ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
Σχολή Ηλεκτρολόγων Μηχανικών και
Μηχανικών Υπολογιστών
ΔΠΜΣ στην Επιστήμη Δεδομένων και Μηχανική
Μάθηση

Υλοποίηση και αξιολόγηση συστημάτων μηχανικής μάθησης παραγωγής με εφαρμογή στη σημασιολογική κατάτμηση γεωχωρικών δεδομένων

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ηλιοπούλου Χρυσοβαλάντω-Θεοδώρα

Επιβλέπων: Κωνσταντίνος Καράντζαλος
Αναπληρωτής Καθηγητής Ε.Μ.Π.

Αθήνα, Μάρτιος 2021

National Technical University of Athens
School of Electrical and Computer Engineering
MSc in Data Science and Machine Learning

Implementation and assessment of machine learning production systems for geospatial data classification tasks

MASTER THESIS

Iliopoulou Chrysovalanto-Theodora

Supervisor: Konstantinos Karantzalos
Associate Professor at NTUA

Athens, March 2021

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
ΔΠΜΣ στην Επιστήμη Δεδομένων και Μηχανική Μάθηση

Υλοποίηση και αξιολόγηση συστημάτων μηχανικής μάθησης παραγωγής με εφαρμογή στη σημασιολογική κατάτμηση γεωχωρικών δεδομένων

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ηλιοπούλου Χρυσοβαλάντω-Θεοδώρα

Επιβλέπων: Κωνσταντίνος Καράντζαλος
Αναπληρωτής Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 30η Μαρτίου 2021.

.....	.....	.....
Κωνσταντίνος Καράντζαλος	Γεώργιος Γκούμας	Γεώργιος Στάμου
Αν. Καθηγητής Ε.Μ.Π.	Αν. Καθηγητής Ε.Μ.Π	Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, Μάρτιος 2021

…….

Ηλιοπούλου Χρυσοβαλάντω-Θεοδώρα
Κάτοχος Διατμηματικού Μεταπτυχιακού Διπλώματος Ειδίκευσης στο επιστημονικό πεδίο «Επιστήμη Δεδομένων και Μηχανική Μάθηση (Data Science and Machine Learning)»

Περίληψη

Το πεδίο της μηχανικής μάθησης έχει παρουσιάσει τεράστια πρόοδο τα τελευταία χρόνια. Χάρη στη συνεχή εξέλιξη των τεχνικών και μεθόδων ταξινόμησης και πρόβλεψης και την ικανοποιητική τους απόδοση, η χρήση τους και η ενσωμάτωσή τους στο βιομηχανικό χώρο κρίνεται αναγκαία. Είναι σημαντικό η ενσωμάτωσή τους σε ένα παραγωγικό περιβάλλον να γίνεται με τρόπο που συμβάλλει στην ομαλή συνεργασία του ανθρώπινου δυναμικού που εξειδικεύεται στην ανάπτυξη εφαρμογών αλλά και τη γρήγορη και ορθώς καθορισμένη διαδικασία ανάπτυξης και ενημέρωσης των εφαρμογών.

Μέχρι σήμερα, έχουν αναπτυχθεί τεχνικές και εργαλεία που αυτοματοποιούν τις επαναλαμβανόμενες διαδικασίες που απαιτούνται στη φάση ανάπτυξης νέου κώδικα και εγκατάστασής του σε ζωντανό περιβάλλον. Ωστόσο, η εφαρμογή τους δεν είναι αρκετή για το χώρο της μηχανικής μάθησης. Οι τρεις βασικές συνιστώσες της μηχανικής μάθησης είναι ο κώδικας, τα δεδομένα και τα μοντέλα. Η αλλαγή έστω και μιας από τις τρεις συνιστώσες μπορεί να οδηγήσει σε διαφορετική πρόβλεψη από το σύστημα που αυτές παράγουν. Συνεπώς, είναι απαραίτητη η ταυτόχρονη διαχείριση και των τριών συνιστωσών ως προς την αποθήκευση, την ιστορικότητα και την προώθηση στο παραγωγικό περιβάλλον του βέλτιστου μοντέλου που προκύπτει από το συνδυασμό τους.

Σκοπός της παρούσας μεταπτυχιακής εργασίας είναι η διερεύνηση και η μελέτη μερικών από τα πολυάριθμα εργαλεία που έχουν αναπτυχθεί για την αντιμετώπιση των δυσκολιών της μηχανικής μάθησης σε παραγωγικό περιβάλλον. Επιλέγονται μερικά εξ αυτών και ελέγχεται ο συνδυασμός τους, με στόχο το σχεδιασμό και την ανάπτυξη μιας υποδομής, ικανής να καλύψει τις προκλήσεις που αντιμετωπίζει η ένταξη και η διαχείριση μοντέλων μηχανικής μάθησης σε παραγωγικές εφαρμογές. Αντικείμενο εφαρμογής των εργαλείων αυτών αποτελεί το πρόβλημα ταξινόμησης των στοιχείων μιας γεωχωρικής εικόνας, που απεικονίζει υποθαλάσσιο έδαφος, σε ορισμένες κλάσεις συγκεκριμένης σύστασης.

Προσεγγίζοντας το πρόβλημα με διαφορετικές λύσεις αλλά και με διαφορετικά σύνολα δεδομένων, αξιολογούνται τα επιλεγμένα εργαλεία και τονίζονται τα οφέλη της χρήσης τους για την εύρεση της καλύτερης δυνατής λύσης, τόσο κατά τη διάρκεια ανάπτυξης νέων μεθόδων ταξινόμησης όσο και στη φάση συνεχούς ενημέρωσης ενός εγκαταστημένου μοντέλου σε μια παραγωγική εφαρμογή.

Λέξεις κλειδιά

Μηχανική μάθηση στην παραγωγή, Επιστήμη Δεδομένων, Διαδικασίες μηχανικής μάθησης, σωληνώσεις συνεχούς ενσωμάτωσης και εγκατάστασης

Abstract

Machine learning as a field has evolved significantly over the last years. Due to the continuous evolution of classification and prediction methods, but also to their satisfying efficiency, the models' usage and integration in industry have become a necessity. It is important that their integration into a production environment occurs in a way that contributes to smooth cooperation of the human resources specializing in developing applications, but also in quick and properly defined operations associated with developing and updating applications.

Up till today, many techniques and tools have been produced, for automating repeated operations needed for phases of developing new code and deploying it into a live environment. However, they can not be simply applied for machine learning applications. Machine learning's three main components are code (algorithms), data and models. Changing one of them may lead to different classification results. Therefore, it is necessary that all the components are handled simultaneously concerning the storage, history and forwarding to the production environment of the optimal model that comes from their combination.

The purpose of the present master thesis is to investigate and study some of the numerous tools that have been developed for coping with the struggles of machine learning when productising. Some of them are chosen in order to check the adequacy of their combination aiming at the design and implementation of an infrastructure, appropriate for handling the challenges of deploying and operating machine learning models in production. These tools are assessed considering the classification of backscatter images of the seabed into certain classes of specific composition.

Approaching the problem according to different solutions but also to different datasets, the tools considered are evaluated. At the same time, the benefits of using them for finding the best possible solution during the development of new classification methods but also during continuous integration of the deployed model, are noticed.

Keywords

Machine learning in production, Data science, ML operations, continuous integration and deployment pipelines

Extended Summary

Nowadays, it's virtually impossible to miss the rapid rise of Artificial Intelligence and Machine Learning fields, both in general and specifically in the context of production systems and solutions. Terms like AI, automation and machine learning are becoming more and more popular. However, industries find it difficult to integrate machine learning (ML) techniques efficiently and in an automated way, because ML is not just about code, but it consists of many more components, such as data and models. Till today more emphasis has been put on ML in research and how to increase accuracy and efficiency based on clean and static data (Kuyen, 2020). ML in production has to overcome a number of challenges associated with automation, resource management, data handling, configuration, model analysis, serving infrastructure and monitoring just like any other application (Sculley et al., 2015). In addition, the fact that data is constantly shifting and users' features change leads to the need of automated pipelines that handle these changes to preserve an optimal and efficient installation of a prediction model.

Many companies and teams have developed state-of-the-art tools to achieve automation in internal services, such as the Airbnb team with Airflow (Wikipedia, 2019) (HG Insights, 2021) and Uber with Michelangelo (Uber et al., 2017). In order to cope with experiment reproducibility, application scalability, team collaboration, quick rollouts and rollbacks, many technologies have been released. These technologies focus on model and data versioning, distributed preprocessing, analytics and training, ML pipeline orchestration and model serving. Some of the most popular are Airflow (The Apache Software Foundation, 2021), Kubeflow (The Kubeflow Authors, 2018-2021), Apache Spark (The Apache Software Foundation, 2018), MLflow (MLflow Project, a Series of LF Projects, LLC, 2021) and Tensorflow (Abadi et al., 2015). Choosing which tools are more appropriate for solving ML problems is difficult and depends on many subjects, such as location of data storage, frameworks and language familiarity but also engineering architectures like microservice model serving. Using this software the goal is still to achieve scaling, parallelism, logging, monitoring, testing, automation, security and rolling upgrades (Karbhari, 2020).

According to the reasons above, this thesis' subject is to design and develop an infrastructure including ML solutionson a Kubernetes cluster in order to adopt ML techniques efficiently and in prospect effortlessly. The image 0.1 shows the structure of the tools that were chosen combined with other services needed for optimal and advanced functionality. The main tools chosen are Airflow, a general platform expertised on orchestrating and scheduling pipeline tasks, and MLflow, another open source software used for managing ML lifecycle, including experimentation, reproducibility, deployment and a central model registry. Airflow consists of two components, the webserver for the user interface and the scheduler for triggering predefined DAGs. Although Airflow provides several possible Executors for running the tasks of a DAG, like Celery or Kubernetes (Ask Solem & contributors, 2009-2018), the LocalExecutor (Pierre, 2020) was selected for the applications examined. Beside these tools, two different instances of postgres databases were installed on the cluster

in order to save the pipelines (directed acyclic graphs - DAGs) of the Airflow system and the experiment details of the MLflow. A minIO server, a Kubernetes-native object storage suit and s3 compatible, was also established responsible for saving objects associated and produced during specific experiment runs on MLflow server. The fact that Seldon software is integrated with MLflow server, a set of tools for deploying ML models at scale, led to the installation of a Seldon (Seldon Technologies Ltd, 2021) deployment based on REST MLflow model service. Another approach is considered, the one of installing a REST service using the Flask framework due to the easy integration that MLflow provides through its API in many ways (Python API, REST API). For the purpose of a toy problem that was applied on the Airflow-MLflow combination in order to evaluate if it is appropriate for coping with ML challenges, Kafka, a messaging system and Zookeeper, a centralized service for maintaining configurations, were also installed. All the services were built behind an Ingress controller.

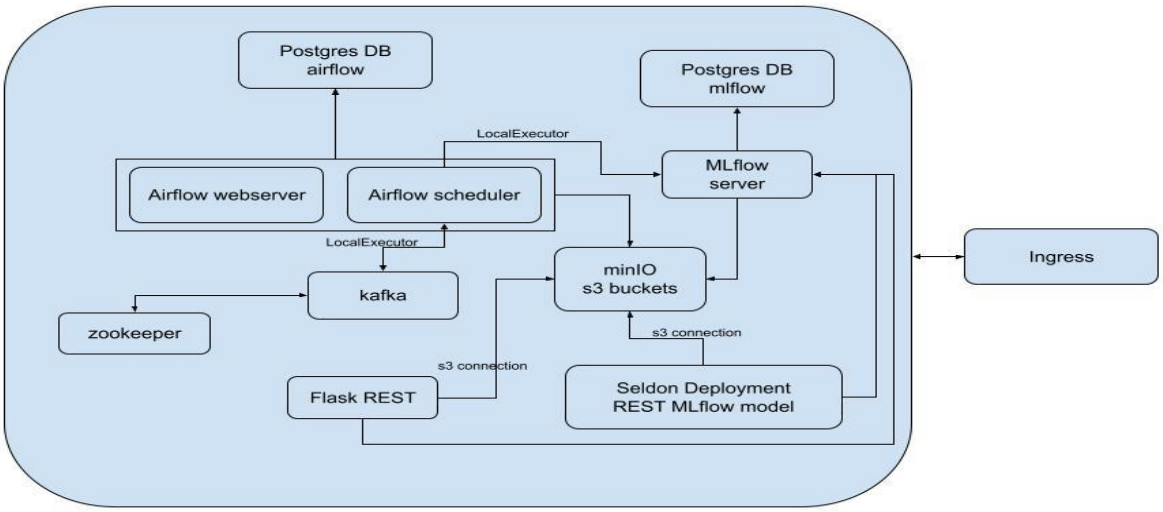


Figure 0.1: Infrastructure diagram displaying the connection between microservices developed for purposes of finding machine learning solutions end-to-end.

A toy problem was examined in order to evaluate the way the tool combination may contribute to continuous enhancement of ML model's efficiency. MNIST dataset was used and three DAGs were developed, one for the model initialization, one for streaming constantly new data, derived from a predefined training dataset, to the kafka service and one for consuming the incoming data in order to retrain the model and check if the produced model is better than the serving one. Through this problem it seems that developing an automated pipeline, like this, the serving model can be updated day by day without requiring manual processes' execution.

There are three ways of logging the execution of a ML pipeline defined in Airflow to MLflow. The first one is by producing the code inside of DAG's tasks including logging calls in the code, the second one is by producing a MLproject, grouping into a folder both the code and the file describing the environment required to be executed successfully, and the last one is by designing a DAG where each task executes only a specific entrypoint of a predefined

MLproject. The best method is the latter, because in that way the execution environment is isolated, avoiding possible conflicts between different library releases a code may require, but also exploiting the combination of different ML techniques and strategies in order to find the best solution that produces the optimal prediction model.

One of the most important advantages of these two tools is that they are constantly supported and updated by their development teams, thus providing the possibility of integration with many more other services. They both are compatible with DVC actions (DVC, 2021) for versioning code, data and models and maintaining history, if more advanced actions than those that MLflow provides through the artifact store, are required. Also, thanks to the available API they both provide, it is possible for other tools to easily integrate with them and eventually promote extended collaboration.

There are plenty of benefits obtained by combining these ML pipeline tools. From the aspect of the code, each project's execution environment is isolated, so developers have the opportunity to develop new ML techniques independently without worrying about library conflicts with pre-existing library dependencies. Moreover, experiment reproducibility and repetition becomes a quick and easy routine due to the tools' features. The ability of scheduling repeated processes that have to be executed periodically on a specific time is extremely helpful because no manual processes, increasing the possibility of faults, are required. Producing caching techniques about experiment runs that have already taken place in MLflow leads to quicker runs and less storage memory used. Both tools provide a user interface resulting in enhanced human usability. When someone visualizes the extremely complex ML pipelines used to produce effective ML models it's easier to understand the processes and experiment details. Access to artifacts, pipeline executions, history, logs, errors becomes a simple task, letting the expert to focus on what really matters, which is solving ML problems. Pipelines, ML tasks and individual runs are grouped in sections, artifacts are saved in folders, everything associated with ML is organised in an automated way saving time but also effort from manual handling.

The established infrastructure was applied on a classification problem in order to be evaluated. This problem deals with seabed classification from multibeam multi-spectral echosounder data. Training data is obtained by the combination of an input image in RGB and their equivalent ground-truth image that contains the classes that each pixel is mapped to. For the purposes of finding the optimal model that maps the pixels to several seabed classes, three preprocessing techniques, the median filter (Church et al., 2008) (Villar et al., 2017), the min-max scaling and the z-score scaling (Κύρκος, 2015), and two classifiers, SVM (Pouteau et al., 2012) (Vapnik, 1982) (Vapnik, 2000) (Zhu & Blumberg, 2002) and Random Forest (RF) (Breiman, 2001) were examined. Training data consists of three images, one corresponding to patricia region and two of the bedford region acquired in 2016 and 2017, respectively. Due to the fact that the corresponding classes of the two regions were different (Fine Sand, Med Sand and Rocky/Gravel for patricia, Muddy Sand with Organic, Muddy Sand, Fine Muddy Sand, Sand/Gravel and Rocky/Gravel for bedford) two approaches were followed. The former was to consider each category as a separate one, while the latter was to group Med Sand and

Sand/Gravel into one class, leave Rocky/Gravel as is and group all the other classes as Fine Sand. Finally, in order to evaluate a model's progress through time, the original dataset was divided into many subsets. One approach was to use only one portion, for example a 50% of total data. The other approach is to create subsets containing only specific classes of the original data for each one of the three images. In that way, a model could be trained using different combinations of the subsets. At first only the data of bedford_2016 image was used. Afterwards, subset of patricia_1_2, meaning data mapped to class 1 and 2, was joined to bedford_2016. Joining the two datasets adds knowledge into the training dataset, in order to retrain the two classifiers for each preprocess and each one of the two class approaches, and study the model's accuracy. At last, bedford_2017_3, meaning data corresponding to class 3, was added to improve inadequacies that the previous model presented.

During every experiment run that produced a prediction model, an evaluation task, responsible for logging evaluation metrics to MLflow database, was completed. The metrics that were computed were accuracy, Cohen's kappa (Μανωλέσου, 2015) (McHugh, 2012), precision and recall from confusion matrix, based on all available data from the ground-truth images. The validation data, that was obtained by splitting the original data into two individual datasets, the training one and the validation one, was used to compute validation error after training the classifier.

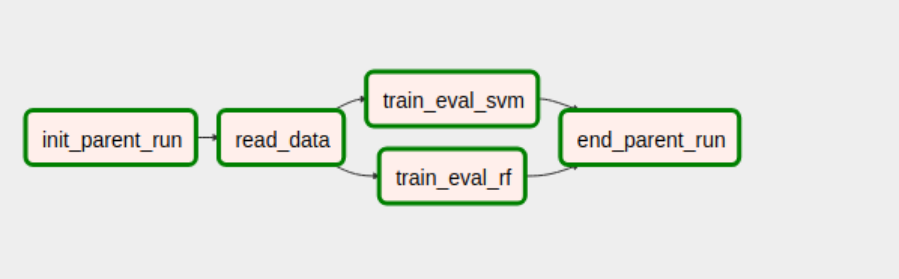


Figure 0.2: The predefined DAG including the ML steps required for producing a prediction model solving the seabed classification

Aiming to execute all different combinations of experiments that come from the multiple choices of preprocessing, classifiers and number of classes, and then find the optimal prediction model that maps an image's pixels to correct seabed categories, two DAGs were defined. The former consists of specific ML tasks required for producing the model (figure 0.2) and the latter consists of three tasks, one for finding the optimal model among the completed runs, one for updating the REST service about the new prediction model that has to be loaded and the one for promoting the chosen model into production in MLflow's model registry (figure 0.3).

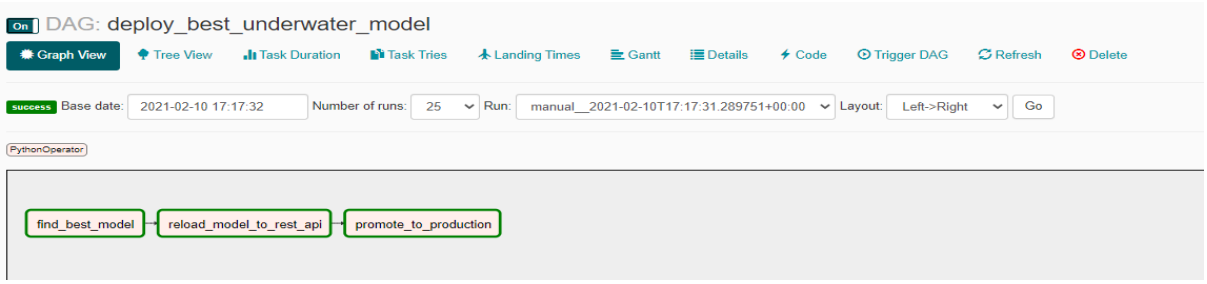


Figure 0.3: The predefined DAG including the tasks required for finding the optimal prediction model and its loading into production.

The first conclusion that can be derived from experiments is that the RF classifier is more efficient in comparison to the SVM, for the problem at hand. This can be justified by the imbalanced data (more available data about the class 1 than the other two), circumstances under which RF seems to be specialized. Also, it seems that using the 50% results into similar metrics for both classifiers regardless of the preprocessing or the approach followed. Thanks to the simulation of constant data enrichment, achieved by combining more and more datasets in sequential experiment runs, what can be concluded is that adding extra information about classes that the model seems to be incapable of detecting, may lead to increasing recall and precision on those "problematic" classes, and thus to higher accuracy in general. The classification results coming from the three different dataset versions are displayed in figure 0.4 next to the reference result derived from the optimal model (trained with the initial data), for the case of median filter and the grouped target classes. The trivial case of using datasets with insufficient information about all target classes, is validated by the zero metrics computed for the missing categories. Although these general conclusions stand on all technique combinations examined, the case of the median filter, with the grouped target classes approach is displayed on table 0.1. When it comes to the performance of the three different preprocessing methods it seems that the two scaling methods result in very similar classification images, while minor differences appear in median's filter result. However, all three methods achieve high accuracy when RF classifier is used, and this is interpreted by the fact that the evaluation takes place based on ground-truth non-zero pixels where all methods have the same classification result (figure 0.5).

Table 0.1: Recording of evaluation metrics for each prediction model produced when the median filter was applied and the target classes were grouped into 3 total categories.

data	classifier	valid_error	accuracy	kappa	precision_class_1	recall_class_1	precision_class_2	recall_class_2	precision_class_3	recall_class_3
all	SVM	0.228	0.77	0.61	0.97	0.87	0.63	0.96	0.93	0.0
all	**RF**	**0.027**	**0.97**	**0.95**	**0.98**	**0.98**	**0.97**	**0.96**	**0.95**	**0.98**
percentage_0.5	**RF**	**0.036**	**0.96**	**0.94**	**0.97**	**0.98**	**0.96**	**0.95**	**0.95**	**0.96**
bedford16	RF	0.451	0.55	0.19	0.56	1.0	0.13	0.03	0.97	0.42

bedford16, patricia_class_1_2	RF	0.176	0.83	0.7	0.93	0.9	0.72	0.97	0.97	0.26
bedford16, patricia_class_1_2, bedford17_class_3	RF	0.121	0.88	0.8	0.94	0.9	0.8	0.96	0.96	0.65
bedford16	SVM	0.484	0.52	0.2	0.51	0.83	0.0	0.0	0.52	0.8
bedford16_class_1_2	SVM	0.519	0.48	0.04	0.55	0.98	0.09	0.04	0.0	0.0
bedford16_class_1_2	RF	0.501	0.5	0.07	0.54	1.0	0.2	0.07	0.0	0.0
bedford16percentage_0.5	RF	0.448	0.55	0.19	0.57	1.0	0.2	0.06	0.97	0.38
bedford16percentage_0.5	SVM	0.524	0.48	0.16	0.5	0.74	0.36	0.02	0.42	0.78
percentage_0.5	SVM	0.379	0.62	0.34	0.56	0.99	0.2	0.0	0.91	0.97

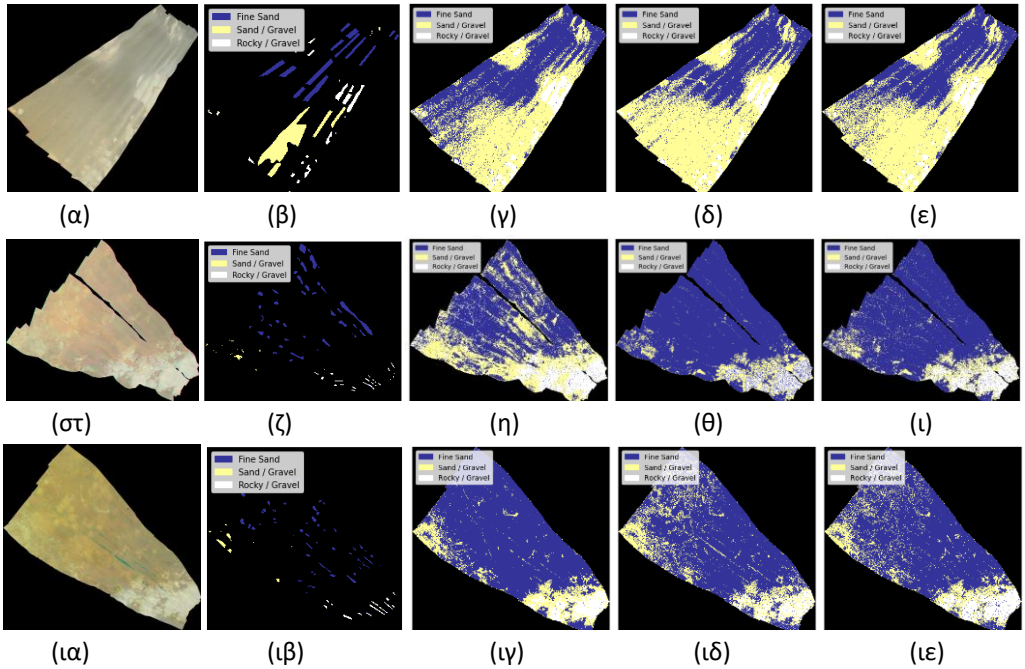


Figure 0.4: The result images produced by the three models, for each preprocessing method, for the case of RF classifier and the 3 grouped target classes. α)patricia, β)gt_patricia, γ)patricia_median, δ)patricia_min_max, ε)patricia_z-score, στ)bedford_2016, ζ)gt_bedford_2016, η)bedford_2016_median, ϑ)bedford_2016_min_max, ι)bedford_2016_z-score, ια)bedford_2017, ιβ)gt_bedford_2017, ιγ)bedford_2017_median, ιδ)bedford_2017_min_max, ιε)bedford_2017_z-score.

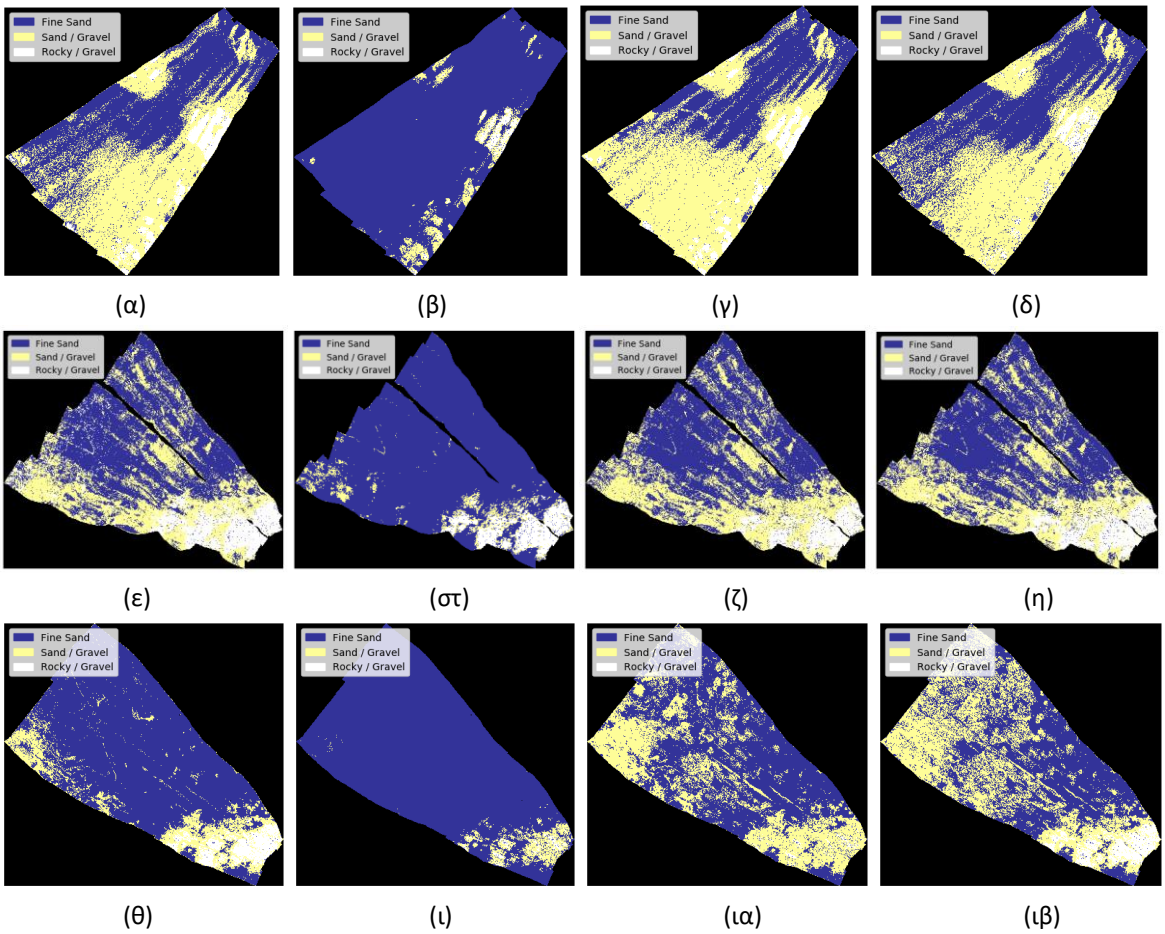


Figure 0.5: The result images by the three prediction models produced by the three datasets, sequentially combining more subsets, for the case of RF classifier and the 3 grouped target classes, per training picture, in comparison with the reference result image of the model trained with all data. α)patricia_ref, β)patricia_dataset_1, γ)patricia_dataset_2, δ)patricia_dataset_3, ε)bedford_16_ref, στ)bedford_16_dataset_1, ζ)bedford_16_dataset_2, η)bedford_16_dataset_3, θ)bedford_17_ref, ι)bedford_17_dataset_1, ια)bedford_17_dataset_2, ιβ)bedford_17_dataset_3.

Finally, the present Master thesis encourages its further elaboration suggesting future study directions to be followed. These are associated with the application of the developed infrastructure on more classification problems or the enhancement of the established tool ecosystem with more technologies covering other fields of machine learning like distributed systems and more resource demanding problems. Using monitoring tools and predefined processes that handle silent failures presented by the prediction models should be considered. Another direction is to create stress tests and conditions in order to examine how these tools should be configured in order to be constantly responding. Scaling is another option, by testing different execution methods in the Airflow system like Celery or Kubernetes Executor.

Ευχαριστίες

Η παρούσα μεταπτυχιακή εργασία, με την οποία ολοκληρώνεται ο κύκλος σπουδών μου στο διατμηματικό μεταπτυχιακό πρόγραμμα με πεδίο “Επιστήμη Δεδομένων και Μηχανική Μάθηση”, δε θα μπορούσε να περατωθεί χωρίς τη συμβολή ορισμένων ανθρώπων. Αρχικά, θα ήθελα να ευχαριστήσω θερμά τον κ. Κωνσταντίνο Καράντζαλο, Αναπληρωτή Καθηγητή του Ε.Μ.Π., για τη δυνατότητα που μου έδωσε να εκπονήσω την παρούσα εργασία προσεγγίζοντας στο μέγιστο δυνατό βαθμό το αντικείμενο ενδιαφέροντός μου. Τον ευχαριστώ επίσης, για την άψογη συνεργασία καθ’ όλη τη διάρκεια εκπόνησής της και την επίβλεψη της εργασίας συνολικά.

Δε θα μπορούσα να παραλείψω ανάλογες ευχαριστίες στον κ. Βαλσάμη Ντούσκο, Μεταδιδακτορικό Ερευνητή στο εργαστήριο τηλεπισκόπησης, για την επίσης άψογη συνεργασία μας, την ευγένειά του, την καθοριστική βοήθεια και καθοδήγηση που μου προσέφερε τόσο κατά την εκπόνηση της εργασίας όσο και κατά τη συγγραφή της.

Τέλος, θέλω να ευχαριστήσω την οικογένεια και τους φίλους μου για την υποστήριξη και τη βοήθεια που μου παρείχαν καθόλη τη διάρκεια που χρειάστηκε για την πραγμάτωση της παρούσας εργασίας.

Περιεχόμενα

Περιεχόμενα πινάκων

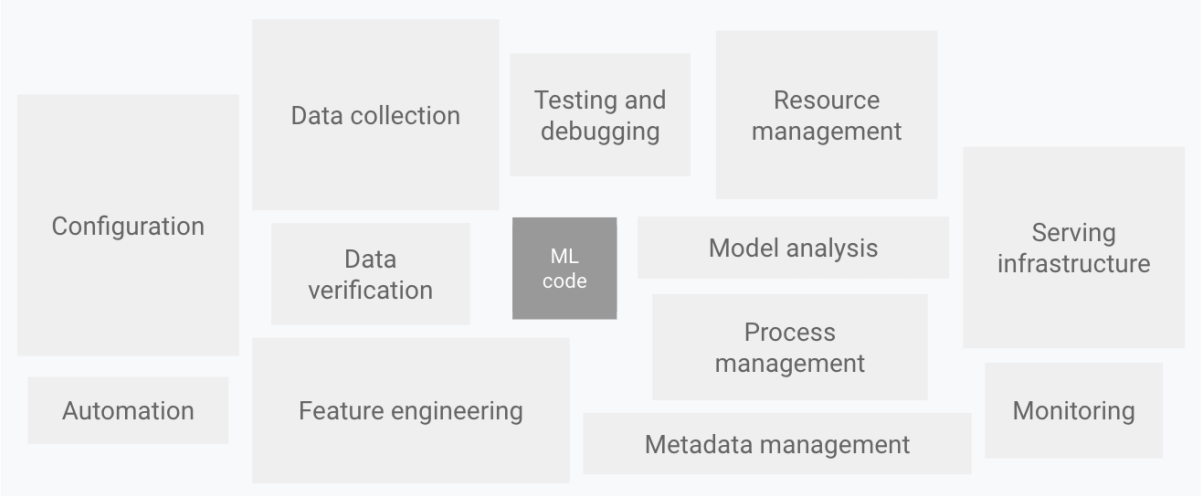
Περιεχόμενα εικόνων

Κεφάλαιο 1 : Εισαγωγή

Το γεγονός ότι η μηχανική μάθηση έχει παρουσιάσει τα τελευταία χρόνια τεράστια πρόοδο και εξέλιξη στο χώρο της πληροφορικής και της τεχνολογίας, έχει οδηγήσει στην ευρεία χρήση τεχνικών ταξινόμησης και πρόβλεψης και την ανάπτυξη αντίστοιχων μοντέλων στο χώρο της βιομηχανίας. Η ανάπτυξη τέτοιων μεθόδων από εξειδικευμένους ανθρώπους, όπως είναι οι επιστήμονες δεδομένων (data scientists) και οι μηχανικοί μηχανικής μάθησης (machine learning engineers) είναι μια διαδικασία σχετικά καλώς ορισμένη. Ωστόσο, επειδή πρόκειται για ένα τεράστιο πεδίο μελέτης με πολλές συνιστώσες που λαμβάνονται υπόψιν κατά την φάση ανάπτυξης των μοντέλων αυτών, η ενσωμάτωσή τους, που απαιτεί τη συνεχή ανάπτυξη και ενημέρωση των εγκατεστημένων εφαρμογών, σε παραγωγικά περιβάλλοντα αντιμετωπίζει πολλές δυσκολίες. Συνεπώς, δημιουργείται το ερώτημα για το ποιος είναι ο σωστός τρόπος, αν υπάρχει, να ολοκληρώνονται αυτές οι διαδικασίες με επιτυχία, ταχύτητα και πλήρη διαφάνεια ως προς τις ομάδες που είναι υπεύθυνες για αυτές.

1.1 Ο χώρος της μηχανικής μάθησης

Ανεξάρτητα από τη μηχανική μάθηση, η ανάπτυξη εφαρμογών σχετίζεται με την ανάπτυξη νέου κώδικα (code development) και την αποθήκευση των απαραίτητων δεδομένων (data storage), τα οποία για καλύτερη απόδοση είθισται να αντιμετωπίζονται ανεξάρτητα και απομονωμένα. Ένα ολοκληρωμένο σύστημα μηχανικής μάθησης ωστόσο, αποτελείται από πολύ περισσότερες συνιστώσες στον πραγματικό κόσμο. Όπως φαίνεται στην εικόνα 1.1, η ανάπτυξη κώδικα που εφαρμόζει τεχνικές μηχανικής μάθησης αποτελεί μόνο ένα μικρό κομμάτι του γενικότερου συνόλου. Συνιστώσες που επίσης περιγράφουν το χώρο της μηχανικής μάθησης είναι η συλλογή δεδομένων, η επαλήθευση των δεδομένων, η εξαγωγή χαρακτηριστικών, η ανάλυση των παραγόμενων μοντέλων, η ρύθμιση και η αυτοματοποίηση διαδικασιών που σχετίζονται με τη διαχείριση των πόρων, τη διαχείριση των διαδικασιών και των τεχνικών ελέγχου και αποσφαλμάτωσης, η υποδομή που ενσωματώνει το μοντέλο πρόβλεψης προκειμένου να εξυπηρετήσει προβλέψεις αλλά και η διαρκής παρακολούθηση όλων των παραπάνω.

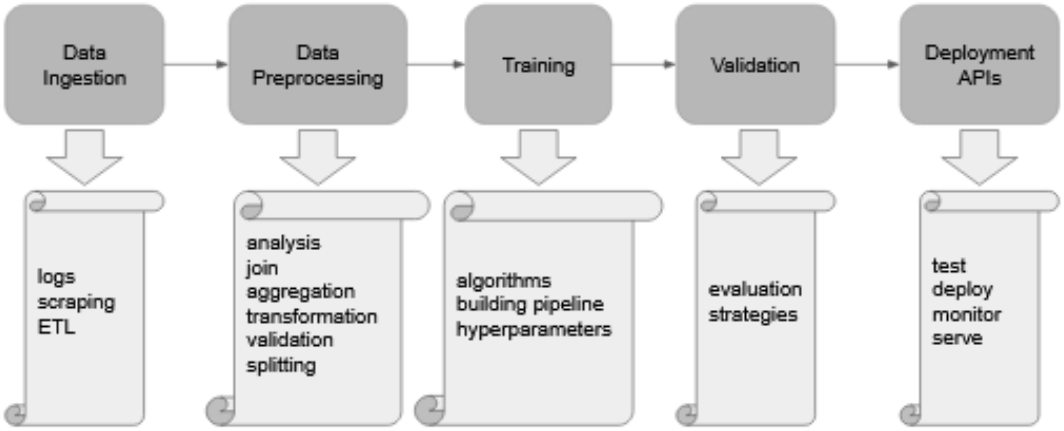


Εικόνα 1.1 Στοιχεία συστημάτων μηχανικής μάθησης (Sculley et al., 2015).

Προϋπόθεση για την παραγωγή ενός συστήματος μηχανικής μάθησης είναι ο σχεδιασμός της λύσης που μπορεί να προσεγγίσει όσο το δυνατόν καλύτερα το υπό εξέταση πρόβλημα. Κατά το σχεδιασμό μελετώνται, διερευνώνται και σταδιακά ορίζονται λεπτομέρειες και τεχνικές που σχετίζονται τόσο με τα διαθέσιμα δεδομένα όσο και με το παραγόμενο μοντέλο. Το σύνολο μελέτης είναι διαφορετικό από πρόβλημα σε πρόβλημα. Ωστόσο, βασικά αντικείμενα μελέτης είναι οι μετρικές και οι συνθήκες αξιολόγησης της αξιοπιστίας του μοντέλου που παράγεται. Συγκεκριμένα, μια διαδικασία παραγωγής ενός συστήματος μηχανικής μάθησης, όπως φαίνεται και στην εικόνα 1.2, αποτελείται από τα παρακάτω βασικά στάδια:

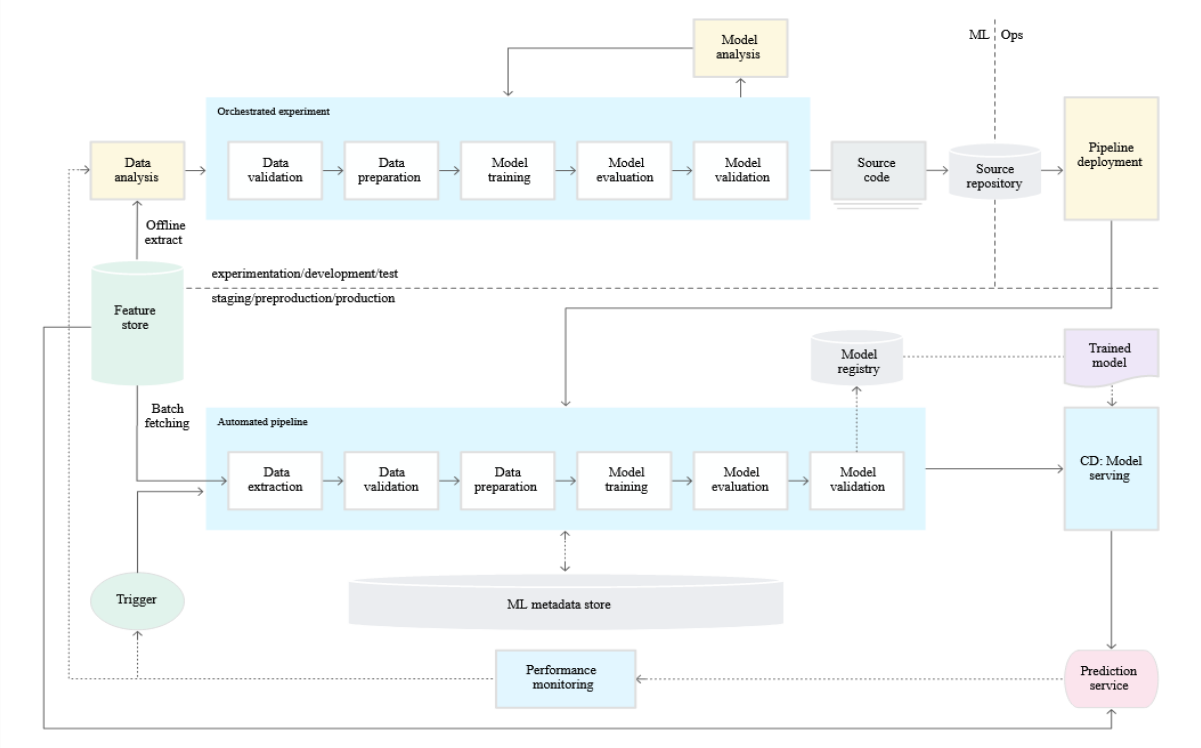
- Συλλογή δεδομένων (data ingestion): Πρόκειται για το αρχικό στάδιο μια σωλήνωσης επιμέρους σταδίων, στο οποίο συλλέγονται τα απαιτούμενα δεδομένα για την εκπαίδευση ενός μοντέλου από πολλές και διαφορετικές πηγές. Η συλλογή μπορεί να γίνεται από αρχεία κειμένου (logs), με την τεχνική της ιστοσυγκομιδής (web scraping), μέσω ETL (extract, transform, load) ακολουθιών από διαθέσιμες πηγές δεδομένων ή και άλλων λιγότερο αυτοματοποιημένων διαδικασιών που πραγματοποιούν τη συλλογή, τη μετατροπή και την αποθήκευση των δεδομένων χειροκίνητα. Μάλιστα, όταν τα δεδομένα είναι εικόνες, στις διαδικασίες συγκαταλέγονται και η σημασιολογική κατάτμηση (semantic segmentation), επισήμανση (labeling) ή και ανίχνευση (detection) στις θέσεις (pixels) των εικόνων που πραγματοποιούνται χειροκίνητα από εξειδικευμένα άτομα.
- Προεπεξεργασία Δεδομένων: Σε αυτό το στάδιο πραγματοποιείται συνένωση των συλλεχθέντων δεδομένων προκειμένου να διερευνηθούν και να αναλυθούν τα διαθέσιμα δεδομένα (Explanatory Data Analysis - EDA). Ύστερα, εφαρμόζονται σε αυτά τεχνικές επεξεργασίας (αφαίρεση, ομαδοποίηση, κανονικοποίηση, γέμισμα μη προσδιορισμένων τιμών, μετασχηματισμοί). Τέλος, επαληθεύονται και χωρίζονται σε επιμέρους σύνολα δεδομένων για τις φάσεις εκπαίδευσης, επικύρωσης και ελέγχου κατά την εκπαίδευση και την αξιολόγηση ενός μοντέλου.

- Εκπαίδευση: Ακολουθεί το στάδιο στο οποίο ορίζονται οι αλγόριθμοι ταξινόμησης που εξετάζονται, η ακολουθία ενεργειών που απαιτείται για την πρόβλεψη με βάση το μοντέλο, και ρυθμίζονται κατάλληλα οι υπερ-παράμετροι του μοντέλου για την επίτευξη της βέλτιστης απόδοσης.
- Επικύρωση: Το σύνολο δεδομένων ελέγχου, εφαρμόζεται στο ανεπτυγμένο μοντέλο και με βάση προκαθορισμένα κριτήρια προκύπτουν οι μετρικές που αξιολογούν το μοντέλο ως προς την ικανότητά του να προβλέπει, την ευαισθησία του και την ακρίβεια.
- Εγκατάσταση στην παραγωγή: Αυτό το στάδιο είναι αρκετά μεγάλο στάδιο και περιλαμβάνει την εγκατάσταση του καλύτερου μοντέλου που προκύπτει από τα προηγούμενα στάδια συνολικά. Εμπεριέχει, επίσης, την αυτοματοποίηση διαδικασιών που αφορούν τον έλεγχο της σωστής λειτουργίας ενός μοντέλου (testing), την κατανάλωσή του από άλλες εφαρμογές (serving APIs) και τη διαρκή παρακολούθησή του (monitoring) (Breck et al., 2017).



Εικόνα 1.2: Τα 5 στάδια μιας αυτοματοποιημένης διαδικασίας μηχανικής μάθησης.

Η παραπάνω ακολουθία σταδίων ή αλλιώς σωλήνωση, στη φάση ανάπτυξης ενός συστήματος ταξινόμησης, επαναλαμβάνεται διαρκώς, με αποτέλεσμα να δημιουργείται η ανάγκη για αυτοματοποίησή της (Chauhan et al., 2020), όπως φαίνεται στην εικόνα 1.3.



Εικόνα 1.3: Διάγραμμα απεικόνισης των συνηθέστερων διαδικασιών που απαιτούνται για την παραγωγή και την εγκατάσταση ενός μοντέλου πρόβλεψης στην παραγωγή (Abadi et al., 2015).

1.2 Η μηχανική μάθηση στην έρευνα έναντι στην παραγωγή

Δεδομένου ότι η χρήση της μηχανικής μάθησης στο βιομηχανικό χώρο βρίσκεται σε πρώιμο στάδιο, οι άνθρωποι που ειδικεύονται στο πεδίο της, προέρχονται κατά βάση από τον ακαδημαϊκό χώρο. Ο τρόπος που προσεγγίζουν ένα πρόβλημα, με στόχο την εύρεση της βέλτιστης δυνατής λύσης, συνήθως έχει ως θεμέλιο τη μελέτη σχετικών αναφορών και τη διεξαγωγή έρευνας. Η ιδιοσυγκρασία αυτή διαφέρει όμως ως προς τις ανάγκες και τις απαιτήσεις στον χώρο των επιχειρήσεων. Παρακάτω παρατίθενται μερικές παράμετροι ως προς τις οποίες διαφοροποιούνται.

	Research	Production
Objectives	Model performance	Different stakeholders have different objectives
Computational priority	Fast training, high throughput	Fast inference, low latency
Data	Static	Constantly shifting
Fairness	Good to have (sadly)	Important
Interpretability	Good to have	Important

Πίνακας 1.1: Διαφορές της εφαρμογής της μηχανικής μάθησης στην έρευνα και τα παραγωγικά περιβάλλοντα (Kuyen, 2020).

Η εφαρμογή της μηχανικής μάθησης στην έρευνα στοχεύει στη βέλτιστη απόδοση του παραγόμενου μοντέλου και στη βελτίωση ακόμα και κατά μικρών ποσοστών των μετρικών απόδοσης. Έχει ως προτεραιότητα τη γρήγορη εκπαίδευση και την υψηλή διεκπεραιωτική ικανότητα. Επίσης, στην έρευνα είθισται αρχικά να συλλέγονται όλα τα διαθέσιμα δεδομένα, και ύστερα αυτά να υποβάλλονται στις κατάλληλες διαδικασίες επεξεργασίας, επαλήθευσης με στόχο τη χρήση τους ως δεδομένα εκπαίδευσης, επαλήθευσης και ελέγχου. Συνεπώς, από τη στιγμή που ολοκληρωθεί το προηγούμενο βήμα και κατά τη φάση αξιολόγησης των αναπτυγμένων τεχνικών μηχανικής μάθησης που εφαρμόστηκαν για την αντιμετώπιση του προβλήματος, τα δεδομένα παραμένουν στατικά και καλά σχηματισμένα.

Στην παραγωγή όμως, τα παραπάνω δεν έχουν ισχύ. Σε μια επιχειρησιακή εφαρμογή, η απόδοση του μοντέλου πρόβλεψης που επιδιώκεται εξαρτάται από τη φύση του προβλήματος. Στόχος είναι και σε αυτήν την περίπτωση το βέλτιστο μοντέλο, αλλά παράγοντες όπως το κέρδος, και το κόστος για την εύρεση αυτού μπορεί να περιορίσουν την επιμέρους βελτίωσή του (Kang et al., 2020). Επιπλέον, σχετικά με την ταχύτητα, οι διαδικασίες δεν επικεντρώνονται στη γρήγορη εκπαίδευση του μοντέλου αλλά αυτό που παίζει καθοριστικό ρόλο είναι η ταχύτητα πρόβλεψης ή ταξινόμησης (inference). Δεδομένου ότι πρόκειται για εφαρμογές που εξυπηρετούν ταυτόχρονα πολλούς χρήστες στόχος είναι ο χαμηλός χρόνος απόκρισης (low latency). Ωστόσο, από τις πιο σημαντικές παραμέτρους που λαμβάνονται υπόψιν είναι το γεγονός ότι τα δεδομένα συνεχώς μεταβάλλονται. Σε βάθος χρόνου, ανάλογα και τη φύση του προβλήματος που επιλύει ένα μοντέλο, είτε γιατί άλλαξαν οι συνήθειες των χρηστών ή γιατί η εφαρμογή προσθέτει ως στόχο μια ομάδα χρηστών με διαφορετικό σύνολο χαρακτηριστικών, τα δεδομένα μπορεί να αλλάξουν ριζικά κατανομή με αποτέλεσμα το μοντέλο να πρέπει να ενημερωθεί με τη νέα πληροφορία. Η διαρκής παρακολούθηση και η ενημέρωση του μοντέλου είναι μια διαδικασία που δε σχετίζεται με την έρευνα (Tuggener et al., 2019).

1.3 Προκλήσεις της μηχανικής μάθησης στην παραγωγή

Σύμφωνα με τα παραπάνω, η είσοδος της μηχανικής μάθησης σε παραγωγικό περιβάλλον αντιμετωπίζει αρκετές προκλήσεις. Ένα σύστημα μηχανικής μάθησης απαρτίζεται από τις εξής τρεις βασικές συνιστώσες: τον κώδικα, τα δεδομένα και το μοντέλο εκπαίδευσης (Karbhari, 2020). Στη φάση ανάπτυξης ενός τέτοιου συστήματος η αλλαγή έστω και μιας από τις τρεις αυτές συνιστώσες μπορεί να οδηγήσει σε διαφορετικό σύστημα πρόβλεψης και κατ' επέκταση σε διαφορετικό αποτέλεσμα. Μία από τις προκλήσεις που καλείται να αντιμετωπίσει, λοιπόν, είναι η εξάρτηση μεταξύ των τριών συνιστωσών και η από κοινού διατήρηση του ιστορικού αλλαγών τους (versioning).

Μια ακόμη δυσκολία σχετική με ένα σύστημα μηχανικής μάθησης είναι η δυνατότητα της αναπαραγωγισιμότητας τόσο ως προς τη σκοπιά των μεθόδων που χρησιμοποιήθηκαν, αλλά και ως προς τα αποτελέσματα που προέκυψαν από την εκπαίδευση ενός τέτοιου συστήματος. Είναι σύνηθες φαινόμενο οι επιστήμονες δεδομένων σε προσωπικά αρχεία (scripts ή notebooks), να παράγουν κάποιο κώδικα, να ελέγχουν τα αποτελέσματά του, να δοκιμάζουν διαφορετικές παραμέτρους ή αλγορίθμους ταξινόμησης με στόχο την εύρεση του συστήματος με τη βέλτιστη απόδοση και επαναληπτικά να οδηγούνται σε πολλές και διαφορετικές εκτελέσεις πειραμάτων. Συνεπώς, είναι σημαντικό να έχουν τη δυνατότητα να αναπαράξουν ένα παρελθοντικό πείραμα ύστερα από μερικές αλλαγές εύκολα και γρήγορα. Με αυτόν τον τρόπο καθίσταται δυνατή η αναπαραγωγή των πειραμάτων ειδικά σε προβλήματα όπου για την ικανοποιητική λύση τους απαιτείται η εφαρμογή μοντέλων πρόβλεψης με μη-ντετερμινιστικά χαρακτηριστικά (π.χ. γενετικοί αλγόριθμοι), είτε στη φάση αρχικοποίησής τους ή κατά την εκπαίδευση, πράγμα εξαιρετικά δύσκολο να επιτευχθεί με μη αυτοματοποιημένες ενέργειες.

Στο ίδιο μήκος κύματος προκλήσεων εντάσσεται η ταχύτητα προσαρμογής των ατόμων που εισέρχονται σε μια καινούρια ομάδα ή ένα καινούριο πρότζεκτ. Είναι απαραίτητο νεοεισαχθέντες ειδικοί να εγκλιματίζονται όσο το δυνατόν γρηγορότερα και να κατανοούν τις ήδη αναπτυγμένες τεχνικές χωρίς να χρειάζεται να ενημερωθούν από τα άτομα που τις ανέπτυξαν. Η οπτικοποίηση των μεγάλων και σύνθετων σωληνώσεων (pipelines) που παράγουν το εκάστοτε σύστημα μηχανικής μάθησης είναι αποδοτικότερη από την αναζήτηση σε μεγάλα αρχεία κειμένου ή κώδικα. Στη βιομηχανία, κάθε πρότζεκτ υλοποιείται από μια ομάδα ατόμων με ίδια ή και διαφορετική ειδίκευση, καθιστώντας αναγκαία τη δυνατότητα συνεργασίας, την εύκολη και άμεση πρόσβαση στα πειράματα των μελών της ομάδας, το διαμοιρασμό των δεδομένων και του κώδικα και την από κοινού απεικόνιση και διατήρησή τους σε μια ενιαία υποδομή (είτε πρόκειται για κάποια πλατφόρμα ή κάποιο εργαλείο).

Εκτός από τα προαναφερθέντα, που θα μπορούσε κανείς να τα ομαδοποιήσει ως προκλήσεις κατά τη φάση ανάπτυξης ενός συστήματος πρόβλεψης, εξίσου σημαντικές προκλήσεις αποτελούν ορισμένες δυσκολίες κατά τη φάση της εγκατάστασης του συστήματος (deployment) και της διαρκούς διαθεσιμότητάς του για να εξυπηρετεί τους χρήστες (serving) (Paleyes et al., 2021). Μια μεγάλη πρόκληση για κάθε είδους εφαρμογή

είναι η κλιμακωσιμότητα (scalability) (Weichert et al., 2019). Λόγω της πολύπλοκης σύνθεσης τέτοιων συστημάτων και του όγκου των δεδομένων που συνεχώς αυξάνονται, είναι σημαντικό να περιορίζονται οι πόροι που χρησιμοποιούνται σε φάσεις χαμηλής κίνησης και να αυξάνονται με την αύξηση της κίνησης.

Επιπλέον, η εγκατάσταση νέων ενημερωμένων μοντέλων πρόβλεψης σε ένα περιβάλλον που ήδη εξυπηρετεί μια σκοπιμότητα, πρέπει να γίνεται γρήγορα και χωρίς να διακόπτεται έστω και στιγμιαία η λειτουργία της εφαρμογής. Αντίστοιχα, εάν το παραγωγικό περιβάλλον ενημερωθεί με κάποιο νέο μοντέλο που τελικά δεν αποδίδει καλύτερα από μια προηγούμενη εκδοχή, η ύπαρξη αυτοματοποιημένων διαδικασιών που επαναφέρουν την προηγούμενη συνολική κατάσταση κρίνεται καθοριστική (Yao et al., n.d.).

Τέλος, είναι γεγονός ότι ένα σύστημα μηχανικής μάθησης όπως και αν είναι ορισμένο πάντα δίνει μια απάντηση σε μια είσοδο. Ωστόσο, η απάντηση αυτή δε σημαίνει ότι είναι και η σωστή. Τα δεδομένα σε μια παραγωγική εφαρμογή αλλάζουν συνεχώς στην πορεία του χρόνου (Cadavid et al., 2020), καθιστώντας δύσκολο των εντοπισμό αθόρυβων αποτυχιών (silent failures) (Carvajal Soto et al., 2019). Ένα τέτοιο σύστημα σε παραγωγικό περιβάλλον πρέπει να συνοδεύεται και από διαδικασίες που επαναληπτικά ελέγχουν την ορθή λειτουργία του (Mayr et al., 2019).

1.4 Αντικείμενο διπλωματικής και δομή

Πολλές από τις προκλήσεις με τις οποίες έρχεται αντιμέτωπη η ενσωμάτωση τεχνικών μηχανικής μάθησης στην παραγωγή αποτελούν προκλήσεις που αντιμετωπίζει εδώ και χρόνια η ανάπτυξη λογισμικού και εφαρμογών. Τη λύση την έδωσε ο ορισμός διαδικασιών υποδομών και εφαρμογών (operations) στοχεύοντας στην ευέλικτη ανάπτυξη της Πληροφορικής (agile), σε σταθερότερες υπηρεσίες (maintainability) και στη δυνατότητα για συχνότερες αλλαγές (quick rollouts-rollbacks). Οι ευρέως χρησιμοποιούμενες αυτές διαδικασίες, γνωστές και ως DevOps (development operations) δεν αρκούν για τις ανάγκες της μηχανικής μάθησης λόγω της σύνθετης φύσης της. Για αυτό το λόγο, έχουν αναπτυχθεί διαδικασίες που κατ' αντιστοιχία ονομάζονται MLOps (machine learning operations). Η παρούσα διπλωματική εργασία στη συνέχεια, εστιάζει και μελετά μερικές από τις τεχνικές και τα πολυάριθμα εργαλεία που έχουν αναπτυχθεί για την κάλυψη των αναγκών της μηχανικής μάθησης. Έπειτα, με βάση κάποιες παραμέτρους και με στόχο την επίλυση ενός προβλήματος ταξινόμησης, σχετικά με την κατηγοριοποίηση των θέσεων στοιχείων εικόνων από υποθαλάσσια εδάφη σε προκαθορισμένες κλάσης σύστασης, επιλέγονται και συνδυάζονται μερικά από αυτά τα εργαλεία. Σχεδιάζεται και αναπτύσσεται μια υποδομή που περιλαμβάνει τα εν λόγω εργαλεία, αλλά και επιπλέον υπηρεσίες που εξυπηρετούν σε μια ολοκληρωμένη εγκατάσταση ενός συστήματος μηχανικής μάθησης, προκειμένου να επισημανθούν τα οφέλη που προσκομίζονται από τη χρήση τους. Προσομοιώνονται, επίσης, οι συνθήκες κατά τις οποίες ένα ήδη εγκατεστημένο μοντέλο χρήζει ενημέρωσης, προκειμένου να δύναται να αποδώσει ικανοποιητικά στα νέα δεδομένα με την πάροδο του χρόνου, περιγράφονται τα βήματα που διαμορφώνουν αυτοματοποιημένες διαδικασίες, με

τις οποίες παράγεται κάθε φορά ένα μοντέλο με μεγαλύτερη ευαισθησία και καλύτερη διακριτική ικανότητα, και τελικά εγκαθίσταται στην παραγωγή.

Κεφάλαιο 2: Θεωρητικό υπόβαθρο

Μπορεί μέχρι σήμερα η εφαρμογή της μηχανικής μάθησης στη βιομηχανία να μην έχει ωριμάσει πλήρως, ωστόσο πολλές επιχειρήσεις και ομάδες μηχανικών έχουν προχωρήσει σε καινοτόμες μεθόδους και στην ανάπτυξη εργαλείων για την απλοποίηση της διαδικασίας παραγωγής μοντέλων προβλέψεων και τον ομαλό κύκλο ζωής της μηχανικής μάθησης στις εφαρμογές. Στο κεφάλαιο αυτό αναφέρονται μερικά τέτοια παραδείγματα και διερευνώνται ορισμένα από τα εργαλεία που έχουν υλοποιηθεί και γίνονται ολοένα πιο δημοφιλή.

2.1 Ομάδες και ανεπτυγμένα εργαλεία

2.1.1 Airbnb και Apache Airflow

Μία από τις πιο γνωστές εφαρμογές είναι η ηλεκτρονική πλατφόρμα της Airbnb που ασχολείται με την εύρεση και την ενοικίαση καταλυμάτων. Η ομάδα της Airbnb το 2014 ανέπτυξε ένα λογισμικό ανοιχτού κώδικα για τη διαχείριση αυτόματων ροών εργασιών (Wikipedia, 2019). Το 2016 ενσωματώθηκε στο πρόγραμμα Apache Incubator. Δεδομένου ότι η μηχανική μάθηση σχετίζεται άμεσα με προκαθορισμένες ακολουθίες εργασιών, το Airflow έχει βρει εφαρμογή στο συγκεκριμένο πεδίο και χρησιμοποιείται από χιλιάδες επιχειρήσεις που δραστηριοποιούνται στο πεδίο της. Μερικές από αυτές είναι οι PayPal, Facebook, CloudFlare, Slack και Robinhood (HG Insights, 2021).

2.1.2 Uber και Michelangelo

Η Uber είναι μια επιχείρηση παροχής υπηρεσιών μεταφοράς προσώπων, η οποία εδραιώνεται σε πολλές πόλεις παγκοσμίως. Προκειμένου να βελτιώσει την εμπειρία των χρηστών και να προσωποποιήσει την εφαρμογή για κάθε χρήστη έχει επενδύσει στη μηχανική μάθηση και την τεχνητή νοημοσύνη. Μια ομάδα εξειδικευμένων μηχανικών προχώρησε στην υλοποίηση μια εσωτερικής πλατφόρμας μηχανικής μάθησης ως υπηρεσία (ML-as-a-service), που ονομάζεται Michelangelo (Uber et al., 2017), για να κλιμακώσει τις δυνατότητες της μηχανικής μάθησης και να καλύψει τις απαιτήσεις της πλατφόρμας υπηρεσίας “ταξί”. Κίνητρο για την ανάπτυξή του, αποτέλεσε η ανάγκη που υπήρχε για τη συνεργασία μεταξύ των επιστημόνων δεδομένων και των μηχανικών για την εγκατάσταση των μοντέλων, που παρήγαγαν οι πρώτοι, από τους δεύτερους σε συγκεκριμένο χρονικό διάστημα. Στόχευαν στη δημιουργία ενός συστήματος που οδηγεί σε αξιόπιστες, ομοιόμορφες και εύκολα αναπαράξιμες σωληνώσεις εργασιών σχετικές με την εκπαίδευση και τη δημιουργία αποτελεσματικών μοντέλων πρόβλεψης.

2.1.3 Spotify και Luigi

Αντίστοιχα με την Airbnb, η γνωστή σε όλους πλατφόρμα αναπαραγωγής μουσικής Spotify ανέπτυξε ένα πακέτο γραμμένο σε python, ανοικτού κώδικα, με στόχο την κατασκευή σύνθετων pipelines. Ενδεικτικά, μερικές από τις εταιρείες που έχουν επενδύσει στο συγκεκριμένο εργαλείο για την κάλυψη των αναγκών τους είναι οι Foursquare, Red Hat, Glossier, Big Data, Grovo και Movio (Spotify AB, 2012-2019).

2.2 Εργαλεία διαδικασιών μηχανικής μάθησης (MLOps tools)

Σύμφωνα με όσα αναφέρθηκαν στο προηγούμενο κεφάλαιο για την κάλυψη των απαιτήσεων που παρουσιάζει η μηχανική μάθηση στην παραγωγή, έχουν αναπτυχθεί εργαλεία με στόχο τις ανάγκες για:

- διατήρηση της ιστορικότητας του τρίπτυχου κώδικας, δεδομένα, μοντέλα
- κατανεμημένη (προ-)επεξεργασία και ανάλυση των δεδομένων
- κατανεμημένη εκπαίδευση μοντέλων
- οργάνωση και διαχείριση των ml pipelines
- διαθεσιμότητα μοντέλων ως υπηρεσία (model serving/executors)

Στη συνέχεια παρουσιάζονται και αντιπαραβάλλονται τα πιο δημοφιλή για κάθε κατηγορία.

2.2.1 Καταγραφή διαφορετικών εκδόσεων κώδικα και δεδομένων (Model Data versioning)

Η ανάγκη για τη διατήρηση ενός ιστορικού διαφορετικών εκδόσεων σχετικά με τον κώδικα, τα δεδομένα, τις παραμέτρους που ορίζουν τα μοντέλα πρόβλεψης και κατ' επέκταση και τα παραγόμενα μοντέλα είναι αναμφισβήτητη. Τρία από τα πιο συχνά χρησιμοποιούμενα εργαλεία είναι τα εξής:

- Pachyderm: Πρόκειται για μια δωρεάν και ανοικτού κώδικα πλατφόρμα. Παρέχει τη γνώση της γενεαλογίας των δεδομένων (data lineage), δηλαδή με πλήρη βεβαιότητα τις σχέσεις και την πορεία του τρίπτυχου ΔΚΜ (δεδομένα, κώδικας, μοντέλα) σε βάθος χρόνου. Προσφέρει επίσης, δυνατότητες αυτοματοποίησης χειροκίνητων διεργασιών (Pachyderm Inc, 2021).
- DVC: Είναι συντομογραφία της ορολογίας Data Version Control (DVC, 2021). Είναι επίσης, εργαλείο ανοικτού κώδικα που διαχειρίζεται αρχεία μεγάλου μεγέθους, όπως σύνολα δεδομένων, μοντέλα μηχανικής μάθησης, μετρικές και κώδικα με στόχο να επιτύχει το διαμοιρασμό τους μεταξύ των ομάδων και τη δυνατότητα αναπαραγωγής τους. Είναι βασισμένο στον τρόπο λειτουργίας του Git version control, ενσωματώνοντας τη δυνατότητα της διακλάδωσης, χωρίς να περιορίζεται από το μέγεθος των αρχείων. Μπορεί επίσης να χρησιμοποιηθεί ως εργαλείο διαχείρισης

pipelines με σημαντικό προτέρημα την ανεξαρτησία του από τη γλώσσα προγραμματισμού (language-agnostic).

- Dolt: Πρόκειται για μια σχεσιακή βάση δεδομένων με ιστορικό εκδόσεων, επίσης δωρεάν και ανοικτού κώδικα (DoltHub Inc, 2021). Είναι εμπνευσμένη από γνωστά χαρακτηριστικά του Git και της βάσης δεδομένων MySQL. Σημαντικά χαρακτηριστικά που έχει προσθέσει στη λειτουργία του είναι η αποθήκευση της γενεαλογίας των δεδομένων, η ιστορικότητα χρήσιμων τιμών σε συνάρτηση με το χρόνο και η συνεργατικότητα των επιμέρους οντοτήτων που διαχειρίζεται με τη μορφής σχεσιακών σχημάτων.

2.2.2 Κατανεμημένη (προ-)επεξεργασία / Ανάλυση δεδομένων (Distributed data (pre-)processing / Analytics)

Αρκετά συχνά, τα δεδομένα που χρησιμοποιούνται για την εκπαίδευση μοντέλων πρόβλεψης είναι υπερβολικά μεγάλου μεγέθους, της τάξεως Terabyte (τέραμπαιτ - TB) και Petabyte (πέταμπαιτ - PB), ενώ οι διαθέσιμοι πόροι σε ένα μηχάνημα δεν είναι απεριόριστοι. Η δυνατότητα επεξεργασίας όλων των δεδομένων συνολικά σε ένα μηχάνημα καθίσταται αδύνατη λόγω του απαιτούμενου χρόνου επεξεργασίας αλλά και της απαιτούμενης μνήμης. Συνεπώς, εισάγεται η έννοια της επεξεργασίας τους σε πολλά μηχανήματα κατανεμημένα και η υλοποίηση εργαλείων που την πραγματοποιούν. Τα πιο διαδεδομένα από αυτά είναι:

- Apache Spark: Το Spark εργαλείο χρησιμοποιείται ευρέως από πολλούς οργανισμούς για την επεξεργασία και την ανάλυση μεγάλων συνόλων δεδομένων, επιτυγχάνοντας υψηλή απόδοση (The Apache Software Foundation, 2018). Είναι συμβατό με αρκετές γλώσσες προγραμματισμού, όπως είναι οι Java, Scala, Python και R, ενώ υποστηρίζει την επεξεργασία γράφων, μέσω του χαρακτηριστικού graphX.
- Dask: Πρόκειται για μια βιβλιοθήκη ανοικτού κώδικα, που έχει αναπτυχθεί στη γλώσσα προγραμματισμού Python, σε συνεργασία με άλλα όμοια πακέτα, όπως είναι τα NumPy, pandas και scikit-learn (NumFOCUS, 2019). Οι δυνατότητές της στοχεύουν σε προχωρημένες μεθόδους παράλληλης επεξεργασίας και ανάλυσης των δεδομένων σε μεγάλη κλίμακα. Δεδομένου ότι το Spark είναι ένα ολόκληρο οικοσύστημα, το Dask εργαλείο είναι μικρότερο και λιγότερο ώριμο σε σχέση με το Spark. Μπορεί και χειρίζεται πολυδιάστατους πίνακες και μοντέλα, όμως εκλείπει από αυτό η δυνατότητα διαχείρισης γράφων.
- Ray: Αποτελεί επίσης ανοικτού κώδικα λογισμικό, που παρέχει δυνατότητες κατανεμημένης επεξεργασίας δεδομένων σε μεγάλη κλίμακα σε οποιοδήποτε περιβάλλον, είτε σε τοπικό μηχάνημα ή σε νέφος. Εκτός από την κατανεμημένη επεξεργασία, εφαρμόζεται και στα υπόλοιπα στάδια του κύκλου ζωής της μηχανικής μάθησης, όπως είναι η εκπαίδευση νευρωνικών δικτύων ρυθμίζοντας κατάλληλα τις υπερ-παραμέτρους, η ενισχυτική μάθηση ή ακόμα και εγκατάσταση του μοντέλου εξυπηρέτησης σε κατανεμημένα συστήματα (The Ray Team, 2021).

2.2.3 Κατανεμημένη εκπαίδευση μοντέλων (Distributed model training)

Είναι πιθανό για αρκετά προβλήματα ταξινόμησης και πρόβλεψης οι γνωστοί αλγόριθμοι ταξινόμησης να μην αρκούν για την εύρεση ενός αποδοτικού μοντέλου και να απαιτείται η προσέγγισή τους με βαθιά μηχανική μάθηση. Τα νευρωνικά δίκτυα που προκύπτουν αποτελούνται συχνά από πολλά στρώματα με πολλαπλούς νευρώνες και ακόμα περισσότερα συναπτικά βάρη. Για την αποτελεσματική εκπαίδευσή τους τόσο από την άποψη του χρόνου αλλά και της μνήμης που απαιτείται για τον υπολογισμό των βαρών τους, καθιστάται αναγκαία η χρήση περισσότερων μηχανημάτων. Εργαλεία κατανεμημένης εκπαίδευσης που χρησιμοποιούνται από πολλούς οργανισμούς είναι τα εξής:

- Horovod: Πρόκειται για τεχνολογία ανοιχτού κώδικα που εστιάζει στη γρήγορη και εύκολη εκπαίδευση βαθιών νευρωνικών δικτύων που ορίζονται μέσω TensorFlow, Keras, PyTorch και Apache MXNet (Uber Technologies, Inc., 2018).
- RaySGD: Είναι μια ελαφριά βιβλιοθήκη κατανεμημένης εκπαίδευσης μοντέλων συμβατών με τις βιβλιοθήκες TensorFlow και PyTorch. Με πολύ μικρές αλλαγές στον κώδικα, η κλιμάκωση είναι δυνατό να επιτευχθεί σε πολλαπλούς κόμβους, πολλαπλές μονάδες επεξεργασίας CPUs ή πολλαπλές GPUs. Είναι πολύ εύκολο να συνδυαστεί με τις υπόλοιπες Ray εφαρμογές (Tune, Serve, RLlib) (The Ray Team, 2021).
- Distributed TensorFlow: Η πλατφόρμα TensorFlow παρέχει μια διεπαφή προγραμματισμού εφαρμογών (API) δίνοντας τη δυνατότητα για παράλληλη εκπαίδευση μοντέλων σε πολλαπλές GPUs ή TPUs (Abadi et al., 2015).

2.2.4 Οργάνωση και διαχείριση σωληνώσεων μηχανικής μάθησης (ML pipeline orchestration)

Σχετικά με την κάλυψη της ανάγκης για διαχείριση και οργάνωση αλυσίδων διεργασιών στο πεδίο της μηχανικής μάθησης, έχουν αναπτυχθεί πολλά εργαλεία. Τα πιο δημοφιλή που έχουν υιοθετηθεί και χρησιμοποιούνται περισσότερο είναι τα παρακάτω:

- Airflow: Πρόκειται για μια ώριμη πλατφόρμα ανοιχτού κώδικα με πάρα πολλές δυνατότητες που σχεδιάστηκε για τον ορισμό, τον προγραμματισμό και την παρακολούθηση ροών εργασιών ανεξαρτήτου σκοπού. Παρέχει δυνατότητες κλιμάκωσης, δυναμικής δημιουργίας pipeline και εκτέλεσης των εργασιών ακόμα και σε υπηρεσίες νέφους, όπως είναι η πλατφόρμα της Google (GCP), της Amazon και άλλες υπηρεσίες τρίτων (The Apache Software Foundation, 2021).
- Kubeflow: Το λογισμικό αυτό αναπτύχθηκε για την εγκατάσταση ολοκληρωμένων ροών εργασιών μηχανικής μάθησης σε Kubernetes περιβάλλον. Είναι συμβατό με πολλές τεχνολογίες μηχανικής μάθησης όπως είναι τα TensorFlow, Seldon-core, Jupyter, Pachyderm και πολλά ακόμα. Δεδομένου ότι είναι πολύ πρόσφατο εργαλείο οι δυνατότητές του και οι τεχνολογίες που υποστηρίζει διαρκώς αυξάνονται. Το Kubeflow αποτελεί μια ολοκληρωμένη λύση καθώς αντιμετωπίζει τα ζητήματα της

μηχανικής μάθησης από άκρη σε άκρη (καταγραφή πειραμάτων, διαχείριση σωληνώσεων, εγκατάσταση υπηρεσιών). Ωστόσο, το μεγάλο εύρος των λειτουργιών που παρέχει, οδηγεί σε υψηλές υπολογιστικές και αποθηκευτικές απαιτήσεις (The Kubeflow Authors, 2018-2021).

- MLflow: Η συγκεκριμένη πλατφόρμα ανοιχτού κώδικα εστιάζει στη διαχείριση του κύκλου ζωής της μηχανικής μάθησης (MLflow Project, a Series of LF Projects, LLC, 2021). Χρησιμοποιείται για την εκτέλεση και την καταγραφή πειραμάτων, τη γρήγορη και εύκολη αναπαραγωγή τους, την εγκατάσταση μοντέλων πρόβλεψης σε διάφορα περιβάλλοντα υπηρεσιών και την αποθήκευση, τη σήμανση και τη διαχείριση ανεπτυγμένων μοντέλων ταξινόμησης σε ένα κεντρικό μητρώο.
- Argo: Πρόκειται για ένα ακόμα εργαλείο ανοιχτού κώδικα που υποστηρίζει την εκτέλεση ακολουθιακών διεργασιών και εξυπηρετεί τις ανάγκες συνεχούς ενσωμάτωσης κώδικα και εγκατάστασης εφαρμογών σε Kubernetes περιβάλλον (Argo Project Authors, 2020).
- Luigi: Το συγκεκριμένο πακέτο αναπτύχθηκε από την ομάδα του Spotify στη γλώσσα Python. Συμβάλλει στη δημιουργία σύνθετων pipelines και διευκολύνει την παρακολούθηση και τη διαχείριση των εκτελέσεών τους (Spotify AB, 2012-2019).

2.2.5 Διαθεσιμότητα μοντέλων ως υπηρεσία (Model Serving / Executors)

Για κάθε πρόβλημα ταξινόμησης, μετά το στάδιο εύρεσης του βέλτιστου μοντέλου πρόβλεψης ακολουθεί το στάδιο της εγκατάστασής του σε παραγωγικό περιβάλλον με τη μορφή υπηρεσίας. Αυτό μπορεί να επιτευχθεί οργανωμένα και αυτόματα είτε με τη χρήση εργαλείων που αναλαμβάνουν την διαθεσιμότητα των μοντέλων ως μεμονωμένες εφαρμογές ή με στοιχεία εκτελεστές (executors) που ενσωματώνουν το μοντέλο στη λειτουργία τους. Μερικά από αυτά είναι:

- Seldon-core: Είναι πλατφόρμα ανοιχτού κώδικα που μετατρέπει τα παραχθέντα μοντέλα μηχανικής μάθησης σε μεμονωμένες υπηρεσίες REST[1]/GRPC[2] στην παραγωγή (Seldon Technologies Ltd, 2021).
- Cortex: Πρόκειται για πλατφόρμα με δυνατότητες εγκατάστασης, διαχείρισης και κλιμάκωσης υπηρεσιών μηχανικής μάθησης στην παραγωγή (Cortex Labs, 2021).
- TorchServe: Ειδικεύεται στην εγκατάσταση PyTorch μοντέλων σε παραγωγικό περιβάλλον (PyTorch Serve Contributors, 2020).
- TFX (TensorFlow Extended): Η πλατφόρμα αυτή συμβάλλει στην εγκατάσταση ακολουθιακών διεργασιών στην παραγωγή, όπου κάθε διεργασία έχει οριστεί

[1] REST (Representational state transfer): Η αντιπροσωπευτική μεταφορά κατάστασης είναι ένα αρχιτεκτονικό στυλ λογισμικού που χρησιμοποιεί ένα υποσύνολο HTTP. Συνήθως χρησιμοποιείται για τη δημιουργία διαδραστικών εφαρμογών που χρησιμοποιούν υπηρεσίες Web.

[2] GRPC (gRPC Remote Procedure Calls):Το gRPC είναι ένα σύστημα κλήσης απομακρυσμένης διαδικασίας ανοιχτού κώδικα που αναπτύχθηκε αρχικά στην Google το 2015. Βασίζεται στη χρήση του πρωτοκόλλου HTTP/2 για τη σύζευξη επικοινωνίας μεταξύ των εφαρμογών.

σύμφωνα με βιβλιοθήκες συμβατές με το πακέτο TensorFlow, εξασφαλίζοντας υψηλή απόδοση (Abadi et al., 2015).

- Celery: Είναι ένα ευέλικτο και αξιόπιστο κατανεμημένο σύστημα σχεδιασμένο να διαχειρίζεται πληθώρα μηνυμάτων. Η λειτουργία του βασίζεται στην εκτέλεση ουράς εργασιών σε πραγματικό χρόνο, ενώ ταυτόχρονα υποστηρίζει και εκτέλεση εργασιών σε προγραμματισμένο χρόνο (Ask Solem & contributors, 2009-2018).

2.3 Καθοριστικοί παράγοντες επιλογής των κατάλληλων εργαλείων

Αναμφισβήτητα, υπάρχουν πολλά εργαλεία που μπορούν να αντιμετωπίσουν σε ικανοποιητικό βαθμό τις ανάγκες μιας ομάδας εξειδικευμένης στη μηχανική μάθηση, και σίγουρα εξακολουθούν να αναπτύσσονται ακόμα περισσότερα. Δημιουργούνται, ωστόσο, διάφορα ερωτήματα σχετικά με την επιλογή των καταλληλότερων από αυτά. Τα ερωτήματα έχουν να κάνουν με το ποιες παράμετροι καθορίζουν την επιλογή, το κατά πόσο είναι εφικτό να χρησιμοποιηθεί μόνο ένα ή αν χρειάζεται συνδυασμός περισσότερων. Οι περιοχές κλειδιά που πρέπει να αναλογιστεί η σχετική ομάδα αναφέρονται στη συνέχεια.

Αρχικά, σημαντικό ρόλο στην επιλογή των εργαλείων προς χρήση έχει το περιβάλλον στο οποίο αποθηκεύονται τα δεδομένα της εφαρμογής. Η πρόσβαση στα δεδομένα οφείλει να γίνεται με τρόπο συμβατό ως προς τα εργαλεία. Υπάρχουν τρεις εναλλακτικές ως προς το περιβάλλον και αυτές είναι: on premises, δηλαδή σε μηχανήματα που ανήκουν στην ομάδα, σε νέφη (cloud storage) σε έτοιμους αποθηκευτικούς χώρους μέσω των υπηρεσιών GCS (Google Cloud Storage), AWS (Amazon Web Services) και Azure Storage, αλλά και σε υβριδικό συνδυασμό των δύο επιλογών. Η πρώτη περίπτωση προτιμάται για πολύ μεγάλα μεγέθη δεδομένων και όταν η εκπαίδευση των μοντέλων πραγματοποιείται επίσης σε τοπικά μηχανήματα, ενώ η δεύτερη όταν η εκπαίδευση των μοντέλων και η εγκατάστασή τους λαμβάνει χώρα σε υπολογιστικά νέφη. Αξίζει να σημειωθεί πως στη δεύτερη επιλογή εάν υπάρχουν ευαίσθητα δεδομένα για τα οποία απαιτείται προστασία με προηγμένες μεθόδους ασφάλειας η υβριδική προσέγγιση είναι κατάλληλη για την αποθήκευση των ευαίσθητων δεδομένων τοπικά (in house).

Καθοριστικό παράγοντα στην επιλογή των εργαλείων έχουν τα χαρακτηριστικά των ίδιων των εργαλείων. Αρχικά, ομάδες και εταιρείες, που προχωρούν στην ανάπτυξη μιας εφαρμογής με μεθόδους μηχανικής μάθησης και η ιδεολογία τους βασίζεται σε λύσεις ανοιχτού κώδικα, είναι συνηθέστερο να προτιμούν open source τεχνολογίες. Αντίστοιχα, εάν πρόκειται για εργαλεία επί πληρωμή, το διαθέσιμο οικονομικό ποσό περιορίζει τις πιθανές επιλογές και τον συνδυασμό τους ώστε να είναι εφικτή μια συνολική λύση. Επιπλέον του διαθέσιμου οικονομικού προϋπολογισμού, τα εργαλεία κλειστού κώδικα συνηθίζεται να μην προχωρούν σε ενσωμάτωση υπηρεσιών εργαλείων ανταγωνιστών κατασκευαστών, δυσχεραίνοντας τη διαλειτουργικότητα και τη συνεργασία περαιτέρω λύσεων. Επιπρόσθετα της αξίας των εργαλείων, προτεραιότητα στην επιλογή έχουν τα εργαλεία που είναι περισσότερο αποτελεσματικά, αποδοτικά και δημοφιλή σε σχέση με άλλα. Είναι σημαντικό δηλαδή να έχουν υιοθετηθεί από πολλούς και μεγάλους οργανισμούς στις εσωτερικές τους

διαδικασίες. Τέλος, εξίσου σημαντικές είναι η υποστήριξη και η συχνότητα των ενημερώσεων που παρέχουν οι έτοιμες λύσεις.

Τα τελευταία χρόνια επιλέγεται η δομή των εφαρμογών να βασίζεται στην αρχή των microservices (μονολιθικές υπηρεσίες). Με αυτόν τον τρόπο αποδομείται η συνολική λειτουργία μιας υπηρεσίας σε πολλές και μικρότερες απομονωμένες λειτουργίες. Μερικά από τα θετικά που προκύπτουν από την αποδόμηση είναι η γρήγορη ενημέρωση μιας και μόνο λειτουργίας, η ευέλικτη διαθεσιμότητα υπηρεσιών (deployment releases) σε διαφορετικά περιβάλλοντα ή και με διαφορετικές στρατηγικές (canary deployment[3]), η κλιμακωσιμότητα και η ευελιξία ως προς την κατανάλωση του μοντέλου πρόβλεψης. Ωστόσο, δημιουργούνται και μερικές δυσκολίες που σχετίζονται με την ανάγκη για παρακολούθηση επιπλέον συστημάτων, πολλαπλές εγκαταστάσεις των επιμέρους λειτουργιών, αντί μιας συνολικής, και την ενδεχόμενη αύξηση του χρόνου απόκρισης λόγω της ενδοεπικοινωνίας των επιμέρους συστημάτων. Τα εργαλεία που επιλέγονται είναι σημαντικό να συμβάλλουν στην ομαλοποίηση της microservice αρχιτεκτονικής χωρίς να επιβαρύνουν με επιπλέον φορτίο απαιτώντας περισσότερο χρόνο και έργο από την ομάδα προγραμματιστών.

Τέλος, υπάρχουν βασικές λειτουργίες που δεν πρέπει να εμποδίζονται από τη χρήση των επιλεχθέντων τεχνολογιών. Η κλιμακωσιμότητα τόσο ως προς τα δεδομένα όσο και ως προς τις λειτουργίες, η παραλληλοποίηση των εκτελέσεων κώδικα καθώς και βασικές λειτουργίες παρακολούθησης (monitoring), καταγραφής λεπτομερειών σχετικών με την υπηρεσία (logging) και ασφάλειας της εφαρμογής (security) θα πρέπει να εξασφαλίζονται όσον αφορά τη χρήση τους και τις εσωτερικές τους λειτουργίες.

[3] Πρόκειται για στρατηγική που επιτυγχάνει σταδιακή κυκλοφορία νέων λειτουργιών στην παραγωγή. Συνήθως, το ενημερωμένο λογισμικό διατίθεται μόνο σε μια μικρή μερίδα χρηστών/καταναλωτών ώστε να ελεγχθεί η πλήρης λειτουργικότητά του σε παραγωγικό περιβάλλον και να ληφθούν σχόλια από τους χρήστες.

Κεφάλαιο 3: Προτεινόμενα εργαλεία και Υποδομή

Σε αυτό το κεφάλαιο επιλέγεται και προτείνεται ο συνδυασμός δύο από τα εργαλεία που αναφέρθηκαν στο προηγούμενο κεφάλαιο με στόχο την αυτοματοποίηση των διαδικασιών ανάπτυξης ενός βέλτιστου μοντέλου πρόβλεψης και εγκατάστασής του στο παραγωγικό περιβάλλον. Σχεδιάζεται και αναλύεται η υποδομή που περιλαμβάνει την εγκατάσταση των δύο εργαλείων σε ένα οικοσύστημα μαζί με άλλες υπηρεσίες για τη βέλτιστη λειτουργία τους. Τέλος, παρουσιάζονται τα οφέλη και σχολιάζεται η συμβολή και η αξία της εγκατάστασης μιας τέτοιας υποδομής με στόχο την ανάπτυξη λύσεων σε προβλήματα του πεδίου της μηχανικής μάθησης.

3.1 Εγκατάσταση εργαλείων

3.1.1 MLflow

Δεδομένου ότι η ανάπτυξη ενός μοντέλου μηχανικής μάθησης είναι μια σύνθετη διαδικασία αποτελούμενη από αρκετά στάδια χρειάζεται η εγκατάσταση ενός τουλάχιστον εργαλείου σχετικού με την οργάνωση και την ακολουθιακή εκτέλεση αυτών των σταδίων. Από αυτά επιλέχθηκε το MLflow εργαλείο λόγω της ευκολίας εγκατάστασής του, της ευκολίας ως προς τη χρήση του και την κατανόηση των λειτουργιών του, και των χαμηλών υπολογιστικών πόρων που απαιτεί σε σχέση με άλλα εργαλεία.

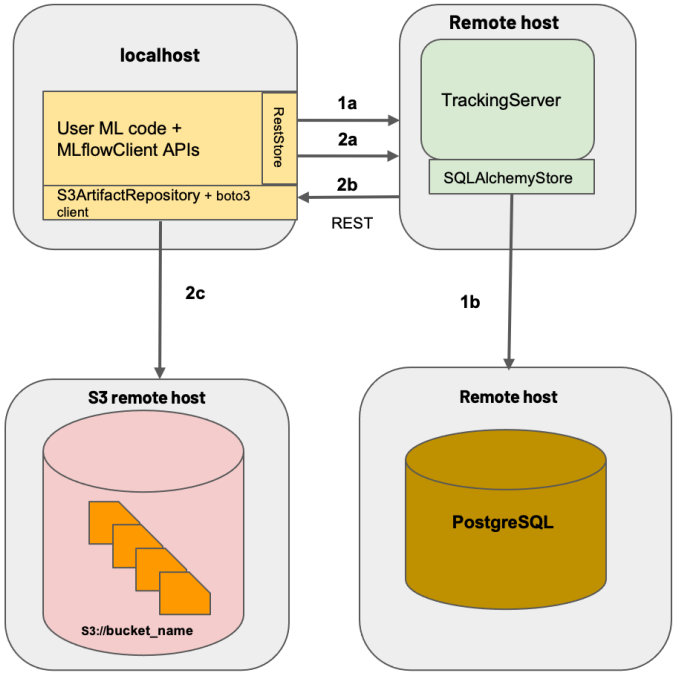
Το MLflow εργαλείο χρησιμοποιεί δύο διαφορετικά στοιχεία για αποθήκευση, το χώρο αποθήκευσης της λειτουργίας της εφαρμογής (backend storage) και το χώρο αποθήκευσης αντικειμένων (artifact store). Ο πρώτος σχετίζεται με την καταγραφή οντοτήτων και λεπτομερειών που αφορούν τις πειραματικές διαδικασίες που εκτελούνται. Τέτοιες είναι οι διαφορετικές εκτελέσεις στο χρόνο, οι παράμετροι, μετρικές, σημειώσεις, μετα-δεδομένα και επισημάνσεις. Ο δεύτερος αφορά την αποθήκευση και διατήρηση αντικειμένων όπως είναι τα αρχεία κειμένου και δεδομένων, εικόνες, δεδομένα στη μνήμη (in-memory) απαραίτητα για την εκτέλεση και τα μοντέλα που παράγονται μέσω των πειραμάτων. Η διασύνδεση και η επικοινωνία μεταξύ του MLflow εξυπηρετητή, των δύο αποθηκευτικών χώρων και του περιβάλλοντος ανάπτυξης του κώδικα των πειραμάτων παρουσιάζεται στην εικόνα 3.1.

Από προεπιλογή, το MLflow αποθηκεύει τα πειράματα σε έναν τοπικό φάκελο στο περιβάλλον εγκατάστασής του με μορφή συμβατή με SQLAlchemy. Ωστόσο, υποστηρίζει τη δυνατότητα καταγραφής των πειραμάτων σε άλλο εξυπηρετητή MLflow, σε περιβάλλον Databricks και σε βάσεις δεδομένων σύμφωνα με τις διαλέκτους mysql, mssql, sqlite και postgressql. Στην παρούσα εργασία επιλέχθηκε η καταγραφή να γίνει σε έναν απομονωμένο εξυπηρετητή με τη βάση δεδομένων postgres.

Για την αποθήκευση αντικειμένων σχετικών με τα πειράματα το MLflow υποστηρίζει τις εξής δυνατότητες:

- Amazon S3 and S3-compatible storage
- Azure Blob Storage
- Google Cloud Storage
- FTP server
- SFTP server
- NFS
- HDFS

Η επιλογή της καταλληλότερης μεθόδου εξαρτάται από τον όγκο και την εμπιστευτικότητα των αντικειμένων. Στα πλαίσια της παρούσας εργασίας επιλέχθηκε η υπηρεσία minIO, δηλαδή μιας στοίβας αποθήκευσης δεδομένων απόλυτα συμβατής με την τεχνολογία της Amazon S3, που είναι ευρέως διαδεδομένη.



Εικόνα 3.1: Δομή και διασύνδεση των συνιστωσών που απαιτούνται για την καταγραφή των πειραμάτων και των αντικειμένων του MLflow server σε εξωτερικά συστήματα (MLflow Project, a Series of LF Projects, LLC, 2021).

Το MLflow εργαλείο είναι ιδανικό για την εκτέλεση διαφορετικών πειραμάτων με διαφορετικές προσεγγίσεις για το εκάστοτε πρόβλημα ορίζοντας το αντίστοιχο πρότζεκτ που ονομάζεται MLproject. Οι κύριες ιδιότητες ενός MLproject είναι οι παρακάτω:

- Όνομα (name): Ένας λακωνικός αλλά περιγραφικός τίτλος του πρότζεκτ.
- Σημεία εισόδου (entry points): Πρόκειται για λέξεις εντολές που συμπληρώνονται από επιπλέον πληροφορίες σχετικά με τις παραμέτρους τους (τύπος και προκαθορισμένες τιμές). Κάθε πρότζεκτ έχει ένα τουλάχιστον τέτοιο σημείο που χρησιμοποιείται για την κλήση του εκτελέσιμου κώδικα που αντιστοιχεί στο

συγκεκριμένο entry point. Κάθε entry point μπορεί να αντιστοιχηθεί σε μια λειτουργία ή ένα στάδιο ενός workflow μηχανικής μάθησης.

- Περιβάλλον (environment): Περιλαμβάνει το σύνολο των βιβλιοθηκών και των εξαρτήσεων που απαιτεί ο κώδικας προκειμένου να εκτελεστεί επιτυχώς. Τα περιβάλλοντα αυτά μπορούν να οριστούν είτε ως conda environments ή ως docker containers.

Κατά την εκτέλεση ενός πρότζεκτ κατασκευάζεται, ενεργοποιείται το κατάλληλο περιβάλλον και ύστερα εκτελείται ο κώδικας, επιτυγχάνοντας απομόνωση του πρότζεκτ από άλλα χωρίς να μεταβάλλεται το ήδη προϋπάρχον περιβάλλον. Με βάση τα παραπάνω μπορεί κανείς να αναπτύξει διαφορετικές υλοποιήσεις για κάθε στάδιο ενός αυτοματοποιημένου κύκλου μηχανικής μάθησης και να τα συνδυάσει κατάλληλα για να επιτύχει το βέλτιστο μοντέλο.

3.1.2 Airflow

Αν και το MLflow εργαλείο λύνει τα προβλήματα ενός pipeline ανάπτυξης μοντέλων πρόβλεψης με μερικές επιπλέον λειτουργίες που αφορούν την οργάνωση των πειραμάτων, τη συσχέτιση του κώδικα με τα δεδομένα και τα μοντέλα που προκύπτουν από αυτά, ακόμα και τη διαθεσιμότητα τους για πρόβλεψη, δημιουργήθηκε το ερώτημα πώς μπορεί να αυτοματοποιηθεί και η διαδικασία ενημέρωσης των μοντέλων σε παραγωγικό περιβάλλον. Γι' αυτό το λόγο εγκαταστήθηκε επιπλέον η πλατφόρμα Airflow, που αποτελεί πιο ώριμη τεχνολογία με δυνατότητες χρονοπρογραμματισμού εκτέλεσης γράφων διεργασιών, επίβλεψης αποτυχίας εκτελέσεων, δημιουργία συναγερμών για αποτυχημένες διεργασίες, συνεργασίας και ενσωμάτωσης λειτουργιών άλλων τεχνολογιών (Apache Tools, Databricks, Google, Microsoft Azure κ.α.). Το Airflow αποτελείται από δύο βασικά στοιχεία, τον εξυπηρετητή ιστού που επιτρέπει την οπτικοποίηση των ροών εργασιών, τη διαχείρισή τους και άλλες λειτουργίες, και τον χρονοπρογραμματιστή ο οποίος είναι υπεύθυνος για την εκτέλεση των προκαθορισμένων διαδικασιών από εργάτες με τη σωστή σειρά και στο σωστό χρόνο. Σχετικά με την αποθήκευση των απαραίτητων οντοτήτων του εργαλείου όπως είναι οι ροές διεργασιών δίνεται η επιλογή μιας εκ των τριών βάσεων δεδομένων SQLite, που είναι η προεπιλεγμένη, η MySQL και η PostgresSQL. Και σε αυτήν την περίπτωση εγκαταστάθηκε άλλο ένα περιβάλλον postgres.

Η κύρια οντότητα της πλατφόρμας είναι οι κατευθυνόμενοι ακυκλικοί γράφοι (Directed Acyclic Graphs - DAG). Σε κάθε DAG, οι κόμβοι του γράφου αντιστοιχούν στις διαδικασίες που ορίζονται σε ένα pipeline (tasks), και οι ακμές στις μεταξύ τους εξαρτήσεις εκτέλεσης. Λόγω της φύσης αυτής της κατηγορίας γράφων ορίζοντας κανείς ένα DAG μπορεί να ορίσει σύνθετες ακολουθιακές διαδικασίες με πολλές διακλαδώσεις και εξαρτήσεις μεταξύ των διαδικασιών. Κατά τον ορισμό του εκάστοτε task ρυθμίζονται οι απαραίτητες παράμετροι που απαιτούνται για την εκτέλεσή του και επιλέγεται ο τελεστής operator μεταξύ ενός εύρους προκαθορισμένων επιλογών (python, bash, docker, kubernetes και άλλα). Επιλέγοντας έναν τελεστή στην ουσία επιλέγεται ο πυρήνας που απαιτείται για την εκτέλεση του task. Πέρα από τους προκαθορισμένους operators, μέσω του Airflow μπορεί

κανείς να ορίσει εξατομικευμένους operators που εξυπηρετούν εξατομικευμένες λειτουργίες, όπως μπορεί και να ορίσει δυναμικά νέα DAGs επιλέγοντας ήδη ορισμένους γράφους ως υπογράφους.

Το Airflow διαθέτει επίσης, διαφορετικά είδη εκτελεστών - executors για την εκτέλεση των καθορισμένων γράφων. Οι executors ενημερώνονται από τον scheduler του Airflow για την εκτέλεση μιας εργασίας. Οι πιο αποδοτικοί ως προς την κλιμακωσιμότητα των εκτελέσεων είναι οι Celery και Kubernetes (Pierre, 2020). Στην πρώτη περίπτωση, προϋπόθεση είναι να υπάρχει εκ των προτέρων ένας εργάτης, ενώ η επίτευξη της κλιμάκωσης των εργατών είναι περισσότερο δύσκολη σε σχέση με τον Kubernetes Executor. Στη δεύτερη περίπτωση το περιβάλλον και οι παράμετροι του εκτελεστή ορίζονται στις παραμέτρους της διεργασίας που εκτελείται και δεν απαιτείται η ύπαρξη εκ των προτέρων κάποιου εργάτη. Υπάρχει επίσης ο LocalExecutor που μπορεί να εκτελέσει παράλληλα διεργασίες χρησιμοποιώντας τους πόρους του μηχανήματος, ο οποίος και επιλέχθηκε για την εκτέλεση των πειραμάτων εφόσον δεν υπάρχουν αυξημένες ανάγκες για παραλληλοποίηση στα πειράματα που ακολουθούν στο επόμενο κεφάλαιο.

3.1.3 Airflow - MLflow και MNIST dataset (toy problem)

Προκειμένου να αξιολογηθεί εάν είναι εφικτό να συνδυαστούν τα δύο αυτά εργαλεία και να αυτοματοποιήσουν από άκρη σε άκρη (end-to-end) τις διαδικασίες της μηχανικής μάθησης και κυρίως την ανάγκη διαρκούς παρακολούθησης και ενημέρωσης ενός μοντέλου ταξινόμησης σε πραγματικό χρόνο, εφαρμόστηκε ο συνδυασμός τους στο σύνολο δεδομένων MNIST. Το σύνολο των δεδομένων διασπάστηκε σε δύο μέρη, το σύνολο δεδομένων εκπαίδευσης και το σύνολο δεδομένων ελέγχου. Το μοντέλο που αναπτύχθηκε και κατασκευάστηκε μέσω της βιβλιοθήκης keras είναι ένα βαθύ νευρωνικό δίκτυο με δύο δισδιάστατα συνελικτικά στρώματα τα οποία ακολουθούν ένα επίπεδο MaxPooling και δύο πλήρως συνδεδεμένα επίπεδα. Δε δόθηκε έμφαση στην εύρεση του βέλτιστου μοντέλου, δηλαδή στην αναζήτηση της βέλτιστης αρχιτεκτονικής του νευρωνικού δικτύου (στρώματα, νευρώνες, συναρτήσεις ενεργοποίησης) σε συνδυασμό με τη βέλτιστη ρύθμιση των υπερ-παραμέτρων του, αλλά στη συνεχή βελτίωση του δεδομένου μοντέλου εμπλουτίζοντας τη γνώση του με νέα δεδομένα σε βάθος χρόνου. Για αυτό το λόγο δημιουργήθηκαν τα τρία DAGs που ακολουθούν:

- initial DAG: Σε αυτόν το γράφο ορίζεται το pipeline σύμφωνα με το οποίο στο πρώτο στάδιο φορτώνονται και χωρίζονται τα δεδομένα σε δύο σύνολα δεδομένων, εκπαίδευσης και ελέγχου, ενώ στο δεύτερο κατασκευάζεται, εκπαιδεύεται και αποθηκεύεται το αρχικό μοντέλο. Το συγκεκριμένο DAG εκτελείται μία μόνο φορά.
- stream DAG: Σύμφωνα με αυτό το pipeline, το οποίο εκτελείται επαναληπτικά με χρονοπρογραμματισμό, ένα μικρό ποσοστό από το αρχικό σύνολο δεδομένων, χρησιμοποιείται για ζωντανή μετάδοση (live streaming) ως επιπλέον πληροφορία. Για το λόγο αυτό εγκαταστάθηκε το Kafka εργαλείο, δηλαδή ένα σύστημα ανταλλαγής μηνυμάτων στο οποίο κάθε φορά που εκτελείται ο συγκεκριμένος κώδικας,

αποστέλλονται μερικά από τα δεδομένα αυτά. Τα δεδομένα αυτά με τη μορφή μηνυμάτων εισέρχονται σε ένα προκαθορισμένο κανάλι. Για τη διαχείριση των καναλιών (ουρές και θέματα - queues και topics) του Kafka και τον συντονισμό της τοπολογίας του εγκαταστάθηκε το zookeeper (συγκεντρωτική υπηρεσία διατήρησης ρυθμίσεων, ονοματολογίας και πληροφοριών που συμβάλλει στον κατανεμημένο συγχρονισμό και την πρόσβαση σε ομαδικές υπηρεσίες).

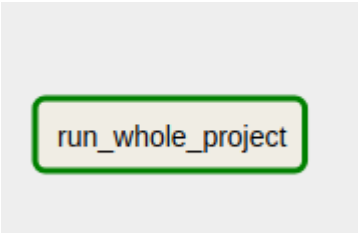
- update DAG: Αυτός ο γράφος, που εκτελείται επίσης επαναληπτικά με μικρότερη συχνότητα από τον προηγούμενο, καταναλώνει τα μηνύματα που εισέρχονται στο κανάλι του Kafka, προκειμένου να τα χρησιμοποιήσει για να επανεκπαιδεύσει το ήδη υπάρχον μοντέλο και να ενισχύσει την απόδοσή του βελτιώνοντας την ακρίβειά του. Υπολογίζει τις αντίστοιχες μετρικές απόδοσης και αν είναι μεγαλύτερες από το υπάρχον μοντέλο τότε ενημερώνει το τρέχον μοντέλο.

Από τα παραπάνω συμπεραίνεται ότι χρησιμοποιώντας το MLflow ως εργαλείο καταγραφής και επισήμανσης των πειραμάτων σε συνδυασμό με το Airflow για την οργάνωση και τον προγραμματισμό γενικότερων ροών εργασιών η αυτοματοποιημένη παραγωγή και ενημέρωση ενός μοντέλου πρόβλεψης είναι εφικτή.

3.1.4 Τρόποι καταγραφής πειραμάτων στο MLflow μέσω Airflow

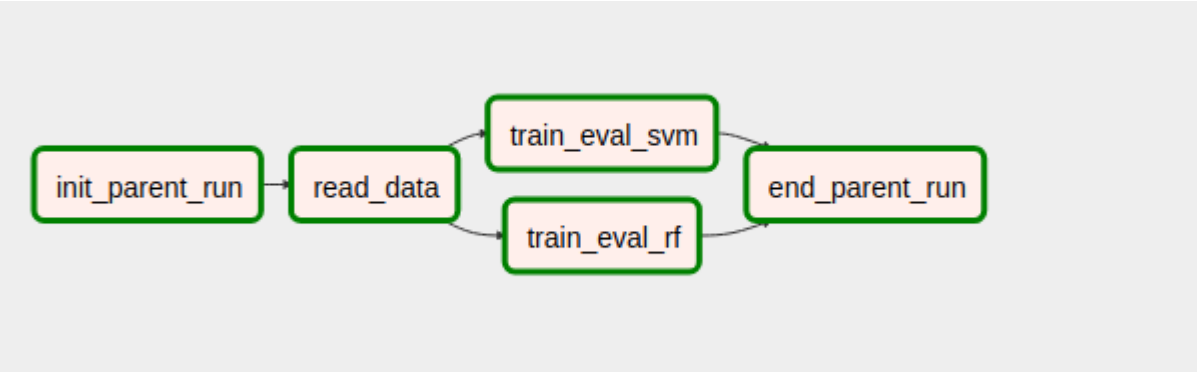
Όπως προαναφέρθηκε, στο Airflow ορίζονται ροές διεργασιών με τη μορφής γράφων που αποτελούνται από tasks, ενώ στο MLflow υπάρχουν τα αντίστοιχα πρότζεκτ με τα αντίστοιχα entry points. Υπάρχουν τρεις τρόποι να συνδυάσει κανείς τα δύο αυτά εργαλεία για την εκτέλεση και την καταγραφή των πειραμάτων στο MLflow μέσω του Airflow και παρουσιάζονται στη συνέχεια.

- Ο πρώτος και λιγότερο ευέλικτος τρόπος σχετίζεται με την ανάπτυξη του κώδικα στον οποίο ορίζεται το πείραμα, εσωτερικά του στις διεργασίες (tasks) του γράφου, χρησιμοποιώντας το MLflow μόνο ως εργαλείο καταγραφής παραμέτρων και μετρικών μέσω κατάλληλων κλήσεων της διεπαφής (API) προς τον MLflow server. Με αυτόν τον τρόπο δεν ορίζεται κάποιο πρότζεκτ με αποτέλεσμα να μην ενεργοποιείται το κατάλληλο περιβάλλον εκτέλεσης καθιστώντας αναγκαία την εγκατάσταση εκ των προτέρων των απαιτούμενων βιβλιοθηκών στο περιβάλλον του Airflow.
- Ο δεύτερος τρόπος περιλαμβάνει τον ορισμό ενός MLproject και την εκτέλεσή του από το Airflow ορίζοντας ένα μόνο task. Το πρότζεκτ πρακτικά ενσωματώνει στο προκαθορισμένο entry point, την ακολουθιακή εκτέλεση των επιμέρους διεργασιών που ενδεχομένως απαιτούνται για την ανάπτυξη του μοντέλου πρόβλεψης. Με αυτόν τον τρόπο επιτυγχάνεται η απομόνωση του περιβάλλοντος του Airflow από το περιβάλλον εκτέλεσης του πειράματος λύνοντας τις συγκρούσεις εξαρτήσεων (dependency conflicts) που μπορεί να προκύψουν. Ωστόσο, πολύπλοκες ροές διεργασιών με διακλαδώσεις είναι δύσκολο να οριστούν προγραμματιστικά.

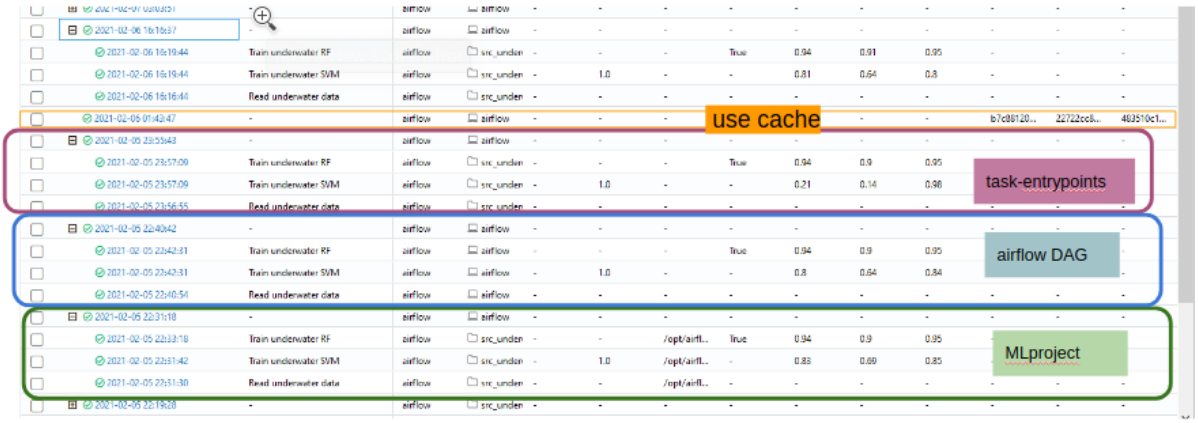


Εικόνα 3.2: Απεικόνιση του διαγράμματος DAG που ορίζει την εκτέλεση ενός MLproject.

- Ο τρίτος τρόπος και περισσότερο ευέλικτος είναι ο ορισμός ενός κατευθυνόμενου γράφου που να αποτελείται από επιμέρους διαδικασίες. Κάθε task του DAG μπορεί να αναλαμβάνει την εκτέλεση ενός από τα στάδια ανάπτυξης ενός μοντέλου πρόβλεψης. Στην πράξη κάθε ένα task θα πυροδοτεί την εκτέλεση ενός entry point από το καθορισμένο πρότζεκτ που σχετίζεται με το συγκεκριμένο πρόβλημα ταξινόμησης. Με αυτόν τον τρόπο, εκτός του απομονωμένου περιβάλλοντος, μπορεί κανείς να επωφεληθεί τη δυνατότητα σύνθεσης πολύπλοκων ροών διεργασιών και τον ορισμό των μεταξύ τους εξαρτήσεων μέσω του Airflow. Επίσης, έχοντας υλοποιήσει διαφορετικές μεθόδους για κάθε στάδιο ενός αυτοματοποιημένου pipeline μηχανικής μάθησης, είναι εύκολη η δημιουργία διαφορετικών συνδυασμών υλοποιήσεων ορίζοντας διαφορετικούς γράφους. Επιπλέον, για κάθε task ορίζονται προκαθορισμένες τιμές των παραμέτρων που απαιτούνται για την εκτέλεσή του. Σε περίπτωση που η εκτέλεση ενός γράφου πυροδοτηθεί κατά απαίτηση χειροκίνητα (μέσω της εφαρμογής ή με API κλήση), οι προκαθορισμένες τιμές των παραμέτρων μπορούν να μεταβληθούν ορίζοντας νέες με ισχύ την τρέχουσα εκτέλεση. Συνεπώς, έχοντας ορίσει ένα γενικευμένο pipeline παραγωγής μοντέλου ταξινόμησης, όπως αυτό της εικόνας 3.3, όπου το στάδιο ανάγνωσης των δεδομένων μπορεί να περιλαμβάνει και την προεπεξεργασία των δεδομένων, και θέτοντας ως παράμετρο το όνομα της υλοποίησης (συνάρτησης προεπεξεργασίας) κατά την εκτέλεση του γράφου, μπορεί κανείς να πειραματιστεί με διαφορετικές εκδοχές του πειράματος καθώς η μέθοδος επεξεργασίας διαφέρει.



Εικόνα 3.3: Απεικόνιση του διαγράμματος DAG που ορίζει την εκτέλεση και την καταγραφή ενός pipeline μέσω επιμέρους σημείων εισόδου (entrypoints) ενός MLproject.



Εικόνα 3.4: Απεικόνιση διαφορετικών εκτελέσεων ενός πειράματος με τους τρεις διαφορετικούς τρόπους: i) ανάπτυξη κώδικα στην πλατφόρμα του Airflow, ii) ανάπτυξη και εκτέλεση κώδικα σε δομή MLproject και iii) ανάπτυξη και εκτέλεση κώδικα σε δομή MLproject συνδυάζοντας διάφορα entrypoints.

3.1.5 Άλλα εργαλεία

Κατά την επιλογή του συνδυασμού των δύο εργαλείων τέθηκε το ερώτημα αν αυτά αρκούν για να καλύψουν όλες τις ανάγκες που προκύπτουν σε ένα ζήτημα μηχανικής μάθησης. Μερικές από τις ανάγκες σχετίζονται με την ιστορικότητα των δεδομένων, την κατανεμημένη εκτέλεση μερικών διαδικασιών, την κλιμακωσιμότητα και τη διάθεση των μοντέλων πρόβλεψης προς άλλες υπηρεσίες. Το να βρεθεί ένα εργαλείο που να μπορεί να καλύψει όλες τις ανάγκες της μηχανικής μάθησης είναι δύσκολο, καθώς κάθε εργαλείο εξειδικεύεται σε μια λειτουργία προκειμένου να επιτύχει υψηλή απόδοση για παραγωγικές υπηρεσίες. Ωστόσο, το προτέρημα και των δύο εργαλείων είναι ότι υπάρχει συνεχής υποστήριξη και ενημέρωση του λογισμικού από τις ομάδες τους δίνοντας τη δυνατότητα ενσωμάτωσης με άλλα κατάλληλα εργαλεία προσθέτοντας ολοένα και περισσότερες λειτουργίες. Στη συνέχεια, προτείνονται, ανά κατηγορία, ενδεικτικά εργαλεία που μπορούν να χρησιμοποιηθούν για την κάλυψη των παραπάνω αναγκών.

Ιστορικότητα εκδόσεων και δεδομένων

Η ιστορικότητα των εκδόσεων, πέρα από τις δυνατότητες που παρέχει το MLflow ως προς τα αντικείμενα που αποθηκεύει, επιτυγχάνεται με την εισαγωγή εργασιών που εκτελούν εντολές του DVC εργαλείου, ως επιμέρους tasks στα pipelines του Airflow.

Κατανεμημένες διαδικασίες και κλιμακωσιμότητα

Επίσης, η παραλληλοποίηση μπορεί να πραγματοποιηθεί μέσω της βιβλιοθήκης Dask και από τα δύο συστήματα ή αν οι απαιτήσεις είναι αυξημένες με την ενσωμάτωση του Apache Spark, με το οποίο είναι συμβατά. Ενδεικτικά, η κλιμακωσιμότητα μπορεί να επιτευχθεί με τη βοήθεια της πλατφόρμας Kubernetes.

Διαθεσιμότητα μοντέλων (model serving)

Τέλος, το MLflow έχει αναπτύξει κατάλληλη διεπαφή (API) σε Python, R, Java και REST, με στόχο την πρόσβαση στις απαραίτητες πληροφορίες που καταγράφει, και την εύκολη ενσωμάτωσή του από εξωτερικές υπηρεσίες. Κατά την αποθήκευση των μοντέλων που αναπτύσσονται πακετάρει ταυτόχρονα τα απαραίτητα στοιχεία του, όπως είναι το περιβάλλον που απαιτεί για να εγκατασταθεί. Με αυτόν τον τρόπο το μοντέλο μπορεί να φορτωθεί και να εγκατασταθεί σε εξωτερικά συστήματα. Συνεπώς, η διαθεσιμότητα των μοντέλων πραγματοποιείται είτε με μια εξατομικευμένη υπηρεσία, για παράδειγμα μια νέα εφαρμογή με REST αρχιτεκτονική, ή έτοιμα εργαλεία serving όπως είναι το Seldon-core που υποστηρίζει την εγκατάσταση συστημάτων MLflow Server σε Kubernetes υποδομή. Στις εικόνες 3.5 και 3.6 φαίνονται αντίστοιχα οι κλήσεις που γίνονται στα δύο αυτά συστήματα προκειμένου να πάρει κανείς την πρόβλεψη για μια είσοδο σύμφωνα με το βέλτιστο μοντέλο που επιλέχθηκε προς εγκατάσταση.

```
krysa@alkaios-rslab:~$ curl http://10.1.179.126:5004/predict/Underwater_model_production?image_name=b
edford_2016.tif > result.txt
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 5029k  100 5029k    0     0  2849k      0  0:00:01  0:00:01 --:--:-- 2850k
```

Εικόνα 3.5: Παράδειγμα HTTP κλήσης προς την υπηρεσία REST για τη λήψη του αποτελέσματος ενός προβλήματος ταξινόμησης.

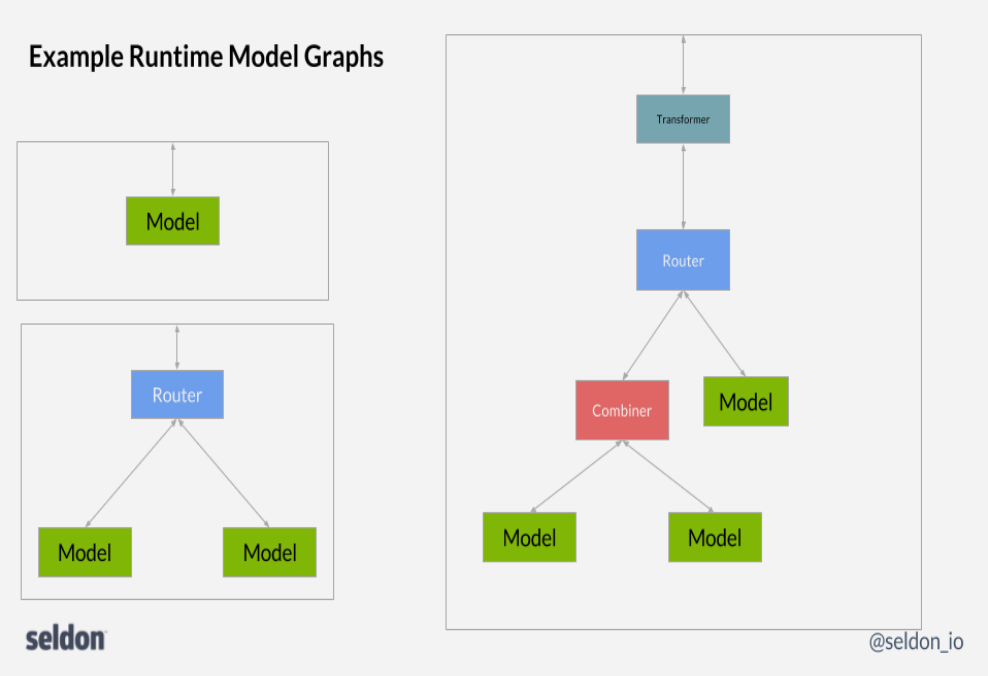
```
(thesis) chili[redacted]:~$ curl -X POST -H 'Content-Type: application/json'  -d '{"data": {"nd
array": [[[23,99,123],[2,199,23],[123,129,223],[23,99,123]], [[23,99,123],[2,199,23],[123,129,223],[2
3,99,123]],[[23,99,123],[2,199,23],[209,208,188],[1,0,1]]]}}'  http://[redacted]/prd/seldon/seldo
n/mlflow-deployment-aa/api/v1.0/predictions
{"data":{"names":[],"ndarray":[0,0,0,0,0,0,0,0,0,0,0]},"meta":{"requestPath":{"wines-classifier-aa"
:"seldonio/mlflowserver:1.6.0"}}}
```

Εικόνα 3.6: Παράδειγμα HTTP κλήσης προς την εγκατάσταση ενός μοντέλου πρόβλεψης μέσω του εργαλείου Seldon-core για τη λήψη του αποτελέσματος ενός προβλήματος ταξινόμησης.

Το Seldon-core λογισμικό παρέχει δυνατότητες εγκατάστασης μοντέλων σε συστήματα με τη μορφή γράφων. Όπως φαίνεται στην εικόνα 3.7, υπάρχουν τέσσερα είδη κόμβων που υποστηρίζει και μπορούν να συνδυαστούν σε ένα γράφο, τα οποία είναι τα εξής:

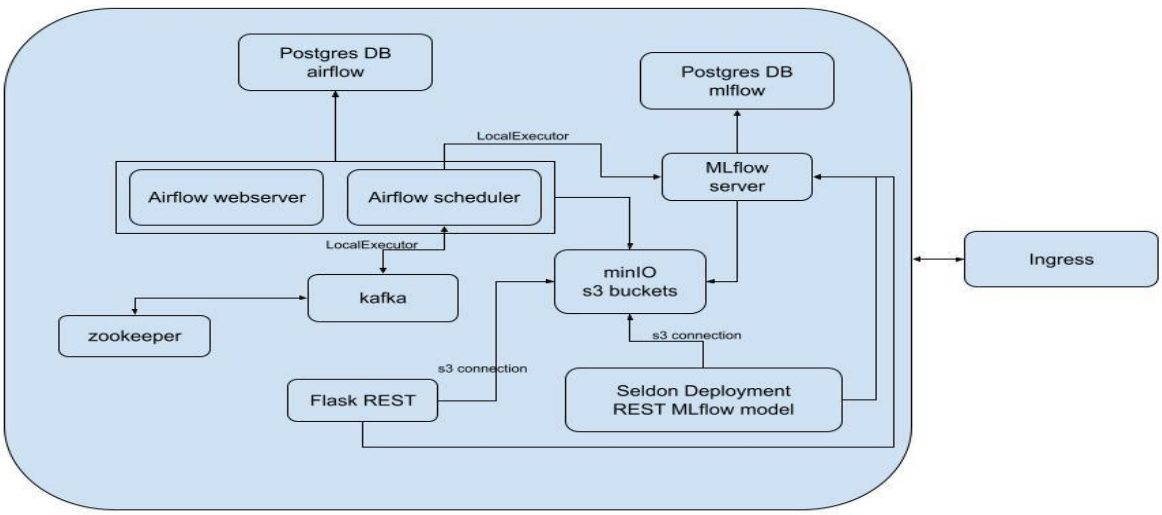
- Μοντέλο (model): Πρόκειται για τον κόμβο που περιλαμβάνει την εγκατάσταση του μοντέλου πρόβλεψης. Συνήθως είναι μια μονολιθική υπηρεσία (microservice) με το μοντέλο συσκευασμένο σε μια εικόνα docker. Λόγω της ολοκλήρωσης της ενσωματωμένης λειτουργίας του MLflow Server αυτό που χρειάζεται είναι μόνο ο φάκελος που περιλαμβάνει το μοντέλο.
- Δρομολογητές (router): Αυτός ο κόμβος αναλαμβάνει τη διαχείριση των κλήσεων αιτημάτων που δέχεται η εγκατάσταση Seldon. Στην πράξη είναι πιθανό να υπάρχουν δύο ή περισσότερα εγκατεστημένα μοντέλα και μια στρατηγική με την οποία ένα ποσοστό των κλήσεων δρομολογείται σε διαφορετικό μοντέλο πρόβλεψης.
- Συνδυαστές (combiner): Πολλές φορές για τη βελτίωση της ακρίβειας της πρόβλεψης συνδυάζονται οι προβλέψεις που προκύπτουν από δύο ή περισσότερα μοντέλα, σύμφωνα με μεθόδους συνόλου (ensemble methods), όπως είναι η μέθοδος Bagging και ο Voting ταξινομητής.

- Μετασχηματιστές (transformer): Αυτός ο κόμβος είναι υπεύθυνος για το μετασχηματισμό, τόσο της εισόδου (κανονικοποίηση χαρακτηριστικών, εντοπισμός ακραίων τιμών) όσο και της εξόδου, στην επιθυμητή μορφή για το μοντέλο και το χρήστη αντίστοιχα.



Εικόνα 3.7: Παραδείγματα διαφορετικών γράφων συμπερασμού συνδυάζοντας τα διαφορετικά δομικά στοιχεία που παρέχει το εργαλείο Seldon-core.

3.1.6 Υποδομή σε σύστημα Kubernetes



Εικόνα 3.8: Διάγραμμα απεικόνισης της υποδομής και της διασύνδεσης των μονολιθικών υπηρεσιών που αναπτύχθηκαν για την παραγωγή λύσεων στη μηχανική μάθηση από άκρο σε άκρο.

Στην εικόνα 3.8 παρουσιάζεται η υποδομή που σχεδιάστηκε επιλέγοντας τα εργαλεία που κρίθηκαν κατάλληλα προκειμένου να καλυφθούν οι ανάγκες της μηχανικής μάθησης. Η

εγκατάσταση κάθε υπηρεσίας πραγματοποιήθηκε σε ένα συγκρότημα υπηρεσιών (cluster) με τη βοήθεια της πλατφόρμας Kubernetes[4]. Η υποδομή βασίζεται στην τεχνική μονολιθικών υπηρεσιών (microservices) απομονώνοντας κάθε εργαλείο για γρήγορη και εύκολη εγκατάσταση νέων λειτουργιών. Για την εγκατάσταση κάθε υπηρεσίας χρησιμοποιήθηκε το Helm, ένα εργαλείο διαχείρισης πακέτων και εφαρμογών σε Kubernetes. Για λόγους εξατομικευμένης χρήσης των εργαλείων που επιλέχθηκαν κατασκευάστηκαν τρία διαφορετικά docker images, ένα για το Airflow περιλαμβάνοντας το λογισμικό του Airflow μαζί με το πακέτο miniconda για την αποτελεσματική εκτέλεση πειραμάτων MLflow σε κατάλληλο περιβάλλον, ένα για το MLflow προσθέτοντας ορισμένες βιβλιοθήκες και ένα για την εφαρμογή REST που αναπτύχθηκε για τη διαθεσιμότητα του μοντέλου παραγωγής με χρήση του Flask framework σε Python.

Όπως είναι εμφανές, η υποδομή περιλαμβάνει την εγκατάσταση των δύο τεχνολογιών Airflow και MLflow μαζί με μια εξωτερική βάση δεδομένων PostgresSQL για κάθε σύστημα. Η εγκατάσταση του Airflow περιλαμβάνει τις δύο συνιστώσες webserver και scheduler. Αποτελείται, επίσης, από τον MLflow server και τον minIO server. Έχουν στηθεί οι δύο εναλλακτικές εφαρμογές για την κατανάλωση του μοντέλου παραγωγής, οι οποίες είναι απαραίτητο να επικοινωνούν τόσο με τον MLflow Server για την εύρεση της τοποθεσίας του φακέλου που περιέχει το μοντέλο που επιλέγεται για εγκατάσταση στην παραγωγή, όσο και με το minIO server για την πρόσβαση στο φάκελο με το μοντέλο. Περιλαμβάνονται και τα δύο συστήματα που υλοποιήθηκαν για την υλοποίηση του προβλήματος αναφοράς με το MNIST dataset, kafka και zookeeper. Όλες αυτές οι εφαρμογές τοποθετούνται “πίσω” από έναν Ingress Controller, ο οποίος είναι διαχειριστής του φορτίου των HTTP κλήσεων σε μια υποδομή, εξειδικευμένος για περιβάλλοντα Kubernetes.

Η εγκατάσταση της υποδομής, και συνεπώς των εργαλείων, έγινε σε ένα τοπικό μηχάνημα με επεξεργαστή 7ης γενιάς και 12 υπολογιστικούς πυρήνες. Η διαθέσιμη μνήμη του μετάλλου είναι 12GB RAM, ενώ ο δίσκος είναι 256 GB. Δημιουργήθηκε ένας κόμβος για την εγκατάσταση των εργαλείων μέσω της πλατφόρμας Kubernetes.

3.2 Οφέλη από τη χρήση των επιλεγμένων εργαλείων

Η διεξαγωγή πειραμάτων που απαιτείται για την επίλυση ενός προβλήματος μηχανικής μάθηση, όπως είναι και αυτά που ακολουθούν στο επόμενο κεφάλαιο, θα μπορούσαν να πραγματοποιηθούν και να εκτελεστούν χειροκίνητα από κάποιο άτομο εξειδικευμένο χωρίς τη χρήση της υποδομής που αναπτύχθηκε. Κάτι τέτοιο όμως θα δυσκόλευε σε μεγάλο βαθμό την οργάνωση των πειραμάτων, τη διατήρηση των

[4] Δε σχετίζεται με τον τελεστή Kubernetes Executor του Airflow. Ο LocalExecutor που έχει επιλεχθεί χρησιμοποιεί τους πόρους του Airflow container για την εκτέλεση των tasks, σε αντίθεση με τον KubernetesExecutor που εν γένει απαιτεί δυναμικά τη δημιουργία νέων PODs (υπολογιστικές μονάδες - ομάδα από containers).

αποτελεσμάτων και των συνθηκών εκπαίδευσης, την ιστορικότητα των εκδόσεων του κώδικα αλλά και των αντικειμένων που σχετίζονται με στάδια της μηχανική μάθησης και πολλά ακόμα, καθιστώντας τελικά την ολοκλήρωσή τους πολύ χρονοβόρα. Τα οφέλη από τη χρήση των επιλεγμένων εργαλείων είναι πολλά και παρουσιάζονται στη συνέχεια.

Οφέλη ως προς την ανάπτυξη κώδικα

- Απομόνωση περιβάλλοντος κώδικα: Στη μηχανική μάθηση είναι πολύ συχνό φαινόμενο να εξετάζονται νέες τεχνικές επεξεργασίας των δεδομένων και εκπαίδευσης μοντέλων, καθιστώντας αναγκαία την εγκατάσταση νέων βιβλιοθηκών και πακέτων στη γλώσσα προγραμματισμού που συντάσσεται ο αντίστοιχος κώδικας. Διαφορετικές βιβλιοθήκες και εκδόσεις αυτών μπορεί να οδηγήσουν σε συγκρούσεις και αδυναμίες εγκατάστασης. Ο τρόπος που εκτελούνται τα πειράματα μέσω του MLflow, με τον ορισμό ενός νέου και σαφώς ορισμένου περιβάλλοντος εγκατάστασης των απαραίτητων για την εκτέλεση των πειραμάτων πακέτων, διατηρώντας αμετάβλητο το τρέχον περιβάλλον, λύνει τα προβλήματα αυτά και εξοικονομεί χρόνο ανάπτυξης από τους προγραμματιστές. Δίνει, επίσης, τη δυνατότητα στα εξειδικευμένα άτομα να αναπαράξουν εύκολα και γρήγορα τα πειράματα χωρίς χειροκίνητη εγκατάσταση και ρύθμιση του περιβάλλοντος εκτέλεσης.
- Εύκολη ενοποίηση με τις MLflow λειτουργίες: Η ύπαρξη ενός API του MLflow εργαλείου βοηθάει στην ένταξη των λειτουργιών του σε άλλα συστήματα. Με αυτόν τον τρόπο οι πληροφορίες των πειραμάτων, οι παράμετροι, τα αποτελέσματα, τα αρχεία και τα μοντέλα είναι εφικτό να καταναλωθούν με αυτοματοποιημένο τρόπο από οποιοδήποτε σύστημα υποστηρίζει ένα από τους διαθέσιμους τρόπους ενοποίησης.
- Συσχέτιση με τον κώδικα: Το MLflow παρέχει τη δυνατότητα συσχέτισης των πειραμάτων με το αναγνωριστικό της έκδοσης του κώδικα, το οποίο μπορεί να είναι το hash[5] που αντιστοιχεί στο commit της έκδοσης του τρέχοντος κώδικα. Αντίστοιχα το Airflow παρέχει τη δυνατότητα οπτικοποίησης του κώδικα με τον οποίο ορίζεται το εκάστοτε DAG.

Οφέλη ως προς την εκτέλεση πειραμάτων

- Αναπαραγωγή πειραμάτων: Η καταγραφή στο MLflow εργαλείο των παραμέτρων, του κώδικα και όλων των συνιστωσών που επηρεάζουν ένα πρόβλημα μηχανικής μάθησης συμβάλλει στην εύκολη αναπαραγωγή ενός ήδη εκτελεσμένου πειράματος. Ανάλογη είναι και η συμβολή του Airflow στο οποίο επίσης γίνεται καταγραφή των παραμέτρων που λαμβάνει ένας γράφος τη στιγμή που πυροδοτείται για να εκτελέσει

[5] Πρόκειται για ένα μοναδικό κλειδί συγκεκριμένης κωδικοποίησης που προσδιορίζει κάθε διαφορετική καταχώρηση κώδικα.

την αλληλουχία των εργασιών που ορίζονται σε αυτόν. Παρέχει επίσης, μέσω της διεπαφής χρήστη την επιλογή να πυροδοτήσει την επανεκτέλεση μιας διεργασίας ενός DAG.

- Προγραμματισμός εκτέλεσης ροών εργασιών: Αρκετές φορές υπάρχει η ανάγκη επανεκτέλεσης συγκεκριμένων διαδικασιών σε τακτά χρονικά διαστήματα. Το Airflow είναι μια ολοκληρωμένη πλατφόρμα στην οποία μπορεί να ορίσει κανείς χρονικά μοτίβα επανάληψης διαδικασιών, παρακολουθώντας την επιτυχή ή ανεπιτυχή εκτέλεσή τους και καταγράφοντας σχετικές πληροφορίες. Δίνει τη δυνατότητα ενημέρωσης μέσω ειδοποιήσεων για το αποτέλεσμα της εκτέλεσης.
- Αποφυγή πανομοιότυπης εκτέλεσης πειραμάτων: Λόγω της ύπαρξης διεπαφής API, για τα ολοκληρωμένα πειράματα στο εργαλείο του MLflow, είναι εύκολο να ελεγχθεί αν ένα πείραμα έχει ήδη ολοκληρωθεί με τις ίδιες συνθήκες και παραμέτρους. Με αυτόν τον τρόπο δίνεται η δυνατότητα να αναπτυχθεί μια λογική με βάση την οποία ένα ίδιο πείραμα δεν επαναλαμβάνεται αλλά χρησιμοποιείται το καταγεγραμμένο. Με αυτόν τον τρόπο, αν εκτελούνται pipelines των οποίων κάποιο χρονοβόρο στάδιο έχει επαναληφθεί στο παρελθόν, αποφεύγεται η επανεκτέλεσή του και η συνολική εκτέλεση απαιτεί λιγότερο χρόνο. Επίσης, εξοικονομείται μνήμη ως προς τα αντικείμενα που αποθηκεύονται και σχετίζονται με το συγκεκριμένο πείραμα, εξαλείφοντας διπλότυπα αρχεία.

Οφέλη ως προς την ανθρώπινη χρηστικότητα

- Διεπαφή χρήστη - Οπτικοποίηση: Με την εγκατάσταση και των δύο εργαλείων, MLflow και Airflow, παρέχεται η δυνατότητα στο χρήστη να διαχειρίζεται τα πειράματα που εκτελεί και να μπορεί με οπτικά ερεθίσματα να κατανοήσει λεπτομέρειες και στοιχεία σχετικά με αυτά. Η κατανόηση των πολύπλοκων σωληνώσεων που δημιουργούνται είναι πιο εύκολη από τον ανθρώπινο οργανισμό όταν η χρονική αλληλουχία και οι μεταξύ τους εξαρτήσεις οπτικοποιούνται. Η περιήγηση σε διαφορετικά πρότζεκτ, εκτελέσεις, πειράματα και οντότητες είναι περισσότερο εύκολη μέσω μιας διεπαφής. Συνήθεις διαδικασίες, όπως η πρόσβαση σε αρχεία καταγραφής των εκτελέσεων, σε διαφορετικά στιγμιότυπα των εκτελεσμένων σωληνώσεων διεκπεραιώνονται γρήγορα και εύκολα. Επίσης, υπάρχει δυνατότητα σύγκρισης δύο διαφορετικών εκτελέσεων των πειραμάτων καθιστώντας περισσότερο κατανοητές τις διαφορές τους, προτέρημα καθοριστικό ειδικά στη φάση ανάπτυξης νέων τεχνικών (εικόνα 3.9).
- Οργάνωση και ομαδοποίηση ροών εργασιών: Στο MLflow λογισμικό παρέχεται η δυνατότητα δημιουργίας πειραμάτων κάτω από οποία εκτελούνται συγκεκριμένα πρότζεκτ. Για κάθε ένα από αυτά μπορεί να οριστεί διαφορετική τοποθεσία αποθήκευσης των αντικειμένων που σχετίζονται με τα πειράματα. Με αυτόν τον τρόπο επιτυγχάνεται αυτόματα η οργάνωση των πρότζεκτ σε διαφορετικούς φακέλους και ενότητες, ενώ ταυτόχρονα η πρόσβαση στα πρότζεκτ μιας ομάδας είναι εφικτή από το ίδιο σημείο διεπαφής. Επιπλέον, μέσα από το συγκεκριμένο εργαλείο,

είναι εφικτή η εκτέλεση και η δημιουργία εμφωλευμένων εκτελέσεων των διαφορετικών σταδίων που απαρτίζουν ένα pipeline, επιτυγχάνοντας ομαδοποίηση των εκτελέσεων ανά διαφορετικό στιγμιότυπο ενός DAG. Εάν η οργάνωση των πειραμάτων γινόταν χειροκίνητα εκτός από το γεγονός ότι η διαδικασία θα ήταν χρονοβόρα, θα απαιτούσε προσεκτικές κινήσεις ώστε να αποφευχθούν τυχόν λάθη.

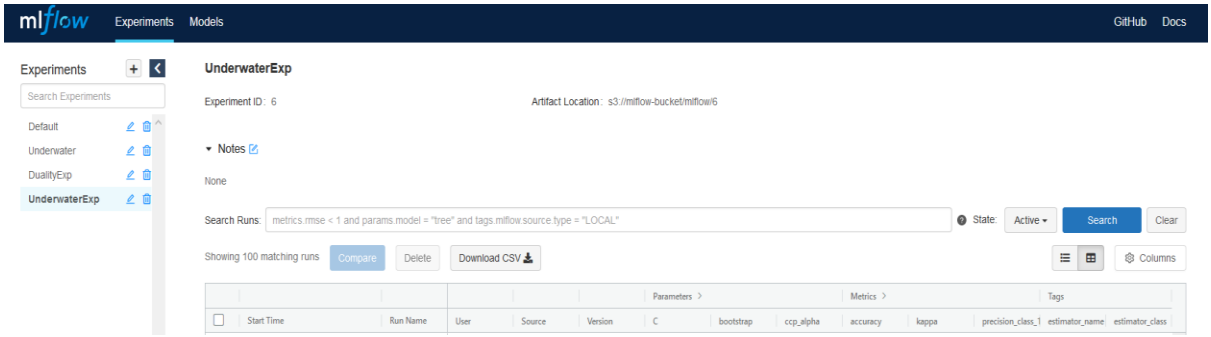
Οφέλη ως συνεργασία των ομάδων σχεδιασμού και ανάλυσης

Στο συγκεκριμένο πεδίο επιστήμης, αυτό της μηχανικής μάθησης, η συνεργασία μεταξύ ομάδων διαφορετικών ειδικοτήτων είναι απαραίτητη. Με την εγκατάσταση μιας κοινής υποδομής με μεγάλη χρηστικότητα το χάσμα μεταξύ των γνωστικών πεδίων των διαφορετικών ομάδων γεφυρώνεται και μπορούν να συμβάλλουν από κοινού στην επίτευξη ενός καλύτερου αποτελέσματος. Μάλιστα, η συνεργασία είναι σημαντική ακόμα και στην ίδια ομάδα καθώς η ανάγκη ύπαρξης ενός περιβάλλοντος μέσω του οποίου είναι εφικτή η ολιστική παρακολούθηση των τεχνικών που αναπτύσσονται για την επίλυση ενός προβλήματος, εξοικονομεί χρόνο στο ανθρώπινο δυναμικό μειώνοντας την επικοινωνία που θα απαιτείτο υπό άλλες συνθήκες για ενημέρωση ως προς την ανάπτυξη υλοποιήσεων και μοίρασμα των πειραμάτων.

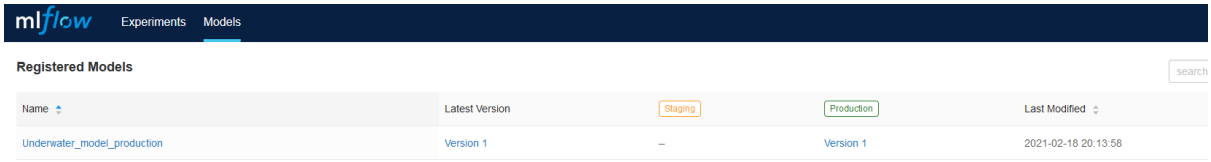
Underwater › Comparing 2 Runs

Run ID:	b8f9fe40449240afb18898a7f4270853	17523cbdc13a40998566bae157a241bd
Run Name:	Train underwater SVM	Train underwater RF
Start Time:	2021-02-07 03:04:09	2021-02-07 03:04:09
Parameters		
C	1.0	
bootstrap		True
ccp_alpha		0.0
class_weight	None	None
criterion		gini
dual	True	
filter	median	median
fit_intercept	True	
gt	/opt/airflow/dags/src/src_underwater/bb16_gt.tif	/opt/airflow/dags/src/src_underwater/bb16_gt.tif
input_image	/opt/airflow/dags/src/src_underwater/bedford_2016.tif	/opt/airflow/dags/src/src_underwater/bedford_2016.tif
intercept_scaling	1	
loss	squared_hinge	
max_depth		None
max_features		auto
max_iter	1000	
max_leaf_nodes		None
max_samples		None
method	SVM	RF

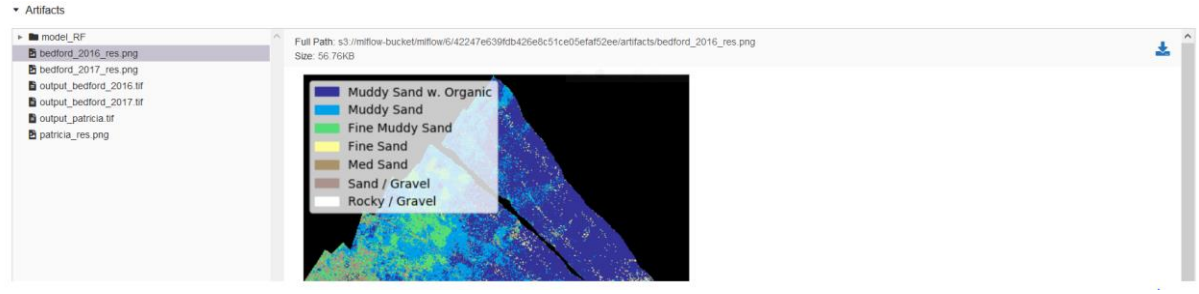
Εικόνα 3.9: Απεικόνιση της όψης κατά τη σύγκριση δύο εκτελέσεων ενός πειράματος μέσω της πλατφόρμας του MLflow.



Εικόνα 3.10: Απεικόνιση της όψης του MLflow server κατά την αρχική σελίδα. Σε αυτήν την όψη εμφανίζεται αριστερά η ομαδοποίηση και τα ενεργά διαφορετικά πειράματα που καταγράφει.



Εικόνα 3.11: Απεικόνιση της όψης που παρέχει το MLflow εργαλείο για την καταγραφή και την αποθήκευση της ιστορικότητας των εκδόσεων και της κατάστασης των μοντέλων που παράγονται για την επίλυση ανά πείραμα που εξετάζεται



Εικόνα 3.12: Απεικόνιση της όψης που περιλαμβάνει τη λίστα με τα αντικείμενα που αποθηκεύονται κατά την εκτέλεση ενός στιγμιότυπου του πειράματος. Είναι δυνατή τόσο η περιήγηση στα αρχεία όσο και η άμεση πρόσβαση σε αυτά με τη δυνατότητα λήψης τους.

Κεφάλαιο 4: Αποτελέσματα

Σε αυτό το κεφάλαιο περιγράφεται το πρόβλημα της μηχανικής μάθησης στο οποίο βρίσκει εφαρμογή η υποδομή που αναπτύχθηκε. Αναλύεται η μεθοδολογία που αναπτύσσεται για την εύρεση της βέλτιστης λύσης του προβλήματος, εξετάζοντας ένα σύνολο από διαφορετικές τεχνικές προεπεξεργασίας των δεδομένων, των ταξινομητών και άλλων τεχνικών μηχανικής μάθησης. Καταγράφονται σημαντικές πληροφορίες σχετικά με τα δεδομένα που χρησιμοποιήθηκαν για την εκπαίδευση των ταξινομητών και περιγράφονται η στρατηγική και οι μετρικές αξιολόγησης των ταξινομητών. Στη συνέχεια, παρουσιάζονται τα αποτελέσματα των επιδόσεων των μοντέλων πρόβλεψης που εξετάστηκαν. Συγκεκριμένα αξιολογείται η απόδοση των δύο διαφορετικών ταξινομητών, Random Forest (RF) και Support Vector Machine (SVM), για κάθε συνδυασμό που προκύπτει από τη μέθοδο της προεπεξεργασίας στα δεδομένα, το φίλτρο μεσαίας τιμής, την κανονικοποίηση ελαχίστου-μεγίστου και την κανονικοποίηση z-score, και το σύνολο των κατηγοριών ταξινόμησης, τρεις και εφτά κλάσεις αντίστοιχα. Επιπροσθέτως, για κάθε συνδυασμό χρησιμοποιούνται διαφορετικά σύνολα δεδομένων εκπαίδευσης, τα οποία αποτελούν υποσύνολα των αρχικών διαθέσιμων δεδομένων, τονίζοντας με αυτόν τον τρόπο τη σημαντικότητα συνεχούς ενημέρωσης ενός μοντέλου με νέα δεδομένα. Τέλος, μελετάται, μέσω της προσομοίωσης της συνεχούς ενημέρωσης ενός μοντέλου, η εξέλιξή του στο χρόνο. Οι πίνακες παρουσίασης των αποτελεσμάτων δημιουργήθηκαν με κώδικα ο οποίος συλλέγει τα δεδομένα από τα πειράματα που ολοκληρώθηκαν και τα αποθηκεύει σε αρχεία csv.

4.1 Πρόβλημα εφαρμογής και Δεδομένα

4.1.1 Περιγραφή του προβλήματος

Στην ενότητα αυτή παρουσιάζεται το πρόβλημα ταξινόμησης πάνω στο οποίο εφαρμόστηκε η υποδομή που σχεδιάστηκε με τα εργαλεία που κρίθηκαν κατάλληλα. Το πρόβλημα σχετίζεται με την ταξινόμηση του θαλάσσιου βυθού με δεδομένα που προέρχονται από ηχοβολιστικά μηχανήματα πολλαπλών δεσμών και πολλαπλών συχνοτήτων. Χρησιμοποιώντας νέες προηγμένες τεχνικές μηχανικής μάθησης, στόχος είναι η κατασκευή ενός περιβάλλοντος παραγωγής του βέλτιστου μοντέλου κατάλληλου για τη χαρτογράφηση πολλών κατηγοριών του θαλάσσιου πυθμένα με ισχύ σε εφαρμογές αρχαιολογίας, ενέργειας, γεωλογικής επικινδυνότητας και άλλες. Στην πράξη, στόχος είναι η κατασκευή μιας υπηρεσίας που λαμβάνει ως είσοδο μια εικόνα που απεικονίζει τον πυθμένα της γης και παράγει ως αποτέλεσμα μια εικόνα ίδιων διαστάσεων με τιμή, σε κάθε θέση-στοιχείο της (pixel), την κλάση στην οποία ταξινομείται το αντίστοιχο pixel.

Για την εκπαίδευση του μοντέλου, χρησιμοποιήθηκαν δεδομένα που προήλθαν από εικόνες υπερήχου σε συνδυασμό με τις αντίστοιχες εικόνες μελέτης πεδίου (ground-truth

images). Δηλαδή, σε κάθε εικόνα υπερήχου αντιστοιχεί μια εικόνα που ως πληροφορία σε κάθε pixel περιέχει την πραγματική κλάση στην οποία ανήκει το κάθε pixel.

4.1.2 Μεθοδολογία

Για την εκπαίδευση του μοντέλου χρησιμοποιήθηκαν συνδυαστικά τα δεδομένα από τις δύο εικόνες. Για κάθε θέση στοιχείο της εικόνας, ως χαρακτηριστικά θεωρήθηκαν οι τρεις τιμές (στην κλίμακα του γκρι) που προέρχονται από τα τρία κανάλια RGB της εικόνας (X_R, X_G και X_B) και ως έξοδος η τιμή στην αντίστοιχη θέση στην ground-truth εικόνα. Θέσεις στοιχεία στα οποία υπάρχει μηδενική τιμή σε ένα από τα τρία κανάλια αφαιρούνται από το σύνολο δεδομένων, όπως και στοιχεία για τα οποία η τιμή στην ground-truth εικόνα είναι μηδενική (no-data pixels).

Προεπεξεργασία:

Στις εικόνες, προτού χρησιμοποιηθούν για την εκπαίδευση του μοντέλου, εφαρμόστηκε μια διαδικασία προεπεξεργασίας. Εξετάστηκαν τρία διαφορετικά είδη προεπεξεργασίας τα οποία είναι: η εφαρμογή του φίλτρου μεσαίας τιμής (median filter) (Villar et al., 2017) από τη βιβλιοθήκη scikit-image, η κανονικοποίηση ελάχιστου-μέγιστου (min-max) και η κανονικοποίηση z-score (Κύρκος, 2015).

- Φίλτρο μεσαίας τιμής (Church et al., 2008): Το φιλτράρισμα με ένα φίλτρο μεσαίας τιμής είναι μια μη γραμμική τεχνική. Η τιμή median ενός συνόλου A είναι ίση με τη μεσαία τιμή του συνόλου. Το φίλτρο μεσαίας τιμής χρησιμοποιείται για την εξομάλυνση (smoothing) των ακμών και τη μείωση του θορύβου μιας εικόνας. Η σχέση που περιγράφει το μετασχηματισμό είναι η εξής:

$$median(A) = \begin{cases} a_{\frac{n+1}{2}} \lfloor n, περιττος \\ \frac{1}{2}\left(a_{\frac{n}{2}} + a_{\frac{n}{2}+1} \right) \lfloor n, αρτιος \end{cases}$$

- Κανονικοποίηση ελάχιστου-μέγιστου: Με αυτήν την μέθοδο κανονικοποίησης οι αριθμητικές τιμές στοιχίζονται με άλλες, οι οποίες κυμαίνονται εντός μιας προκαθορισμένης περιοχής τιμών (Κύρκος, 2015). Η αντιστοίχιση γίνεται με γραμμικό μετασχηματισμό, που ορίζεται από τη σχέση:

$$x' = \frac{x - min}{max - min}(new_{max} - new_{min}) + new_{min},$$

όπου x η εκάστοτε τιμή της μεταβλητής (χαρακτηριστικού), x' η νέα τιμή, $min \, και \, max$η μικρότερη και η μεγαλύτερη τιμή της μεταβλητής και $new_{min} \, και \, new_{max}$ το νέο εύρος τιμών. Με τη μέθοδο αυτή διατηρείται η αναλογία μεταξύ των τιμών που υπήρχε και στα δεδομένα αρχικά. Το διάστημα που επιλέχθηκε για αυτό το μετασχηματισμό είναι το [0,1].

- Κανονικοποίηση z-score: Η μέθοδος αυτή πραγματοποιεί μετασχηματισμό των αριθμητικών τιμών, χρησιμοποιώντας τη μέση τιμή και την τυπική απόκλισή τους. Για μια μεταβλητή με μέση τιμή μκαι τυπική απόκλιση σ, ο μετασχηματισμός των τιμών γίνεται με την εξής σχέση:

$$x' = \frac{x - \mu}{\sigma},$$

όπου x η εκάστοτε τιμή της μεταβλητής (χαρακτηριστικού) και x' η νέα τιμή. Η μέθοδος αυτή είναι κατάλληλη σε περιπτώσεις που τα δεδομένα περιέχουν ακραίες τιμές. Η μέση τιμή των τιμών που δίνει μετά το μετασχηματισμό ισούται με 0 (Κύρκος, 2015).

Ταξινομητές:

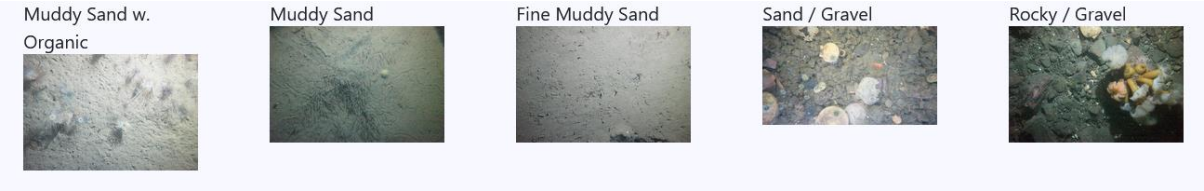
Για την ταξινόμηση των εικόνων εξετάστηκαν δύο είδη ταξινομητών, η μηχανή υποστήριξης διανυσμάτων (SVM) και ο αλγόριθμος RF των τυχαίων δασών (Random Forest).

- SVM: Ο αλγόριθμος SVM είναι μια από τις πιο πρόσφατα αναπτυσσόμενες μεθοδολογίες στον τομέα της μηχανικής μάθησης που χρησιμοποιείται σε προβλήματα ταξινόμησης και παλινδρόμησης και αποτελεί μια εφαρμογή της αρχής Structural Risk Minimization που εισήγαγε ο Vapnik (Vapnik, 2000). Το πλεονέκτημά του είναι η δυνατότητα μοντελοποίησης πολύπλοκων, μη γραμμικών ορίων διαχωρισμού των τάξεων (ή στην περίπτωση της παλινδρόμησης, της σχέσης μεταξύ των εξαρτώμενων μεταβλητών) σε πολυδιάστατους χώρους με τη χρήση kernel συναρτήσεων και συστηματοποίησης. Στην περίπτωση που εφαρμόζεται σε προβλήματα ταξινόμησης, αποτελεί ένα γραμμικό ταξινομητή ο οποίος αναθέτει σε ένα δείγμα εκπαίδευσης την περιγραφή διαφορετικών τάξεων προσαρμόζοντας ένα βέλτιστο υπερεπίπεδο (optimal separating hyperplane) διαχωρισμού, το οποίο αποτελεί εκείνο το βέλτιστο όριο που προσδιορίζεται έπειτα από επαναληπτικές διαδικασίες μάθησης και διαχωρίζει το δείγμα εκπαίδευσης σε διακριτό και προκαθορισμένο αριθμό τάξεων, ενώ παράλληλα ελαχιστοποιεί το σφάλμα ταξινόμησης (Vapnik, 1982), (Zhu & Blumberg, 2002). Με αυτό τον τρόπο, ο αλγόριθμος SVM προσπαθεί να μεγιστοποιήσει το περιθώριο διαχωρισμού (margin), δηλαδή την απόσταση μεταξύ των πλησιέστερων δειγμάτων εκπαίδευσης τα οποία ονομάζονται Support Vectors. Έπειτα, διαχωρίζει κατά τον ίδιο τρόπο και τα υπόλοιπα άγνωστα για τον αναλυτή δεδομένα (Pouteau et al., 2012), (Mutlu, 2014).
- Random Forest: Ο αλγόριθμος αυτός αναπτύχθηκε αρχικά από τον Breiman (Breiman, 2001) και είναι ένα σύνολο μεθόδων που εφαρμόζεται σε προβλήματα επιβλεπόμενης ταξινόμησης και παλινδρόμησης και βασίζεται στα δέντρα ταξινόμησης και παλινδρόμησης (CART) αντίστοιχα. Οι RF ταξινομητές βασίζονται στην υπόθεση ότι διαφορετικοί και ανεξάρτητοι προγνωστικοί παράγοντες προβλέπουν εσφαλμένα στα διάφορα πεδία μελέτης. Συνδυάζοντας όμως τα αποτελέσματα των προβλέψεων αυτών, είναι δυνατή η βελτίωση της συνολικής ακρίβειας πρόβλεψης. Τα CARTs παρουσιάζουν σημαντικές διαφορές ως προς τη δομή τους, εάν τα δείγματα εκπαίδευσης διαφέρουν ελαφρώς. Το χαρακτηριστικό

αυτό σε συνδυασμό με το «Bagging» ή «bootstrap aggregating» και την τυχαία επιλογή οδηγούν στη δημιουργία των προγνωστικών παραγόντων.

4.1.3 Δεδομένα και προσέγγιση της ταξινόμησης

Τα δεδομένα που χρησιμοποιήθηκαν για την εκπαίδευση του κάθε ταξινομητή προήλθαν από τις τρεις εικόνες που παρουσιάζονται στην εικόνα 4.3. Οι δύο από τις τρεις εικόνες αντιστοιχούν σε δύο διαφορετικά χρονικά στιγμιότυπα της περιοχής bedford (2016 και 2017 χρονολογικά), ενώ η τρίτη στην περιοχή patricia. Όπως φαίνεται και στην εικόνα 4.1 τα pixels της περιοχής bedford αντιστοιχίζονται στις εξής πέντε κατηγορίες: Muddy Sand w. Organic (λασπώδης άμμος με οργανικά στοιχεία), Muddy Sand (λασπώδης άμμος), Fine Muddy Sand (λεπτή λασπώδης άμμος), Sand/Gravel (αμμοχάλικο) και Rocky/Gravel (χαλίκι/βραχώδης πυθμένας). Αντίστοιχα, τα pixels της περιοχής patricia αντιστοιχίζονται στις εξής τρεις κλάσεις (εικόνα 4.2): Fine Sand (λεπτή άμμος), Med Sand (εν μέρει άμμος) και Rocky/Gravel (χαλίκι). Δεδομένου ότι οι μοναδικές κατηγορίες διαφέρουν για κάθε εικόνα και ότι στόχος ήταν η συνένωση των δεδομένων από όλες τις εικόνες συνολικά για την εκπαίδευση των μοντέλων, το πρόβλημα ταξινόμησης προσεγγίστηκε με δύο τρόπους.



Εικόνα 4.1: Απεικόνιση των κατηγοριών στις οποίες ανήκουν τα στοιχεία της περιοχής bedford.

Εικόνα 4.2: Απεικόνιση των κατηγοριών στις οποίες ανήκουν τα στοιχεία της περιοχής patricia.

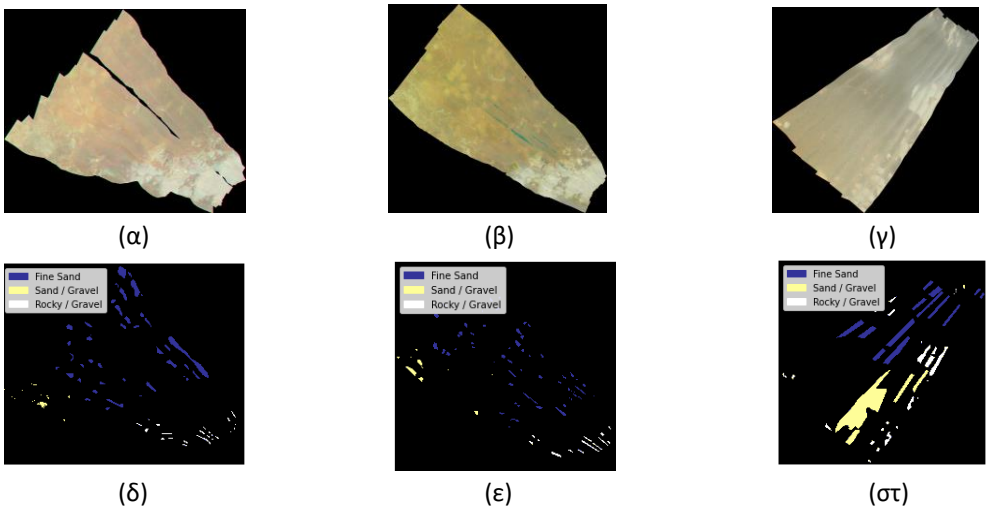
Α' τρόπος: Ο πρώτος τρόπος ήταν κάθε κλάση να θεωρηθεί μοναδική, δηλαδή να επιτευχθεί η ταξινόμηση των θέσεων στοιχείων των εικόνων σε μία από τις 7 κατηγορίες:

- 1 - Muddy Sand w. Organic
- 2 - Muddy Sand
- 3 - Fine Muddy Sand
- 4 - Fine Sand

- 5 - Med Sand
- 6 - Sand/Gravel
- 7 - Rocky/Gravel

Β’ τρόπος: Ο δεύτερος τρόπος ήταν να γίνει ομαδοποίηση των κλάσεων που παρουσιάζουν αρκετές ομοιότητες ως προς τη δομή του εδάφους και να επιλεχθεί μια κατηγορία εκπρόσωπος για κάθε ομάδα που δημιουργήθηκε. Συγκεκριμένα, οι ομάδες κλάσεων και η κατηγορίες που προέκυψαν είναι οι εξής τρεις:

- 1 - Fine Sand (Muddy Sand w. Organic, Muddy Sand, Fine Muddy Sand και Fine Sand)
- 2 - Sand/Gravel (Sand/Gravel και Med Sand)
- 3 - Rocky/Gravel



(α) (β) (γ)

(δ) (ε) (στ)

Εικόνα 4.3: Απεικόνιση των εικόνων που χρησιμοποιήθηκαν για την εκπαίδευση των ταξινομητών και της αντίστοιχης ground-truth εικόνας τους. α) bedford_2016, β)bedford_2017, γ)patricia, δ)gt_bedford_2016, ε)gt_bedford_2017, στ)gt_patricia.

Για κάθε τρόπο προσέγγισης, πριν την προεπεξεργασία των δεδομένων έγινε αντιστοίχιση των τιμών της ground-truth εικόνας, σε μία από τις [1,7] για τον α’ τρόπο και σε μία από τις [1,3] για τον β’. Στην εικόνα 4.3 παρουσιάζονται οι εικόνες που χρησιμοποιήθηκαν για την εκπαίδευση του μοντέλου και οι αντίστοιχες ground-truth εικόνες για το β’ τρόπο προσέγγισης. Είναι εμφανές ότι για τις δύο bedford εικόνες τα δεδομένα που αντιστοιχούν σε κάθε κλάση δεν είναι ισοζυγισμένα και το σύνολο των δεδομένων είναι ανισομερές, με τα δεδομένα που αντιστοιχούν στην κλάση 1 (Fine Sand) να είναι πολύ περισσότερα των άλλων δύο. Κάτι ανάλογο συμβαίνει και στην τρίτη εικόνα, patricia, όπου η κλάση για την οποία δεν υπάρχει μεγάλο πλήθος δεδομένων είναι η 3 (Rocky/Gravel). Αυτό αναμένεται να οδηγήσει σε πιθανή αδυναμία ικανοποιητικής εκπαίδευσης του μοντέλου με τη μέθοδο SVM, καθώς ο ορισμός των διαχωριστικών καμπυλών μπορεί να μην είναι ο

βέλτιστος. Η μέθοδος RF μπορεί και αντιμετωπίζει ικανοποιητικά μη ισοζυγισμένα σύνολα δεδομένων, οπότε και η απόδοσή της αναμένεται να είναι καλύτερη.

Στον πίνακα 4.1 καταγράφεται το πλήθος των δεδομένων (pixels) που αναλογούν σε κάθε εικόνα, καθώς και πόσα από αυτά αντιστοιχούν σε κάθε κατηγορία ταξινόμησης τόσο για την προσέγγιση με τον α' τρόπο με τις 7 κλάσεις, όσο και για το β' τρόπο με τις 3.

Πίνακας 4.1: Πίνακας καταγραφής του πλήθους των στοιχείων ανά εικόνα που ανήκουν σε κάθε κατηγορία ταξινόμησης για τις δύο προσεγγίσεις ταξινόμησης σε 3 και 7 κλάσεις, με βάση το αρχικό σύνολο εκπαίδευσης και το υποσύνολό του διατηρώντας ένα μερίδιο της τάξεως των 50%.

	3_median	7_median	3_median (50%)	7_median (50%)
bedford2016	35251 (29948: 1, 1751: 2, 3552: 3)	35251 (20739: 1 , 5374: 2, 3835: 3, 1751: 6, 3552: 7)	17625 (15008: 1, 823: 2, 1794: 3)	17625 (10445: 1, 2722: 2, 1880: 3, 849: 6, 1729: 7)
bedford2017	47713 (31484: 1, 7260: 2, 8969: 3)	47713 (14258: 1 , 6738: 2, 10488: 3, 7260: 6, 8969: 7)	23856 (15689: 1, 3600: 2, 4567: 3)	23856 (7042: 1 , 3423: 2, 5247: 3, 3662: 6, 4482: 7)
patricia	218272 (81125: 1, 103313: 2, 33834: 3)	218272 (81125: 4, 103313: 5, 33834: 7)	109136 (40641: 1, 51744: 2, 16751: 3)	109136 (40646: 4, 51617: 5, 16873: 7)

Προκειμένου να αξιολογηθεί η υποδομή που σχεδιάστηκε για το ρόλο και τη σημασία που έχει σε ένα παραγωγικό περιβάλλον, έγινε προσομοίωση των συνθηκών που επικρατούν σε ένα σύστημα παραγωγής, όπου με δεδομένο ένα μοντέλο πρόβλεψης που έχει προαχθεί στην παραγωγή, αποκτώνται νέα δεδομένα που μπορούν να συμβάλλουν στη βελτίωση της προβλεπτικής ικανότητας του παραγωγικού μοντέλου. Γι' αυτό το λόγο, από το αρχικό σύνολο δεδομένων για κάθε εικόνα λήφθηκαν διαφορετικές εκδοχές μικρότερου μεγέθους διατηρώντας μόνο ένα μέρος από αυτά σύμφωνα με κάποια λογική.

Αρχικά, η μια στρατηγική ήταν να διατηρηθεί ένα ποσοστό από τα αρχικά δεδομένα με τυχαίο τρόπο επιλογής τους. Στον πίνακα 4.1 έχει υπολογιστεί το πλήθος των δεδομένων που αναλογούν στο 50% από το αρχικό σύνολο και το πλήθος τους ανά κατηγορία. Με την τυχαία διαδικασία επιλογής, το ποσοστό των pixels που αναλογεί σε κάθε κατηγορία είναι ίδιο (με μικρές αποκλίσεις) σε σχέση με το ποσοστό ανά κατηγορία στο αρχικό σύνολο δεδομένων. Συνεπώς, δε δημιουργείται διαφορετική κατανομή των κλάσεων.

Η άλλη στρατηγική ήταν από το αρχικό σύνολο δεδομένων να προκύψουν σύνολα που περιέχουν pixels που αντιστοιχίζονται σε συγκεκριμένες κλάσεις αφαιρώντας τα υπόλοιπα. Ένα τέτοιο παράδειγμα είναι το σύνολο δεδομένων patricia_1_2 στο οποίο έχει αφαιρεθεί η κλάση 3. Ο λόγος που επιλέχθηκε αυτού του είδους ο διαχωρισμός βασίζεται στη λογική ότι σε ένα παραγωγικό μοντέλο το οποίο δεν έχει υψηλή διακριτική ικανότητα για μια συγκεκριμένη κλάση, αλλά πολύ καλή στις υπόλοιπες, με την απόκτηση ενός νέου συνόλου δεδομένων και απομονώνοντας την "προβληματική" κλάση να καταφέρει να βελτιώσει τις μετρικές ακρίβειας για την κλάση αυτή και τελικά να προκύψει ένα καλύτερο μοντέλο. Συνεπώς, στα πειράματα που πραγματοποιήθηκαν περιλαμβάνεται μια

προσομοίωση κατά την οποία αρχικά παράγεται ένα μοντέλο πρόβλεψης, χρησιμοποιώντας τα δεδομένα μόνο από την εικόνα bedford_2016. Στη συνέχεια, παράγεται και αξιολογείται το μοντέλο που προκύπτει από τον εμπλουτισμό της γνώσης του αρχικού μοντέλου με την πληροφορία που προέρχεται από το σύνολο δεδομένων patricia_1_2, δηλαδή μόνο από τις κλάσεις 1 και 2 της εικόνας patricia, για την προσέγγιση με τις 3 κλάσεις (β' τρόπος). Τέλος, κατ' αντιστοιχία, επιλέγεται να εξεταστεί η ενημέρωση του μοντέλου με την πληροφορία από το σύνολο δεδομένων bedford_2017_3. Ανάλογη προσομοίωση γίνεται και για τον α' τρόπο προσέγγισης ταξινόμησης. Το αν η επιλογή των υποσυνόλων των δεδομένων είναι βέλτιστη δεν αποτελεί αντικείμενο μελέτης. Αυτό το οποίο είναι άξιο έρευνας, είναι πώς μπορεί να βελτιώσει κανείς την απόδοση ενός μοντέλου στην παραγωγή, έστω και σε μικρό ποσοστό, καθώς νέα δεδομένα γίνονται διαθέσιμα, εξασφαλίζοντας την ολοκλήρωση των ενεργειών που απαιτεί η ενημέρωση ενός μοντέλου (ιστορικότητα, οργάνωση, παραγωγή, έλεγχο καλής λειτουργίας και εγκατάσταση).

Σε κάθε διαφορετική εκτέλεση του συνολικού πειράματος, για την αξιολόγηση του εκάστοτε μοντέλου, το αρχικό σύνολο δεδομένων χωρίζεται σε δύο επιμέρους σύνολα, το σύνολο εκπαίδευσης και το σύνολο επαλήθευσης. Συγκεκριμένα, για καθεμία από τις διαφορετικές κλάσεις του αρχικού συνόλου, επιλέγεται με τυχαίο τρόπο το 90% των δεδομένων της κλάσης το οποίο και αντιστοιχίζεται στο σύνολο εκπαίδευσης, ενώ το υπόλοιπο 10% στο σύνολο επαλήθευσης. Συνεπώς, συνολικά προκύπτουν δύο σύνολα, μεγέθους, 90% και 10% αντίστοιχα, επί του αρχικού συνόλου, με ίδια αναλογία ως προς τις κλάσεις. Η αξιολόγηση των δύο ταξινομητών γίνεται σύμφωνα με τις μετρικές που περιγράφονται στη συνέχεια. Αρχικά, για κάθε ταξινομητή υπολογίζεται το σφάλμα επαλήθευσης, το οποίο προκύπτει από το σύνολο δεδομένων επαλήθευσης, και περιγράφεται από το λόγο των λανθασμένων προβλέψεων προς το πλήθος των δεδομένων του συνόλου επαλήθευσης ($valid_{err} = \frac{sum(y_{pred} != y_{actual})}{sum(y_{actual})}$).

Οι υπόλοιπες μετρικές προκύπτουν από τη σύγκριση των προβλεπόμενων τιμών από τον ταξινομητή με τις αντίστοιχες πραγματικές τιμές από την ground-truth εικόνα. Οι μετρικές που καταγράφονται για κάθε μέθοδο ταξινόμησης είναι:

- ακρίβεια (accuracy): Η μετρική αυτή περιγράφει το λόγο των σωστών προβλέψεων προς το συνολικό πλήθος των προβλέψεων που πραγματοποιήθηκαν.
- Cohen's kappa: Ο συντελεστής k Cohen (Μανωλέσου, 2015) είναι ένα στατιστικό μέτρο της συμφωνίας μεταξύ των αξιολογήσεων δύο βαθμολογητών όταν και οι δύο βαθμολογούν το ίδιο αντικείμενο. Δηλαδή, μετρά τη συμφωνία/ασυμφωνία με βάση την απόσταση των προβλεπόμενων κλάσεων από τον ταξινομητή και των πραγματικών κλάσεων των προτύπων. Λαμβάνει τιμές στο διάστημα [-1, 1], με την τιμή 1 να δηλώνει τέλεια συμφωνία. Αρνητική τιμή kappa δηλώνει συμφωνία χειρότερη από το αναμενόμενο, ή αλλιώς ασυμφωνία (McHugh, 2012).
- συνέπεια (precision): Πρόκειται για το δείκτη που προκύπτει από τον πίνακα σύγχυσης (confusion matrix) και περιγράφει το ποσοστό των θέσεων στοιχείων που

ο ταξινομητής έχει αναγνωρίσει ότι αντιστοιχούν στην κλάση X και ανήκουν πράγματι στην κλάση X.

- ανάκληση (recall): Η μετρική αυτή προκύπτει επίσης από τον πίνακα σύγχυσης και περιγράφει για κάθε κατηγορία το ποσοστό των θέσεων στοιχείων που ανήκουν στην κατηγορία αυτή και κατάφερε να εντοπίσει.

Με στόχο τον προγραμματισμό των παραπάνω ενεργειών δημιουργήθηκαν δύο DAGs στην πλατφόρμα Airflow, το underwater_run_as_entrypoints και το deploy_best_underwater_model που αναλύονται στη συνέχεια.

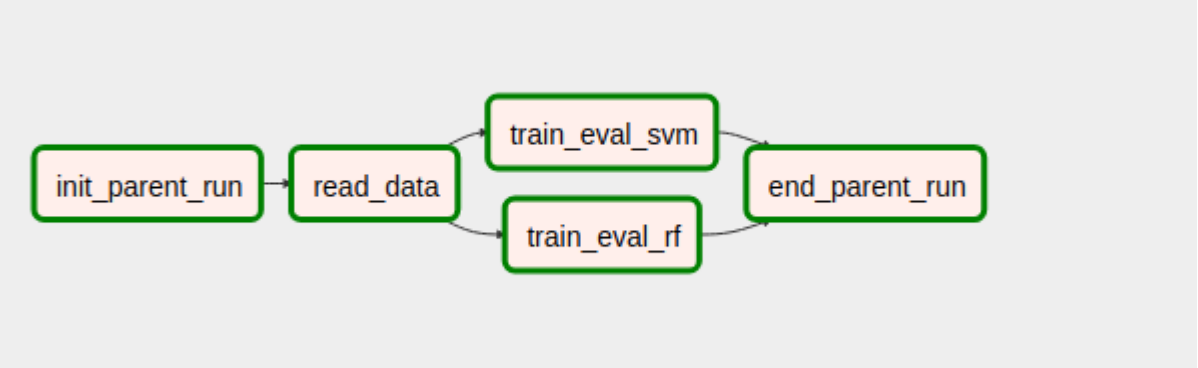
underwater_run_as_entrypoints DAG: Η μορφή του συγκεκριμένου γράφου οπτικοποιείται στην εικόνα 4.4. Σε αυτό το γράφο ορίζονται οι διαδικασίες και οι εξαρτήσεις τους ως προς τη χρονική και ακολουθιακή σειρά εκτέλεσής τους, που σχετίζονται με τη ροή εργασιών παραγωγής ενός μοντέλου πρόβλεψης. Αποτελείται από πέντε διαδικασίες με μια διακλάδωση. Οι διαδικασίες που εντάσσονται στο DAG είναι:

- init_parent_run: Με αυτό το βήμα επιτυγχάνεται η οργάνωση και η ομαδοποίηση των εκτελέσεων που ακολουθούν στα επόμενα βήματα του γράφου. Πρακτικά, αρχικοποιείται μια εκτέλεση “πατέρας”, ως ρίζα των επιμέρους βημάτων, στο περιβάλλον MLflow με ισχύ σε ένα συγκεκριμένο πείραμα (συγκεκριμένα το UnderwaterExp).
- load_data_step: Σε αυτό το βήμα φορτώνονται και συνενώνονται τα σύνολα δεδομένων που δίνονται ως παράμετρος κατά την πυροδότηση του συγκεκριμένου γράφου. Προεπιλογή του συνόλου δεδομένων που χρησιμοποιείται για την εκπαίδευση των μοντέλων, είναι όλες οι εικόνες με όλα τα διαθέσιμα δεδομένα τους. Τη συνένωση ακολουθεί ο διαχωρισμός σε σύνολο εκπαίδευσης και σύνολο επαλήθευσης για την αξιολόγηση των μοντέλων. Τα κατάλληλα αρχεία που δημιουργούνται για κάθε εκτέλεση αποθηκεύονται όπως έχει περιγραφεί στην αποθήκη αντικειμένων του MLflow. Το στάδιο αυτό ακολουθεί μια διακλάδωση για παράλληλη εκπαίδευση δύο διαφορετικών μοντέλων με διαφορετική μέθοδο ταξινόμησης.
- train_eval_rf: Σε αυτό το βήμα κατασκευάζεται ένα μοντέλο ταξινόμησης με τη μέθοδο RF και εκπαιδεύεται με το σύνολο δεδομένων εκπαίδευσης. Ακολουθεί η αξιολόγησή του υπολογίζοντας το σφάλμα επαλήθευσης σύμφωνα με το σύνολο επαλήθευσης (valid_error). Για την αξιολόγηση της διακριτικής του ικανότητας εξάγονται ορισμένες τιμές μετρικών συνολικά και για τις τρεις εικόνες. Στην πράξη για κάθε εικόνα, αρχικά παράγεται η εικόνα ίδιου μεγέθους με τις προβλεπόμενες κλάσεις και στη συνέχεια συγκρίνονται με την αντίστοιχη ground-truth εικόνα.
- train_eval_svm: Η διαδικασία σε αυτό το βήμα είναι όμοια με την παράλληλη προς αυτή εργασία. Η διαφορά εντοπίζεται στη μέθοδο ταξινόμησης εφαρμόζοντας την τεχνική SVM.

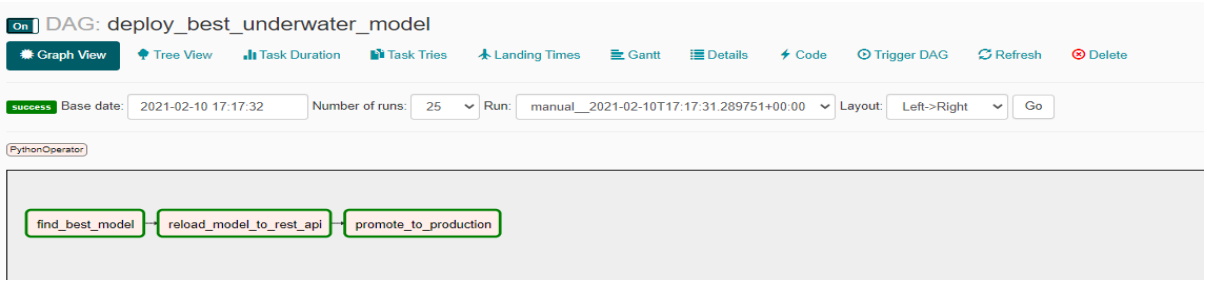
- end_parent_run: Σε αυτό το τελικό βήμα, όταν όλα τα προγενέστερα έχουν επιτευχθεί, ολοκληρώνεται με επιτυχία η εκτέλεση του “πατέρα”.

deploy_best_underwater_model DAG: Η μορφή του γράφου αυτού οπτικοποιείται στην εικόνα 4.5. Εδώ ορίζονται οι κατάλληλες ενέργειες που αποσκοπούν στην εύρεση του βέλτιστου μοντέλου και την προαγωγή του στο παραγωγικό περιβάλλον:

- find_best_model: Σε αυτό το βήμα συλλέγονται όλες οι διαφορετικές εκτελέσεις που σχετίζονται με το πείραμα UnderwaterExp. Φιλτράρονται οι εκτελέσεις που πληρούν συγκεκριμένες συνθήκες, για παράδειγμα όσες εκτελέσεις έχουν σαν μετρική το κλειδί accuracy (ακρίβεια) και τιμή μεγαλύτερη από 0.87. Η λεκτική διαλογής των εκτελέσεων είναι συμβατή με τον τρόπο που εκτελούνται ερωτήσεις (queries) σχετικά με τα πειράματα στην πλατφόρμα MLflow. Από τις επιλεγμένες εκτελέσεις αναζητείται το αναγνωριστικό κλειδί (id) που περιέχει το μοντέλο με τη βέλτιστη απόδοση. Αν βρεθεί μοντέλο να ικανοποιεί τις παραπάνω συνθήκες, τότε καταγράφεται στο μητρώο των μοντέλων του MLflow, με τον κατάλληλο αριθμό έκδοσης, και προάγεται στην κατάσταση stage (πρόκειται για ένα βήμα πριν την προαγωγή σε περιβάλλον παραγωγής). Αν δε βρεθεί μοντέλο να πληροί αυτές τις προϋποθέσεις τα επόμενα βήματα του γράφου αγνοούνται.
- reload_model_to_rest_api: Αν και εφόσον βρεθεί η εκτέλεση με το βέλτιστο μοντέλο, η εφαρμογή που ενσωματώνει τη λειτουργία του βέλτιστου μοντέλου, ενημερώνεται μέσω μιας κλήσης HTTP για την ύπαρξη ενός νέου παραγωγικού μοντέλου, το οποίο θα πρέπει να φορτώσει εκ νέου στη μνήμη του και να αντικαταστήσει το παλιό.
- promote_to_production: Όταν ο server του προηγούμενου βήματος ενημερωθεί με επιτυχία, δηλαδή η απάντηση της κλήσης είναι 200, τότε το μοντέλο του προηγούμενου βήματος προάγεται από την κατάσταση stage στην κατάσταση production, δηλώνοντας έτσι το μοντέλο που εξυπηρετεί τις προβλέψεις.



Εικόνα 4.4: Απεικόνιση του γράφου στον οποίο ορίζονται τα στάδια που απαιτούνται για την παραγωγή του μοντέλου πρόβλεψης του προβλήματος εφαρμογής.



Εικόνα 4.5: Απεικόνιση του γράφου στον οποίο ορίζονται οι διαδικασίες που απαιτούνται για την εύρεση του βέλτιστου μοντέλου και την εγκατάστασή του στο παραγωγικό περιβάλλον.

4.2 Σχολιασμός πειραμάτων

4.2.1 Σύγκριση μοντέλων με εφαρμογή του μεσαίου φίλτρου και 3 κλάσεις ταξινόμησης

Στον πίνακα 4.2 παρουσιάζονται οι μετρικές των μοντέλων ταξινόμησης που παράγονται σύμφωνα με τη διαδικασία που έχει ήδη περιγραφεί. Όπως είναι εμφανές, τα δύο καλύτερα μοντέλα που προκύπτουν είναι αυτά με τον ταξινομητή RF, χρησιμοποιώντας όλα τα δεδομένα των εικόνων, ή χρησιμοποιώντας το 50% τους. Υπενθυμίζεται ότι στην περίπτωση του 50% των δεδομένων η αναλογία ανά κατηγορία για την κάθε εικόνα είναι όμοια, συνεπώς τα δύο αυτά μοντέλα εκπαιδεύονται ανάλογα. Οι μετρικές ακριβείας μεταξύ των δύο αυτών μοντέλων παρουσιάζουν ανεπαίσθητες διαφορές της τάξεως 1%. Στην περίπτωση του ταξινομητή SVM οι διαφορές είναι λίγο μεγαλύτερες, πράγμα λογικό, καθώς περισσότερα δείγματα μπορούν να συμβάλλουν σε καλύτερο προσδιορισμό των διαχωριστικών καμπυλών των κλάσεων.

Πίνακας 4.2: Πίνακας καταγραφής των αποτελεσμάτων αξιολόγησης των διαφορετικών μοντέλων για την περίπτωση εφαρμογής του μεσαίου φίλτρου και της ταξινόμησης στο σύνολο των τριών κλάσεων.

data	classifier	valid_error	accuracy	kappa	precision_class_1	recall_class_1	precision_class_2	recall_class_2	precision_class_3	recall_class_3
all	SVM	0.228	0.77	0.61	0.97	0.87	0.63	0.96	0.93	0.0
all	**RF**	**0.027**	**0.97**	**0.95**	**0.98**	**0.98**	**0.97**	**0.96**	**0.95**	**0.98**
percentage_0.5	**RF**	**0.036**	**0.96**	**0.94**	**0.97**	**0.98**	**0.96**	**0.95**	**0.95**	**0.96**
bedford16	RF	0.451	0.55	0.19	0.56	1.0	0.13	0.03	0.97	0.42
bedford16, patricia_class_1_2	RF	0.176	0.83	0.7	0.93	0.9	0.72	0.97	0.97	0.26
bedford16, patricia_class_1_2, bedford17_class_3	RF	0.121	0.88	0.8	0.94	0.9	0.8	0.96	0.96	0.65

bedford16	SVM	0.484	0.52	0.2	0.51	0.83	0.0	0.0	0.52	0.8
bedford16_class_1_2	SVM	0.519	0.48	0.04	0.55	0.98	0.09	0.04	0.0	0.0
bedford16_class_1_2	RF	0.501	0.5	0.07	0.54	1.0	0.2	0.07	0.0	0.0
bedford16percentage_0.5	RF	0.448	0.55	0.19	0.57	1.0	0.2	0.06	0.97	0.38
bedford16percentage_0.5	SVM	0.524	0.48	0.16	0.5	0.74	0.36	0.02	0.42	0.78
percentage_0.5	SVM	0.379	0.62	0.34	0.56	0.99	0.2	0.0	0.91	0.97

Όσον αφορά την απόδοση των δύο ταξινομητών, με δεδομένα εκπαίδευσης το αρχικό σύνολο δεδομένων, φαίνεται ότι η μηχανή διανυσμάτων υποστήριξης δεν αποδίδει πολύ ικανοποιητικά παρουσιάζοντας μεγάλη αδυναμία να εντοπίσει σημεία των εικόνων που ανήκουν στην κατηγορία 3. Αυτό, ενδεχομένως να οφείλεται στο γεγονός ότι τα δεδομένα εκπαίδευσης δεν είναι ισοπληθή για κάθε κατηγορία.

Άξια επισήμανσης είναι η προσομοίωση των συνθηκών ενός παραγωγικού μοντέλου που ελέγχεται η εκπαίδευσή του με περισσότερα δεδομένα. Στον πίνακα 4.2 στο κίτρινο πλαίσιο παρουσιάζονται οι μετρικές απόδοσης των νέων μοντέλων. Αρχικά, θεωρείται ως σημείο αναφοράς το παραγωγικό μοντέλο που προέκυψε με τη μέθοδο RF και εκπαιδεύτηκε με τα δεδομένα της εικόνας bedford_2016. Είναι εμφανές ότι το μοντέλο παρουσιάζει αρκετά προβλήματα ταξινόμησης κυρίως ως προς την κλάση 2. Στην πορεία που αποκτάται και η εικόνα patricia επιλέγεται να επανεκπαιδευτεί το μοντέλο με τα προηγούμενα δεδομένα εμπλουτίζοντάς τα με τα δεδομένα της νέας εικόνας απομονώνοντας τις κλάσεις 1 και 2. Συγκρίνοντας τις δύο γραμμές παρατηρεί κανείς ότι τόσο το recall όσο και το precision της αρχικά “προβληματικής” κλάσης 2 έχει αυξηθεί, αυξάνοντας επίσης τη συνέπεια της κλάσης 1. Μειώνεται ωστόσο, η ικανότητα εντοπισμού της κλάσης 3. Δημιουργείται έτσι η ανάγκη για περισσότερα δεδομένα της κλάσης 3. Με την απόκτηση και της τρίτης εικόνας, bedford_2017, επιλέγεται, μιας και η επίδοση των κλάσεων 1 και 2 είναι σχετικά ικανοποιητική, να απομονωθεί μόνο η κλάση 3, πράγμα που οδηγεί σε αύξηση της ανάκλησης για την κλάση 3, και της συνέπειας των άλλων δύο κλάσεων, όπως είναι λογικό αφού πλέον αντιμετωπίζεται η λανθασμένη ταξινόμηση (misclassification) της κλάσης 3 στις 1 και 2. Ακολουθούν ορισμένες τετριμμένες εκτελέσεις, όπως είναι η εκπαίδευση του μοντέλου με δεδομένα μόνο από τις κλάσεις 1 και 2 οδηγώντας προφανώς, σε πλήρη αδυναμία κατηγοριοποίησης της κλάσης 3.

4.2.2 Σύγκριση μοντέλων με εφαρμογή του μεσαίου φίλτρου και 7 κλάσεις ταξινόμησης

Πίνακας 4.3: Πίνακας καταγραφής των αποτελεσμάτων αξιολόγησης των διαφορετικών μοντέλων για την περίπτωση εφαρμογής του μεσαίου φίλτρου και της ταξινόμησης στο σύνολο των εφτά κλάσεων.

data	classifier	valid_error	accuracy	kappa	precision_class_1	recall_class_1	precision_class_2	recall_class_2	precision_class_3	recall_class_3	precision_class_4	recall_class_4	precision_class_5	recall_class_5	precision_class_6	recall_class_6	precision_class_7	recall_class_7
all	SVM	0.595	0.41	0.12	0.0	0.0	0.95	0.21	0.78	0.75	0.99	0.07	0.37	1.0	0.0	0.0	0.0	0.0
all	**RF**	**0.036**	**0.96**	**0.95**	**0.91**	**0.95**	**0.91**	**0.86**	**0.96**	**0.97**	**0.99**	**0.98**	**0.97**	**0.97**	**0.92**	**0.85**	**0.95**	**0.98**
percentage_0.5	**RF**	**0.049**	**0.95**	**0.94**	**0.89**	**0.93**	**0.86**	**0.83**	**0.92**	**0.96**	**0.99**	**0.98**	**0.97**	**0.95**	**0.88**	**0.81**	**0.95**	**0.96**
bedford_2016	RF	0.79	0.21	0.12	0.15	0.98	0.46	0.34	0.3	0.27	0.0	0.0	0.0	0.0	0.06	0.19	0.97	0.42
bedford16, patricia_class_4_5	RF	0.266	0.73	0.63	0.79	0.55	0.78	0.34	0.61	0.26	0.79	0.98	0.7	0.97	0.19	0.16	0.98	0.26
bedford16, patricia_class_4_5, bedford17_class6_7	RF	0.188	0.81	0.75	0.85	0.55	0.77	0.33	0.67	0.26	0.81	0.98	0.85	0.97	0.39	0.9	0.96	0.64
bedford16, patricia_class_4_5, bedford17_class6_7	SVM	0.54	0.46	0.23	0.07	0.01	0.0	0.0	0.0	0.0	0.99	0.38	0.44	1.0	0.13	0.42	0.0	0.0
bedford16, patricia_class_4_5	SVM	0.495	0.5	0.36	0.26	0.96	0.0	0.0	0.0	0.0	0.97	0.86	0.47	0.47	0.0	0.0	0.0	0.0
bedford_2016	SVM	0.701	0.3	0.21	0.15	0.99	0.26	0.23	0.6	0.81	0.0	0.0	0.0	0.0	0.33	0.05	0.86	0.88
percentage_0.5	SVM	0.459	0.54	0.32	0.93	0.0	0.92	0.03	0.98	0.57	0.98	0.6	0.43	1.0	0.0	0.0	0.96	0.04

Όμοια είναι τα αποτελέσματα που προκύπτουν αν η ταξινόμηση γίνει με την προσέγγιση των εφτά κλάσεων. Και με αυτήν την προσέγγιση το πλήθος των δεδομένων (αρχικό μέγεθος ή 50% αυτών) δεν επηρεάζει σημαντικά την απόδοση των ταξινομητών, ειδικά όσον αφορά τον ταξινομητή RF. Σε αυτήν την περίπτωση, όπως είναι λογικό εκπαιδεύοντας το μοντέλο πρόβλεψης μόνο με την εικόνα bedford_2016 οι τιμές των μετρικών για τις κλάσεις 4 και 5 είναι μηδενικές καθώς η εικόνα δεν περιέχει σημεία που αντιστοιχούν σε αυτές τις κλάσεις. Προσθέτοντας δεδομένα από την εικόνα patricia για τις κλάσεις 4 και 5 παρατηρείται ικανοποιητική βελτίωση του μοντέλου, μειώνεται ωστόσο η επίδοσή του για την κλάση 7. Τέλος, προστίθενται δεδομένα των κλάσεων 6 και 7, στις οποίες είναι αδύναμο το μοντέλο, από την εικόνα bedford_2017, βελτιώνοντας σημαντικά την τελική του ακρίβεια από 73% σε 81%. Αξίζει να σημειωθεί ότι η επίδοση των μοντέλων που εκπαιδεύονται με όλα τα δεδομένα από τις εικόνες, είναι πολύ μεγαλύτερη (96%). Η εύρεση του βέλτιστου συνδυασμού υποσυνόλων των δεδομένων είναι αντικείμενο μελέτης του σταδίου της προεπεξεργασίας ενός αυτοματοποιημένου pipeline. Με αυτήν την προσομοίωση και τη δημιουργία αυτοματοποιημένων ροών εργασιών στόχος είναι ο συμπερασμός της ευκολίας και της ταχύτητας με τις οποίες μπορεί κανείς να επιτύχει την εύρεση της βέλτιστης λύσης.

4.2.3 Σύγκριση λοιπών μοντέλων

Σε αυτήν την ενότητα παρουσιάζονται οι αντίστοιχοι πίνακες για τα μοντέλα που αναπτύχθηκαν με τις προαναφερθείσες διαφορετικές μεθόδους. Τα συμπεράσματα ως προς την απόδοση των μοντέλων είναι όμοια με τα προηγούμενα.

Κανονικοποίηση ελαχίστου-μεγίστου και 3 κλάσεις ταξινόμησης

Πίνακας 4.4: Πίνακας καταγραφής των αποτελεσμάτων αξιολόγησης των διαφορετικών μοντέλων για την περίπτωση εφαρμογής της κανονικοποίησης ελαχίστου-μεγίστου και της ταξινόμησης στο σύνολο των τριών κλάσεων.

data	classifier	valid_error	accuracy	kappa	precision_class_1	recall_class_1	precision_class_2	recall_class_2	precision_class_3	recall_class_3
percentage_0.5	**RF**	**0.025**	**0.97**	**0.96**	**0.98**	**0.99**	**0.97**	**0.97**	**0.97**	**0.94**
all	SVM	0.109	0.89	0.82	0.93	0.97	0.85	0.91	0.88	0.62
all	**RF**	**0.014**	**0.98**	**0.98**	**0.99**	**0.99**	**0.98**	**0.98**	**0.97**	**0.98**
bedford16	RF	0.838	0.16	-0.31	0.23	0.27	0.18	0.06	0.04	0.08
bedford16, patricia_class_1_2	RF	0.235	0.77	0.6	0.92	0.82	0.65	0.99	0.96	0.07
bedford16, patricia_class_1_2, bedford17_class_3	RF	0.212	0.79	0.65	0.92	0.81	0.68	0.98	0.92	0.26
bedford16, patricia_class_1_2, bedford17_class_3	SVM	0.21	0.79	0.64	0.95	0.91	0.66	0.97	0.0	0.0
bedford16_class_1_2	RF	0.629	0.37	-0.16	0.41	0.75	0.13	0.05	0.0	0.0
bedford16_class_1_2	SVM	0.522	0.48	0.16	0.99	0.23	0.42	1.0	0.0	0.0
bedford16percentage_0.5	SVM	0.658	0.34	0.16	0.98	0.39	0.14	0.0	0.19	1.0
bedford16percentage_0.5	RF	0.674	0.33	0.06	0.98	0.26	0.45	0.52	0.03	0.08
percentage_0.5	SVM	0.11	0.89	0.82	0.93	0.97	0.85	0.91	0.88	0.62

Κανονικοποίηση ελαχίστου-μεγίστου και 7 κλάσεις ταξινόμησης

Πίνακας 4.5: Πίνακας καταγραφής των αποτελεσμάτων αξιολόγησης των διαφορετικών μοντέλων για την περίπτωση εφαρμογής της κανονικοποίησης ελαχίστου-μεγίστου και της ταξινόμησης στο σύνολο των εφτά κλάσεων.

data	classifier	valid_error	accuracy	kappa	precision_class_1	recall_class_1	precision_class_2	recall_class_2	precision_class_3	recall_class_3	precision_class_4	recall_class_4	precision_class_5	recall_class_5	precision_class_6	recall_class_6	precision_class_7	recall_class_7
percentage_0.5	**RF**	**0.038**	**0.96**	**0.95**	**0.93**	**0.96**	**0.86**	**0.81**	**0.95**	**0.95**	**0.98**	**0.98**	**0.97**	**0.98**	**0.93**	**0.86**	**0.97**	**0.94**
all	**RF**	**0.022**	**0.97**	**0.97**	**0.95**	**0.97**	**0.91**	**0.85**	**0.97**	**0.97**	**0.99**	**0.99**	**0.98**	**0.99**	**0.97**	**0.93**	**0.97**	**0.98**
all	SVM	0.172	0.83	0.77	0.83	0.98	0.78	0.18	0.85	0.92	0.85	0.96	0.78	0.89	0.76	0.01	0.92	0.63
bedford_	RF	0.842	0.16	0.05	0.64	0.58	0.58	0.33	0.09	0.33	0.0	0.0	0.0	0.0	0.2	0.66	0.07	0.27

2016																		
bedford16, patricia_class_4_5	RF	0.28	0.72	0.62	0.92	0.58	0.8	0.33	0.71	0.33	0.88	0.99	0.67	0.99	0.09	0.19	0.93	0.07
bedford16, patricia_class_4_5, bedford17_class6_7	RF	0.228	0.77	0.69	0.93	0.58	0.82	0.33	0.66	0.36	0.88	0.99	0.76	0.99	0.31	0.96	0.97	0.26
bedford16, patricia_class_4_5, bedford17_class6_7	SVM	0.304	0.7	0.59	0.88	0.58	0.79	0.2	0.9	0.22	0.87	0.98	0.67	0.96	0.16	0.59	0.0	0.0
bedford16, patricia_class_4_5	SVM	0.281	0.72	0.61	0.89	0.58	0.79	0.21	0.56	0.89	0.86	0.98	0.65	0.98	0.07	0.04	0.43	0.0
bedford_2016	SVM	0.719	0.28	0.17	0.89	0.58	0.79	0.22	0.43	0.98	0.0	0.0	0.0	0.0	0.09	0.09	0.2	1.0
percentage_0.5	SVM	0.174	0.83	0.77	0.83	0.98	0.79	0.18	0.84	0.92	0.85	0.96	0.78	0.9	0.75	0.0	0.92	0.62

Κανονικοποίηση z-score και 3 κλάσεις ταξινόμησης

Πίνακας 4.6: Πίνακας καταγραφής των αποτελεσμάτων αξιολόγησης των διαφορετικών μοντέλων για την περίπτωση εφαρμογής της κανονικοποίησης z-score και της ταξινόμησης στο σύνολο των τριών κλάσεων.

data	**classifier**	**valid_error**	**accuracy**	**kappa**	**precision_class_1**	**recall_class_1**	**precision_class_2**	**recall_class_2**	**precision_class_3**	**recall_class_3**
percentage_0.5	**RF**	**0.023**	**0.98**	**0.96**	**0.98**	**0.99**	**0.98**	**0.97**	**0.97**	**0.97**
all	SVM	0.103	0.9	0.83	0.93	0.87	0.84	0.9	0.95	0.96
all	**RF**	**0.015**	**0.98**	**0.98**	**0.99**	**0.99**	**0.99**	**0.98**	**0.97**	**0.98**
bedford16	RF	0.32	0.68	0.44	0.62	0.99	0.79	0.24	0.97	0.76
bedford16, patricia_class_1_2	RF	0.164	0.84	0.72	0.91	0.94	0.75	0.97	0.98	0.18
bedford16, patricia_class_1_2, bedford17_class_3	RF	0.146	0.86	0.75	0.92	0.94	0.77	0.98	0.97	0.29
bedford16, patricia_class_1_2, bedford17_class_3	SVM	0.183	0.82	0.7	0.98	0.84	0.68	0.97	0.96	0.38
bedford16, patricia_class_1_2	SVM	0.229	0.77	0.61	0.98	0.85	0.62	0.98	0.94	0.02
bedford16	SVM	0.38	0.62	0.34	0.56	1.0	0.85	0.01	0.91	0.94
bedford16_class_1_2	RF	0.451	0.55	0.16	0.52	1.0	0.8	0.2	0.0	0.0
bedford16_class_1_2	SVM	0.466	0.53	0.14	0.6	0.99	0.29	0.17	0.0	0.0
bedford16percentage_0.5	SVM	0.379	0.62	0.34	0.56	1.0	0.81	0.01	0.91	0.94

bedford16percentage_0.5	RF	0.335	0.66	0.42	0.6	0.99	0.76	0.21	0.96	0.76
percentage_0.5	SVM	0.102	0.9	0.83	0.94	0.87	0.84	0.91	0.95	0.96

Κανονικοποίηση z-score και 7 κλάσεις ταξινόμησης

Πίνακας 4.7: Πίνακας καταγραφής των αποτελεσμάτων αξιολόγησης των διαφορετικών μοντέλων για την περίπτωση εφαρμογής της κανονικοποίησης z-score και της ταξινόμησης στο σύνολο των εφτά κλάσεων.

data	classifier	valid_error	accuracy	kappa	precision_class_1	recall_class_1	precision_class_2	recall_class_2	precision_class_3	recall_class_3	precision_class_4	recall_class_4	precision_class_5	recall_class_5	precision_class_6	recall_class_6	precision_class_7	recall_class_7
percentage_0.5	**RF**	**0.044**	**0.96**	**0.94**	**0.91**	**0.95**	**0.81**	**0.8**	**0.89**	**0.94**	**0.98**	**0.98**	**0.97**	**0.97**	**0.93**	**0.83**	**0.97**	**0.97**
all	RF	0.024	0.97	0.97	0.94	0.97	0.9	0.84	0.95	0.97	0.99	0.98	0.98	0.98	0.96	0.92	0.97	0.98
all	SVM	0.254	0.75	0.65	0.0	0.0	0.03	0.0	0.69	0.11	0.69	0.98	0.74	0.96	0.18	0.01	0.94	0.97
percentage_0.5	SVM	0.254	0.75	0.65	0.0	0.0	0.07	0.0	0.68	0.1	0.69	0.98	0.74	0.96	0.18	0.01	0.94	0.97
bedford_2016	RF	0.702	0.3	0.23	0.26	0.97	0.11	0.48	0.21	0.82	0.0	0.0	0.0	0.0	0.16	0.55	0.96	0.71
bedford16, patricia_class_4_5	RF	0.226	0.77	0.69	0.85	0.8	0.71	0.45	0.72	0.53	0.82	0.99	0.72	0.99	0.89	0.25	0.97	0.16
bedford16, patricia_class_4_5, bedford17_class6_7	RF	0.189	0.81	0.74	0.86	0.76	0.72	0.44	0.88	0.57	0.83	0.99	0.76	0.99	0.92	0.94	0.98	0.29
bedford16, patricia_class_4_5, bedford17_class6_7	SVM	0.339	0.67	0.52	0.0	0.0	0.0	0.0	0.0	0.0	0.68	0.98	0.62	0.98	0.03	0.0	0.96	0.42
bedford16, patricia_class_4_5	SVM	0.391	0.61	0.43	0.0	0.0	0.0	0.0	0.0	0.0	0.68	0.98	0.56	0.99	0.0	0.0	0.96	0.05
bedford_2016	SVM	0.683	0.32	0.24	0.21	0.97	0.1	0.37	0.3	0.96	0.0	0.0	0.0	0.0	0.21	0.18	0.92	0.89

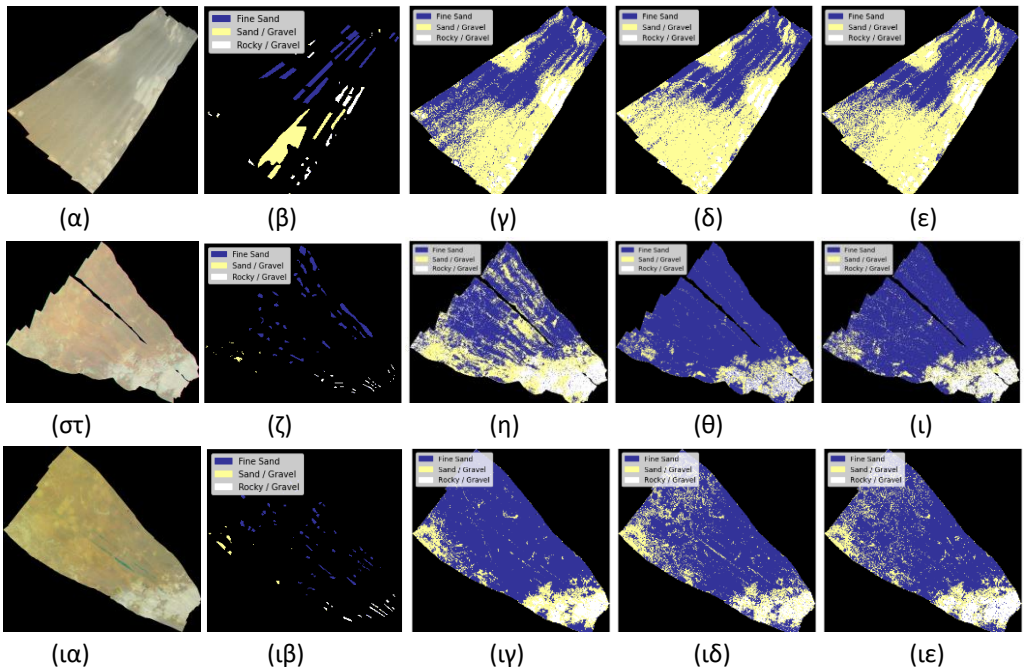
4.2.4 Σύγκριση των διαδικασιών προεπεξεργασίας

Προκειμένου να μελετηθεί το αποτέλεσμα της κατηγοριοποίησης των στοιχείων μιας εικόνας σε έναν προκαθορισμένο αριθμό κλάσεων για τις τρεις διαφορετικές τεχνικές

προεπεξεργασίας, παρουσιάζονται οι μετρικές απόδοσης των μοντέλων που παρήχθησαν από την εκπαίδευση με όλα τα διαθέσιμα δεδομένα από τις τρεις εικόνες συνολικά, για τον ταξινομητή RF.

Πίνακας 4.8: Πίνακας σύγκρισης των αποτελεσμάτων των τριών διαφορετικών τεχνικών προεπεξεργασίας για την περίπτωση του ταξινομητή RF, της ταξινόμησης στις 3 κλάσεις και χρησιμοποιώντας όλα τα στοιχεία του αρχικού συνόλου δεδομένων.

name	classifier	valid_error	accuracy	kappa	precision_class_1	recall_class_1	precision_class_2	recall_class_2	precision_class_3	recall_class_3
median	RF	0.027	0.97	0.95	0.98	0.98	0.97	0.96	0.95	0.98
minmax	RF	0.014	0.98	0.98	0.99	0.99	0.98	0.98	0.97	0.98
meanstd	RF	0.015	0.98	0.98	0.99	0.99	0.99	0.98	0.97	0.98



Εικόνα 4.6: Απεικόνιση των παραγόμενων εικόνων από τα τρία μοντέλα πρόβλεψης για κάθε τεχνική προεπεξεργασίας, με αντιστοίχιση σε 3 κατηγορίες-κλάσεις και χρήση του ταξινομητή RF για κάθε μια από τις τρεις εικόνες εκπαίδευσης. α)patricia, β)gt_patricia, γ)patricia_median, δ)patricia_min_max, ε)patricia_z-score, στ)bedford_2016, ζ)gt_bedford_2016, η)bedford_2016_median, ϑ)bedford_2016_min_max, ι)bedford_2016_z-score, ια)bedford_2017, ιβ)gt_bedford_2017, ιγ)bedford_2017_median, ιδ)bedford_2017_min_max, ιε)bedford_2017_z-score.

Στον πίνακα 4.8 που παρουσιάζονται οι τιμές των μετρικών που μελετώνται, είναι εμφανές ότι η μέθοδος κανονικοποίηση ελαχίστου-μεγίστου έχει ίδια ποσοστά επιτυχίας με

τη μέθοδο κανονικοποίησης z-score. Η μόνη διαφορά είναι στη συνέπεια της κλάσης 2, ωστόσο είναι πολύ μικρή της τάξεως 1%, που ενδεχομένως να οφείλεται και σε στρογγυλοποίηση της τιμής. Εξίσου ικανοποιητική είναι η επίδοση του μεσαίου φίλτρου με αμελητέες διαφορές ως προς τη συνέπεια και την ανάκληση των κλάσεων, ενώ συνολικά η ακρίβεια που επιτυγχάνει είναι 97% έναντι της 98% των άλλων δύο μεθόδων. Στην εικόνα 4.6 συμπεριλαμβάνονται οι εικόνες που παράγονται από τα μοντέλα ταξινόμησης για τις διαφορετικές αυτές προσεγγίσεις.

Παρατηρείται ότι οι εικόνες που παράγονται από το μοντέλο πρόβλεψης όταν είσοδος είναι η patricia μοιάζουν αρκετά μεταξύ τους. Αυτό όμως, δε συμβαίνει στον ίδιο βαθμό με είσοδο τις δύο εκδοχές της περιοχής bedford. Αν και οι εικόνες που παράγονται έπειτα από προεπεξεργασία με μεθόδους κανονικοποίησης έχουν ανεπαίσθητες διαφορές, σύμφωνα με το ανθρώπινο μάτι, όταν συγκρίνονται με την εικόνα που παράγεται με τη μέθοδο του μεσαίου φίλτρου οι διαφορές είναι αισθητές σε μια εκτεταμένη περιοχή της εικόνας. Ωστόσο, η απόδοση και των τριών εικόνων είναι εξίσου ικανοποιητική. Αξίζει να σημειωθεί ότι η επαλήθευση και οι τιμές των μετρικών αξιολόγησης προκύπτουν από τις ground-truth εικόνες. Δηλαδή ως μάσκα στις παραγόμενες εικόνες χρησιμοποιούνται μόνο τα σημεία στα οποία υπάρχει η γνώση της ταξινόμησης. Αν παρατηρήσει κανείς τις τρεις εκδοχές των εικόνων που παράγονται για τις τρεις τεχνικές προεπεξεργασίας, θα δει ότι στα σημεία αυτά η πρόβλεψη είναι ίδια και στις τρεις εκδοχές, δικαιολογώντας έτσι την πολύ καλή επίδοση που έχουν και οι τρεις τεχνικές.

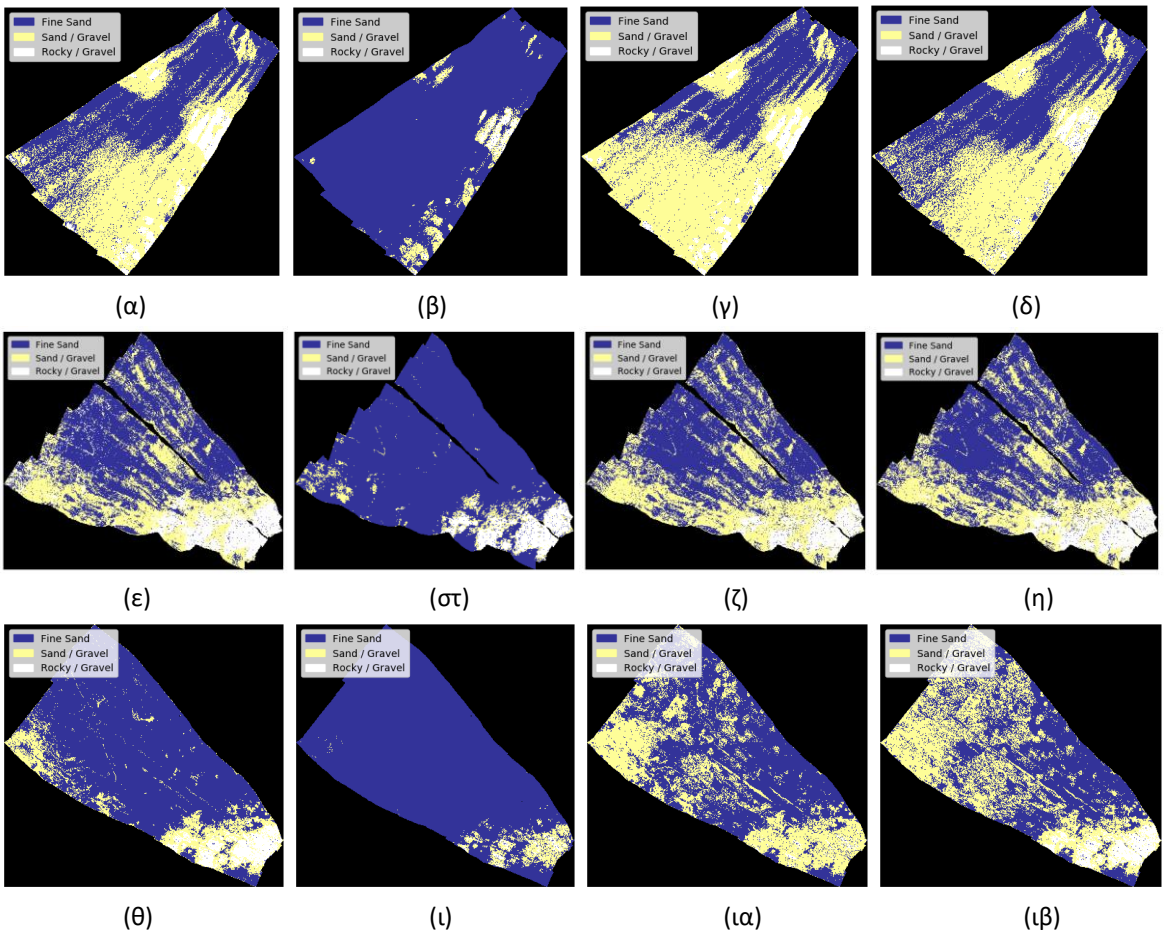
4.2.5 Εξέλιξη του μοντέλου με διαδοχική εισαγωγή νέων δεδομένων

Ένα κομμάτι των πειραμάτων εστίασε στην εξέλιξη ενός παραγωγικού μοντέλου χρησιμοποιώντας επιπλέον πληροφορία που γίνεται διαθέσιμη στην εφαρμογή σε βάθος χρόνου, οδηγώντας σταδιακά στην ενημέρωση και τη βελτιστοποίησή του. Προκειμένου να μελετηθεί η χρονική του εξέλιξη εξετάστηκαν τα εξής τρία διαφορετικά σύνολα δεδομένων, όπως αυτά ορίστηκαν στο προηγούμενο κεφάλαιο:

- bedford_2016 (dataset_1)
- bedford_2016 και patricia_1_2 (dataset_2)
- bedford_2016, patricia_1_2 και bedford_2017_3 (dataset_3)

Η διαδικασία αυτή της προσομοίωσης πραγματοποιήθηκε για κάθε συνδυασμό παραμέτρων εκπαίδευσης (αλγόριθμος ταξινόμησης, τεχνική προεπεξεργασίας, ομαδοποίηση κατηγοριών ταξινόμησης). Στην εικόνα 4.7 παρατίθεται για καθεμία από τις τρεις εικόνες που χρησιμοποιήθηκαν για την εκπαίδευση, η παραγόμενη εικόνα ταξινόμησης από το μοντέλο που παράγεται με τον αλγόριθμο RF και την τεχνική προεπεξεργασίας μεσαίου φίλτρου για την προσέγγιση με τις τρεις κατηγορίες ταξινόμησης και με δεδομένα εκπαίδευσης τα αρχικά δεδομένα συνολικά. Δίπλα στην εικόνα αναφοράς, για κάθε μια από τις εικόνες, παρουσιάζεται η εικόνα που προκύπτει από το καθένα μοντέλο των τριών διαφορετικών συνόλων εκπαίδευσης με τη σειρά. Είναι εμφανές ότι, όσο το σύνολο

δεδομένων εκπαίδευσης εμπλουτίζεται με επιπλέον γνώση, τόσο η προβλεπόμενη εικόνα τείνει να μοιάσει στην εικόνα αναφοράς του μοντέλου με τη βέλτιστη επίδοση.



Εικόνα 4.7: Απεικόνιση ανά εικόνα εκπαίδευσης, των παραγόμενων εικόνων από τα τρία μοντέλα πρόβλεψης που προκύπτουν από εκπαίδευση με τα τρία διαφορετικά σύνολα εκπαίδευσης ακολουθιακά, με αντιστοίχιση σε 3 κατηγορίες-κλάσεις και χρήση του ταξινομητή RF. α)patricia_ref, β)patricia_dataset_1, γ)patricia_dataset_2, δ)patricia_dataset_3, ε)bedford_16_ref, στ)bedford_16_dataset_1, ζ)bedford_16_dataset_2, η)bedford_16_dataset_3, θ)bedford_17_ref, ι)bedford_17_dataset_1, ια)bedford_17_dataset_2, ιβ)bedford_17_dataset_3.

Κεφάλαιο 5: Συμπεράσματα και μελλοντικές ενέργειες

Η παρούσα διπλωματική εργασία εστιάζει στο σχεδιασμό και την υλοποίηση μιας υποδομής με εργαλεία της τελευταίας τεχνολογίας προκειμένου να καλυφθούν οι προκλήσεις και οι δυσκολίες που αντιμετωπίζει η ένταξη των τεχνικών της μηχανικής μάθησης σε ένα παραγωγικό περιβάλλον από ομάδες εξειδικευμένων ατόμων. Στόχος είναι η αυτοματοποίηση και ο προγραμματισμός διαδικασιών που χρειάζεται να επαναλαμβάνονται στο πέρασμα του χρόνου για τη διαρκή βελτίωση των μοντέλων πρόβλεψης και την αυτοματοποιημένη εγκατάστασή τους σε ζωντανές υπηρεσίες. Με εφαρμογή της ανεπτυγμένης υποδομής σε ένα πρόβλημα μηχανικής μάθησης, αυτό της ταξινόμησης των στοιχείων μιας εικόνας υπερήχου που απεικονίζει το θαλάσσιο πυθμένα σε προκαθορισμένο αριθμό κατηγοριών, από το προηγούμενο κεφάλαιο συμπεραίνεται η σημαντικότητα της υλοποίησης μιας τέτοιας υποδομής.

5.1 Συμπεράσματα

Η χρήση των εργαλείων που επιλέχθηκαν αποτελεί μια ικανοποιητική λύση για την αυτοματοποίηση των ενεργειών που απαιτούνται για την ανάπτυξη εφαρμογών μηχανικής μάθησης. Τα εργαλεία Airflow και MLflow σε συνδυασμό με τις υπηρεσίες που εγκαταστάθηκαν για την εύλογη χρήση τους, παρέχουν αρκετά οφέλη στο συγκεκριμένο επιστημονικό πεδίο. Αυτά σχετίζονται τόσο με τη συνεργατικότητα, τη γρήγορη κατανόηση των ήδη ανεπτυγμένων μεθόδων και τη γρήγορη προσαρμογή των ειδικών σε νέα πρότζεκτ. Η γρήγορη ανάπτυξη νέων μεθόδων και αναπαραγωγής των υπαρχόντων είναι σημαντική σε μια ζωντανή υπηρεσία καθώς η διαρκής ενημέρωση του μοντέλου με τα νέα δεδομένα είναι σημαντική για μια αποδοτική υπηρεσία, απαιτώντας γρήγορη αλλά και ασφαλή επανεκπαίδευση των μοντέλων πρόβλεψης ή ανάπτυξη νέων υλοποιήσεων. Η διατήρηση της ιστορικότητας των εκδόσεων και των αρχείων που σχετίζονται με τα μοντέλα που προάγονται στην παραγωγή είναι απαραίτητη ώστε να εξασφαλίζεται η ενδεχόμενη επανεγκατάσταση ενός παρελθοντικού αλλά λειτουργικού μοντέλου. Η οπτικοποίηση των ενεργών διαδικασιών, των βέλτιστων προσεγγίσεων και η δυνατότητα σύγκρισης διαφορετικών υλοποιήσεων είναι ωφέλιμη στη φάση ανάπτυξης μειώνοντας το χρόνο που απαιτείται, συνεπώς και το κόστος. Αυτοματοποιημένες διαδικασίες οδηγούν σε διαφάνεια των ενεργειών που απαιτούνται για την επίλυση ενός προβλήματος ταξινόμησης, αλλά και σε σιγουριά για την εγκυρότητα και την ομαλή λειτουργία των υπηρεσιών που επιλύουν το πρόβλημα. Η φύση της συγκεκριμένης υποδομής στο περιβάλλον που εγκαταστάθηκε και η δυνατότητα ενσωμάτωσής της με λοιπά εργαλεία που επιλύουν άλλες ανάγκες, όπως είναι η κατανεμημένη επεξεργασία και εκπαίδευση ή η κλιμακωσιμότητα των συστημάτων, εφησυχάζουν ότι πρόκειται για μια ορθή επιλογή με πολύ περισσότερες δυνατότητες. Ωστόσο, πρόκειται για ένα πεδίο που δεν έχει ωριμάσει πλήρως, συνεχώς αναπτύσσονται νέες τεχνικές και τεχνολογίες και εξελίσσονται οι υπάρχουσες, πράγμα που ενδεχομένως

μελλοντικά να οδηγήσει σε αντικατάσταση αυτών των εργαλείων με άλλα. Στην παρούσα φάση, είναι αρκετοί οι οργανισμοί που έχουν στραφεί και στρέφονται σε αυτές τις τεχνολογίες, γεγονός που τονίζει την αναγκαιότητα για το σχεδιασμό ανάλογων υποδομών.

5.2 Μελλοντικές ενέργειες

Μελλοντικά, η συγκεκριμένη υποδομή που αναπτύχθηκε στα πλαίσια της διπλωματικής εργασίας, μπορεί να βρει εφαρμογή σε οποιοδήποτε πρόβλημα ταξινόμησης προκύψει. Για την ακρίβεια, θα μπορούσαν να προστεθούν νέες πειραματικές ενότητες, με διαφορετικού είδους προβλήματα ταξινόμησης και διαφορετικές απαιτήσεις, οδηγώντας τελικά στην εγκατάσταση περισσότερων εργαλείων για την κάλυψη διαφορετικών αναγκών. Πιθανά τέτοια εργαλεία είναι τα Optuna και Sigopt, εξειδικευμένα στη ρύθμιση με αυτοματοποιημένο τρόπο, των υπερ-παραμέτρων βαθιών νευρωνικών δικτύων.

Η υποδομή θα μπορούσε να ενισχυθεί με τεχνολογίες που εξειδικεύονται στην παρακολούθηση των παραγωγικών συστημάτων, του χρόνου απόκρισης των μοντέλων ταξινόμησης αλλά και της επίδοσής τους σε βάθος χρόνου. Τέτοια εργαλεία είναι τα Prometheus, Hydrosphere και το Cortex.

Επιπλέον, μια ακόμα κατεύθυνση εξέλιξης της παρούσας εργασίας θα ήταν να ελεγχθεί η κλιμακωσιμότητα των συστημάτων στην πράξη, εφαρμόζοντας συνθήκες καταπόνησης των συστημάτων εξυπηρέτησης (stress test), ώστε να μελετηθούν τα όρια που παρουσιάζουν τα συγκεκριμένα εργαλεία για συγκεκριμένα χαρακτηριστικά πόρων.

Τέλος, κατά την εγκατάσταση των εργαλείων ελήφθησαν αποφάσεις που σχετίζονται με την εσωτερική λειτουργία των εργαλείων όπως είναι ο LocalExecutor του Airflow και το περιβάλλον conda για την εκτέλεση των πειραμάτων. Θα μπορούσε να γίνει έρευνα ως προς το ποιος Executor είναι περισσότερο αποτελεσματικός και εύκολα υλοποιήσιμος σε σχέση με το βαθμό κλιμάκωσης που μπορεί να επιτύχει, για παράδειγμα μεταξύ των Celery και Kubernetes Executors.

Βιβλιογραφικές Αναφορές

Μανωλέσου, Α. (2015). Παραμετρικοί και μη παραμετρικοί έλεγχοι υποθέσεων. In *Μεθοδολογία έρευνας και εισαγωγή στη Στατιστική Ανάλυση Δεδομένων με το IBM SPSS STATISTICS*. Εκδόσεις Κάλλιπος. http://hdl.handle.net/11419/5081

Κύρκος, Ε. (2015). Προεπεξεργασία Δεδομένων. In *Επιχειρηματική ευφυΐα και εξόρυξη δεδομένων*. Εκδόσεις Κάλλιπος. https://repository.kallipos.gr/handle/11419/1226

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., & et Al. (2015). *Distributed training with TensorFlow*. TensorFlow: Large-scale machine learning on heterogeneous systems. https://www.tensorflow.org/guide/distributed_training

The Apache Software Foundation. (2018). *Apache Spark*. Apache Spark - Unified Analytics Engine for Big Data. Retrieved 3 3, 2018, from https://spark.apache.org/

The Apache Software Foundation. (2021). *Apache Airflow*. Apache Airflow. Retrieved 2 22, 2021, from https://airflow.apache.org/

Argo Project Authors. (2020). *Argo*. Get stuff done with Kubernetes. Retrieved 3 2, 2021, from https://argoproj.github.io/

Ask Solem & contributors. (2009-2018). *Introduction to Celery — Celery 5.0.5 documentation*. https://docs.celeryproject.org/en/stable/getting-started/introduction.html

Breck, E., Cai, S., Nielsen, E., Salib, M., & Sculley, D. (2017). The ML Test Score: A Rubric for ML Production Readiness and Technical Debt Reduction. *Proceedings of IEEE Big Data*.

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*, 5-32. https://doi.org/10.1023/A:1010933404324

Cadavid, J. P. U., Lamouri, S., Pellerin, R., Grabot, B., & Fortin, A. (2020, August). Machine learning applied in production planning and control: a state-of-the-art in the era of industry 4.0. *Intelligent Manufacturing*, *31*. 10.1007/s10845-019-01531-7

Carvajal Soto, J. A., Tavakolizadeh, F., & Gyulai, D. (2019). An online machine learning framework for early detection of product failures in an Industry 4.0 context. *International Journal of Computer Integrated Manufacturing*, *32*(4-5 Smart Cyber-Physical System Applications in Production and Logistics), 452-465. https://doi.org/10.1080/0951192X.2019.1571238

Chauhan, K., Jani, S., Thakkar, D., Dave, R., Bhatia, J., Tanwar, S., & Obaidat, M. S. (2020). Automated Machine Learning: The New Wave of Machine Learning. *Innovative Mechanisms for Industry Applications (ICIMIA) 2020 2nd International Conference*, 205-212.

Church, J., Chen, Y., & Rice, D. S. (2008). A Spatial Median Filter for Noise Removal in Digital Images. *Southeastcon, IEEE*.

Cortex Labs. (2021). *Cortex*. Cortex - Deploy machine learning models to production. Retrieved 3 2, 2021, from https://www.cortex.dev/

DoltHub Inc. (2021). *Dolt*. Introduction - Dolt. Retrieved 3 4, 2021, from https://docs.dolthub.com/

DVC. (2021). *DVC*. Data Version Control · DVC. Retrieved 3 3, 2021, from https://dvc.org/

HG Insights. (2021). *Apache Airflow*. Companies Using Apache Airflow, Market Share, Customers and Competitors. Retrieved 3 3, 2021, from https://discovery.hgdata.com/product/apache-airflow

Kang, Z., Catal, C., & Tekinerdogan, B. (2020, November). Machine learning applications in production lines: A systematic literature review. *Computers & Industrial Engineering*, *149*(106773). https://doi.org/10.1016/j.cie.2020.106773

Karbhari, V. (2020). *ML Ops: Data Science Version Control*. medium.com. Retrieved 3 1, 2020, from https://medium.com/acing-ai/ml-ops-data-science-version-control-5935c49d1b76

The Kubeflow Authors. (2018-2021). *Kubeflow*. Kubeflow.

Kuyen, C. (2020). Understanding machine learning production. In *CS 329S: Machine Learning Systems Design*. Stanford.

Li, S. Z., & Jain, A. (2009). Score Normalization. *Encyclopedia of Biometrics*, *Springer*. https://link.springer.com/referenceworkentry/10.1007%2F978-0-387-73003-5_767

Mayr, A., Kißkalt, D., Meiners, M., Lutz, B., Schäfer, F., Seidel, R., Selmaier, A., Fuchs, J., Metzner, M., Blank, A., & Franke, J. (2019, October). Machine Learning in Production – Potentials, Challenges and Exemplary Applications. *Procedia CIRP*, *86*(Towards shifted production value stream patterns through inference of data, models, and technology), 49-54. 10.1016/j.procir.2020.01.035

McHugh, M. L. (2012). Interrater reliability: the kappa statistic. *Biochem Med(Zagreb)*. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3900052/

MLflow Project, a Series of LF Projects, LLC. (2021). *MLflow*. MLflow - A platform for the machine learning lifecycle. Retrieved 2 22, 2021, from https://mlflow.org/

Mutlu, O. (2014). A Practical and Automated Approach to Large Area Forest Disturbance Mapping with Remote Sensing. *PLOS ONE*. https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0078438

NumFOCUS. (2019). *Dask*. Dask: Scalable analytics in Python. Retrieved 2 28, 2021, from https://dask.org/

Pachyderm Inc. (2021). *Pachyderm*. Pachyderm | Version-controlled data science. Retrieved 3 3, 2021, from https://www.pachyderm.com/

Paleyes, A., Urma, R.-G., & Lawrence, N. D. (2021, January 18). Challenges in Deploying Machine Learning: a Survey of Case Studies.

Pierre, R. (2020). *An Apache Airflow MVP: Complete Guide for a Basic Production Installation Using LocalExecutor*. Apache Airflow in Production with LocalExecutor | Towards Data Science. Retrieved 1 18, 2021, from https://towardsdatascience.com/an-apache-airflow-mvp-complete-guide-for-a-basic-production-installation-using-localexecutor-beb10e4886b2

Pouteau, R., Meyer, J.-Y., Taputuarai, R., & Stoll, B. (2012). Support vector machines to map rare and endangered native plants in Paci fi cislands forests. *Ecological Informatics*, *9*(May 2012), 37-46. https://www.sciencedirect.com/science/article/abs/pii/S1574954112000210?via%3Dihub

PyTorch Serve Contributors. (2020). *TorchServe — PyTorch/Serve master documentation*. https://pytorch.org/serve/

Ray. (2021). *Ray*. Fast and Simple Distributed Computing. Retrieved 2 27, 2021, from https://ray.io/

The Ray Team. (2021). *RaySGD*. RaySGD: Distributed Training Wrappers. Retrieved 2 28, 2021, from https://docs.ray.io/en/latest/raysgd/raysgd.html

Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J. F., & Dennisson, D. (Eds.). (2015). Hidden Technical Debt in Machine Learning Systems. *NIPS'15: Proceedings of the 28th International Conference on Neural Information Processing Systems*, *2*, 1808. https://papers.nips.cc/paper/2015/file/86df7dcfd896fcaf2674f757a2463eba-Paper.pdf

Seldon Technologies Ltd. (2021). *Seldon*. Seldon Core Open-source platform for rapidly deploying machine learning models on Kubernetes. Retrieved 2 22, 2021, from https://www.seldon.io/tech/products/core/

Spotify AB. (2012-2019). *Luigi*. GitHub - spotify/luigi: Luigi is a Python module that helps you build complex pipelines of batch jobs. It handles dependency resolution, workflow management, visualization etc. It also comes with Hadoop support built in. Retrieved 3 2, 2021, from https://github.com/spotify/luigi

Tuggener, L., Amirian, M., Rombach, K., Lörwald, S., Varlet, A., Westermann, C., & Stadelmann, T. (2019). Automated Machine Learning in Practice: State of the Art and Recent Results. *6th Swiss Conference on Data Science (SDS)*, 31-36. 10.1109/SDS.2019.00-11

Uber, Hermann, J., & Del Balso, M. (2017). *Uber Engineering*. Meet Michelangelo: Uber's Machine Learning Platform. Retrieved 3 4, 2021, from https://eng.uber.com/michelangelo-machine-learning-platform/

Uber Technologies, Inc. (2018). *Horovod*. GitHub - horovod/horovod: Distributed training framework for TensorFlow, Keras, PyTorch, and Apache MXNet. Retrieved 2 27, 2021, from https://github.com/horovod/horovod

Vapnik, V. (1982). *Estimation of Dependences Based on Empirical Data*. Springer.

Vapnik, V. N. (2000). *The Nature of Statistical Learning Theory*. Springer.

Villar, S. A., Torcida, S., & Acosta, G. (2017). Median Filtering: A New Insight. *Journal of Mathematical Imaging and Vision*, *58*(1), 1-17. https://link.springer.com/referenceworkentry/10.1007%2F978-0-387-73003-5_767

Weichert, D., Link, P., Stoll, A., Rüping, S., Ihlenfeldt, S., & Wrobel, S. (2019). A review of machine learning for the optimization of production processes. *The International Journal of Advanced Manufacturing Technology*, *104*(1889-1902).

Wikipedia. (2019). *Airbnb*. Wikipedia. Retrieved 3 1, 2021, from https://el.wikipedia.org/wiki/Airbnb

Yao, Q., Wang, M., Chen, Y., Dai, W., Li, Y.-F., Tu, W.-W., Yang, Q., & Yu, Y. (n.d.). Taking the Human out of Learning Applications:A Survey on Automated Machine Learning. *Artificial Intelligence*. https://arxiv.org/abs/1810.13306

Zhu, G., & Blumberg, D. G. (2002). Classification using ASTER data and SVM algorithms;The case study of Beer Sheva, Israel. *Remote Sensing of Environment*, *80*, 233-240. https://www.academia.edu/15797348/Classification_using_ASTER_data_and_SVM_algorithms