, demonstrating skill areas like dough preparation and dessert decoration. Likewise, a data scientist's skill areas in data cleaning, exploratory data analysis, and model creation can be assessed by guiding them through a project that encapsulates the entire pipeline from data cleaning to model development.

1.6.2 Focused Skill Scenarios

There are situations where specific skill sets require targeted evaluation scenarios due to either the complexity of the job tasks associated with the skill sets or the specialized nature of the skill sets, making it so that there is a loss of generalizability if the skill sets are included as part of a role-specific evaluation. For instance, a pastry chef's sugar-sculpting capability might be assessed in a setup distinct from regular kitchen tasks. Similarly, a data scientist might be given scenarios focusing solely on using natural language or computer vision algorithms rather than the typical machine learning algorithms that are more standard across industries.

1.7 Evaluation Scoring

As described in the Mathematical Representation section above we denote the function that represents a numerical representation of an individual's skill set in a skill area S as $M(S)$. $M(S)$ has values ranging from 0 to 1. Depending on the evaluation context (e.g., hiring, early education, practice, internal mobility, etc), the $M(S)$ values across different skill sets can be combined into a recommendation score that can be used for the given purpose. Below, we describe the approach used on the CodeSignal platform to provide a Job Fit Score (JFS) for hiring use cases. Note that in its current form, JFS is only based on Hard Skills; however, an ideal Job Fit Score needs to also take into account the other two categories of skill areas (Tools Skills and Soft Skills), and we plan on incorporating those into the score in future iterations as well.

The Job Fit Score quantifies a candidate's skills and knowledge, distilled from the aggregation of Skill Areas grouped into Skill Sets and expressed through the candidate's ability to complete the Job Tasks relevant to the targeted role. The Job Fit Score provides a numerical scale ranging from 200 (lowest) to 600 (highest). The score is designed to be a valid and reliable representation of the candidate's skills and knowledge relative to the requirements of a particular job that is simple to interpret. Job Fit Scores increase as candidates partially or fully complete each task and subtasks targeting a particular Skill Area.

The Job Fit Score includes a two-tiered scoring system. Within the two-tiered system, the first tier (i.e., score base points) treats the Job Tasks equally and accounts for 80% of possible points that a candidate can earn; the second tier (i.e., score bonus points) is awarded to candidates upon full completion of each question and accounts for 20% of all possible points the candidate can earn. The formula for calculating the Job Fit Score is presented next.

$$JFS = 400 \cdot (\sum_{i=0}^{|S|} (.80 \cdot \frac{M(S_i)}{|S|} + .20 \cdot BS_i)) + 200$$

Where JFS is the Job Fit Score representing a candidate's overall job fit based on their demonstrated Hard Skills. S is the Job Task targeted within a framework. BS_i is the bonus score rewarded to a candidate if the $M(S_i)$ equals 1.00, indicating the candidate successfully completed the Job Task perfectly. Bonus score amounts are framework-specific and tied to how difficult a Job Task is, as well as how well a Job Task differentiates between ability levels.

Bonus points (the second tier) help differentiate the degree of fit for a role between candidates by awarding candidates marginally more points for successfully demonstrating their ability to execute a Job Task at 100%, thereby awarding candidates that display an Expert-level of competence. Treating each Job Task relatively equally (the first tier) motivates candidates to meet as many requirements as possible with their current skills and knowledge.

This equal awarding of base points maximizes fairness and equity of the scoring process, resulting in a favorable candidate experience. Most importantly, by encouraging candidates to complete all Job Tasks they are able, the scoring approach optimizes the precision in identifying candidate skill levels and expands the quality and depth of signal regarding candidate skills and knowledge, captured for determining the degree of overall job fit. For further explanation of the scoring mechanism please see the knowledge base [3].

1.8 Content Validation

In the realm of skills-based assessment, particularly when designing simulation-based tests, the importance of content validation cannot be understated. To effectively capture signals about skills within simulation-based assessments, ensuring that the simulation content faithfully mirrors real-world job demands is critical to their relevance and utility. By engaging subject matter experts in the validation process, we introduce a level of rigor and specificity, ensuring that our simulations are not just hypothetical exercises but genuine replicas of job tasks that would effectively capture on-the-job behaviors, thereby allowing us to infer Skill Sets tied to these job tasks.

For example, to simulate software development projects in engineering roles, we can create a series of project-based tasks that require candidates to write code based on predefined user requirements. Since the underlying Skill Areas that enable candidates to do this are latent psychological attributes, we can only measure Skill Area proficiency by observing candidates' coding behaviors. As such, we must validate that the simulation content accurately reflects relevant job tasks for engineering roles, and that the job tasks being reflected are tied to the Skill Areas we are interested in evaluating. Both of these validation requirements necessitate input from subject matter experts for the given job role (e.g., job incumbents and hiring managers).

Subject matter experts bring a depth of experience and understanding of the nuances and intricacies of a job role. By validating the connections between Skill Areas and Job Tasks with their expertise, we make sure that the assessment remains rooted in the day-to-day realities of the job. This not only bolsters the confidence of both the assessors and the candidates in the process but also improves the predictive validity by making sure the tasks are relevant across roles/situations.

2 Designing Essential Skills Evaluations for Data Scientists

To design a skills evaluation for Data Scientists following the Universal Skills Evaluation Framework outlined above, we first started by identifying the Foundational, Essential, and Specialized Skill Sets that are most commonly associated with this job.

Additionally, we need to understand the primary input to which a Data Scientist would apply their Core Skills to create the primary output of the job.

- Input(s): Dataset
- Output(s): Models, Insights, Recommendations, Predictions

In tables 1, 2, and 3, we provide an overview of all Core Hard Skill Sets and their sub-skills for a Data Scientist that's been identified following the process outlined in section 1.4.

Since our evaluations are designed to be completed in under 2 hours to create a good test-taker experience, including all of the above skills into one evaluation would either result in a shallow assessment OR require a much longer assessment time, leading to a significant drop-off in completion rates. Instead, we prioritize the Skill Sets (referred to as Target Skill Sets in table 4) that are most popular within the industry in terms of demand as well as those that are more possible to simulate in a virtual assessment environment.

Next, we describe sample tasks designed to evaluate each of the Target Skill Sets along with a list of which sub-skills are aimed to be evaluated by the provided task and which are not. The choice of which sub-skills to evaluate and which not to is based on the popularity of the sub-skill that's determined as part of the Skill Set identification process described above.

Note that while Skill Sets aim to be tool-agnostic, tasks designed to evaluate the area permit and often encourage the use of tools to create a more realistic scenario.

3 Task 1

Skills Module

- Exploratory Data Analysis

✓ What's Evaluated

- Combining multiple datasets into a single table
- Handling numeric conversions and data consistency
- Calculating metrics and saving them in a CSV file

✗ What's Not Evaluated

- Advanced feature engineering
- Production environment considerations

Description

You are provided with several files containing information about used cars and their viewing records. The data, for each chunk of cars, resides in four CSV files named `cars_0.csv` to `cars_3.csv`. Each file contains:

- `car_id` (unique identifier)

	Skills Module	Sub-Skills
F1	**Mathematics and Statistics**	Probability Theory and Statistics, Linear Algebra and Multivariate Calculus, Numerical Computation and Optimization Methods, Discrete Mathematics and Graph Theory, Stochastic Processes and Time Series Analysis, Mathematical Modeling and Simulation, Inferential Statistics and Hypothesis Testing, Bayesian Thinking and Statistical Decision Theory
F2	**Coding and Data Algorithms**	Python Programming, R Programming, SQL Programming, Data Structures and Algorithms for Data Science, Handling Large Datasets in Memory, Pattern Matching and Text Processing, Data Transformation Methods, Parallel and Distributed Data Processing Algorithms, Web Scraping and Data Gathering Techniques, Dealing with Semi-structured and Unstructured Data, APIs and Libraries for Data Manipulation, Optimizing Code Performance for Data Manipulation Tasks, Error Handling and Debugging Techniques for Data Processing, Ensuring Data Quality and Integrity in Data Processing, Version Control Practices for Data Science, Algorithm Complexity Analysis and Time Complexity, Fundamentals of Functional Programming for Data Manipulation Tasks
F3	**Data Querying and Retrieval**	SQL and NoSQL Database Query Techniques, Database Management Systems (DBMS), API-Based Data Retrieval Methods, Data Retrieval in Distributed Systems, Web Scraping Techniques, Handling Semi-Structured and Unstructured Data, Big Data Querying Tools and Techniques, File IO and Streaming Data Handling, Data Querying Optimization Techniques, Integrating Queries from Multiple Data Sources

Table 1: Overview of Foundational Skills Modules and their sub-skills for a Data Scientist

- `doors` (number of doors)
- `engine_size` (size of the car's engine)
- `mileage` (current mileage of the car)
- `horsepower` (engine horsepower)
- `has_sunroof` (1 if sunroof is present, 0 otherwise)
- `age` (age of the car in years)
- `price` (the car's price)

You also have a file `car_views.csv` with the following columns:

- `viewer_id` (identifier for the viewer)
- `car_id` (identifier of the car viewed)
- `view_date` (date of the view in YYYY-MM-DD format)

Your tasks are as follows:

1) Compute the average car price across all available data.
2) Determine the percentage of cars that have a sunroof.
3) Calculate the average number of views per car over the last 30 days, from March 16th, 2023, to April 15th, 2023, inclusive.

You should store these three results in `analysis_results.csv`, each with an `insight_type` column and a `value` column (rounded to two decimal places if applicable). Each row should represent one of the above metrics.

Solution (pseudocode)

```
FUNCTION performAnalysis():
    # Combine data from cars_0.csv, cars_1.csv,
    ↪  cars_2.csv, cars_3.csv
    all_car_data = mergeAllCarCSVs(["cars_0.csv",
    ↪  "cars_1.csv", "cars_2.csv", "cars_3.csv"])

    # Convert price to a numeric type and handle missing
    ↪  entries
    clean car price column in all_car_data

    # Compute average car price
    average_price = mean(all_car_data.price)

    # Compute percentage of cars with sunroof
    sunroof_percentage = (count where has_sunroof is 1 /
    ↪  total cars) * 100

    # Read car_views data
    car_views_data = readCSV("car_views.csv")
    parseDates(car_views_data.view_date)

    # Filter views from March 16, 2023 to April 15, 2023
    last_30_days_views =
    ↪  filterByDateRange(car_views_data, "2023-03-16",
    ↪  "2023-04-15")

    # Group by car_id and count views
    views_per_car = groupAndCount(last_30_days_views,
    ↪  "car_id")

    # Compute average views per car
    average_views = mean(views_per_car.countOfEachCar)

    # Save results to analysis_results.csv
    resultData = [
        ["average_car_price", average_price],
        ["percentage_cars_with_sunroof",
        ↪  sunroof_percentage],
        ["average_views_per_car_last_30_days",
        ↪  average_views]
```

	Skills Module	Sub-Skills
E1	**Exploratory Data Analysis and Visualization**	Descriptive Statistical Analysis, Data Distributions and Correlations, Hypothesis Testing and Interpretation, Data Visualization Principles, Data Visualization Tools and Libraries, Visualizing Multivariate Data, Geospatial Data Visualization, Time Series Data Visualization, Interactive Visualization Techniques, Communicating Insights derived from Analysis
E2	**Data Cleaning and Preprocessing**	Identification and Handling Missing Data, Data Cleansing Techniques, Patterns and Exceptions, Data Type Conversions, Data Normalization and Standardization Processes, Outlier Detection and Treatment, Handling Noisy Data, Dataset Partitioning Strategies for Model Building, Feature Engineering Strategies, Dimensionality Reduction Techniques, Text Data Cleaning and Preprocessing, Implementation of Data Imputation Methods, Dealing with Imbalanced Data, Effective Data Sampling Techniques, Categorical Data Preprocessing Techniques, Applying Data Validation Rules
E3	**Machine Learning and Predictive Modeling**	Supervised Machine Learning Techniques, Unsupervised Machine Learning Techniques, Reinforcement Learning Methods, Feature Engineering and Selection Strategies, Predictive Modeling Methods, Machine Learning Algorithms Optimization, Machine Learning Model Interpretability, Ensemble Machine Learning Techniques, Applications of Machine Learning in Data Science, Handling Imbalanced Data in Machine Learning, Cross-Validation Techniques in Machine Learning, Hyperparameter Tuning in Machine Learning
E4	**Model Evaluation, Validation, and Selection**	Descriptive Metrics and Measures, Evaluating Model Accuracy, Evaluating Model Robustness and Stability, Evaluating Model Explainability and Interpretability, Performance Metrics for Classification Models, Performance Metrics for Regression Models, Performance Metrics for Clustering Models, Handling Overfitting and Underfitting, Hyperparameter Tuning and Optimization Methods, Validation Techniques – Cross-Validation, Bootstrapping, Selection of Best Performing Model, Model Comparison and Benchmarking, Power Analysis and Sample Size Calculation, A/B Testing and Experimental Design, Real-time Model Performance Monitoring, Model Retraining Strategies.

Table 2: Overview of Essential Skills Modules and their sub-skills for a Data Scientist

```
    ]
    writeToCSV("analysis_results.csv", resultData)
```

4 TASK 2

Skills Module

- Data Cleaning and Preprocessing

✓ What's Evaluated

- Combining multiple CSV files with potential typographical differences in headers
- Calculating aggregated counts over a specific time range
- Writing final combined data into a single CSV

✗ What's Not Evaluated

- Complex filtering beyond the specified date range
- Integration with external APIs or live databases

Description

You have the same data about used cars split across four CSV files (`cars_0.csv` to `cars_3.csv`), each containing slightly different column name spellings. Each CSV provides the following columns (not necessarily spelled correctly):

- `car_id`
- `doors`
- `engine_size`
- `mileage`
- `horsepower`
- `has_sunroof`
- `age`
- `price`

You also have `car_views.csv`:

- `viewer_id`
- `car_id`

	Skills Module	Sub-Skills
S1	**Deep Learning and Neural Networks**	Understanding Neural Networks and their Architecture, Back-Propagation and Gradient Descent for Neural Networks, Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) and Long Short-Term Memory Units (LSTMs), Generative Adversarial Networks (GANs), Transfer Learning in Deep Learning, Autoencoders and their Applications, Deep Learning Libraries and Frameworks, Hyperparameter Tuning in Deep Learning, Training and Validation of Neural Networks, Overfitting and Dropout in Deep Learning, Model Interpretability in Deep Learning
S2	**Big Data Processing**	Data Warehousing Concepts and Data Lake Architectures, Distributed Computing Principles, Big Data Processing Frameworks (like Hadoop and Spark), Large Scale Data Storage (such as NoSQL Databases), Stream Processing and Real Time Analytics, Batch Processing Techniques, Data Partitioning and Sharding Strategies, Data Processing in Cloud Environments, Big Data Pipelines and ETL Processes, Optimization Techniques for Big Data Processing, Parallel and Distributed Algorithms for Data Processing, Scaling of Data Processing Systems
S3	**Data Privacy, Governance, and Ethics**	Understanding Ethical Implications in Data Science, Maintaining Data Privacy and Security, Regulatory Compliance in Data Handling, Development of Ethical Guidelines in Data Science, Responsible Use of Data and AI, Implementing Data Governance Frameworks, Data Rights and Consent Management, Ethical Considerations in Predictive Modeling, Transparency and Explainability in Machine Learning Models

Table 3: Overview of Specialized Skills Modules and their sub-skills for a Data Scientist

	Skills Module	Sub-Skills
T1	**Exploratory Data Analysis and Visualization**	Descriptive Statistical Analysis, Data Distributions and Correlations, Hypothesis Testing and Interpretation, Data Visualization Principles, Data Visualization Tools and Libraries, Visualizing Multivariate Data, Geospatial Data Visualization, Time Series Data Visualization, Interactive Visualization Techniques, Communicating Insights derived from Analysis
T2	**Data Cleaning and Preprocessing**	Identification and Handling Missing Data, Data Cleansing Techniques, Patterns and Exceptions, Data Type Conversions, Data Normalization and Standardization Processes, Outlier Detection and Treatment, Handling Noisy Data, Dataset Partitioning Strategies for Model Building, Feature Engineering Strategies, Dimensionality Reduction Techniques, Text Data Cleaning and Preprocessing, Implementation of Data Imputation Methods, Dealing with Imbalanced Data, Effective Data Sampling Techniques, Categorical Data Preprocessing Techniques, Applying Data Validation Rules
T3	**Machine Learning and Predictive Modeling**	Supervised Machine Learning Techniques, Unsupervised Machine Learning Techniques, Reinforcement Learning Methods, Feature Engineering and Selection Strategies, Predictive Modeling Methods, Machine Learning Algorithms Optimization, Machine Learning Model Interpretability, Ensemble Machine Learning Techniques, Applications of Machine Learning in Data Science, Handling Imbalanced Data in Machine Learning, Cross-Validation Techniques in Machine Learning, Hyperparameter Tuning in Machine Learning

Table 4: Target Skill Sets

- `view_date`

You need to retrieve the number of views for each car over the last 30 days (from March 16th, 2023, to April 15th, 2023) and merge it with the car data. The final result should reside in `collected.csv` and must have:

- `car_id`
- `doors`

- engine_size
- mileage
- horsepower
- has_sunroof
- age
- price
- num_views

Solution (pseudocode)

```
FUNCTION retrieveData(carData, carViews, daysToQuery):
    # Calculate the threshold date as 30 days before a
    ↪  reference date of 2023-04-15
    start_date = "2023-04-15" minus daysToQuery
    parsedViews = parseDates(carViews, "view_date")
    recentViews = filterViewsAfter(parsedViews,
    ↪  start_date)

    # Count how many views each car received
    viewsCount = groupAndCountById(recentViews, "car_id")

    # Merge the aggregated view counts with the car data
    merged = mergeLeft(carData, viewsCount, on="car_id")
    fillMissingCounts(merged, "num_views", 0)
    return merged


FUNCTION solution():
    # Standardize column names for each CSV of cars
    carCSVs = ["cars_0.csv", "cars_1.csv", "cars_2.csv",
    ↪  "cars_3.csv"]
    correctedCarData = []
    for file in carCSVs:
        df = readCSV(file)
        fixColumnNames(df, ["car_id", "doors",
        ↪  "engine_size", "mileage", "horsepower",
        ↪  "has_sunroof", "age", "price"])
        correctedCarData.append(df)
    allCars = concatenate(correctedCarData)

    carViews = readCSV("car_views.csv")
    combined = retrieveData(allCars, carViews, 30)

    # Write final combined data into collected.csv
    writeCSV("collected.csv", combined)
```

5 TASK 3

Skills Module

- Data Cleaning and Preprocessing

✓ What's Evaluated

- Handling missing data
- Encoding categorical data using an ordinal scheme
- Applying Standard Scaling to numerical columns

✗ What's Not Evaluated

- Producing graphs or visual summaries
- Interpretation of EDA results

Description

You have a dataset describing used cars, split into training and testing sets at `data/train.csv` and `data/test.csv`. Each set includes:

- car_id
- doors
- engine_size
- mileage
- horsepower
- has_sunroof
- age
- num_views
- condition_name (categorical: "Excellent", "Good", "Fair", "Poor")
- price

Perform the following operations:

1) Fill missing values in `age` with the mean age (round down to the nearest integer).
2) Convert `condition_name` into numeric values using a consecutive integer encoding starting from 0.
3) Apply standard scaling to the `mileage` and `horsepower` columns, using only statistics from the training set to avoid data leakage.

Finally, save the resulting `train` and `test` data into `processed_train.csv` and `processed_test.csv`, respectively.

Solution (pseudocode)

```
FUNCTION processData(trainData, testData):
    # 1. Fill missing values in 'age' with the floor of
    ↪  the mean age from the training set
    meanAgeTrain = calculateMean(trainData.age)
    trainData.age = replaceMissing(trainData.age,
    ↪  floor(meanAgeTrain))
    testData.age  = replaceMissing(testData.age,
    ↪  floor(meanAgeTrain))

    # 2. Ordinal encoding for 'condition_name'
    encodingMap = {
        "Poor": 0,
        "Fair": 1,
        "Good": 2,
        "Excellent": 3
    }
    trainData.condition_name =
    ↪  mapValues(trainData.condition_name, encodingMap)
    testData.condition_name  =
    ↪  mapValues(testData.condition_name,  encodingMap)

    # 3. Standard scaling for 'mileage' and 'horsepower'
    mileageMean = mean(trainData.mileage)
    mileageStd  = std(trainData.mileage)
    horsepMean  = mean(trainData.horsepower)
    horsepStd   = std(trainData.horsepower)

    trainData.mileage    = roundToSix((trainData.mileage
    ↪  - mileageMean) / mileageStd)
    testData.mileage     = roundToSix((testData.mileage
    ↪  - mileageMean) / mileageStd)
```

```

    trainData.horsepower = roundToSix((trainData.horsepower - horsepMean) / horsepStd)
    testData.horsepower  = roundToSix((testData.horsepower  - horsepMean) / horsepStd)

    return (trainData, testData)


FUNCTION solution():
    originalTrain = readCSV("data/train.csv")
    originalTest  = readCSV("data/test.csv")

    processedTrain, processedTest = processData(originalTrain, originalTest)

    writeCSV("processed_train.csv", processedTrain)
    writeCSV("processed_test.csv", processedTest)
```

6 Task 4

Skills Module

- Machine Learning and Predictive Modeling

✓ What's Evaluated

- Applying a regression model to predict car prices
- Combining training and validation sets before final training
- Producing predictions and saving them into a CSV file

✗ What's Not Evaluated

- Model explainability and interpretability
- Hyperparameter tuning beyond a minimal setup

Description

You have a processed dataset for used cars. It is divided into:

- `train.csv` (70% of data)
- `val.csv` (15% of data)
- `test.csv` (remaining 15% of data)

The `train.csv` and `val.csv` files include `price` values for each car (the target you want to predict). The `test.csv` file omits the `price` column. Your objective is to build a predictive model that yields car price predictions with minimal error. The performance is calculated by the Mean Absolute Error (MAE). You can evaluate your approach on `val.csv`, then produce predictions against `test.csv`. Output your predictions as a single column named `price` to a file called `predictions.csv`.

Solution (pseudocode)

```
FUNCTION predictPrices(trainSet, valSet, testSet):
    # Separate features and target in train and validation sets
    X_train = dropColumn(trainSet, "price")
    y_train = getColumn(trainSet, "price")

    X_val = dropColumn(valSet, "price")
    y_val = getColumn(valSet, "price")

    # Define a regression model (e.g., random forest)
    model = createRandomForestRegressor(numTrees=200, seed=42, metric="mae")

    # (Optional) combine X_train & X_val + y_train & y_val for final training
    X_combined = combineRows(X_train, X_val)
    y_combined = combineRows(y_train, y_val)

    # Fit model on combined data
    model.fit(X_combined, y_combined)

    # Predict on test data
    predictions = model.predict(testSet)

    return predictions


FUNCTION solution():
    # Load each dataset
    trainData = readCSV("train.csv")
    valData   = readCSV("val.csv")
    testData  = readCSV("test.csv")

    # Generate predictions
    y_pred = predictPrices(trainData, valData, testData)

    # Write to predictions.csv
    writePredictions("predictions.csv", y_pred)
```

7 Conclusion

This research paper presents a novel framework, the CodeSignal Skills Taxonomy, established for the comprehensive evaluation of skills across a multitude of IT roles, particularly focusing on the role of a Data Scientist. The framework effectively captures the rigors of real-world job tasks, maps associated psychological and cognitive attributes onto specific skill areas, and further categorizes these skill areas as either foundational, essential, or specialized. It also introduces the Job Fit Score, a robust scoring system measuring an individual's hard skills compatibility for a job, thereby providing an empirical indicator of their qualification for a specific role. The framework adheres to a methodical content validation process, ensuring authentic replication of true job demands, thereby better capturing and representing on-the-job behaviors and skills. This research serves as a benchmark in the IT recruitment space, revolutionizing candidate qualification assessment by providing a standard, comprehensive, and precise skills evaluation model.

References

[1] B. Bloom, "Taxonomy of educational objectives: The classification of educational goals," *Handbook; Cognitive domain*, vol. 1, 1956.

[2] L. W. Anderson, D. R. Krathwohl, P. Airasian, K. Cruikshank, R. Mayer, P. Pintrich *et al.*, "A revision of bloom's taxonomy of educational objectives," *A Taxonomy for Learning, Teaching and Assessing. Longman, New York*, 2001.

[3] CodeSignal, "Understanding coding score 2023." *CodeSignal Knowledge Base*, 2023. [Online]. Available: https://support.codesignal.com/hc/en-us/articles/13261190299287-Understanding-Coding-Score-2023