Компетентності рівня junior для Data Science	Загальна оцінка компетентності (max. 15)	Компанії				
		DataArt	GlobalLogic	IT-Jim	Quantum	Avenga
1. Аналіз вимог та методології розробки програмного забезпечення в контексті Data Science (DS)						
1.1. Методології Data Science. Розуміння життєвого циклу та гнучких методологій розробки програмного забезпечення Розуміє основні моделі життєвого циклу розробки програмного забезпечення та їх застосування в контексті Data Science. Знає концепції та процеси Scrum: - ролі (Roles and responsibilities); - артефакти (Artifacts: Scrum board, Cards, Epics, Stories, Tasks, Backlogs, Sprint, Increment, etc.); - заходи (Ceremonies: Release planning, Sprint planning, Daily Scrum, Sprint demo, Sprint retrospective); - оцінки (Estimation: Story Points, Velocity, Sprint Burndown Chart, Feature Burnup Chart). Розуміє методологію Kanban, здатний створювати та підтримувати Kanban-дошки з відображенням завдань, їх статусів та прогресу. Знає методологію Waterfall та її відмінності від гнучких методологій. Розуміє життєвий цикл проекту та основні принципи CRISP-DM (project lifecycle, phases).	**10**	1	3	2	3	1
1.2. Оцінка обсягу робіт та ризиків в розробці програмного забезпечення Знає концепцію обсягу робіт (Scope Concept) у розробці програмного забезпечення. Розуміє відмінність між переоцінкою та недооцінкою (Overestimate vs Underestimate) труднощів проекту, здатний визначати та розрізняти вартість та складність завдань для правильної оцінки їх тривалості. Вміє застосовувати процес декомпозиції та рекомпозиції (Decomposition and Recomposition) завдань для ефективної управління проектом. Розуміє методи оцінки на основі аналогій (Analogy-based estimations) та їх застосування в процесі планування робіт. Використовує принципи оцінки робіт на основі історій (Story-based estimations) для покращення точності прогнозування ресурсів.	**8**	2	2	1	2	1
1.3. Вимоги та характеристики в розробці програмного забезпечення Знає визначення вимог до програмного забезпечення та їх роль у процесі розробки. Розуміє характеристики вимог і їх вплив на весь життєвий цикл проекту. Розуміє різні рівні вимог у розробці програмного забезпечення, такі як бізнес-вимоги, функціональні вимоги, технічні вимоги тощо. Вміє виявляти та аналізувати найпоширеніші ризики, пов'язані з вимогами. Здатний встановлювати та ідентифікувати осіб або команди, які відповідають за визначення, уточнення та затвердження вимог. Розуміє важливість детальних та конкретних характеристик вимог для ефективної і успішної реалізації проекту.	**10**	2	2	3	2	1

2. Робота з базами даних у контексті Data Science						
2.1. Розробка моделей даних для аналізу в Data Science Розуміє концепції транзакцій та їх ролі у збереженні даних. Знає типів зв'язків між таблицями: Один-до-одного, Один-до-багатьох, Багато-до-багатьох і їх вплив на аналіз даних. Вміє використовувати концепції нормалізації даних для забезпечення ефективного збереження та опрацювання інформації.	**11**	3	3	1	2	2-3
2.2. Мова структурованих запитів SQL Має навички створення та видалення таблиць та схем, оновлення схем для побудови оптимізованої структури бази даних. Вміє виконувати операції CRUD (створення, читання, оновлення, видалення) для маніпулювання даними. Розуміє та ефективно використовує унікальні (UK), зовнішні (FK) та первинні (PK) ключі для забезпечення цілісності даних. Здатний використовувати запит SELECT для вибору потрібної інформації з бази даних та виконання різних операцій з даними, таких як з'єднання, агрегації та сортування. Вміє об'єднувати результати кількох запитів за допомогою операторів UNION, EXCEPT, INTERSECT, MINUS. Має досвід використання агрегацій (ORDER BY, GROUP BY, HAVING, SUM, COUNT, AVG тощо) для впорядкування, групування, фільтрації даних та обчислення статистичних показників. Застосовує оператор WITH для створення спільних частин запитів або тимчасових наборів даних. Знає принципи використання підзапитів для отримання додаткової інформації або обробки даних, має навички роботи з представленнями (views) для створення віртуальних таблиць на основі запитів.	**13**	3	3	2	2	3
2.3. Використання NoSQL баз даних у Data Science Розуміє відмінності між реляційними (SQL) та документальними чи графовими (NoSQL) базами даних та їх відповідність типу даних у Data Science. Знає особливості NoSQL баз даних та їх можливості у збереженні та обробці структурованих та неструктурованих даних.	**10**	3	2	1	2	2-3
3. Технології та алгоритми у Data Science						
3.1. Методи машинного навчання (Machine Learning). Підготовка та оптимізація даних Усвідомлює проблеми перетренування та недотренування у моделях машинного навчання. Вміє застосовувати основні методи попередньої обробки даних (нормалізація, стандартизація, бінаризація, кодування міток, one-hot encoding, заповнення пропущених значень) Використовує основні методи оптимізації (градієнтний спуск, стохастичний градієнтний спуск, вибір швидкості навчання, ініціалізація ваг, міні-підхід у оптимізації) Здатний здійснювати валідацію результатів за допомогою методів hold-out, leave-one-out, перехресної валідації (cross validation), включаючи time series cross validation та оцінювати їх ефективність	**15**	3	3	3	3	3

3.2. Методи машинного навчання (Machine Learning). Аналіз даних Здатний застосовувати різні варіанти застосування методів машинного навчання для розв'язання конкретних завдань (класифікація, регресія, кластеризація, прогнозування, ранжування, виявлення аномалій та викидів). Знає методи вимірювання відстаней та їх використання (Eucledian, Cosine, Manhattan, Levenshtein) Використовує різні методи класифікації (логістична регресія, дерево рішень, k-nearest neighbors) та здатний оцінювати їх результати за допомогою метрик якості (Precision, Recall, F1 measure, Confusion matrix). Використовуває різні методи регресії (лінійна регресія, дерево регресії, KNN регресія) та метрики оцінки регресії (MAE, MSE, MAPE). Використовує методи кластеризації (K-means, Agglomerative Clustering, DBSCAN) для групування даних. Розуміє методи редукції розмірності даних, зокрема PCA (Principal Component Analysis). Розуміє основи прогнозування часових рядів та методи їх аналізу (Exponential moving average, Autoregression, PACF, Stationary time series, Seasonality, Differencing). Знає основні методи виявлення викидів (Outlier Detection) та вміє їх застосовувати(z-score, IQR, Isolation forest, KNN distance).	**15**	3	3	3	3	3
3.3. Комп'ютерний зір Розуміє поняття кольорового простору та основні методи проекції зображень: подібність, евклідова, афінна, проективна, та мінімальна кількість точок для оцінки кожної з них. Вміє вирішувати задачі у комп'ютерному зорі: сегментація, виявлення об'єктів, класифікація, вирівнювання зображень. Знає основи арифметики зображень: переповнення/недоповнення, конвертація між типами (int, float) та операції векторизації та растрування. Вміє використовувати ключові точки та описи для зображень та розуміє принципи роботи алгоритму ORB. Здатний здійснювати зменшення та збільшення розміру зображень за допомогою найближчого сусіда та білінійної інтерполяції. Використовує Canny edge detector для виявлення меж на зображенні. Розуміє принципи дії Hough detectors для виявлення ліній та кола на зображенні та методів морфологічної обробки зображень (erosion, dilation, open, close, kernel) Розуміє сновні поняття та методи фільтрації та ядер, такі як гаусівське розмиття та медіанне розмиття.	**13**	3	2	3	3	2
3.4. NLP Знає принципи базової підготовки тексту, такої як очищення, токенізація, вилучення стоп-слів та автокорекція. Використовує методи представлення тексту, такі як векторні вкладення, мішок слів, матриця термін-документ та TF-IDF. Вміє вирішувати задачі обробки природної мови, такі як моделювання авторегресії мови, визначення іменованих сутностей, аналіз настроїв, сегментація тем та визначення відносин. Здатний використовувати Техніки скрапінгу та кроулінгу для отримання текстових даних з веб-сторінок.	**14**	3	2	3	3	3
3.5. Deep Learning Знає основні архітектурні елементи нейронних мереж, такі як багатошаровий персептрон (MLP), автоенкодери та Unet. Використовує різні функції активації, такі як ReLU, сигмоїд, тангенс гіперболічний та інші. Здатний застосовувати методи оптимізації для навчання нейронних мереж, такі як стохастичний градієнтний спуск (SGD), SGD з моментом, Adam. Використовує техніки регуляризації для запобігання перенавчанню, такі як Dropout, зменшення ваги, раннє зупинення та аугментація даних. Знає принципи роботи згорткових та пулінгових шарів у зображеннях. Розуміє ключові поняття та ідеї застосування рекурентних нейронних мереж (RNN) та самоуваг. Усвідомлює значення різних функцій втрат та методів покращення навчання нейронних мереж.	**15**	3	3	3	3	3

3.6. GEO Знає основні системи координат, такі як WGS84 (EPSG:4326) та UTM, та їхнє призначення. Використовує різні формати зберігання геоданих у відповідності до потреб проекту. Застосовує бібліотеку rasterio для зчитування та запису растрових даних та роботи з профілями та метаданими. Здатний використовувати бібліотеку GeoPandas для роботи з геоданими у форматі GeoJSON, керування системами координат, виконання просторових об'єднань та зовнішніх пошуків. Усвідомлює принципи роботи бібліотеки Shapely для роботи з геометричними об'єктами та їх властивостями. Знає основні характеристики та особливості доступу до супутникових зображень, таких як Sentinel, Landsat. Використовує основні інструменти ГІС, такі як QGIS, для завантаження та обробки зображень та векторних даних. Застосовує спектральні індекси для виведення корисної інформації з супутникових зображень.	**8**	2	1	2	1	2
4. Робота з кодом						
4.1. Якість коду Знає стандарт оформлення коду PEP 8 для Python. Розуміє призначення та особливості використання інструментів для форматування коду: Black та isort. Вміє застосовувати лінтери, такі як flake8, pylint для забезпечення високої якості коду та дотримання стандартів. Використовує файл requirements.txt для управління залежностями проекту та його відтворюваністю. Розуміє основні складові файлу README та його роль у документуванні проекту та забезпеченні зручності співпраці та використання проекту.	**14**	2	3	3	3	3
4.2. Code Review Розуміє поняття перегляду коду (code review) і його роль у процесі розробки програмного забезпечення. Знає основні аспекти, на які слід звертати увагу під час кодового огляду. Усвідомлює важливість колективної перевірки коду для покращення якості, стабільності та безпеки програмного забезпечення. Вміє користуватись інструментами проведення кодового огляду, такими як GitHub Pull Requests або кодові огляди в IDE, знає їх переваги та обмеження.	**12**	2	2	3	2	3
4.3. Рефакторинг коду Знає основні концепції рефакторингу. Вміє виконувати рефакторинг коду для покращення його читабельності, розширюваності та підтримуваності. Використовує різні прийоми та підходи до рефакторингу, такі як зміна імені змінних, екстракція методів або класів, уникання дублювання коду тощо, для поліпшення якості коду та прискорення розробки.	**11**	2	2	3	2	2
5. Практична обробка даних та програмування в Data Science						
5.1. Використання операційної оболонки (Shell) Знає основні команди для навігації по файловій системі, такі як ls, cd, cp, mv, а також команди для роботи з вмістом файлів, такі як cat, tail, head, less. Розуміє концепцію wildcards. Вміє використовувати команди для отримання довідки про інші команди за допомогою man/--help, а також для встановлення програм через менеджери пакунків. Вміє встановлювати дозволи на файли за допомогою chown та chmod. Розуміє концепції користувачів, груп, root та sudo. Здатний працювати з віддаленими машинами через протокол ssh та використовувати різні утиліти, такі як scp, rsync, tmux та screen для керування процесами на віддалених машинах. Застосовує команди для перевірки використання ресурсів, такі як htop, top, а також для керування процесами. Має базові навички використання bash для написання скриптів, включаючи виконання команд в певному порядку, роботу зі змінними та аргументами командного рядка.	**11**	2	2	2	3	1-2

5.2. Основи мови програмування Python Знає синтаксис мови програмування Python (відступи, лапки, імена, коментарі), ключові слова. Вміє використовувати конструкції мови Python для написання програм та обробки даних, працювати з типами даних та змінними, використовувати умовні оператори та цикли/ Здатний виконувати роботу з файлами, включаючи читання та запис даних, а також введення та виведення інформації в консоль. Використовує вбудовані функції та методи для опрацювання рядків, списків та інших структур даних.	**15**	3	3	3	3	3
5.3. Розширена робота з даними та об'єктами Python Розуміє принципи та особливості роботи з функціями, класами та об'єктами в Python. Вміє працювати зі списками, кортежами, рядками та іншими послідовностями даних, використовуючи умовні вирази, сортування та вбудовані функції. Використовує регулярні вирази для пошуку та заміни текстових даних. Усвідомлює вплив імен та областей видимості на роботу програми. Знає основні поняття класів, об'єктів, наслідування та поліморфізму. Використовує інтроспекцію для отримання інформації про об'єкти та модулі. Здатний обробляти винятки та використовувати власні класи винятків. Знає основи роботи з бітами та бітовими операціями. Працює з серіалізацією об'єктів у форматах JSON та pickle. Розуміє важливість тестування та журналювання коду для підтримки якості програмного забезпечення. Знає основні функції та можливості стандартної бібліотеки Python для роботи з файлами, датою та часом, мережею та командним рядком.	**13**	3	3	3	3	1-3
5.4. Бібліотеки обробки даних та візуалізації Знає бібліотеки OpenCV, NumPy, Pandas та відповідні функції для завантаження, обробки та аналізу зображень та даних. Використовує інструменти візуалізації для побудови графіків та діаграм з даних, такі як Matplotlib, Seaborn або Plotly. Знає функції бібліотеки OpenCV для завантаження, збереження та обробки зображень. Вміє змінювати кольорові моделі та виконувати порогову обробку та морфологічні операції.Здатний змінювати розмір зображення з різними типами інтерполяції для отримання необхідних результатів. Знає оновні концепції NumPy, такі як зрізи, трансляція, транспонування та індексація. Застосовує операції з масивами та трансформації в бібліотеці NumPy. Здатний виконувати індексацію масивів NumPy за допомогою інших масивів або масок. Використовує методи роботи з даними в бібліотеці Pandas, такі як groupby, merge, apply і звернення за допомогою loc і iloc. Застосовує різні способи об'єднання, фільтрації та агрегування даних у Pandas	**13**	3	3	3	2	2-3
5.5. Фреймворки машинного навчання та обробки тексту Знає основні фреймворки та бібліотеки для машинного навчання та обробки тексту, такі як PyTorch/PL, Tensorflow/Keras, scikit-learn, xgboost, albumentations та spaCy. Вміє використовувати тензори та здійснювати операції з ними у вибраному фреймворку (PyTorch/PL або Tensorflow/Keras). Здатний завантажувати дані, підготовлювати набори даних та створювати завантажувачі даних для навчання моделей. Вміє навчати моделі за допомогою методу fit та здійснювати передбачення за допомогою методу predict. Використовує базові оцінювачі, такі як класифікатори та регресори, застосовувати різні методи оцінки моделей, такі як train_test_split, KFold та різні метрики. Вміє створює конвеєри для обробки даних та моделювання, включаючи попередню обробку, вибір ознак, навчання моделі та післяобробку результатів. Здатний зберігати та завантажувати навчені моделі з використанням відповідних методів.	**15**	3	3	3	3	3

5.6. Концепції асинхронного програмування та паралельного виконання Знає основні принципи роботи з потоками в операційних системах та програмуванні. Розуміє природу проблеми блокування(deadlock) та її наслідки для програми. Здатний виявляти потенційні ситуації deadlock та розробляти стратегії їх уникнення. Здатний оптимізовувати виконання Python-коду з урахуванням GIL(Global Interpreter Lock), використовуючи асинхронний підхід. Знає основні функції та методи для роботи з потоками в операційних системах, що підтримують стандарт POSIX. Вміє створювати, запускати, синхронізувати та керувати потоками, використовуючи бібліотеку POSIX у своїх програмах, в тому числі з використанням умовних змінних. Створювати, ініціалізувати, чекати та сигналізувати умовні змінні для ефективної синхронізації роботи потоків. Розуміє поняття mutex (взаємне виключення) та як використовувати його для захисту критичних секцій коду в багатопотокових програмах. Здатний створювати, ініціалізувати, захоплювати та звільняти семафори, забезпечуючи правильну синхронізацію між потоками. Знає види блокування доступу до ресурсів у багатопотокових програмах та як використовувати блокування читання/запису (r/w lock) для ефективного управління доступом до ресурсів Застосовує корутини для створення легковагових операцій та виконання багатозадачних програм.	**9**	2	1	2	2	2
5.7. Алгоритми Вміє визначати складність алгоритмів та використовувати O-нотацію для оцінки швидкості виконання. Використовує алгоритми сортування для оптимізації та вирішення завдань у реальних проектах(сортування бульбашкою, швидке сортування, сортування злиттям тощо). Розуміє логіку побудови та обходу дерев, а також їх використання в різних алгоритмічних завданнях. Здатний здійснювати бінарний пошук та використовувати хеш-таблиці для швидкого доступу до даних. Розуміє роль та значення стеків, черг та зв'язаних списків у програмуванні та можливість їх використання для оптимізації алгоритмів.	**13**	2	3	2	3	3
6. Розмовна англійська						
6.1. Правильна побудова фрази з узгодженням часу Вміє використовувати відповідні часові форми дієслів і структури речень для точного вираження часу в комунікації. Здатний правильно використовувати часові форми, такі як Present Simple, Present Continuous, Past Simple, Past Continuous, Future Simple, для передачі правильного значення в повідомленні.	**13**	2	2	3	3	3
6.2 Професійна it термінологія в розмові з замовником Знає і використовує професійні IT-терміни та скорочення, які використовуються в IT-індустрії, для чіткого та зрозумілого спілкування зі замовником. Вміє перекладати складні технічні концепції на доступну мову для замовника, не знайомого з IT-галуззю.	**13**	2	3	3	3	2
6.3 Професійна it термінологія у діловому звіті Знає і використовує спеціалізовану IT-термінологію та фразеологію при написанні ділових звітів. Вміє передати інформацію про технічні питання та проєкти зрозуміло та конкретно, використовуючи адекватні IT-терміни та абревіатури.	**12**	2	3	2	3	2
6.4 Професійна термінологія для Data Science Знає та розуміє специфічну термінологію, що використовується в контексті Data Science Розуміє та використовує специфічні терміни, патерни, алгоритми, фреймворки, бібліотеки та інші інструменти, які використовуються в Data Science	**13**	2	3	2	3	3

6.5 Неформальне спілкування (small talks) Вміє вести неформальну розмову, розпочинати діалоги та підтримувати невимушену атмосферу. Знає загальновживану термінологію та фрази, які використовуються у неформальних розмовах, таких як загальні питання про погоду, цікаві події, особисті захоплення, спорт, фільми тощо.	**11**	2	1	3	3	1-2

Зауваження від компаній

DataArt

3.3 - 3.6 Залежить від проекту, на який подається кандидат. Якщо проект стосується робраної теми - обов'язково треба знати перелічену інформацію. Якщо проект на іншу тему - бажано знати, але не обов'язково.

GlobalLogic

Пропонована матриця трохи зміщує акценти сучасних вимог до компетенцій джуніора. Якщо це буде цікаво - ось мій варіант, який (на мою суб"єктивну думку) відображає актуальний рівень вимог до цієї позиції.
https://docs.google.com/document/d/1cuECm4Hv2r5F-i6lomNPC_m98KnPK55H6kJud8nzT8k/edit?usp=sharing

Quantum

Англійська дуже важлива. Частіше за все ми відмовимо кандидату через відсутність достатнього рівня англійської, незалежно від рівня технічних скілів інженера.

Avenga

1.1 На позицію Junior Data Scientist важливим є розуміння циклу CRISP-DM. Тому я б це виділила окремо і поставила 3. Однак знання концепцій і процесів scrum, kanban чи waterfall достатньо лише поверхневих, зазвичай це більше робота Project Manager, особливо по створенню і підтримання дошок прогресу, плануванню проекту, від junior engineer цього не вимагається.

1.2 Для позиції Junior Data Scientist важливість навичок, пов'язаних з оцінкою обсягу робіт та ризиків у розробці програмного забезпечення, може бути дещо меншою порівняно з іншими технічними навичками, такими як аналіз даних, програмування та моделювання.

1.3 Загалом, ці знання корисні, але для початкової позиції Junior Data Scientist вони не є критичними. Набагато важливіше володіти базовими навичками роботи з даними, знаннями статистики, програмування та основними інструментами для аналізу даних. В подальшому, коли виникає необхідність працювати над більш складними проектами або управлінням проектами, ці знання можуть стати більш релевантними.

2.1 Щодо основ нормалізації, важливо знати концепцію щоб ефективно працювати з базами даних, уникати дублювання даних та забезпечувати їх цілісність. Але це не є критично необхідним на початковому етапі, більше уваги можна приділити в процесі роботи.

2.2 ці навички є критично важливими для роботи Junior Data Scientist і забезпечують основу для ефективної роботи з даними

2.3 Із стрімким розвитком GenerativeAI важливо виокремити Vector Databases для ефективного зберігання ембедінгів

3.3 Для позиції Junior Data Scientist знання і навички в комп'ютерному зорі можуть бути важливими, але їх значення залежить від конкретної ролі та вимог компанії.

3.4 Важливим є вміння працювати з сучасними методами роботи з текстом за допомогою generative AI і LLMs: моделі трансформерів, GPT-3/4, BERT

3.6 для позиції Junior Data Scientist загалом важливими є знання бібліотек GeoPandas та основ систем координат. Інші навички є нішовими, дуже залежить від конкретної сфери діяльності компанії.

4.3 для Junior Data Scientist, першочергові навички будуть зосереджені на розумінні базових концепцій машинного навчання, обробці даних і написанні функціонального коду. Рефакторинг коду є важливою навичкою, але не є обов'язковою умовою для початкового рівня

5.1 для Junior Data Scientist знання базових команд є важливими (оцінка 2-3), інші навички, такі як робота з віддаленими машинами, можуть бути корисними, але не є критичними для початкового рівня.

5.3 3 (Розуміння принципів роботи з функціями, класами та об'єктами в Python, Вміння працювати зі списками, кортежами, рядками та іншими послідовностями даних, Знання основних функцій та можливостей стандартної бібліотеки Python); 2 (Використання регулярних виразів для пошуку та заміни текстових даних, Знання основ роботи з бітами та бітовими операціями)

5.4 Вміння працювати з бібліотеками, такими як NumPy та Pandas, є критично важливим для більшості позицій Data Scientist, оскільки ці бібліотеки широко використовуються для обробки даних. Знання OpenCV може бути додатковим плюсом, але не завжди є обов'язковим, особливо для початкових позицій, якщо роль не фокусується на обробці зображень.