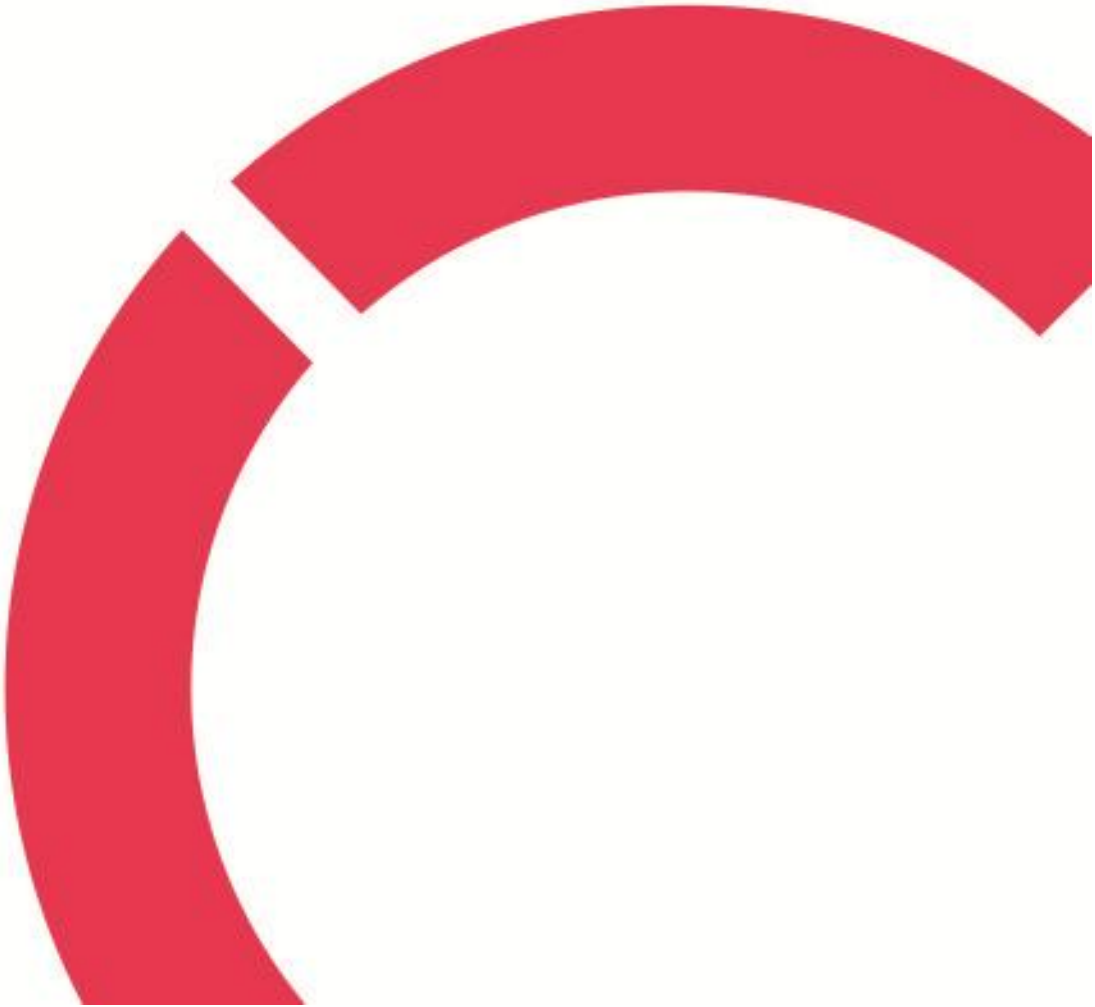
Md Shahil Ahmed Ayon

AI-ENHANCED CAREER PATHWAYS

Personalized Job Recommendations and Growth Suggestions

Thesis
CENTRIA UNIVERSITY OF APPLIED SCIENCES
Business Intelligence Technologies
November 2025



ABSTRACT

Centria University of Applied Sciences	Date	Author
	November 2025	Md Shahil Ahmed Ayon
Degree programme Business Intelligence Technologies		
Name of thesis AI-ENHANCED CAREER PATHWAYS. Personalized Recommendations and Growth Suggestions		
Centria supervisor Henry Paananen		**Pages** 22
Instructor representing commissioning institution or company Henry Paananen		

The aim of this project was to make a recommendation system for the students and job seekers who are willing get a guidance for AI-related job field. An artificial intelligence job postings dataset from open source was taken and cleaned and then used for training the model. This model has used simple content based filtering methods to make the model reliable and easy to understand for the user. Here all the job's required skills and user skills were encoded as vectors and used for calculating cosine similarity. Based on the cosine score the user will be recommended top 5 jobs along with matching and missing skills.

The outcome of this project has proven that this is a successful recommendation model for suggesting jobs to the user along clear explanation of matched and unmatched skills. Also with updating data and adding more interactive features, this model can be capable of providing career guidance in all the practical cases.

Key words
Artificial Intelligence, AI job market, career guidance, content-based filtering, job recommendation, skill matching

ABSTRACT

CONTENTS

FIGURES

1 INTRODUCTION

In this era of Artificial Intelligence (AI), AI is dominating all the industries and everyday it is changing how people work and what kind of work is coming in the market. This fast changing market is creating different types of new roles that never existed before and all these new jobs are coming in the market with new skill requirements. Specially in terms of AI, every other day new software and technologies are being created which are creating new skill requirements in this fast-changing job market. The OECD shows that the way of organizing work, performing task and skills demand in the job market is being dominated by AI. All this new changes comes with both opportunities and confusions with skills requirement for students and new AI professionals. And job skill requirement confusions often arise in the AI field because students see different names for the same job and do not understand what skills are really needed. Overall all these situations make job searching harder for the students and young professionals to choose a perfect job title based on their current skills. (Wilson, Daugherty & Morini-Bianzino, 2017, p. 14-16; OECD, 2023, p. 246-248.)

The aim of this thesis is to work on all the skills requirement confusions with job postings and make a solution for the students and new AI-based job candidates. To address these job searching challenges and to provide them a clear and reliable career suggestions, the goal of this thesis is to make a career recommendation model. This skill based career recommendation system has been trained based on analysing a public dataset of 15000 AI job postings. This recommendation model has used all the skills and the locations of every job to suggest a list of top five job titles to the user by comparing skills requirements of every jobs with the skills provided by the user. This model also provides an extra option to get suggestions based on their preferred locations. For these top five matched job titles, the model provides clear reasons why each was suggested and what skills the user can learn in future for preparing himself for that job.

This job recommendation model has been made based on the method which is called content-based filtering which means this model is trained only based on some data and it will directly focus on the skills from the dataset and skills input from the user. The good part of content-based filtering is that it keeps the model easy to understand, reliable and gives clear suggestions for the students and the job

seekers. But on the other hand it can not analyse previous choices, user history and also unable to update it's data at all. The final outcome of this thesis is a functional recommendation model which takes country choice and skill input from the user and in return provides a simple and clear career guidance.

This project has used open-source AI based job posting dataset from Kaggle. This project tried to develop the model without any complex techniques and use very simple matching logic. The overall idea is to help the IT students and job seekers along with educational institutions by introducing them with the connections between user skills and real world jobs and skill requirements. Then the outcome should be able to help the user with finding a perfect job and also helping them for the career growth in the field of AI.

2 OVERVIEW OF PHENOMENON

AI has opened a large number of jobs for the IT students and newly IT graduates in tech field as things are getting automated everywhere and mostly driven by AI. But the confusion appears when it comes to selecting the perfect job title based on the current skills of applicants or students (Wilson, H.J., Daugherty, P. and Bianzino, N., 2017). After the graduation or even for the internship IT students start looking for their job based on the job title in different job postings but unfortunately there are very few job available postings where employers mention their actual required job skills. Things get even more confusing when it comes to jobs related to AI. In this new era of AI, almost every company needs to have an AI sector along with their IT team. As they are also in the early stages of introducing their AI sector, they also sometimes need guidance to make the job post more accurate according to the skills and job title.

This project is addressing the challenges and confusions that AI-related job seekers and students often face when they see that required skills for AI-related jobs are changing very frequently. Often employers describe similar skills with different words or sometimes most of the job postings do not have proper description of required skills. This recommendation model is focused on helping students and early-career job seekers to find which job title which job titles best fit their current skills using simple demand signals from public job-post data. In this project Kaggle open source dataset "Global AI Job Market & Salary Trends 2025" was used but for this project only job title, company location and skills field were used excluding other data (Sajjad, 2025).

This project aimed to be a simple and explainable prototype model that can be helpful for students and AI-related job seekers to get suggestions of job title based on their current skills and location . This model can be helpful as a quick suggestions for checking one's own skills in short conversations instead of long reports. Besides job recommendation, this model will also show the missing skills for required job title. As this tool has simple approach and only public data , it will be easy to maintain and share within the student society and also the people who are trying to enter AI field at very beginning of their career or they want suggestions what to learn for driving their career to some AI-related jobs. Even though this model is focused to be helpful for students and recent graduates, educational institutions and career services also will be able to take the advantage of it by checking the jobs based on skills around their location.

3 THEORETICAL BACKGROUND

This chapter explains the main ideas and theories that support this thesis and makes all the following chapters easier to understand. The project is about connecting people's skills with job postings , so it is important to explain what is meant by artificial intelligence and how AI systems can be built and trained using raw data. It is also necessary to explain how machines can learn patterns from examples or data because the recommendation model in this thesis depends on training the model from open job posting data. Another point is that the quality of the input matters because unclear or messy skill text can lead to weak matching and confusing results. This background also helps to explain why the model is able to suggest in a clear way instead of acting like a black box. Overall, the chapter gives simple definitions and a shared base of understanding before the practical work is described. (Rahm & Do 2000, pp. 3–6.)

3.1 AI Basics

Artificial Intelligence (AI) is a large sector including machine learning and software development and the ultimate goal of AI is to make computers smarter and increase its efficiency. In 1956 , AI was initiated formally with coining the name, even though AI related work had already been ongoing for almost five years (Russell and Norvig, 1995). The goal of AI is to creating such computer systems that can perform task almost at a human intelligence level. AI can be defined like computer programming work including machine learning which is able to operate tasks on its own, understand the environment and not only able to handle repetitive tasks but also adapt to new changes. So overall AI helps computers and machines to copy the human activity and takes decisions independently to solve complex problems. (Russell and Norvig, 1995, p.1-4.)

Right now human life is dependent on AI for almost every single task starting from the bulb of our bedroom to the car we drive. Nowadays even for searching a small information AI has already taken oven that responsibility to help us. Devices like home pods or phone voice assistants are so advance that they can recognize their owner's voice and respond only to them. AI has done all these achievements with some core idea of learning using related data and also analysing the previous experiences for behaving intelligently. AI mostly includes the machine learning models such as this recommendation model, where the system is built only based on data. On the other hand, deep learning uses neural networks which are inspired by the human brain. Now computers are being able to analyse, learn or

adapt new things is not a miracle but it is the results of a lot of data and complex algorithms developed by AI. So AI is basically helping computers for being capable of thinking and behaving like humans. (Russell and Norvig, 1995, p.1-4.)

3.2 Machine Learning and Model Training

Machine Learning helps the computer to get trained from data or examples rather than just being programmed with some specific rules . It can also be explained as putting a lot of data into the computer so it can figure out its own way to find patterns. In practical example if a model need to be learned how to recognize a car , instead of telling it directly which one is car machine learning will learn by analysing many car photos or also analysing the structure of the car as well then step by step it will recognize the car by its own . Overall , machine learning helps the computers with improving efficiency and accuracy in any task by analysing experience and data just like a student gets better over time through practice. (Samuel, 1959.)

For making predictions and providing decisions , a machine learning model needs to be trained and this process is defined as Model Training. The previous example of recognizing a car can be taken into consideration and here model will train with the car images and data of car structure. Like practicing a skill, model will be able to adjust itself to provide more accurate answer along with gathering more data. In this project model will provide more accurate jobs to the user based on the job data. Overall , model training is basically feeding data the algorithms and makes the model more mature to provide a accurate result. (Mitchell, 1997, p.1-3.)

3.3 Algorithms and Libraries

Even if in the beginning of past decade, AI and machine learning was critical but all the open-source libraries and tools played a vital role in making them easier to learn and use. All of these libraries are the pre-build implementation of commonly used libraries and algorithms. As developers can get the premade and convenient frameworks , they do not need to reinvent the base from beginning. Scikit-learn can be a good example of a one of the most popular Python library which offers great prebuild machine learning algorithms with a tidy and simple interface. According to it’s creators’ claim,

“Scikit-learn is a Python module integrating a wide range of state-of-the-art machine learning algorithms for medium-scale... problems,” with an emphasis on ease of use and performance (Pedregosa et al. 2011, p. 2825).

Beside the scikit-learn, there are many other libraries that have been prepared to fulfil different AI model needs. Such as TensorFlow or PyTorch are being widely used for really complex models in deep learning and deep learning is like a sector of machine learning which involves neural networks . Also some most common libraries are Pandas and NumPy . There is also utilities like Joblib which are used for saving the trained model. All of this libraries have been made with the aim of saving developer’s time and help them with focusing on better problem solving aspect instead of making low level implementation of algorithms. So for faster development along with maintaining the standards, AI or machine learning development in totally depended on them. (Pedregosa et al. 2011, 2825–2826.)

3.4 Data Cleaning Basics

For AI and machine learning most important element is clean data and here cleaning refers to identifying problems or errors in data, then fixing them to make the data clear and reliable. The most common errors in an uncleaned dataset are spelling mistakes, NaN values or sometimes same things are written differently. So data cleaning deals with all these errors from detecting these to removing them and hand over a consistent and tidy data for model training . This step is really essential as a messy data can lead to a poor result, so often called “garbage in, garbage out” (Rahm & Do 2000). And some of the data cleaning process are like trimming extra spaces from text, filling empty values with a safe empty one to keep the data consistent and also proper spacing or separating them by comma and so on. All these actions make the data consistent and tidy before modelling for the expected consistent results. (Rahm & Do 2000, pp. 4–6.)

4 METHODS IMPLEMENTATION

This chapter describes how the project ideas were turned into a working model. The goal was not to build any complex system, but to offer a clear and useful prototype that can be understood by students and early-career job seekers. For that reason, the implementation focuses on a direct comparison between user skills and job skill requirements. The method was selected so that the model can explain its results in plain language such as showing what skills matched and what skills were missing. The chapter also explains the main steps of the implementation, from preparing skill lists to scoring and ranking job titles. This is important because a recommendation system is not only about the final output but also about how the output is produced. Overall, this section shows the logic behind the system and why it fulfils the goal of this project.

4.1 AI in Recommendation Model

The basic theory of this model is a recommender system which suggests items to the user based on how perfectly items match with the user's need. For any recommender system, there are two types of methods available. One of them is collaborative filtering where system learns from user's previous behaviour and analysing similar patterns. Another one is content-based filtering which focuses on user profile and item attributes such as skills or title in a job post. And this second method is a perfect match for this project as this project has solid job skills data and supports transparent reasoning. In the content based filtering method, both of the user data and job skills data has to be turned into some feature vectors and mostly cosine similarity helps the systems to score similarity and rank the jobs based on users' skills. Even though sometimes content based filtering might have the issue of overspecialization, still it is preferable in recommender-systems for it's explainability and almost zero dependency on historical clicks. (Lops, de Gemmis & Semeraro, 2011, pp. 75–79; Ricci, Rokach & Shapira, 2015).

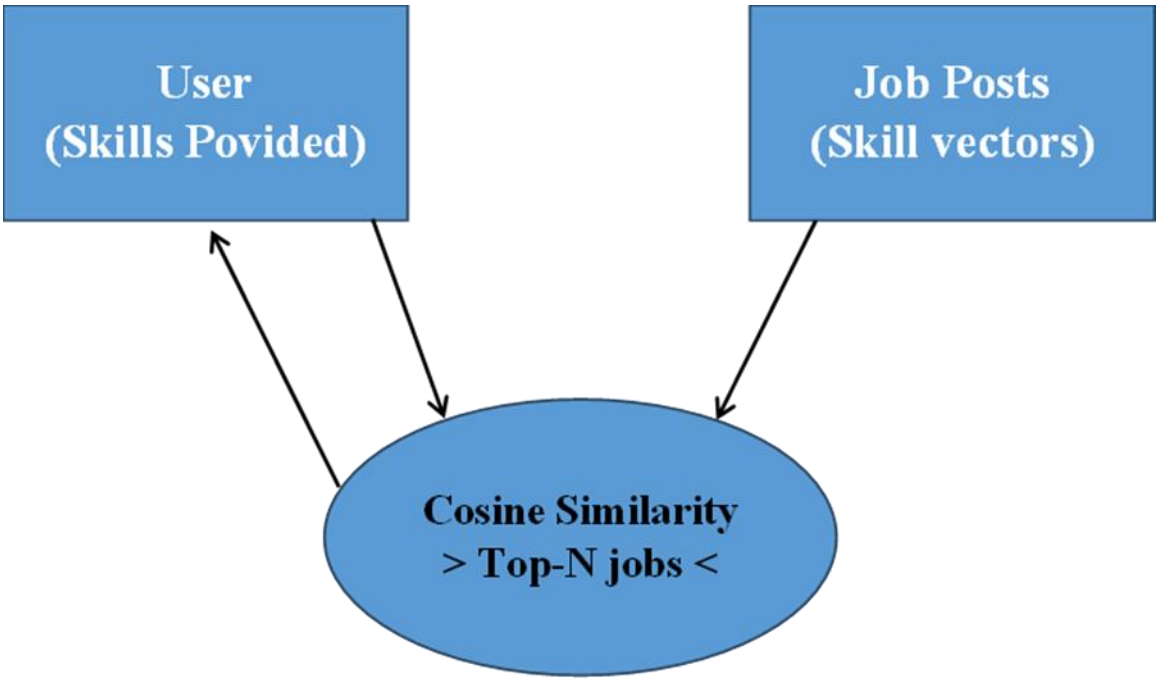


Figure 1Content Based Recommender

For pointing the matched and unmatched skills, Cosine similarity focus on the angle between the user and every job as vectors to rank the jobs and cosine only count the direction but no length. So, longer skills doesn't get extra point but only the overlapping ones. This is the standard content-based pattern: encode profiles as feature vectors and score relevance with a similarity function like cosine (Aggarwal, Recommender Systems: The Textbook, Ch. 4, pp. 139–141, 150–151).

4.2 Recommendation Model Training

In this ML recommendation model, all the job data of required skills must be converted into a row of binary matrix such as (jobs * skills) and here each skill is marked as 1, if it is required for a job else it is 0. Then the skills input by the user are encoded in the same space. After that this model checks the cosine similarity very easily and constantly as the required skills and user's skill both are present at the same place. Then for learning the skill vocabulary and converting skills data sets into a binary indicator matrix , the standard sciki-learn transformer MultiLabelBinarizer has been used in this model. This is a easily understandable and explainable as each column= a named skills and also very much compatible for the classic vector-space scoring. (Pedregosa et al., 2011, pp. 2825–2830; Manning, Raghavan & Schütze, 2009, p. 121.)

This model is working on data with many skills per user and jobs so all these user skills and jobs are eventually multi-level objects and the binary representation helps with avoiding unreliable frequency weight and also help with keeping the explanations straightforward. It works like “matched skills”= intersection and the “missing skills”= job minus user. Then a perfectly interpreted similarity comes with the cosine score which is mostly used in content-based recommendations. (Lops, de Gemmis & Semeraro, 2011, pp. 75–79; Manning, Raghavan & Schütze, 2009, p. 121.)

4.3 Libraries & Algorithms in Recommendation

For a successful recommendation model, It requires tools to clean data, convert text into numbers and calculate similarity in a stable way. Python libraries are some pre-made functions which are helpful for doing common tasks. Python libraries ensure the quality of the tasks and save user’s time. Algorithms are the sequential methods used in code to solve problems such as comparing two skill sets or transforming skills into a format (vector) that can be used. This project has used most of the common libraries and Algorithms, as these are already tested and proven their accuracy. So, in order to make a clear job recommendation model, all these libraries and algorithms were essential to use. (Pedregosa et al. 2011, 2825–2826.)

4.3.1 NumPy

In this model NumPy in one of the most essential part for handling maths quickly. For cosine similarity this model has used a lot of dot products and vector lengths. This is important because cosine similarity needs repeated calculations for many job rows, so maths take long time to run that would make the system feel laggy**.** NumPy runs these operations in optimized C and it can keep memory use very low even without making any difference in the result by using float32 as result even a normal laptop can run smoothly and NumPy’s core implementation is based on C . In Python, these are the reason why NumPy became the standard tool for mathematical tasks including arrays. (Harris et al., 2020.)

4.3.2 Pandas

For this project the data set which was used was from Kaggle is really big with a lot of irrelevant data for training the model . So for using the data, a cleaned csv file needed to be made with Pandas. As this model is suggesting jobs based on location and skills, only three columns- job title, location and required skills were needed. Then for fix casing and all the extra spaces and standardize separators in the skill field and also managing the missing values is done by Pandas. “load → clean → reshape” workflow is basically the job what Pandas was designed for in data-wrangling contexts. (McKinney, 2010). Pandas provides the result with perfect data frame where each row has the strings of consistent skills which are very important as they are going to be converted in binary vector per job.

4.3.3 Scikit-learn

After cleaning the data , it is necessary to turn every job’s skill into a binary row by using scikit-learn. MultiLabelBinarizer is a scikit-learn transformer, used for making binary matrix from label-data sets. Then Fittibg takes the skill vocabulary and at the same column the transforming converts every job. After saving the vocabulary and matrix both, vector from the user gets perfectly aligned according to the job matrix and helps cosine to compare perfectly. This is a common, well-documented approach for multi-label features in scikit-learn workflow (Pedregosa et al., 2011).

4.3.4 Cosine Similarity

Finally with the vectorization, the model will be able to score each job with the cosine between the user input skill vector and the job skill vector. Cosine just focuses on direction ignoring the raw length of the vector. That is the reason cosine can give more accurate result as listing too many data won’t have effect any way unless skills are overlapping. For a content-based recommendation, cosine similarity is the prior choice as it is naturally good for vector length normalization and perfect for models like this project which are dealing with keywords or skills. Then the job recommendation result comes with matched skills and which skills are missing for the job. (Mana & Sasipraba, 2021.)

4.4 Data Cleaning for Recommendation Model

In the information system, if the goal is a perfect recommendation result, it is important to input some high-quality data and it reflects the importance of data cleaning. The primary goal of data cleaning is not only identifying errors or inconsistencies in the data set but also fixing them. It can be various kind of errors such as typing errors or missing values, unstandardized formats and duplicate values. Here in schema-level , the classic methods are used to solve the problems in grouping fields and aligning and then in the instance-level, methods operate on the actual values.

This model has used those principles to job post data so that it writes everything in the same way based on the concept. First, from the raw dataset only job title, location and required skills were taken and text were normalized with lowercasing and trimming whitespaces. Then it is turn for parse the comma-separated skills into lists and standardizing aliases such as “Microsoft Azure / MS Azure”= “Azure” to avoid duplicates. Also for perfect location filtering , the company location or the country names were standardized otherwise, it might lead to unfair counting or matching. These are the instance-level issues which were fixed in real dataset to achieve the goal of data-cleaning and provide a perfect data set to use. Now before training the recommender the cleaned file can be validate by checking the short sample of rows and all the columns and no empty values which represents the literature’s guidance of cleaning data as quality of the modelling is limited by the quality of data. (Do, H.H. and Rahm, E, 2000.)

5 PROCESS DESCRIPTION

The main goal of this of project is to turn raw job posting data into a trained model where it can be helpful to the students or AI-based job candidates by suggesting job based on their skills and the company locations. And for making this recommendation model successful, the process started with cleaning the data, then training the model with that clean data and doing the inspection. Finally, it comes with the final recommendations.

5.1 Data input and setup

Data preparation is the very first step of any kind of ML model so this project have started with a open source job posting data set from Kaggle which is named as ai_job_dataset.csv. For this process Python library- Pandas has been used and as this dataset has a lot of information which are not required for this project, this data preparation process takes only the most necessary 3 columns which are job_title, company_location and the most important one which is required_skills.

Then for cleaning the job_title and Company_location colums, all the data were converted to string type and trimmed spaces from the beginning and end such as " NLP Engineer " to "NLP Engineer" and also with astype(str) missing cell has turned into the literal string nan to avoid any kind of errors. Then the next step is to clean the required_skills column. Here the nan values will be replaced with empty string ("") and converted to string along with splitting them on commas, trimming every pieces, dropping the empty pieces and again rejoining with a standard format which is comma + single space (", "). Then the output of the new clean data set as the FIGURE 2 below.

ai_job_dataset_clean.csv

```
4   AI Specialist,Switzerland,"Kubernetes, Deep Learning, Java, Hadoop, NLP"
5   NLP Engineer,India,"Scala, SQL, Linux, Python"
6   AI Consultant,France,"MLOps, Java, Tableau, Python"
7   AI Architect,Germany,"Data Visualization, R, SQL, Linux"
8   Principal Data Scientist,United Kingdom,"R, Docker, MLOps"
9   NLP Engineer,France,"Python, SQL, Computer Vision, Java, Azure"
10  Data Analyst,Singapore,"Hadoop, Git, Mathematics, Python"
11  AI Software Engineer,Austria,"MLOps, GCP, Scala, Azure, Linux"
12  Autonomous Systems Engineer,Sweden,"MLOps, Python, SQL"
13  AI Architect,South Korea,"R, Data Visualization, Python, Azure"
14  AI Consultant,France,"Tableau, Python, TensorFlow"
15  Autonomous Systems Engineer,Norway,"Scala, SQL, Statistics"
```

Figure 2 Cleaned csv file

5.2 Model Training

After data cleaning, the next step was model training, which means preparing the dataset in a form the model can use. Now the clean data set is ready with the necessary job title, location and required skills columns for training the model and this process step is to make the actual model that is going to be trained and do the recommendation. Now this ai_job_dataset_clean.csv is going to be loaded with the help of the Python library – Pandas. It is important to check if all three columns exist because it is better to catch errors earlier rather than finding them later while running the code. Now for confirming a reliable structure, the title and country columns, which are strings, need to be checked carefully. This will make the upcoming steps model training easier.

5.2.1 Preparing the clean skills

This normalizing process is meant to turn each job's skills into a tidy set, as even small differences could affect the vocabulary and weaken the matching results. This process begins with lower-casing and trimming the text such as " Python " > "python". Then the extra spaces need to be squeezed inside phrases and a small dictionary has been used to map common aliases under one short name.

```
    aliases = {
        "google cloud": "gcp", "google cloud platform": "gcp",
        "amazon web services": "aws", "ms azure": "azure", "microsoft azure": "azure",
        "tf": "tensorflow", "torch": "pytorch",
        "natural language processing": "nlp", "cv": "computer vision",
        "dl": "deep learning",
    }
```

Figure 3: Skill Dictionary

Then all the Skills cells were separated by commas and any empty pieces were dropped then a set going to be kept, so the inside every jobs there won't be any repetition of skills.

5.2.2 Encoding jobs and country filter

In this step a common space need to be build for both of the jobs and users. Now scikit-learn's MultiLabelBinarizer has been used for scanning all the jobs and learning the whole vocabulary and it encodes the jobs as 0 or 1 row over the master list of vocabulary . Here all job * skill matrix (X) has been stored as Float32 to save memory. Also with that same row order list of job title and country will be saved. It will help to make country filtering more faster by the country index which is a dictionary from each country to row numbers of that specific country and it will help the model to directly get the right row without going through the whole matrix all the time.

5.2.3 Packing everything

This is the last step of model training and here everything is going to pack in a single dictionary and save the model with joblib and the model has been saved as job_model.joblib. Here the element X refers to the job * skill matrix where every row is a single job and they are encoded as 0 or 1 over the skill vocabulary. Then Mlb_Classes is the column order for X and a list of skill names which has been learnend by MultiLabelBinarizer and the Job_tittle is the list of job title strings which is aligned with the rows of X. The element company_location is the list of country strings for every jobs and also aligned with rows of X and Required_skills_list is the sorted lists for every job skills which has been used for match-unmatched explanations. Lastly Countries, country_to_indices are for supporting quick country filtering.

5.3 Recommending from the model

Once training was complete, the model could be used to produce recommendations based on user input. In the recommendation stage, the model is loaded and the user provides a set of skills and an optional location choice. First of all this process will began with loading the model file with joblib.load and it provide the job-skill matrix X which is a NumPy array then comes the skill vocabulary mlb_classes and other readable data such as job_title and company_locations as well. Another important parts is fast country index which is the dictionary from country name to row number named as country_to_indices . Then its turn for the user input and inputs will be cleaned with the helper normalize_skills where words will be lower case and trimming spaces . Along with them a small aliases dictionary has been applied to map common values to one name such as "Google cloud Platfrom"= "gcp". Then again parse_skills_arg() has used re to clean the input data and return a clean set such as ("py-

thon". "sql"). The user's country input is cleaned, and fuzzy matching is used to handle any typing mistakes.

5.3.1 Fuzzy Matching For Country input

When a user enters a specific country input and makes a spelling mistake, this project has included a safety net using difflib.get_close_matches.. When the country has been cleaned with normalize_skill and then it is going to be compared with the keys in country_to_indices and then anything seems similar, it will automatically pick the correct one such as "Canadaa"= "Canada". But if nothing seems similar, the model will recommend from the global list. In Addition, Interactive_loop has been added to show some suggestions and choosing option from there. But fuzzy matching only used for the country and skills will be dependent on the normalize_skill and the aliases dictionary so that the cosine score come fair and reliable with the clean vocabulary.

5.3.2 Scoring with cosine

Here for the cosine similarity, a user vector will be created with NumPy and it is going to line up with the model's vocabulary. From the mlb_classes a quick index, idx = {skill: position} is going to be build which will be starting by a zero vector U and for every user skill that is existing in the vocabulary will make the entries to 1. And if the sum of U comes zero (No matches at all), system will stop early as there is nothing to compare. Then for measuring fit, the script will call cosine_scores(X, U) to count the cosine similarity with the help of NumPy's matrix X @ U and np.linal.norm . If any country was selected, it will only run on these specific rows from the index country_to_indices else it is going to score all the jobs. Then cosine counts only the overlapping skills and give a fair result without giving any extra point to longer skills such as skills with multiple words.

5.3.3 Ranking Results

After the cosine scoring np.argsort helps to keep the top results depending on the –top number which is set as 5 here. Then the main function recommend(…) returns all the information for every job and the informations are- the job title. Country name, matching score and most importantly the explanations of

skills which were matched and which one need to be learned . Here it calculates the common skills which appear with both of the user list and job's required_skills_list from the model and also (missing) the unmatched skills of the job with the help of Python sets. Then print as "match= Python, sql | Learn: AWS". And user instantly get why the job was recommended and what skills he needs to acquire.

5.4 Model Inspection

For inspecting the model job_model.joblib , a small health-check Python program called inspect_model.py was created. This step begins with opening the model with joblib.load and gives a instant overview so that it can be confirmed that everything seems normal in the model. It shows the elements in the model as displayed in the FIGURE 4.

inspect_model.py

```
import joblib

m = joblib.load('job_model.joblib')

print('Top-level keys:', sorted(m.keys()))
X = m['X']
print('X type:', type(X).__name__, 'dtype:', getattr(X, 'dtype', None), 'shape:', getattr(X, 'shape', None))
print('jobs:', len(m['job_title']))
print('countries count:', len(m['countries']))
print('vocab size (skills):', len(m['mlb_classes']))
print('sample titles:', m['job_title'][:3])
print('sample locations:', m['company_location'][:3])
print('sample skills list (first job):', m['required_skills_list'][0][:10])
print('country index example keys:', list(m['country_to_indices'].keys())[:5])

```

Figure 4: Model Inspection

Here, it shows the X matrix, which is a large number table with the data type float32. Its shape is defined by the number of rows (jobs) and columns (skills). It also prints some example job titles, sample locations, a skill list for the first job, and some country names from the country index. If there are no errors or zero values, the model is ready to run. Otherwise, the errors must be fixed.

By analysing FIGURE 5 below, it is possible to confirm that this model is healthy and functioning properly. It shows that with float32 X NumPy with the shape (15000, 24) and this refers to 15000 jobs and vocab size of 24. So here the matrix and skill list is matching correctly. Then the output is showing some job title samples such as AI Research Scientist, AI Software Engineer , AI Specialist and some location samples such as China, Canada, Switzerland and as this looks normal without any errors. Then for the first job AI Research scientist, the sample skills are kubernetes, linux, nlp, Pytorch, tableau are

also lower-case, normalized and matching perfectly. Here it also shows that the country is 20 and few sample country index as well which are also lower case . Overall this inspection confirms that the recommendation model is ready to run.

```
PS D:\centria\2025\Thesis\practical work> python inspect_model.py
Top-level keys: ['X', 'company_location', 'countries', 'country_to_indices', 'job_title', 'mlb_classes', 'required_skills_list']
X type: ndarray dtype: float32 shape: (15000, 24)
jobs: 15000
countries count: 20
vocab size (skills): 24
sample titles: ['AI Research Scientist', 'AI Software Engineer', 'AI Specialist']
sample locations: ['China', 'Canada', 'Switzerland']
sample skills list (first job): ['kubernetes', 'linux', 'nlp', 'pytorch', 'tableau']
country index example keys: ['china', 'canada', 'switzerland', 'india', 'france']
```

Figure 5: Inspection result

6 RESULT ANALYSIS

Here the recommendation model has been tested with user skills of NLP, AWS and Python. Also the user's location interest was on Finland and user also misspelled the country name but the result came as shown in the figure below.

```
PS D:\centria\2025\Thesis\practical work> python recommend_from_model.py
Interactive job recommender (type 'exit' to quit).
Search with a Country name or 'Global'?: Finlandd
Did you mean: 1) Finland, 2) Ireland, 3) Switzerland
Choose 1-3, or type a country name (or 'g' for global): 1
Interpreting country 'Finlandd' as 'Finland'
Skills (comma-separated): NLP, aws, python

Top 5 recommendations:
 1. Autonomous Systems Engineer | Finland | score=0.775 | match=aws, nlp, python | learn: computer vision, data visualization
 2. Deep Learning Engineer | Finland | score=0.667 | match=aws, python | learn: computer vision
 3. Deep Learning Engineer | Finland | score=0.667 | match=aws, python | learn: mathematics
 4. Principal Data Scientist | Finland | score=0.667 | match=aws, nlp | learn: azure
 5. Data Scientist | Finland | score=0.667 | match=aws, python | learn: tensorflow
Another query? [y/N]: N
PS D:\centria\2025\Thesis\practical work>
```

Figure 6: Recommendation from the Model

So this recommendation system is working accurately as it understood the wrong input from the user which was "Finlandd" and turned it to Finland . Then system has taken the user's skill input properly as well as recommended the jobs based on the skills user had. As per the requirements system has suggested Autonomous System engineer, Deep Learning Engineer which was suggested twice as there was two jobs available with the same title in Finland and two more jobs. This model is showing cosine score for each jobs and also the list of the matching skills and for future guidance it is also showing the skills user need to learn for that particular job.

Overall each line shows the title, location, matching skills and what more to learn in such a practical and clear format. The output of this recommendation system is very clean and easy to read along with helpful guidance on which jobs fit user's current skills and what to learn next.

7 DISCUSSION AND FUTURE DEVELOPMENT

This recommendation system was initially successful and seemed effective in matching user skills and suggesting job titles based on the job posting data from Kaggle. In the result analysis part, it was shown that the model had correctly taken the inputs from the user and normalized them. Along with the mis-spelled country input, it returned job suggestions for the top five matched job titles based on the required country name. This is assuring that in the model, the fuzzy matching for the country is working properly and every recommendation line clearly shows the user's matched skills and the unmatched skill which user need to learn to fulfil the job skill requirements. Overall the user can analyse the reason for every match and get a idea skill development path which makes the recommendation model successful with correct cosine and clear career suggestions.

7.1 Model Limitations

Even though the current model is a successful recommendation system but it has some limitations as well such as in someone put less numbers of skills like one or two skills only, this model show as most of them are bad match and it might not show a perfect result. Also, this model doesn't consider or is unable to calculate the importance of any specific skill. So it equally counts every skills for all the jobs even though sometime it can be happen that importance of any skill is negligible but this model does not take this in consideration. In these situations, this model might not be accurate in reflecting the real-world job market situation.

Also this model is trained by a statistic dataset from Kaggle. So, it will not be able to update with any new jobs outside of the dataset, as it only contains data based on AI-related job postings. So new role and expertise in other categories are not included. And this model also does not include any educational qualifications or working experience. So sometimes it can be misleading for the user to understand if they are overqualified or underqualified for the job. As this model has not been tested in a real-life scenario, it is not possible to claim how accurate and helpful it can be for the user.

7.2 Future Development

From the model limitation part it is possible to get an idea where this model can improve. First of all, to ensure its real-life benefit, it can be tested with students and job seekers and their feedback can be used for further improvements. Improving the model so that it can learn new job postings from the internet and able to enrich it's data . Also enabling the system to read user's CV instead of just taking skills one by one can make the model more user friendly and access to more data about the user including their educational qualifications and working experience.

In this model for avoiding typing mistakes in the country input, Fuzzy matching has been used but it is possible to take the model one step ahead by using NLP. If NLP could be used for both country inputs and skill inputs, the model will be more connected with the user. For improving the user experience and make it more accessible to user this simple script can be turned into a website or app which will be helpful for collecting user feedback as well.

8 CONCLUSION

This project is a recommendation prototype tool for the IT students and fresh graduates who are willing to build their career in the field of AI. This recommendation tool aims to provide simple and clear job recommendations based on the user's skills and their location preference. In this project a public data of 15000 jobs has been used to train the model and enables the model to provide clear and practical job suggestions. Along with the job suggestions to the users, it also show the reason of job matching e.g. the matching skills and also provides a guidance what other skills user need to learn for that particular job.

For keeping the model simple, reliable and easy to understand this model is based on simple content-based filtering and cosine similarity. In this basic version of this model it has proven it's success with providing helpful and practical suggestions along with handling location input errors from the user. Even though this model has few limitations but with it's accurate recommendations it also proves that this model has great potential to be helpful for students and educational institutions in practical use. In the future, developments such as using more updated data, making the model interactive, and adding smart features like a user interface can improve its usefulness. Practical testing with user feedback will also help make this model a valuable resource for students, educational institutions and anyone interested in starting a career in AI in today's fast-changing job market.

REFERENCES

Charu, C.A., 2016. *Recommender systems: The textbook*, Ch. 4, pp. 139–141, 150–151. Available at: http://ndl.ethernet.edu.et/bitstream/123456789/88451/1/2016_Book_RecommenderSystems.pdf

Harris, C.R., Millman, K.J., Van Der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J. and Kern, R., 2020. Array programming with NumPy. *nature*, *585*(7825), pp.357-362. Available at: https://www.nature.com/articles/s41586-020-2649-2.pdf

Lops, P., De Gemmis, M. and Semeraro, G., 2010. Content-based recommender systems: State of the art and trends. Recommender systems handbook, pp.73-105. Available at: https://d1wqtxts1xzle7.cloudfront.net/50983966/ContentBasedRS-libre.pdf

Mana, S.C. and Sasipraba, T., 2021, March. Research on cosine similarity and pearson correlation based recommendation models. In *Journal of Physics: Conference Series* (Vol. 1770, No. 1, p. 012014). IOP Publishing. Available at: https://iopscience.iop.org/article/10.1088/1742-6596/1770/1/012014/pdf

McKinney, W., 2010. Data structures for statistical computing in Python. *scipy*, *445*(1), pp.51-56. Available at https://web.archive.org/web/20161014042743id_/http://conference.scipy.org/proceedings/scipy2010/pdfs/mckinney.pdf

Mitchell, T. M. (1997). *Machine Learning*. New York: McGraw-Hill. Available at: https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf

OECD (2023) Skill needs and policies in the age of artificial intelligence. OECD Employment Outlook 2023, pp. 246–248. Available at: https://www.oecd.org/en/publications/oecd-employment-outlook-2023_08785bba-en/full-report/skill-needs-and-policies-in-the-age-of-artificial-intelligence_fe530fbf.html

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, *12*, pp.2825-2830. Available At: https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf?source=post_page

Rahm, E. and Do, H.H., 2000. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, *23*(4), pp.3-13.Available at: https://cs.brown.edu/courses/cs227/archives/2017/papers/data-cleaning-IEEE.pdf

Ricci, F., Rokach, L. and Shapira, B., 2021. Recommender systems: Techniques, applications, and challenges. *Recommender systems handbook*, pp.1-35. Available at : https://www.researchgate.net/profile/Ehsan-Momeni-Bashusqeh/post/What_is_the_newest_topic_in_recommender_systems/attachment/59d6435079197b807799ec94/AS%3A442575841697793%401482529711967/download/introduction-handbook-2015.pdf

Russell, S., Norvig, P. and Intelligence, A., 1995. A modern approach. *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, *25*(27), pp.1-4. Available at: https://www.academia.edu/74861041/Artificial_intelligence_modern_approach_9780131038059

Samuel, A.L., 1959. Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, *3*(3), pp.210-229. Available at : https://ebiquity.umbc.edu/_file_directory_/papers/1430.pdf

Sajjad, B. (2025) *Global AI Job Market & Salary Trends 2025*. Kaggle. Dataset. Available at: https://www.kaggle.com/datasets/bismasajjad/global-ai-job-market-and-salary-trends-2025

Schütze, H., Manning, C.D. and Raghavan, P., 2009. *Introduction to information retrieval* (Chapter 6 (p. 121) – cosine similarity.). Cambridge: Cambridge University Press. . Available at: https://nlp.stanford.edu/IR-book/pdf/06vect.pdf

Wilson, H.J., Daugherty, P. and Bianzino, N., 2017. The jobs that artificial intelligence will create. *MIT Sloan Management Review*, *58*(4), p.14-16. Available at : https://sblog.i-scream.co.kr/data-files/ssamblog/1004/201901140511351501.pdf