Optimization of CAR T-Cell Therapies' Supply Chain with the help of Machine Learning

National Technical University of Athens

Stavros Papaiakovou, 05117618

Supervisor: Antonis Kokossis, Professor

Department of Process Analysis and Plant Design

Academic Year 2021-2022

Περίληψη

Οι θεραπείες Τ-λεμφοκυττάρων με Χιμαιρικό Αντιγονικό Υποδοχέα είναι κυτταρικές ανοσοθεραπείες που εμφάνισαν ενθαρρυντικά αποτελέσματα στη θεραπεία τύπων καρκίνου του αίματος όπως η οξεία λεμφοβλαστική λευχαιμία και το επιθετικό λέμφωμα Β-κυττάρων. Οι αυτόλογες θεραπείες Τ-λεμφοκυττάρων με Χιμαιρικό Αντιγονικό Υποδοχέα είναι εξατομικευμένες για τον ασθενή και επομένως ακολουθούν το επιχειρηματικό μοντέλο 1:1, όπου κάθε θεραπεία παράγεται και διανέμεται ξεχωριστά. Προκειμένου να έχουμε μια ικανή αλυσίδα εφοδιασμού που μπορεί να ικανοποιήσει τη ζήτηση και να ανταποκρίνεται στους διάφορους περιορισμούς που συνεπάγονται με τις θεραπείες αυτές όπως χρονικούς περιορισμούς, χρησιμοποιούνται τα μαθηματικά μοντέλα τα οποία ήδη αποτελούν μεγάλο μέρος ως εργαλεία υποστήριξης αποφάσεων. Ωστόσο, λόγω της φύσης αυτών των θεραπειών, η αύξηση στην παραγωγή δημιουργεί ένα δυσεπίλυτο πρόβλημα βελτιστοποίησης της αλυσίδας εφοδοιασμού όταν μοντελοποιείται με το μοντέλο Μικτού Ακέραιου Γραμμικού Προγραμματισμού. Για να καταπολεμηθεί αυτό, εισάγεται μια αποσύνθεση του προβλήματος σε μακροπρόθεσμο σχεδιασμό και βραχυπρόθεσμο προγραμματισμό. Έτσι, χρησιμοποιούνται τεχνικές Μηχανικής Μάθησης για την επίλυση του μακροπρόθεσμου σχεδιασμού της συγκεκριμένης αλυσίδας εφοδιασμού και στη συνέχεια οι λύσεις τους τροφοδοτούνται στο αρχικό μοντέλο Μικτού Ακέραιου Γραμμικού Προγραμματισμού για την επίλυση του προβλήματος προγραμματισμού. Χρησιμοποιήθηκαν τρεις διαφορετικοί αλγόριθμοι Multi-Layer Perceptron, Random Forest και Support Vector Machines, που καταλήγουν να δίνουν πολύ παρόμοια αποτελέσματα. Οι αλγόριθμοι Μηχανικής Μάθησης μειώνουν την πολυπλοκότητα του αρχικού μοντέλου MILP για κάθε δοκιμασμένο σενάριο, μειώνοντας τις διακριτές μεταβλητές και τους περιορισμούς κατά μέσο όρο 63%. Αυτό με τη σειρά του σημαίνει παρόμοια μείωση στη χρήση του κεντρικού επεξεργαστή κατά την επίλυση του προβλήματος βελτιστοποίησης.

Abstract

Chimeric Antigen Receptor (CAR) T-cell therapies are cellular immunotherapies that displayed encouraging results in the treatment of blood cancer types like acute lymphoblastic leukemia and aggressive B-cell lymphoma. Autologous CAR T-cell therapies are patient-specific and therefore follow a 1:1 business model, where every single therapy is produced and distributed uniquely. In order to have a responsive supply chain that can satisfy the demand under the tight time constraints of the supply chain, decision support tools such as mathematical models are already utilized. However, due to the nature of the autologous CAR T-cell therapies, volumetric scale-up is not possible and a scale-out creates an intractable problem to solve. To combat this drawback, a decomposition solution strategy is introduced, where the model is split into long-term planning and short-term scheduling. To this extent, Machine Learning (ML) techniques are used to solve the long-term planning of the supply chain, and then the results are fed to the original Mixed Integer Linear Programming (MILP) model to solve the scheduling problem. Three different ML algorithms were used, namely Multi-Layer Perceptron, Random Forest, and Support Vector Machines, that ended up yielding very similar results. The ML algorithms reduce the complexity of the original MILP model for every scenario tested, by reducing on average the binary variables and constraints by 63%. This in turn means the same reduction in CPU time usage.

Εκτεταμένη Περίληψη

Η θεραπεία Τ-λεμφοκυττάρων με χιμαιρικό υποδοχέα αντιγόνου είναι μια θεραπεία που ανακατευθύνει τα Τ-λεμφοκύτταρα ενός ασθενούς για να στοχεύσουν και να καταστρέψουν καρκινικά κύτταρα. Η υψηλή πιθανότητα υποτροπής που υπάρχει στις θεραπείες καρκίνου, όπως η χημειοθεραπεία και η ακτινοβολία, ώθησε τους ανθρώπους στην αναζήτηση καινοτόμων θεραπειών. Τα γενετικά τροποποιημένα Τ-λεμφοκύτταρα σηματοδοτούν μια νέα εποχή σε αυτόν τον τομέα. Οι θεραπείες Τ-λεμφοκυττάρων με χιμαιρικό υποδοχέα αντιγόνου έχουν αποδειχθεί αποτελεσματικές σε αιματολογικές κακοήθειες των Β-κυττάρων. Σε κλινικές δοκιμές 1ης Φάσης για νέους ασθενείς με οξεία λεμφοβλαστική λευχαιμία Β-κυττάρων (ALL), τα ποσοστά ανταπόκρισης κυμαίνονταν από 69% έως 90%. Επιπλέον, οι αυτόλογες θεραπείες (χρησιμοποίηση των κυττάρων του ασθενή για την ανάπτυξη θεραπείας για τον ίδιο ασθενή) Τ-λεμφοκυττάρων με χιμαιρικό υποδοχέα αντιγόνου έχουν λάβει εγκρίσεις από τον Οργανισμό Τροφίμων και Φαρμάκων (FDA) των ΗΠΑ το 2017 και τον Ευρωπαϊκό Οργανισμό Φαρμάκων (EMA) το 2018. Από το 2017 κυκλοφορούν στην αγορά έξι εγκεκριμένες από τον FDA θεραπείες. Ακόμα κι αν αυτές οι θεραπείες δίνουν πολύ καλά αποτελέσματα, έχουν υψηλό κόστος, το οποίο μπορεί να ξεπεράσει τα 450.000$, όπου περίπου το 30% είναι κόστη αλυσίδας εφοδιασμού. Είναι προφανές λοιπόν ότι η βελτιστοποίηση της εφοδιαστικής αλυσίδας των θεραπειών είναι ζωτικής σημασίας την βιωσιμότητα των θεραπειών.

Μερικοί περιορισμοί των θεραπειών αυτών που δυσκολεύουν την μοντελλοποίηση της εφοδιαστικής αλυσίδας είναι οι εξής. Ο χρόνος είναι ένας κρίσιμος περιορισμός στην εφοδιαστική αλυσίδα των θεραπειών. Αυτές οι θεραπείες έχουν μικρή διάρκεια ζωής. Για μία μη κατεψυγμένη θεραπεία, ο χρόνος μεταφοράς δεν πρέπει να υπερβαίνει το ένα 24ωρο. Επιπρόσθετα, η καταγραφή και ιχνυλάτηση των δειγμάτων καθ' όλη την αλυσίδα εφοδιασμού είναι αναγκαία αφού πρέπει η σωστή θεραπεία να παραδοθεί στον σωστό ασθενή στο τέλος του κύκλου ζωής της θεραπείας. Επομένως, η ταυτοποίηση του δείγματος αλλά και του ασθενή είναι ζωτικής σημασίας. Τέλος, για να μην τεθεί σε κίνδυνο η ποιότητα των θεραπειών, είναι αναγκαίο να διασφαλιστούν οι συνθήκες κατά τη μεταφορά και αποθήκευσή τους.

Όταν η εφοδιαστική αλυσίδα των θεραπειών Τ-λεμφοκυττάρων με χιμαιρικό υποδοχέα αντιγόνου περιγράφεται ως ένα πρόβλημα Μικτού Ακέραιου Γραμμικού Προγραμματισμού

δημιουργείται ένα πρόβλημα βελτιστοποίησης το οποίο γίνεται δυσεπίλυτο για μεγάλη ζήτηση των θεραπειών. Αυτό συμβαίνει αφού οι θεραπείες αυτές, λόγω του αυτόλογού τους χαρακτήρα, ακολουθούν επιχειρηματικό μοντέλο 1:1, με αποτέλεσμα οι περιορισμοί και οι μεταβλητές στο μοντέλο να αυξάνονται με εκθετικό και πιο γραμμικό τρόπο αντίστοιχα. Επίσης, η πολυπλοκότητα ενός μοντέλου Μικτού Ακέραιου Γραμμικού Προγραμματισμού μπορεί να καθοριστεί περίπου με τον αριθμό των μεταβλητών. Αφού σε κάθε κόμβο υπάρχουν δύο διαφορετικοί κλάδοι που μπορούν ακολουθηθούν, τότε η πολυπλοκότητα και συνάμα η δυσκολία επίλυσης γίνεται της τάξης $O(2^n)$, όπου n είναι οι μεταβλητές του προβλήματος. Έτσι, για περιπτώσεις όπου η ζήτηση των θεραπειών ξεπερνάει τις 1700 θεραπείες το χρόνο, ο επιλυτής CPLEX, ο οποίος χρησιμοποιείται για να λύσει το πρόβλημα Μικτού Ακέραιου Γραμμικού Προγραμματισμού, δεν καταφέρνει να βρει λύση λόγω ανεπαρκής μνήμης του ηλεκτρονικού υπολογιστή.

Με σκοπό να ξεπεράσουμε το εμπόδιο αυτό και να μικρύνουμε την πολυπλοκότητα του προβλήματος, γίνεται αποσύνθεση του προβλήματος σε μακροπρόθεσμο σχεδιασμό και βραχυπρόθεσμο προγραμματισμό. Ακολούθως τα δύο προβλήματα λύνονται ξεχωριστά, πρώτα ο μακροπρόθεσμος σχεδιασμός και αργότερα ο βραχυπρόθεσμος προγραμματισμός. Στα πλαίσια αυτής της διπλωματικής, το πρόβλημα του μακροπρόθεσμου σχεδιασμού λύνεται με τεχνικές μηχανικής μάθησης με αποτέλεσμα τη μείωση των μεταβλητών και περιορισμών του προβλήματος Μικτού Ακέραιου Γραμμικού Προγραμματισμού. Σε αυτή την εφοδιαστική αλυσίδα το πρόβλημα του μακροπρόθεσμου σχεδιασμού είναι η εύρεση των κατάλληλων εγκαταστάσεων παραγωγής των θεραπειών που χρειάζονται για να ικανοποιήσουν την ζήτηση.

Υπάρχουν έξι βασικά στάδια στην ανάπτυξη των προτεινόμενων αλγορίθμων μηχανικής μάθησης: 1) Συγκέντρωση των απαραίτητων δεδομένων από την επίλυση του προβλήματος Μικτού Ακέραιου Γραμμικού Προγραμματισμού 2) Επιλογή των αλγορίθμων μηχανικής μάθησης που θα χρησιμοποιηθούν 3) Προ επεξεργασία των δεδομένων για κάθε αλγόριθμο 4) 'Κούρδισμα' των παραμέτρων των μοντέλων μηχανικής μάθησης 5) Εκπαίδευση των αλγορίθμων μηχανικής μάθησης 6) Χρήση των μοντέλων μηχανικής μάθησης για λύση του μακροπρόθεσμου σχεδιασμού της

αλυσίδας εφοδιασμού και έτσι τη μείωση του χώρου επίλυσης του προβλήματος Μικτού Ακέραιου Γραμμικού Προγραμματισμού.

Πρώτα απ' όλα, για να εκπαιδευτεί ένας αλγόριθμος μηχανικής μάθησης, είναι απαραίτητο να υπάρχουν τα απαραίτητα δεδομένα. Στο τρέχον πρόβλημα, τα δεδομένα αποκτώνται χρησιμοποιώντας το μοντέλο Μικτού Ακέραιου Γραμμικού Προγραμματισμού. Με στόχο να ληφθούν αρκετά δεδομένα για την εκπαίδευση των αλγορίθμων έπρεπε να κατασκευαστούν διαφορετικά σενάρια ζήτησης, να λύθει το πρόβλημα Μικτού Ακέραιου Γραμμικού Προγραμματισμού και στη συνέχεια να παρθούν τα απαραίτητα δεδομένα που χρησιμοποιήθηκαν στην εκπαίδευση των αλγορίθμων μηχανικής μάθησης.

Ακολούθως, επιλέχθηκαν τρεις διαφορετικοί αλγόριθμοι μηχανικής μάθησης για την επίλυση του σχεδιασμού με σκοπό να συγκριθούν και να βρεθεί ποιος δίνει τα καλύτερα αποτελέσματα. Οι τρεις διαφορετικοί αλγόριθμοι που χρησιμοποιήθηκαν είναι οι Multi-Layer Perceptron (MLP), Random Forest και Support Vector Machines (SVM). Αυτοί επιλέχθηκαν επειδή χρησιμοποιούνται ευρέως για προβλήματα ταξινόμησης και διαφέρουν στον τρόπο εκπαίδευσής τους. Ο MLP είναι ένα τεχνητό νευρωνικό δίκτυο που παράγει ένα σύνολο εξόδων από ένα σύνολο εισόδων αφότου τα δεδομένα περάσουν από μία συνάρτηση ενεργοποίησης στους υπολογιστικούς ή κρυφούς νευρώνες. Για την εκπαίδευση του, ο MLP χρησιμοποιά backpropagation, όπου μία συνάρτηση σφάλματος υπολογίζει τη διαφορά μεταξύ της πρόβλεψης του νευρωνικού δικτύου και της αναμενόμενης εξόδου του, αφότου ένα παράδειγμα εκπαίδευσης έχει διαδοθεί μέσω του δικτύου. Ο Random Forest αποτελείται από έναν μεγάλο αριθμό μεμονωμένων Decision Trees που τελικά λειτουργούν ως ένα σύνολο. Κάθε μεμονωμένο δέντρο βγάζει μια πρόβλεψη και η πρόβλεψη με τις περισσότερες ψήφους γίνεται η πρόβλεψη του μοντέλου. Ο SVM είναι ένας αλγόριθμος μηχανικής μάθησης που βασίζεται στην ιδέα της εύρεσης υπερεπιπέδων που χωρίζουν καλύτερα ένα σύνολο δεδομένων στις διάφορες κλάσεις που το απαρτίζουν.

Για την εκπαίδευση των αλγορίθμων μηχανικής μάθησης χρειάζεται να οριστούν προφανώς και οι εξαρτημένες μεταβλητές στις οποίες θα εκπαιδευτούν τα μοντέλα. Για να επιλεχθούν οι μεταβλητές αυτές πρώτα μελετήθηκε το πρόβλημα της αλυσίδας

εφοδιασμού και οι μεταβλητές εισόδου στο πρόβλημα Μικτού Ακέραιου Γραμμικού Προγραμματισμού. Έτσι αρχικά επιλέχθηκαν οι ακόλουθες εξαρτημένες μεταβλητές: η συνολική ζήτηση θεραπειών για ένα έτος, το σημείο όπου γίνεται η αφαίρεση των T-λεμφοκυττάρων και τέλος η συνολική χωρητικότητα όλων των εγκαταστάσεων παραγωγής των θεραπειών. Αργότερα, με σκοπό την βελτίωση των μοντέλων μηχανικής μάθησης, προστέθηκε ακόμα μία μεταβλητή, η μεγαλύτερη στιγμιαία ζήτηση του κάθε σεναρίου.

Πριν την προ επεξεργασία των δεδομένων έγινε παρουσίαση των δεδομένων σε διάφορες μορφές. Η παρουσίαση των δεδομένων είναι σημαντική στη μηχανική μάθηση, επειδή δίνει στον χρήστη μια καλή ιδέα για το πώς κατανέμονται τα δεδομένα και μπορούν να εντοπιστούν τυχόν ανωμαλίες. Αυτό με τη σειρά του επιτρέπει την καλύτερη προ επεξεργασία των δεδομένων και συνεπώς καλύτερη απόδοση των μοντέλων. Στην συγκεκριμένη περίπτωση φάνηκε να υπάρχει κακή κατανομή των δεδομένων αφού οι πρώτες τρεις εγκαταστάσεις παραγωγής θεραπειών επιλέγονται πολύ περισσότερες φορές από τις υπόλοιπες τρεις, με αποτέλεσμα να μην λειτουργούν σωστά. Για να καταπολεμηθεί το παραπάνω πρόβλημα χρησιμοποιήθηκαν δύο τεχνικές για εξισορρόπηση των δεδομένων, SMOTE (Synthetic Minority Over-sampling Technique) για παραγωγή επιπλέον δεδομένων και Tomek Links για διαγραφή δεδομένων στα οποία επιλέχθηκε η $2^{η}$ εγκατάσταση. Επίσης, κατά την προ επεξεργασία των δεδομένων έγινε και κλιμάκωσή τους μεταξύ 0 και 1 μόνο για τα μοντέλα MLP και SVM, αφού για το μοντέλο Random Forest η κλιμάκωση δεν είναι αναγκαία για τη σωστή του λειτουργία.

Ακολούθως έγινε 'κούρδισμα' των παραμέτρων των μοντέλων. Για να γίνει αυτό, πρώτα δημιουργήθηκε ένα πλέγμα με διάφορες τιμές των παραμέτρων για κάθε αλγόριθμο. Ο συνδυασμός των τιμών των παραμέτρων που δίνει τα καλύτερα αποτελέσματα για τα δεδομένα που του παρέχονται, δηλαδή που δίνει την καλύτερη βαθμολογία F1, μετά από 5-fold Cross-Validation, χρησιμοποιήθηκε για την εκπαίδευση των μοντέλων.

Τέλος και τα τρία μοντέλα εκπαιδεύτηκαν για δύο διαφορετικές περιπτώσεις. Η πρώτη περίπτωση ήταν χωρίς περιορισμό στις εγκαταστάσεις που είναι δυνατό να επιλεχθούν, δηλαδή μπορεί να επιλεχθούν όλες οι έξι εγκαταστάσεις. Αυτό δίνει μία πιο αποκεντρωμένη προσέγγιση της εφοδιαστικής αλυσίδας όπου πολλές εγκαταστάσεις παραγωγής βρίσκονται

πιο κοντά στους ασθενείς.
Η δεύτερη περίπτωση είναι μία πιο κεντρωμένη προσέγγιση, όπου μία ή δύο εγκαταστάσεις χειρίζονται την παραγωγή όλων των θεραπειών. Άρα σε αυτή τη περίπτωση το μοντέλο είναι περιορισμένο να επιλέγει το μέγιστο δύο από τις έξι εγκαταστάσεις παραγωγής των θεραπειών.

Αφού τα μοντέλα μηχανικής μάθησης εκπαιδευτούν, χρησιμοποιούνται για την εύρεση των των κατάλληλων εγκαταστάσεων παραγωγής των θεραπειών. Μετά, τα δεδομένα εισόδου του προβλήματος Μικτού Ακέραιου Γραμμικού Προγραμματισμού, που έχουν σχέση με τις εγκαταστάσεις που δεν επιλέχθηκαν, διαγράφονται και έτσι ενημερώνεται το μοντέλο. Καθώς τώρα θα υπάρχουν λιγότερες επιλογές για τις εγκαταστάσεις το μοντέλο Μικτού Ακέραιου Γραμμικού Προγραμματισμού έχει συνολικά λιγότερες μεταβλητές και περιορισμούς σε σχέση με προηγουμένως και έτσι μπορεί να λυθεί σε λιγότερο χρόνο.

Στην περίπτωση όπου δεν υπάρχει περιορισμός στις εγκαταστάσεις παρατηρείται για όλα τα σενάρια ζήτησης μείωση στις μεταβλητές και στους περιορισμούς. Συγκεκριμένα για τα περισσότερα σενάρια που δοκιμάστηκαν υπάρχει μείωση 50-65% στους περιορισμούς και τις διακριτές μεταβλητές που μπορεί να φτάσει έως και το 83% σε κάποια σενάρια αλλά και όσο χαμηλά όσο 16% για ένα σενάριο. Όσον αφορά τις λύσεις του προβλήματος Μικτού Ακέραιου Γραμμικού Προγραμματισμού, η μόνη διαφορά, πριν και μετά την μείωση των μεταβλητών, βρίσκεται μόνο στα σενάρια με ζήτηση 1000 θεραπειών το έτος όπου υπάρχει μία μικρή αύξηση 5% στο κόστος των θεραπειών.

Στην περίπτωση όπου τα μοντέλα μπορούν να επιλέξουν ως δύο εγκαταστάσεις από τις έξι τα αποτελέσματα είναι ακόμα καλύτερα. Όπως και πριν, σε όλα τα σενάρια το μοντέλο Μικτού Ακέραιου Γραμμικού Προγραμματισμού καταλήγει σε εφικτή λύση μετά από την μείωση του χώρου με τη βοήθεια μηχανικής μάθησης. Ακόμα οι λύσεις αυτές είναι οι ίδιες σε όλα τα σενάρια που δοκιμάστηκαν με τις λύσεις που βρίσκει το μοντέλο Μικτού Ακέραιου Γραμμικού Προγραμματισμού όταν βρίσκει τις εγκαταστάσεις το ίδιο. Οπότε σε αυτή τη περίπτωση οι αλγόριθμοι μηχανικής μάθησης προβλέπουν πάντα τις σωστές εγκαταστάσεις που χρειάζονται για να ικανοποιήσουν την ζήτηση σε κάθε σενάριο. Επιπρόσθετα, η μείωση στους περιορισμούς και στις μεταβλητές στο μοντέλο Μικτού Ακέραιου Γραμμικού Προγραμματισμού όταν αφαιρούνται οι εγκαταστάσεις που δεν χρειάζονται, φτάνει στα 83% για μερικά

σενάρια, ενώ στα υπόλοιπα οι παράμετροι μειώνονται κατά 65%. Λόγω της μείωσης αυτής, συνεπάγεται και παρόμοια μείωση στο χρόνο που χρειάζεται για να επιλυθεί το πρόβλημα βελτιστοποίησης της αλυσίδας εφοδοιασμού, αφού όταν τα μοντέλο μηχανικής μάθησης εκπαιδευτούν, δεν χρειάζονται πολύ μικρό χρόνο για να κάνουν μια πρόβλεψη.

Ακόμη, και στις δύο περιπτώσεις η μηχανική μάθηση, λόγω μείωσης των μεταβλητών, βοηθάει στη εύρεση λύσης του προβλήματος Μικτού Ακέραιου Γραμμικού Προγραμματισμού σε σενάρια όπου η ζήτηση είναι τόσο μεγάλη και λόγω ανεπαρκής μνήμης, ο επιλυτής δεν έφτανε σε λύση.

Τα τρία μοντέλα μηχανικής μάθησης τελικά καταλήγουν να δίνουν παρόμοια αποτελέσματα μεταξύ τους οπότε συμπεραίνεται πως ο τρόπος που εκπαιδεύουμε μηχανική μάθηση δεν παίζει ιδιαίτερο ρόλο στα αποτελέσματα. Ωστόσο, ο Random Forest μπορεί να εκπαιδευτεί πολύ γρηγορότερο από τους άλλους δύο αλγόριθμους, ενώ από την άλλη το νευρωνικό δίκτυο, λόγω του μεγάλου αριθμού νευρώνων, χρειάζεται πολύ περισσότερο χρόνο.

Acknowledgments

First of all, I would like to thank from the bottom of my heart the people who helped me complete this dissertation. The present dissertation was prepared in the context of my undergraduate studies at the School of Chemical Engineering of the National Technical University of Athens under the supervision of Mr. A. Kokosis in the field of Process Design and Optimization. I would like to say a big thank you to my supervising professor A. Kokosis, as well as Dr. Maria Papathanasiou and PhD candidate Niki Triantafyllou of Imperial College London for the guidance, advice, and support they provided me throughout the elaboration of this diploma. I was lucky to have them all supervising my work, who despite their limited time, showed interest in my work, answered me immediately and at any time to any question and difficulty I had, and trusted me to prepare this diploma. Moreover, this dissertation would not have been completed without the constant support, help and love from my family and friends, who stood by me and gave me strength throughout, and supported me in all the difficulties I had to face.

Table of Contents

1. Introduction

1.1. CAR T-Cell Therapies

Chimeric antigen receptor (CAR) T cell therapy is a cellular therapy that redirects a patient's T cells to specifically target and destroy tumor cells.

The high probability of relapse or even resistance to non-surgical cancer treatments like chemotherapy and radiation drove people in the search of innovative treatments. Genetically engineered T cells marked a new era in that area [1]. CAR T cell therapies have shown to be efficient in hematological B-cell malignancies. In Phase 1 clinical trials for young patients with B-cell acute lymphoblastic leukemia (ALL), response rates ranged from 69% to 90% after T-cells were engineered to express the CAR specific to the B-lymphocyte antigen CD19. [2] [3]. Furthermore, autologous CAR T-cell therapies have received approvals from the U.S. Food and Drug Administration (FDA) (2017) and the European Medicines Agency (EMA) (2018). Since 2017, six FDA approved therapies are in the market, Kymriah, Yescarta, Tecartus, Breyanzi, Abecma and Carvykti, where in the first 4 the target antigen is CD19 and for the other 2 the target antigen is BCMA [4].

Even if these therapies show great potential, they come with a high cost, which can be more than $450,000 in some cases [5], with approximately 30% of them being supply chain costs [6]. It is evident then, that the optimization of the therapies' supply chain is crucial for them to be financially sustainable and competitive [4].

1.1.1. Lifecycle of CAR T-Cell Therapies

The production of CAR T-cells requires several carefully performed steps, and quality control testing is performed throughout the entire protocol. The main steps of a typical CAR T cell therapy lifecycle are: (a) leukapheresis, (b) manufacturing, (c) Quality Control, (d) therapy administration [5]. Leukapheresis in autologous cell therapies (use of a patient's own cellular material to treat disease [7]), takes place in specialized clinical

centers and involves removing the blood from the patient's body, separating the leukocytes, and then returning the blood to the circulation [8], [9].

After a sufficient number of leukocytes have been harvested, the leukapheresis product is transferred to the manufacturing site, where it undergoes a series of processing steps (enrichment, activation, genetic modification, expansion, formulation, and cryopreservation) until the final product is ready to be shipped to the hospital for administration. CAR T cell therapy manufacturing is the lengthiest and most important step of the lifecycle. Enrichment is where the T cells are washed and concentrated via counterflow centrifugal elutriation, a procedure that separates cells by size and density. T-Cell activation is achieved with stimulation of the T cells with anti-CD3 and anti-CD28 monoclonal antibodies coated paramagnetic beads [2], [10] or plate-bound antibodies.

Next, T-lymphocytes are genetically modified with viral transduction using lentiviral vectors, gammaretroviral vectors or other delivery methods [9], [11] to express the cell surface CAR molecule [12]. Then, in order to grow a large number of cells for clinical use, cell expansion happens, where the cell culture can reach a volume of 5L before it is washed and concentrated for blood infusion in the patient [9].

Before the therapy is taken to the clinical site to be administrated to the patient, Quality Control (QC) takes place. This process ensures that the final product has the critical quality attributes and can either happen in the manufacturing site or a different facility. Lastly, after the patient undergoes a treatment of lymphodepleting chemotherapy, the cryopreserved CAR T-cell therapy is thawed and administrated to the patient [5], [13].

As was mentioned these therapies use the patient's own cellular material. To guarantee that the right therapy is manufactured and delivered to the right patient, the sample and the patient have to be identified and linked and then each product is tracked efficiently at every stage of the process through a chain-of-custody system [5]. The autologous nature of the therapies also renders volumetric scale-up impossible since every therapy is essentially a different batch process. Scale-out solutions then, can be used as an alternative [5], [14].

1.2. Optimization

1.2.1. Basic Concepts

Optimization is used daily in many areas of our lives. Optimization can be found a lot in the industry, when manufacturing a product, process and service design, programming, and resource allocation, either in the manufacturing process or transport. Generally, where there are resources that can be distributed in a variety of ways, optimization is used to find the best way to allocate those resources. That is, the goal of optimization is to provide the user who uses it with the necessary information, which he or she may not otherwise have, to make the best possible decision. In other words, optimization is used to find the most correct and best solution to a problem through a set of possible solutions without, of course, violating the constraints that the problem may have [15]. The vector of the decision variables of the problem which does not violate the constraints of the problem and minimizes a cost function or maximizes a fitness function is the optimal solution to the problem. These two aforementioned functions are called objective functions. An optimization problem can be either multi-criteria i.e., it consists of many objective functions (Multi-Objective Optimization Problem), or single-criteria i.e., only one (Single Objective Optimization Problem) [16]. In case a problem is multi-criteria, sometimes the objective functions that make it up are competitive, that is, the improvement of one makes the others worse. In this case, a set of equally good solutions is calculated which dominate the other solutions to the problem, that is, they have a better result than the others in terms of each function. All of these solutions are called the Pareto Front [17]. First, it is useful to express the function of an optimization-minimization problem.

$$\min \vec{f}(\vec{x}) = \min [f_1(\vec{x}), f_2(\vec{x}), \dots, f_N(\vec{x})] \quad (1\text{-}1)$$

In equation 1.1, $\vec{f}$ symbolizes the vector of objective functions, $\vec{x}$ is denotes the vector of the decision variables of the set problem n and $f_1(\vec{x}), f_2(\vec{x}), \dots, f_N(\vec{x})$ are the individual objective functions. In case the problem is subject to restrictions they are presented in equations 1-2.

$$h_j(\vec{x}) = 0 \qquad j = 1, .., k \qquad \text{Eq. 1-2}$$

$$g_i(\vec{x}) \leq 0 \qquad i = 1, .., m$$

With $g_i(\vec{x})$ are the m functions of inequality constraints and $h_j(\vec{x})$ are the k functions of equality constraints. As mentioned above the vector $\vec{x} = [x_1, x_2, \dots, x_p]$, contains the decision variables of the problem. The vector whose combination of decision variables gives the best value to the respective objective function is considered the solution to the problem. These variables can be either discrete or continuous and each of them is characterized by a threshold and a ceiling x_{pl} and x_{pu} respectively. These limits then, define the search space for the solution to the problem. Such constraints, like the limits of decision variables, are called side constraints and do not fall into the same category as the main constraints of the problem mentioned above. In the general case, the constraint functions and the objective functions can be either linear or non-linear [16], [18].

1.2.2. Optimization methods

Initially, optimization methods can be classified into three main categories. Deterministic methods are based on mathematical programming, stochastic methods which are based on artificial intelligence and machine learning algorithms, and lastly hybrid methods that combine two or more different optimization algorithms. Deterministic methods utilize the analytical properties of a problem to arrive at a global solution [19]. Some classes of problems that can be solved with deterministic approaches are linear programming (LP) where convex linear problems are solved, non-linear programming (NLP) where convex nonlinear and non-convex continuous problems are solved, Mixed Integer Linear Programming (MILP) where non-convex discrete linear problems are solved, as well as Nonlinear Mixed Integer Programming (MINLP) where non-convex discrete nonlinear problems are solved [19], [20], [18]. Some deterministic solvers that can solve the problems mentioned above are CPLEX where a branch and cut and a dynamic search algorithm are used [21], GUROBI where cutting planes algorithm and heuristics and search

techniques are used [22], XPRESS that uses Branch and Bound framework [23]. The allocation of the different problems with the solution methods can be seen in figure 2.

Stochastic methods are often using principles of natural selection and are suitable for the search of nonlinear, large spaces [24]. Additionally, they are general-purpose algorithms and they treat the problems as a "black box". Although stochastic methods seem to be effective in solving specific problems, the solutions are not always optimal [25], [26].

In hybrid methods a stochastic method is usually used first, to explore the space and find the optimal area and then using a mathematical/deterministic method as a local optimizer for better use of the area and therefore improving the results of the stochastic method.

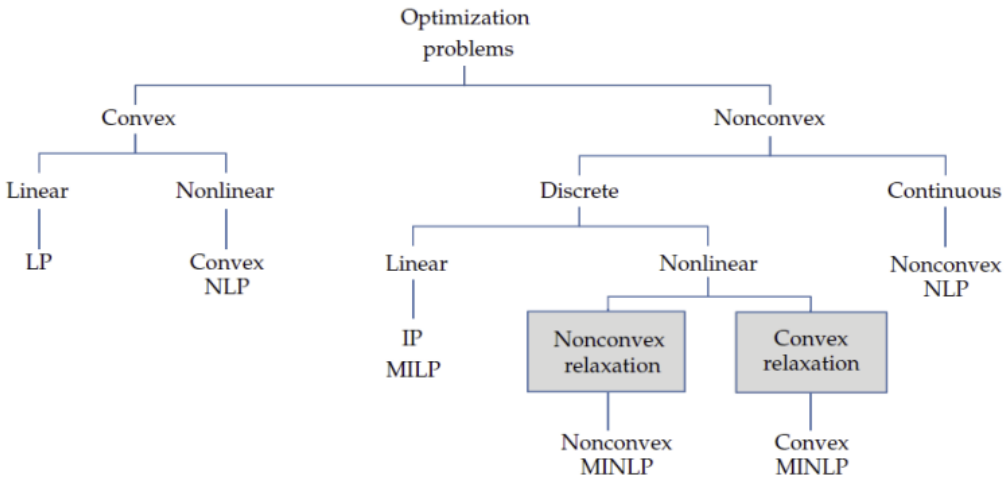


Figure 1: Problems that can be solved using Deterministic methods [19]

1.2.3. Size Reduction Techniques

For large-scale optimization problems, deterministic methods, due to the complexity of the problem, may have a difficult time finding the optimal solution and if they do, they could take a lot of time to do so [19]. A logical approach then is to try and reduce the size of the problem. Decomposition techniques allow to break a problem into smaller ones and then solve them separately, either in parallel or sequentially [27].

One approach for decomposition is to relax a set of complicating constraints. Some examples of this approach are Lagrangian relaxation and cutting-plane methods. On the other hand, Benders' decomposition is a technique that, considering the restrictions of the problem, keeps the value of a set of variables fixed [28]. These are some traditional decomposition approaches that are based on improving the bounds of the problem [29].

Alternatively, recently, meta-heuristic algorithms have been gaining a lot of attention [29]. Meta-heuristic techniques are based on powerful heuristic algorithms and can find close to optimal solutions by guiding heuristic methods over the search space of the problem. Meta-heuristic algorithms examples are Genetic Algorithm, Particle Swarm Optimization, and Variable Neighbourhood Search [6] [30].

In this work, the problem is decomposed in two subproblems where the first one is solved using Machine Learning (ML) techniques and the other using a MILP model.

1.3. Supply Chain

With the term supply chain, we mean the chain or the alignment of different firms that are involved in one way or the other (raw materials, transportation, merchants) in manufacturing a product or providing services for the final consumer. Supply chain was described by Christopher M. as "a network of organizations that are involved, through upstream and downstream linkages, in the different processes and activities that produce value in the form of products and services in the hands of the ultimate consumer" [31]. The supervision and control of the processes in supply chains like planning, manufacturing, and distribution, is called supply chain management.

1.3.1. Supply Chain Management

Supply chain design and management are crucial, as a good design structure can result in considerable savings in both the investment costs as well as the operational costs [32]. However, accomplishing this is not straightforward. A lot of operating factors and constraints like raw materials acquisition, manufacturing and inventory capacities, transportation modes and times, and even expiring dates for food or pharmaceutical products, must be considered when managing a supply chain [33]. Adding to these

factors, changes in demand or adding new products to the market, sum up to an extremely difficult problem to manage for the lowest costs. Optimizing these processes, managers have to have some guidance from decision support tools [34].

1.3.2. Supply Chain Optimization

Optimizing supply chains is really challenging as was mentioned before. First of all, modeling a supply chain is a difficult challenge. One has to develop tractable models that can be solved in a reasonable time range and can depict all the parts of the supply chain. Secondly, multiscale optimization that characterizes the supply chain is another issue. The decision support tool has to be able to make decisions both for long-term, strategic and tactical decisions, like sourcing and planning respectively, and for short-term like scheduling. Uncertainties, such as sudden changes in demand or the availability of raw materials, are again difficult to account for in models [5] [33] [34]. Last but not least is the algorithmic and computational challenge, which includes the development and solving of efficient algorithms [34].

1.3.3. Pharmaceutical Supply Chains

Managing the supply chain of pharmaceutical products can be a way more difficult task than other products and many pharmaceutical companies are still importantly a long way from effectively satisfying market demands [35]. Certain characteristics distinguish pharmaceutical products from the rest of the consumer products. Firstly, it takes a lot of time and money for the completion of the clinical trials that a pharmaceutical product has to pass. However, even after the approval of a product, there can be a high manufacturing time compared to consumer products. Moreover, the demands for pharmaceutical products are often uncertain and can arise problems with capacity, as well as the pharmaceutical industry is vulnerable to disasters, like technological malfunctions or strikes, which can lead to unsatisfying the demand [33]. Adding to these, due to their nature, pharmaceutical products have a limited shelf-life. All of the above contribute to a challenging, to say the least, supply chain [36].

For personalized medicine supply chains and particularly for CAR T-cell therapies' supply chain, the above challenges remain and more are added. One really important aspect of

the supply chain model is sample tracking. The tracking required here is bi-directional as it must be ensured that the right therapy will be delivered to the right patient at the end of the product cycle. Therefore, sample and patient identification are vital and errors in tracking traceability are not allowed. Time is also a critical factor in CAR T-cell therapies' supply chain. These therapies have tight shelf-life windows so processing, storage, and shipping times must be tightly controlled because for a fresh, not frozen, therapy the laytime for transport should not exceed a 24-hour window. Lastly, in order not to risk the quality of the therapies, it is imperative to ensure the conditions during transfer and storage so equipment validation is necessary [5], [2].

Due to the aforementioned reasons, a decentralized manufacturing approach, where several manufacturing facilities are situated closer to patients, instead of a centralized approach, where a single facility handles the production, is attractive. The patient-specific nature of these therapies hinders volumetric scale-up and therefore parallel manufacturing lines are needed. Moreover, the simpler logistics for a final product that a decentralized approach is offering, as well as that this supply chain network is more resilient to unforeseen events, and the fact that fresh products can make it to the patients faster, makes the decentralized approach a preferred choice for personalized medicine in general [37].

Models are being developed to optimize these supply chains. A. Bernardi et al. developed a MILP model to provide candidate solutions concerning the location, number, and capacity of manufacturing sites, storage duration, and the most suitable mode of transport for each therapy [14]. Moschou et al. presented a MILP investment planning model to propose patient-centric, cost-efficient supply chain network structures [38]. Karakostas et al. considered a General Variable Neighbourhood Search algorithm to identify key elements impacting design and operation in CAR T cell supply chains [6].

2. Background and State-of-the-Art

The current supply chain model is currently based on the products' lifecycle that was described in section 1.1.2 and can be seen in Figure 2. Given its sensitive and autologous nature, the decision-making process is based on specific handling conditions and patient schedules, thus making the configuration of the optimal supply chain particularly complicated [39]. The current state-of-the-art model that is used to solve the optimization problem of this supply chain is a Mixed Integer Linear Programming (MILP) formulation that provides candidate solutions with respect to the location, number, and capacity of manufacturing sites and the most suitable mode of transport. The MILP model is used to assess the supply chain network performance under different time constraint scenarios on the total return time of each therapy [14]. In particular, the model gets inputted the capital and variable costs (capital investment and fixed variable cost for every manufacturing facility, unit transport costs, quality control costs), the transport time for the two transport modes (j1, j2) from the leukapheresis sites to manufacturing facilities and then to the hospital, the manufacturing facilities' capacities and the demand profile. Then minimizing the total cost as shown in equation (A-1), while satisfying the constraints in equations (A-15) to (A-29), (A-33), (A-35), the model outputs the time and transport mode that a therapy is leaving a leukapheresis site (A-7), then a manufacturing facility (A-10) and lastly arriving at the hospital (A-12). It also gives the manufacturing (A-2) and transport costs (A-4) for every therapy, as well as their total return time (A-34).

The supply chain network superstructure that includes 4 nodes leukapheresis site, manufacturing site, Quality Control, and hospital can be seen in Figure 3. The complexity of the supply chain network, with the combination of the one-to-one business model that autologous CAR T-cell therapies follow and the time restrictions that are imposed due to the sensitivity to temperature and stress, short shelf-life CAR T-cells create a really complex, computationally expensive optimization problem. Small-scale instances of the CAR T-cell supply chain model, that is a low annual demand, can be solved with exact methods using commercial solvers such as CPLEX. However, as the demand increases and is expected to increase in the following years, it can be easily observed that the MILP model becomes computationally intractable very quickly.

Consequently, there is obviously a need to reduce the complexity of the problem in order to make it feasible for larger instances. Reducing the size of the problem can be done with various techniques that were mentioned before.

3. Purpose of this work

The purpose of this work is to try and train machine learning algorithms to replace the planning part of the supply chain model, which is to predict the best manufacturing sites that need to be established to satisfy the demand, in order to reduce the overall complexity of the problem and therefore the computational cost. The solution of the data models is used to remove the manufacturing facilities that are not needed and then the MILP problem can be solved faster for optimal solutions. Three different ML algorithms were trained for a decentralized approach where all six manufacturing sites are possible to get chosen. Next, the same algorithms were trained with a constrain, a limit of a max of two manufacturing sites that can be established, which suggests a more centralized approach where one or two facilities handle the production of the therapies. An analysis of the results and a comparison of the three ML algorithms will follow.

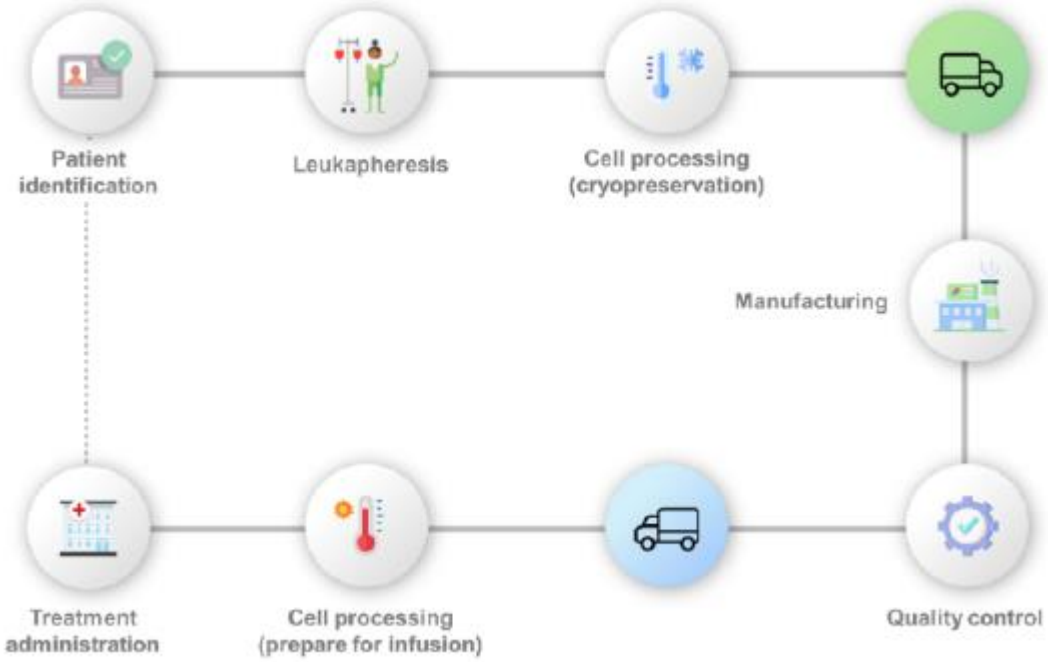


Figure 2: Schematical supply chain of CAR T-Cell therapies

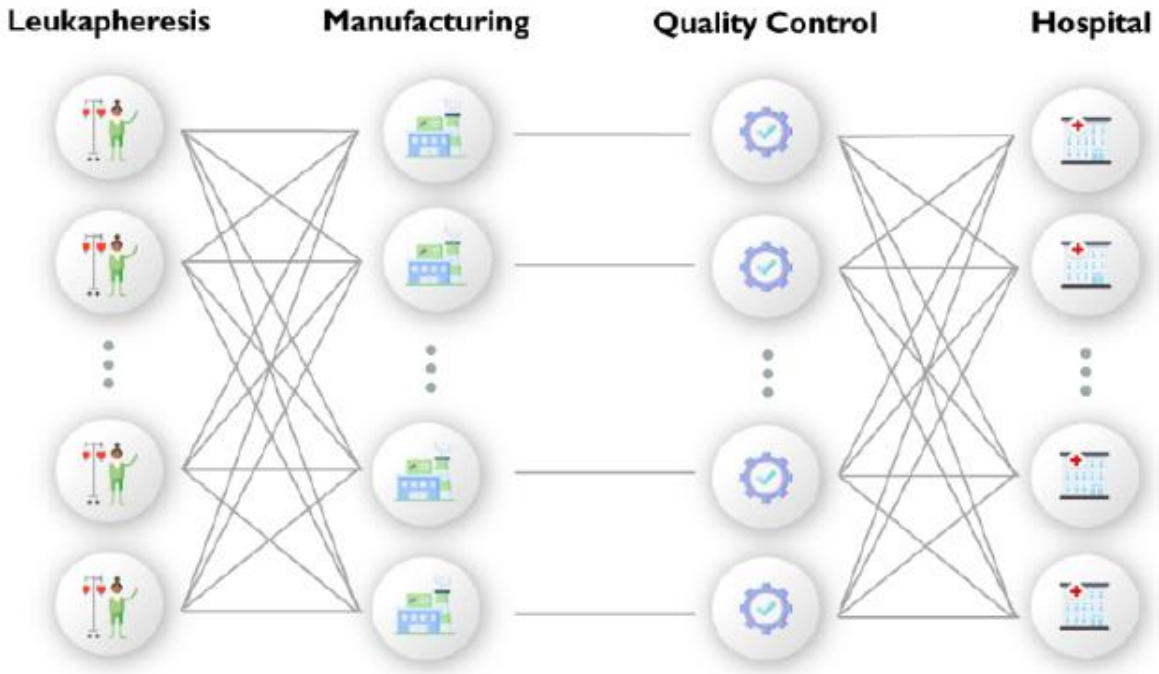


Figure 3: CAR T cell supply chain network with 4 nodes: (a) leukapheresis site, (b) manufacturing site, (c) Quality Control, and (d) hospital

4. Literature Review

Machine learning techniques are very appealing for numerous applications. Supply chain optimization is one of those applications. Even though until now, supply chain optimization is done using mathematical models (MILP, MINLP, etc.) [40], machine learning is now getting a lot of attention in this field and is on the verge of improving and revolutionizing the supply chain management [41]. Research for different supply chains utilizes machine learning for optimizing their processes in different parts of the supply chain (transportation, manufacturing, etc.) [42]. In [43], D. Goettsch et al. used ML to reduce the possible storage depot locations in biomass co-firing supply chain, in coordination with a stochastic MILP model. D. N. Duc and N. Nananukul in [44] trained an artificial neural network in a real case study from Central Vietnam for a biomass supply chain problem to determine the optimal facility locations. Demand forecasting is another application of ML in supply chains. In [45] and [46] ML algorithms were used to predict the demand and thus improve the efficiency of the supply chain in a steel manufacturer and the leading household and personal care manufacturer respectively. ML techniques are also in research for pharmaceutical supply chains. A data model was used to find the efficiency of a pharmaceutical company's supply chain in what orders are delivered on time [47]. Lastly, in [48], four different ML algorithms were trained from optimal solutions of mathematical models, for decision-making on order and transshipment quantities in a network of hospitals.

The ML algorithms used for the above applications can be seen in Table 1.

Table 1: Literature review for the use of Machine Learning in supply chain optimization

Reference	Application	Model
D. Goettsch et al. (2020)	Select Potential Depot Locations for the Supply Chain of Biomass Co-Firing	Logistic Regression, Random Forrest, Multi-Layer Perceptron Neural Network
D. N. Duc and N. Nananukul (2020)	Planning biomass plant locations using a Hybrid Methodology Based on Machine Learning and MIP	Artificial Neural Network
J. Feizabadi (2022)	Demand forecasting	ARIMAX, Artificial Neural Network
M. A. Villegas, D. J. Pedregal and J. R. Trapero (2018)	Demand forecasting	Support Vector Machines
C. Han and Q. Zhang (2021)	Optimization of supply chain efficiency management	BP Neural Network
Kartal, Hasan, et al (2016)	Multi-attribute inventory classification	Naïve Bayes, Artificial Neural Network, Support Vector Machine
B. Abbasi, T. Babaei, Z. Hosseinifard, K. Smith-Miles and M. Dehghani (2020)	A case study in blood supply chain management	Multi-Layer Perceptron Neural Network, k-Nearest Neighbor, Random Forest, Classification and Regression Tree
Manyathi Sakhile (2017)	Robustness of supply chain management system	Decision Tree

5. Methodology

In this section, the process of training the ML algorithms and then using these trained models to solve the planning of the supply chain to reduce the complexity of the MILP model is described.

Six main stages followed in the development of the proposed algorithms.

1) Gathering the necessary data from the MILP state-of-the-art algorithm
2) Choosing the ML algorithms to be used
3) Pre-processing the data for each algorithm
4) Tuning their hyperparameters
5) Training the ML algorithms
6) Using the trained data models for the problem reduction

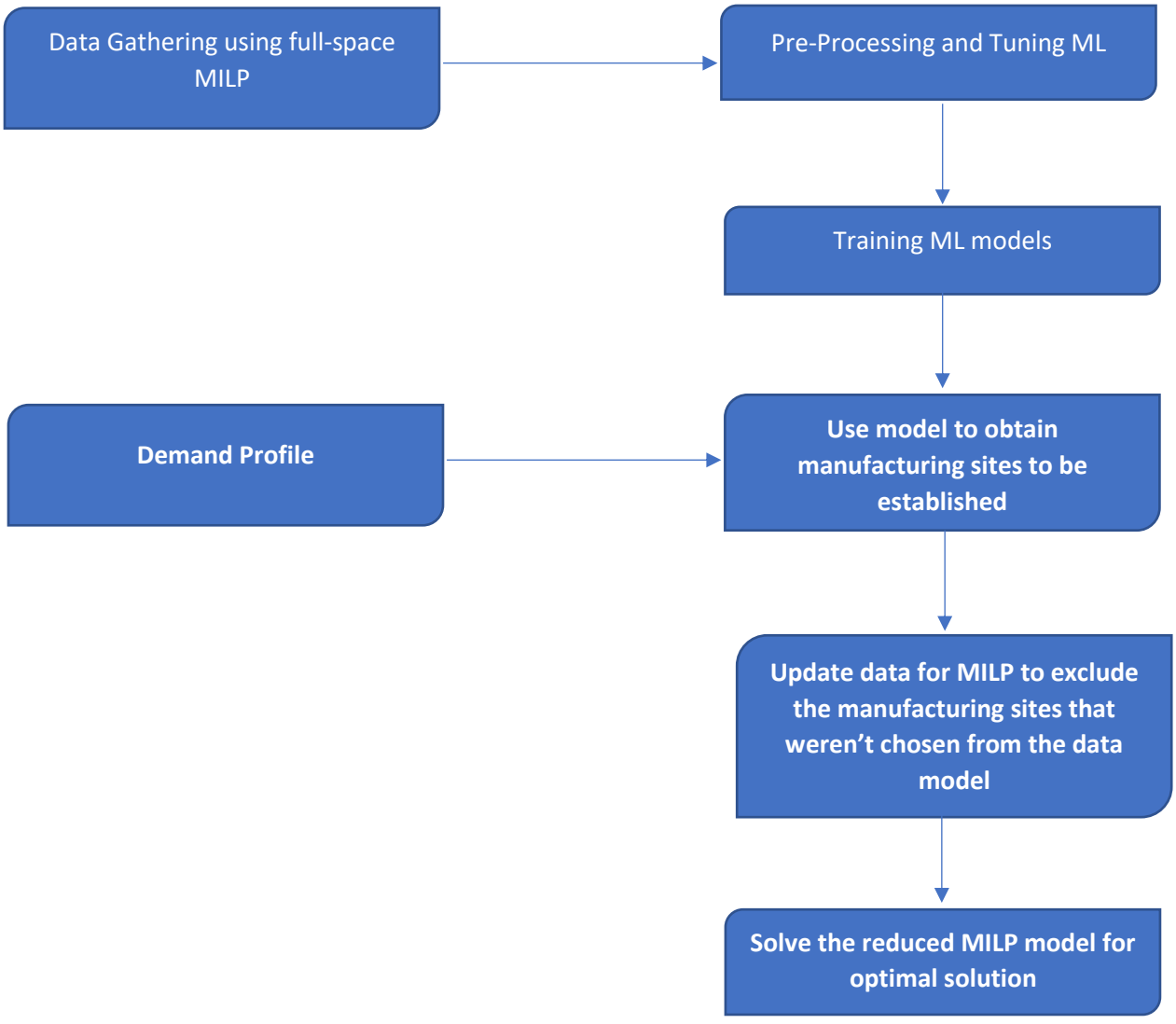


Figure 4: Flowchart for the proposed algorithm

5.1. Data Gathering:

First of all, in order to train a machine learning algorithm, it is necessary to acquire the needed data. In the current problem, the data is acquired using the MILP model proposed by [14] With the aim of getting enough data to train the algorithms different demand scenarios had to be constructed and then ran through the MILP model. Then, when an optimal solution was found, the data was processed for the purpose of training the ML algorithms.

The data was gathered after 26 different demand scenarios for 100 patients/year, 22 scenarios for 200 patients/year, 6 scenarios for 300 patients/year, 6 scenarios for 500 patients/year, 6 scenarios for 640 patients/year, 3 scenarios for 750 patients/year, 5 scenarios for 1000 patients/year, 3 scenarios for 1200 patients/year, 2 scenarios for 1500 patients/year, 1 scenario for 1700 patients/year and finally 2 scenarios for 2000 patients/year. These resulted in having 8663 examples for the ML algorithms to train.

All the scenarios were run for the first quarter of the year i.e. for 200 patients/year, the scenario list 50 patients arriving at a leukapheresis site on the first 90 days of the year. Another point of the scenarios is that the leukapheresis site can serve up to 8 people a day. At the current stage of this work, is worth mentioning that the transportation method is restricted to a single transportation (j2).

5.2. Machine Learning Algorithms

5.2.1. Multi-Layer Perceptron

Multi-layer Perceptron (MLP) is a supervised learning algorithm that learns a function $f(\cdot):R^m \rightarrow R^o$ by training on a dataset, where m is the number of dimensions for input and o is the number of dimensions for output. Given a set of features $X=x_1, x_2,..., x_m$, and a target y, it can learn a non-linear function approximator for either classification or regression. It is different from logistic regression, in that between the input and the output

layer, there can be one or more non-linear layers, called hidden layers. Figure 5 shows a two hidden layer MLP [49].

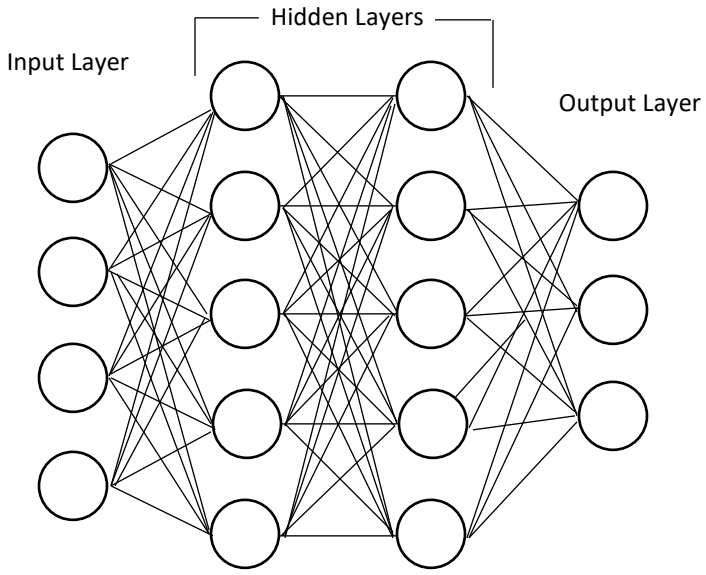


Figure 5: Two hidden layers MLP

The leftmost layer, known as the input layer, consists of a set of neurons $\{x_i \mid x_1, x_2, \ldots, x_m\}$ representing the input features. Each neuron in the hidden layer transforms the values from the previous layer with a weighted linear summation $w_1x_1+w_2x_2+\ldots+w_mx_m$, followed by a non-linear activation function $g(\cdot):R\rightarrow R$. The output layer receives the values from the last hidden layer and transforms them into output values [49]. Multi-Layer Perceptron has its advantages and disadvantages. One really strong point of neural networks is their ability to train and learn non-linear models, due to the non-linear activation functions of the hidden layers. MLP can also be trained in real-time, that is any data added in the training set is used to train the classifier at the time of the addition. Some drawbacks of MLP follow. First of all, MLP is sensitive to feature scaling, so using a scaler is basically mandatory. Moreover, MLP with hidden layers has a non-convex loss function where there exists more than one local minimum. Therefore, different random weight initializations can lead to different validation accuracy. MLP also due to backpropagation has high time complexity. Lastly, MLP, like a lot of other classifiers, requires tuning to a number of

hyperparameters such as the number of hidden neurons, layers, and iterations [49] , [50], [51].

Mathematical Formulation

As a supervised learning algorithm, the classifier is first given a set of training examples (x_1,y_1), (x_2,y_2), …, (x_n,y_n), where $x_i \in R^n$ and $y \in R^l$. For one hidden layer neural network the MLP learns the following function:

$$f(x) = w_2 * g({w_1}^T * x + b1) + b2 \quad (3\text{-}1)$$

where $w_1 \in R^m$ represents the weights of the input layer, $w_2 \in R^h$ represents the weights of the hidden layer, and b1, $b_2 \in R$ represents the bias added to the hidden and output layer.

g(·) is the activation function that the hidden layers pass through and there are a lot of different ones that anyone can choose from. The most popular are:

- **Linear activation function**: $g(z) = z$
- **Rectified Linear Units (ReLU) —** With ReLU, we ensure our output doesn't go below zero (or negative). Therefore if *z* is greater than zero, the output remains *z*, else if *z* is negative, the output is zero: $g(z) = \max(0, z)$
- **Tanh:** $g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$
- **Sigmoid activation:** $g(z) = \frac{1}{1+e^{-z}}$

To obtain the outputs, f(x) passes through one activation function. For binary classification, f(x) passes through the sigmoid function to obtain output values between zero and one. A threshold, often set to 0.5, would assign samples of outputs larger or equal to 0.5 to the positive class, and the rest to the negative class.

If there are more than two classes, f(x) itself would be a vector with a size equal to the classes. Instead of passing through the sigmoid function, it passes through the softmax function [49], [52].

- Sigmoid Function: $softmax(z)_i = \frac{\exp(zi)}{\sum_i^k \exp(z_i)}$

where zi represents the i_{th} element of the input to softmax, which corresponds to class i, and k is the number of classes. The result is a vector containing the probabilities that sample x belongs to each class. The output is the class with the highest probability.

Starting from initial random weights, multi-layer perceptron (MLP) minimizes the loss function by repeatedly updating the weights. After computing the loss, a backward pass propagates it from the output layer to the previous layers, providing each weight parameter with an update value meant to decrease the loss. The loss is given by the Average Cross-Entropy

$$Loss = \frac{1}{2n}\sum_{i=0}^{n}(y_i \log\big(f(x_i)\big) + (1 - y_i)\log\Big(1 - \big(f(x_i)\big)\Big)) + \frac{a}{2n}\|w\|_2^2 \quad (3\text{-}2)$$

where $a\|w\|_2^2$ is an L2-regularization term (aka penalty) that penalizes complex models; and α>0 is a non-negative hyperparameter that controls the magnitude of the penalty.

The update of the weights happens with gradient descent like below:

$$w^{i+1} = w^i - \varepsilon \nabla Loss_w^i \quad (3\text{-}3)$$

where i is the iteration step, and ε is the learning rate with a value larger than 0.

The algorithm stops when it reaches a preset maximum number of iterations; or when the improvement in loss is below a certain, small number [49], [52].

5.2.2. Random Forest

Random forest consists of a large number of individual decision trees that operate as an ensemble. The goal of ensemble methods is to combine the predictions of several base estimators built with a given learning algorithm to improve generalizability/robustness over a single estimator. The prediction of the ensemble is given as the averaged prediction of the individual classifiers [49], [53].

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation [53].

Like all the other algorithms DTs and Random Forest have their advantages and disadvantages. Firstly, DTs and Random Forest are easy to interpret and understand as can be visualized. Secondly, Random Forest is one of few algorithms that doesn't need a lot of data preprocessing [54]. It can give really good results with no scaling or dummy variables. Furthermore, the cost for training each tree is logarithmic in the number of examples in the training set [49]. On the other hand, a negative of DTs and Random Forest is that predictions of decision trees are neither smooth nor continuous, but piecewise constant approximations as seen in figure 6. Therefore, they are not good at extrapolation [55]. Lastly, DTs are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree [53]. Random Forest can mitigate this problem, although global optimality is not guaranteed [49].

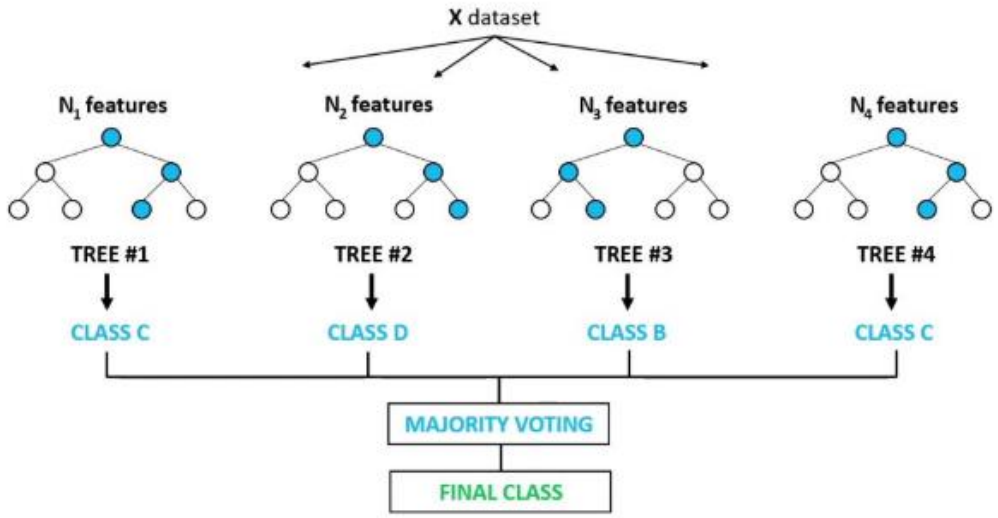


Figure 6: Random Forest Classifier visualization

Mathematical Formulation of Decision Trees

As the MLP, the classifier is first given a set of training examples (x_1,y_1), (x_2,y_2), …, (x_n,y_n), where $x_i \in R^n$ and $y \in R^l$. For every split that happens in a node m, there is a threshold t_m that is used to partition the data Q_m with n_m samples in two subsets $Q_m^{left}(\theta)\ and\ Q_m^{right}(\theta)$ where θ is the candidate split that consists of a feature j and the threshold.

$$Q_m^{left}(\theta) = \{(x,y)|x_j \leq t_m \tag{3-4}$$

$$Q_m^{right}(\theta) = \frac{Q_m}{Q_m^{left}(\theta)}$$

The choice of the best candidate split is taken based on the function $G(Q_m, \theta)$ and the parameters (θ*) are selected in order to minimize G.

$$G(Q_m,\theta) = \frac{n_m^{left}}{n_m} * H\left(Q_m^{left}(\theta)\right) + \frac{n_m^{right}}{n_m} * H\left(Q_m^{right}(\theta)\right) \tag{3-5}$$

where H is the impurity function and can be measured with the "Gini" or "Entropy" function.

- Gini: $H(Q_m) = \sum_k p_{mk}(1 - p_{mk})$
- Entropy: $H(Q_m) = -\sum_k p_{mk}\log(p_{mk})$

and $p_{mk} = \frac{1}{n_m}\sum_{y \in Q_m} I(y = k)$

The splitting process happens until the maximum allowable depth is reached ($n_m < min_{samples}$) or $n_m = 1$. [49]

5.2.3. Support Vector Machine

Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression, and outliers detection. SVMs are based on the idea of finding hyperplanes that best divides a dataset into classes. There are a lot of different methods that one can implement SVMs for classification purposes like C-Support Vector

Classification, Nu-Support Vector Classification and Linear-Support Vector Classification. SVC and NuSVC are similar methods, but accept slightly different sets of parameters and have different mathematical formulations. On the other hand, LinearSVC is a faster implementation of Support Vector Classification for the case of a linear kerne [49]. SVMs are generally effective in high-dimensional spaces and have good results in the cases where the number of features is greater than the number of examples [56]. However, if the number of features is much greater than that of the samples, overfitting the data becomes common and regularization of the data is important. A huge advantage that the SVMs have is their versatility because the decision function can be specified using different Kernel functions, which are customizable. Unfortunately, SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation [57] [58].

For multi-class classification, like this problem, SVC and NuSVC follow the "one-versus-one" approach for choosing the class. This means that the classifier compares the probability that a class is the right one between two classes and after the comparison happens between all the classes, the one that 'won' everyone is the predicted class. This means that $(n_{classes} * (n_{classes} - 1))/2$ have to be constructed [49].

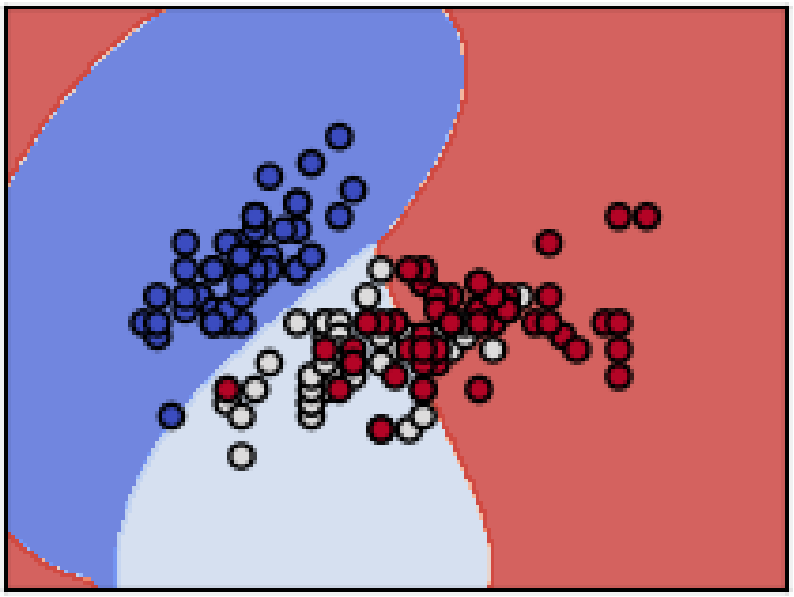


Figure 7: Example of splitting the data into classes using SVC

Mathematical Formulation of SVC

The mathematical formulation of SVMs is based on the intuition that a good separation is achieved by the hyper-plane that has the largest distance to the nearest training data points of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.

Just as the other classifiers first the classifier is given a training set which consists of $x_i \in R^p$ and a vector $y \in \{-1,1\}^n$, i=1,…,n. As was mentioned before the prediction follows a one-vs-one approach so the classification happens between just two classes. To find the best hyperplanes SVC solves the following problem:

$$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i \qquad (3\text{-}6)$$

$$s.t. \, y_i (w^T f(x_i) + b \geq 1 - \zeta_i$$

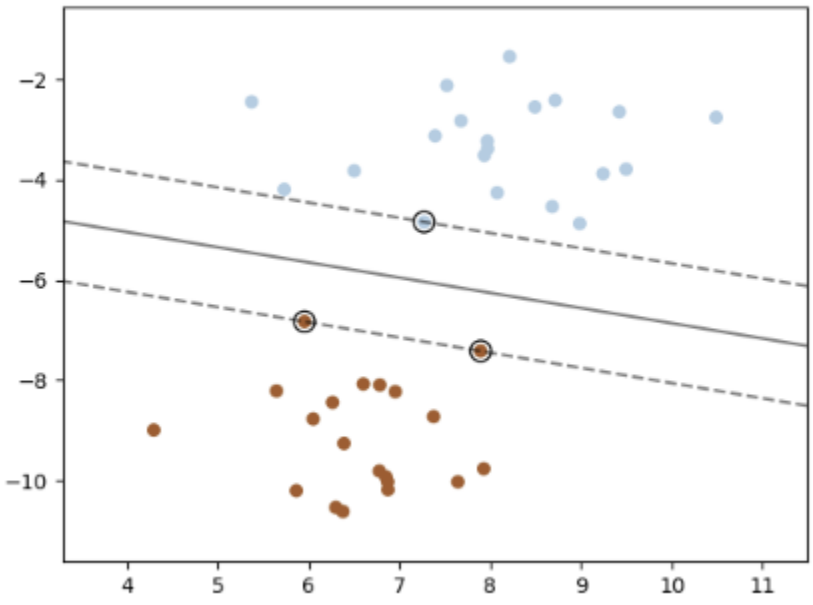


Figure 8: Example of a linearly separated problem with three support vectors

where w are the weights for the training examples and ζ is the distance between a wrong predicted point to its correct hyperplane and C is a regularization parameter and basically controls the strength of the penalty when a sample is misclassified.

The above problem can be rewritten:

$$\min_{a} \frac{1}{2} a^T Q a - e^T a \quad (3\text{-}7)$$

$$s.t. y^T a = 0$$

where α_i are the dual coefficients and $0 < \alpha_i < C$, i=1,...,n, e is the vector of all ones and Q is an n by n semidefinite matrix. Q is given by:

$$Q_{ij} = y_i y_j K(x_i, x_j) \quad (3\text{-}8)$$

$$K(x_i, x_j) = f(x_i)^T f(x_j)$$

and K is the kernel function that measures similarity between any pair of inputs (x_i, x_j). Lastly, the predicted value is given by

$$\hat{y} = sign(\sum_{i=1}^{n} w_i y_i K(x_i, x_j) \quad (3\text{-}9)$$

Some common kernels that are used are:

- Linear Kernel: $K = \langle x_i, x_j \rangle$
- Polynomial Kernel: $K = (\gamma * \langle x_i, x_j \rangle + r)^d$, where d is the degree of the polynomial and gamma is a parameter that defines how much influence a single example has
- Radial basis function (rbf) kernel: $K = \exp\left(-\gamma * \|x_i - x_j\|^2\right)$
- Sigmoid kernel: $K = \tanh(\gamma \langle x_i, x_j \rangle + r)$ [49]

Input features

The input features for the machine learning algorithms have to be associated with the result that the algorithms produce. At this point of the work the features selected are:

- The total demand for CAR-T cell therapies for a year i.e. 200 patients/year
- The total capacity of all manufacturing facilities available at the time a patient arrives at a leukapheresis site
- The leukapheresis site where a sample has been taken and left for a manufacturing facility.

The result for each example is the manufacturing facility that the MILP model selected for the current therapy to be manufactured.

5.3. Visualization

Visualization is significant in machine learning because it gives a good idea of how the data is distributed and finds any abnormalities or imbalances. This, in turn, allows for a better preprocessing of the data and therefore better performance of the models. After plotting and analyzing figures 9-12, one can make some observations. First of all, we can conclude from both figures the data is imbalanced. Manufacturing site 'm2' is picked a lot more times than any other facility on both the constrained and unconstrained models. This is due to the fact that in scenarios with more than 1000 patients per year, a manufacturing facility with more capacity is needed and thus it gets picked, as well as those scenarios contribute more examples than the rest. The imbalance continues between manufacturing facilities 1, 3 and 4, 6. This happens because the model prioritizes manufacturing facilities 1 and 3 and then picks 4 or 6 only after the former are full. Lastly, manufacturing facility 5 is never picked on the unconstained model with the current data since for 2000 patients per year there is no need for establishing a second 31-capacity manufacturing facility.

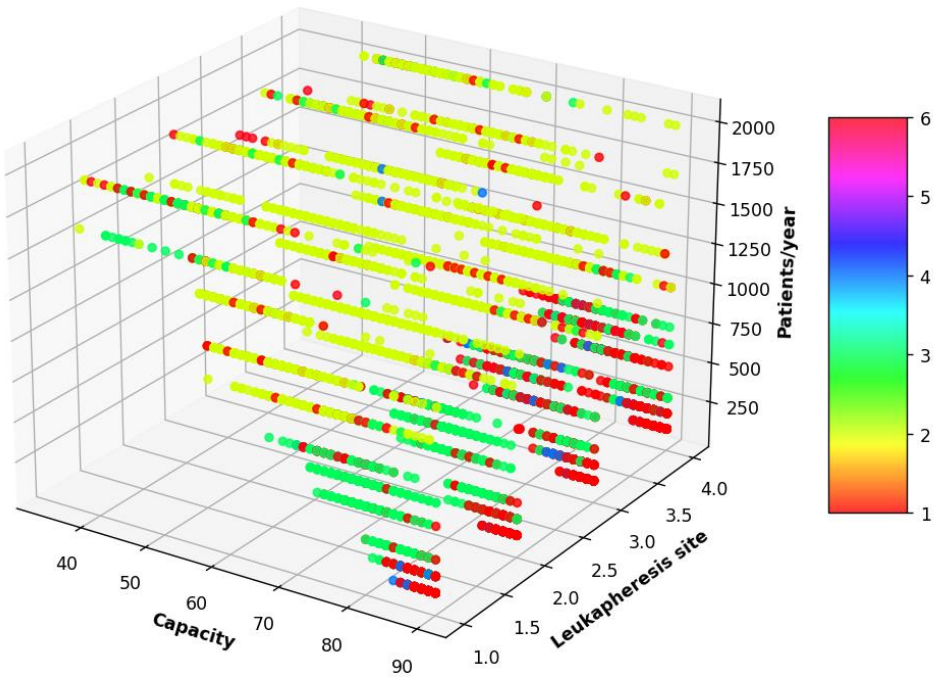


Figure 9: 3-D representation of the data gathered in respect of the input features for the unconstrained model

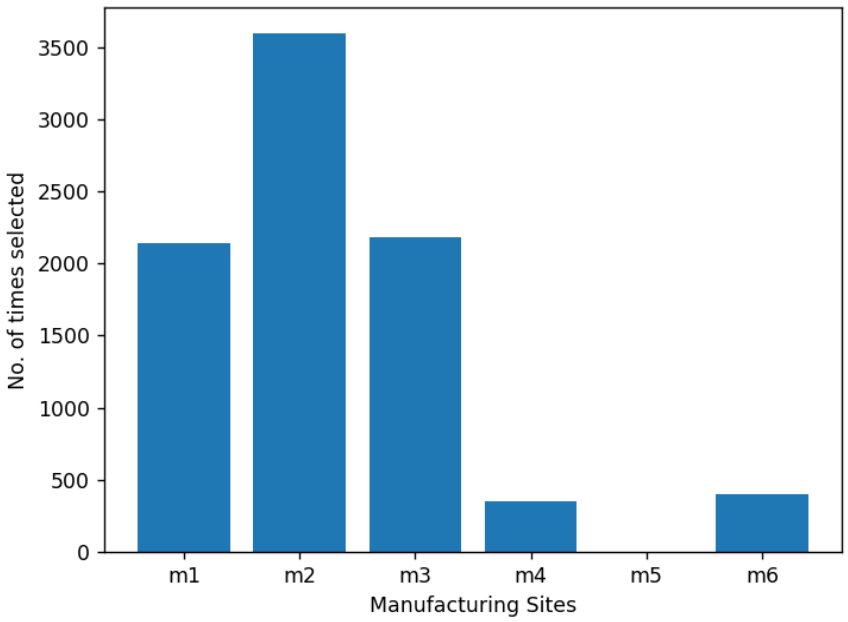


Figure 10: Number of times each manufacturing site was selected for each therapy in all the scenarios for the unconstrained model

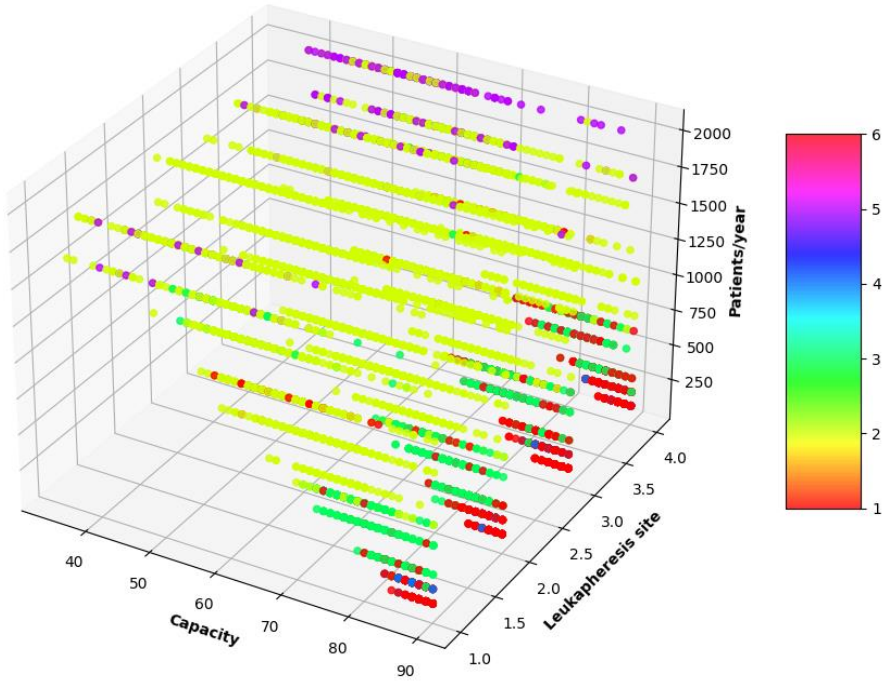


Figure 12: 3-D representation of the data gathered in respect of the input features for the constrained model

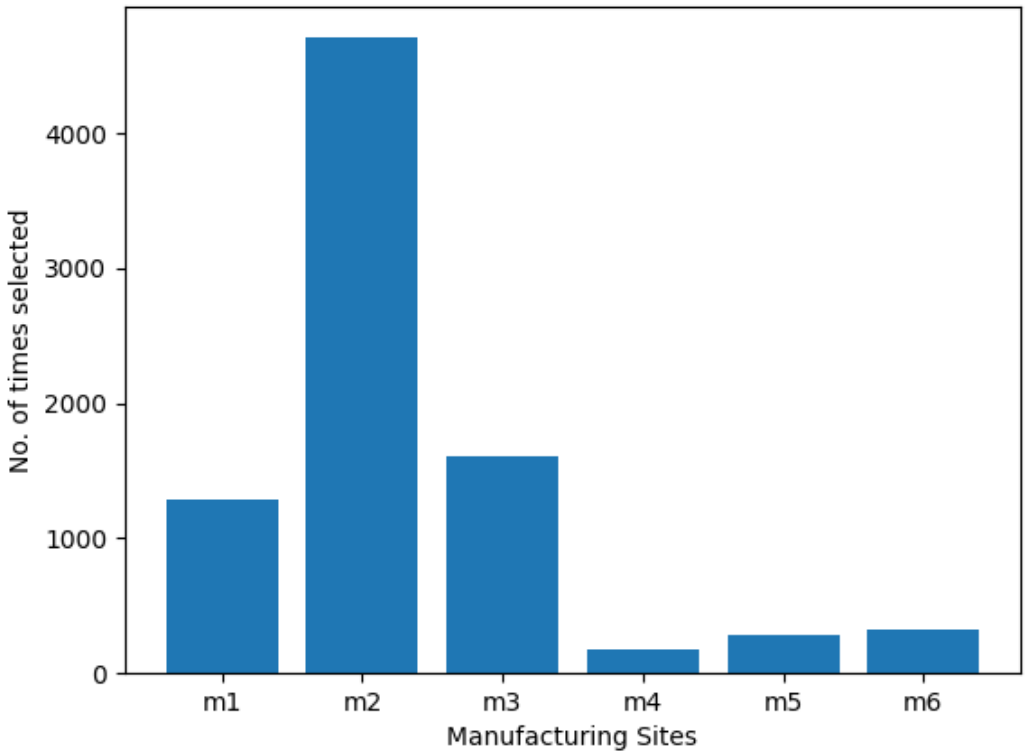


Figure 11:Number of times each manufacturing site was selected for each therapy in all the scenarios for the constrained model

5.4. Preprocessing

5.4.1. Scaling

During sections 3.2.1. and 3.2.2. it was mentioned that scaling the data is crucial for MLP and SVM classifiers. The two common approaches to bringing different features onto the same scale are normalization and standardization.

Normalization refers to rescaling and squeezing the features to a range of [0, 1], which is a special case of min-max scaling. The scaling formula is shown below [59] [60]

$$x_{norm}^{i} = \frac{x^{i} - x_{min}}{x_{max} - x_{min}} \tag{3-10}$$

The standardization technique is used to center the feature columns at mean 0 with a standard deviation of 1 so that the feature columns have the same parameters as a standard normal distribution. Unlike Normalization, standardization maintains useful information about outliers and makes the algorithm less sensitive to them in contrast to min-max scaling, which scales the data to a limited range of values. Here is the formula for standardization.

$$x_{std}^{i} = \frac{x^{i} - \mu_{x}}{\sigma_{x}} \tag{3-11}$$

where μx is the mean of each feature over all data points and σx is the square root of the variance.

For this work normalization of the features was chosen.

5.4.2. Data balancing

Often real-world data sets are composed of imbalanced data, that is there is an unequal distribution of the classes. Imbalanced data pose a challenge for predictive modeling as most of the machine learning algorithms used for classification were designed around the assumption of an equal number of examples for each class. This results in models that have poor predictive performance, specifically for the minority class. This is shown in the data in this particular problem that we examine. There are three methods for balancing

one's data. Oversampling, which replicates examples of the minority class to achieve a more balanced distribution, and under-sampling, which aims to balance the data set by eliminating examples of the majority class, or a combination of the two [61]. In order to fight the aforementioned imbalance, a method that combines oversampling using the method SMOTE and cleaning of the data using the method of Tomek links is used. The results after the oversampling/cleaning are shown in figure 5.

Synthetic Minority Over-sampling Technique (SMOTE) is an oversampling technique that does not just replicate other examples to over-sample the minority class. SMOTE generates synthetic minority examples. For every minority example, its k nearest neighbors of the same class, often set to 5, are calculated, then some examples are randomly selected from them according to the over-sampling rate. After that, new synthetic examples are generated along the line between the minority example and its selected nearest neighbors. By doing that, better and more general regions can be learned for the minority class. [62] [63]

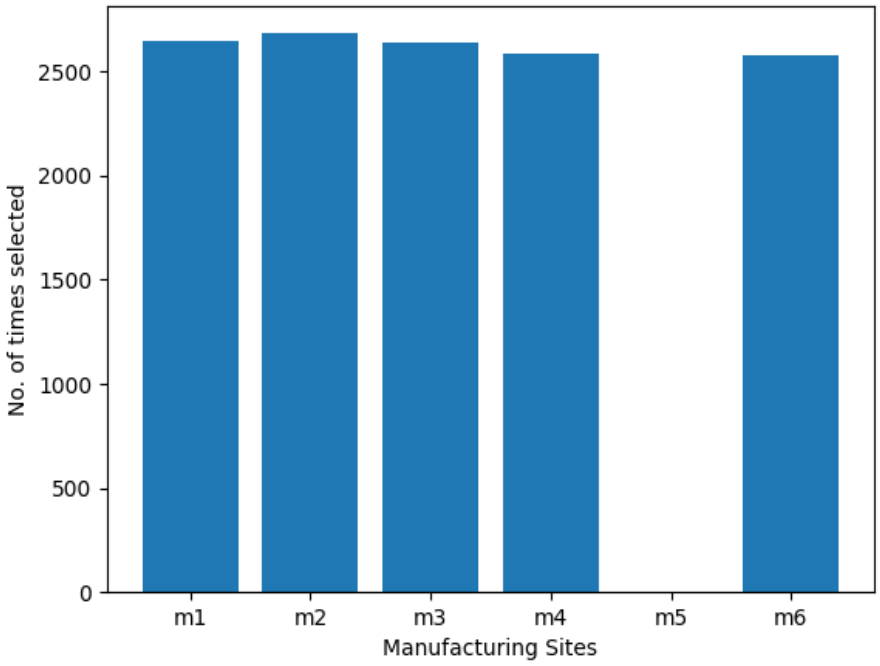


Figure 13: Number of times each manufacturing site was selected for each therapy in all the scenarios after oversampling the training set for unconstrained model

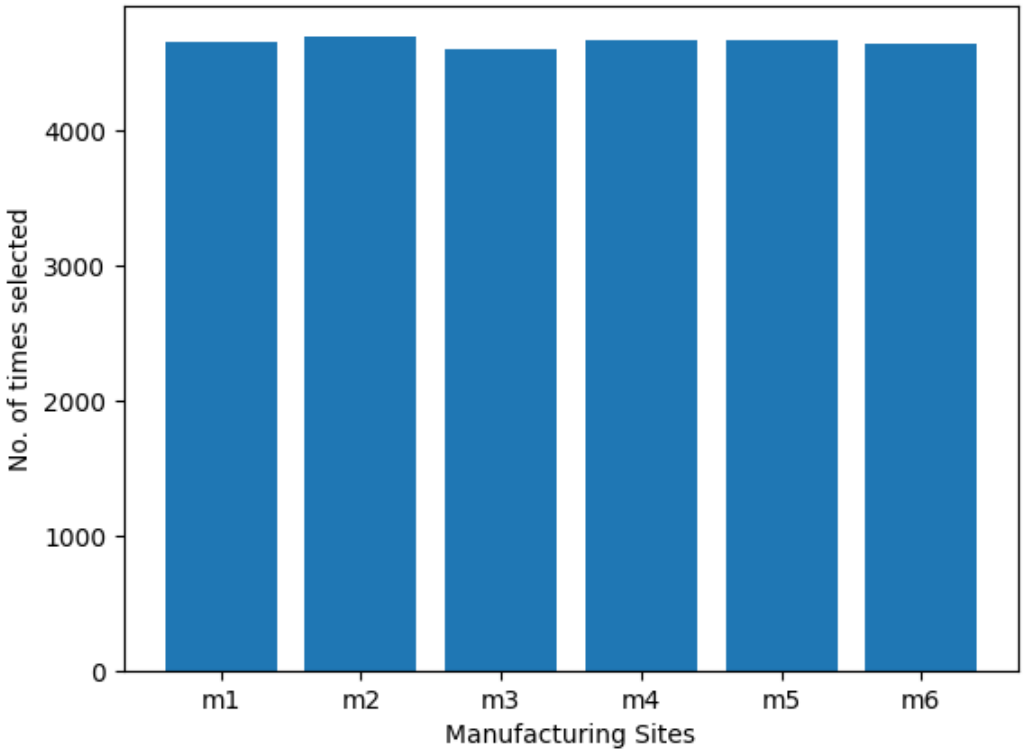


Figure 14:Number of times each manufacturing site was selected for each therapy in all the scenarios after oversampling the training set for the constrained model

Tomek Links is an approach used as an under-sampling technique or a post-processing cleaning step. In this case, Tomek Links were used to under-sample the majority class of the data (m2). Tomek Links is based on the Nearest-Neighbor Rule (NNR). A Tomek's link between two samples essentially links two samples that are the nearest neighbors and is defined by the distance(d) between two samples of different classes x, y, and another one z as:

$$d(x,y) < d(x,z) \;\; and \;\; d(x,y) < d(y,z)$$

After the link is established if one of the samples belongs to the majority class, that sample will get removed, therefore cleaning the data. The process is depicted in the figures below. [64] [65] [66]

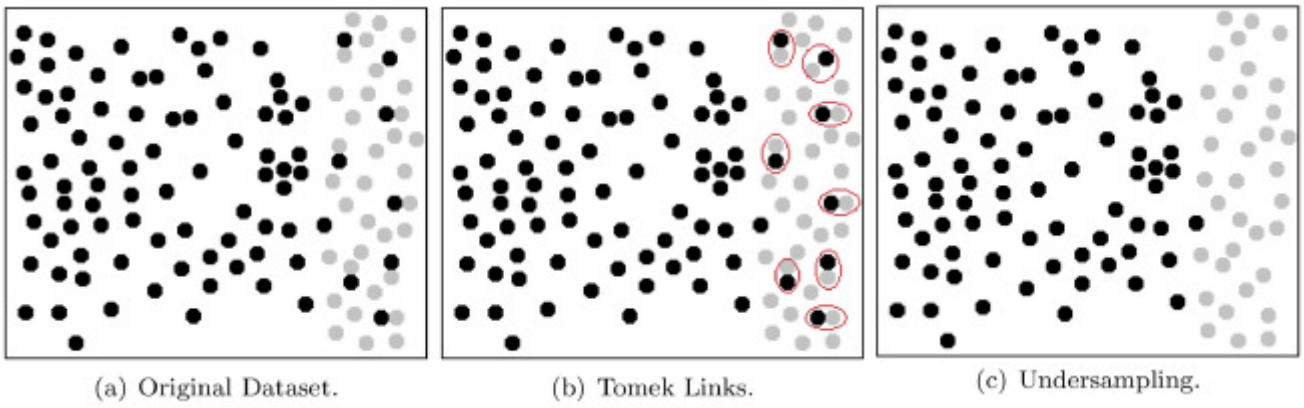


Figure 15: Tomek Links under-sampling technique

5.5. Tuning and Training the ML algorithms

A very significant part of every ML algorithm is its hyperparameters. In order to have a data model that makes the right predictions, one has to find the best hyperparameters that help the model understand the data that is given. For every ML algorithm, different hyperparameters need to be tuned. Some were shown in section 3.2. Below, there is a more specific analysis of which and what the hyperparameters the three algorithms were tuned to.

Tuning commences by constructing a grid with the hyperparameters of each classifier. After constructing the grid there are various but similar ways to find the best hyperparameters. There is no standard way to optimize the classifier in terms of hyperparameters and because of that, the major method of finding them is just searching for the ones that give the best results.

Specifically, a k-fold Cross-Validation technique is used for the tuning. Cross-Validation is a procedure that resamples the data of the model into several groups (k-folds) that are specified by the programmer. After the division happens, for each fold, the model is trained on the k-1 folds of the dataset and then tested on the k_{th} fold. After the training and testing of all the number of groups are finished, the average performance of all the different folds is the cross-validation performance of the model. We follow this procedure so as to estimate the skill of the ML model on unseen data. Subsequently, by varying the hyperparameters of the model and using a k-fold Cross-Validation technique, we can find the best parameters to fit the model.

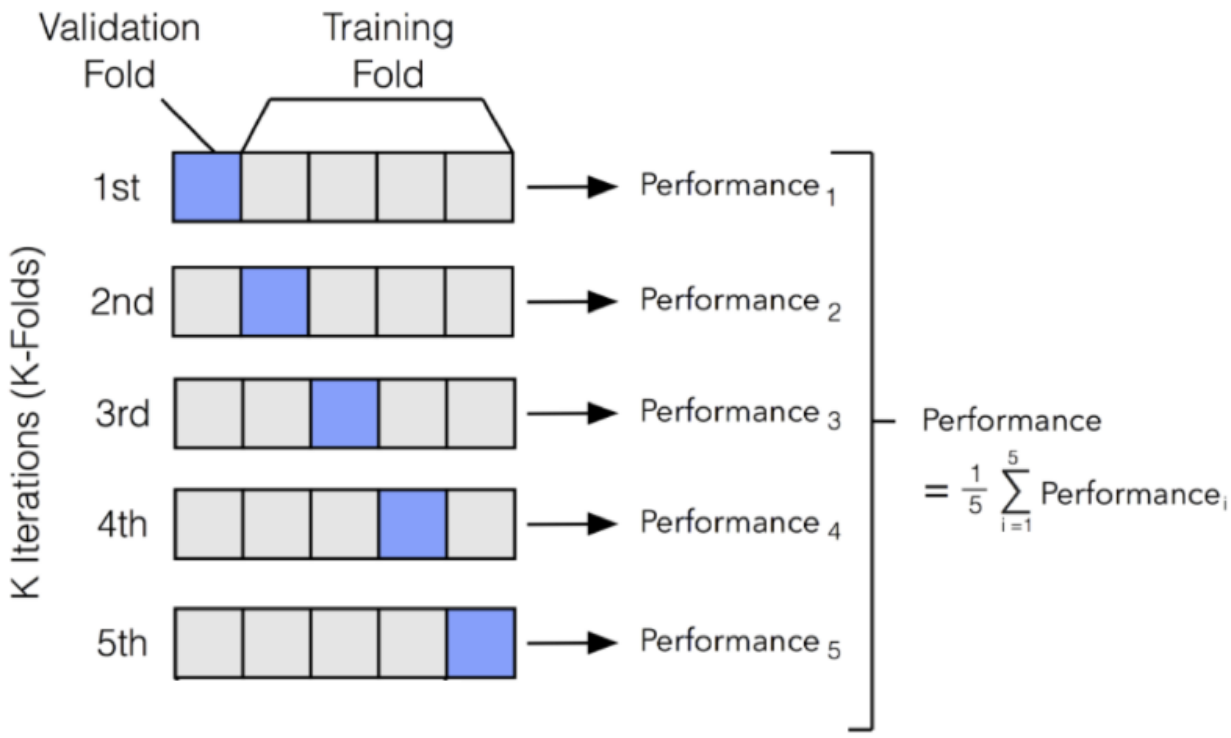


Figure 16: Example of a 5-fold Cross-Validation splitting of the dataset

Table 2: Description of the Hyperparameters of MLPClassifier algorithm

Parameters	Description
hidden_layer_sizes	The number of the hidden layers and the number of nodes of each one
solver	The solver for weight optimization. 'sgd' refers to stochastic gradient descent 'adam' refers to a stochastic gradient-based optimizer proposed by Kingma, Diederik, and Jimmy Ba 'lbfgs': optimizer in the family of quasi-Newton methods.
activation function	Activation function for the hidden layer.
alpha	L2 penalty (regularization term) parameter
learning_rate	Learning rate schedule for weight updates. 'constant' is a constant learning rate 'adaptive': each time two consecutive epochs fail to decrease the training loss, the current learning rate is divided by 5

Table 3: Description of the Hyperparameters of the SVM algorithm [2]

Parameters	Description
C	The penalty parameter of the error term or regularization term
Gamma	The kernel coefficient.
Kernel	Specifies the kernel type to be used in the algorithm (e.g., linear, rbf, poly).
Class_weight	Weights associated with classes
degree	Degree of the polynomial kernel function ('poly')

Table 4: Description of the Hyperparameters of RandomForestClasssifier algorithm

Parameters	Description
n_estimators	The number of trees in the forest.
criterion	The function to measure the quality of a split. Supported criteria are "gini" for the Gini impurity and "entropy" for the information gain. Note: this parameter is tree-specific.
max_depth	The maximum depth of the tree.
max_features	The number of features to consider when looking for the best split:
class_weight	Weights associated with classes
min_samples_leaf	The minimum number of samples required to be at a leaf node
min_samples_split	The minimum number of samples required to split an internal node

The performance of ML algorithms can be evaluated through numerous metrics. The simpler are the Accuracy, Precision, Recall, and F1 score metrics. The Accuracy metric gives the number of right predictions on the entire dataset. The Precision quantifies the number of positive class predictions that belong to the positive class. Recall shows the number of positive class predictions made out of all positive examples in the dataset. The F1 score is a performance metric that gives a balance of the precision and recall scores in one number. The equations that calculate the above metrics can be seen below.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (3\text{-}12)$$

$$Precision = \frac{TP}{TP+FP} \quad (3\text{-}13)$$

$$Recall = \frac{TP}{TP+FN} \quad (3\text{-}14)$$

$$F1\ score = \frac{2*Precision*Recall}{Precision+Recall} \quad (3\text{-}15)$$

where TP, TN, FP, and FN are True Positive, True Negative, False Positive, and False Negative respectively. Another performance metric that is also used and a really good visual representation of the performance is the confusion matrix. The confusion matrix depicts how many times, for each class, the classifier predicted one class in relation to the others. In Figure 17 there is an example of a confusion matrix that shows that the classifier predicted the class '1' seventeen times when the true class was '1', twelve times when the true class was '3', and two times when the true class was '4' and so on.

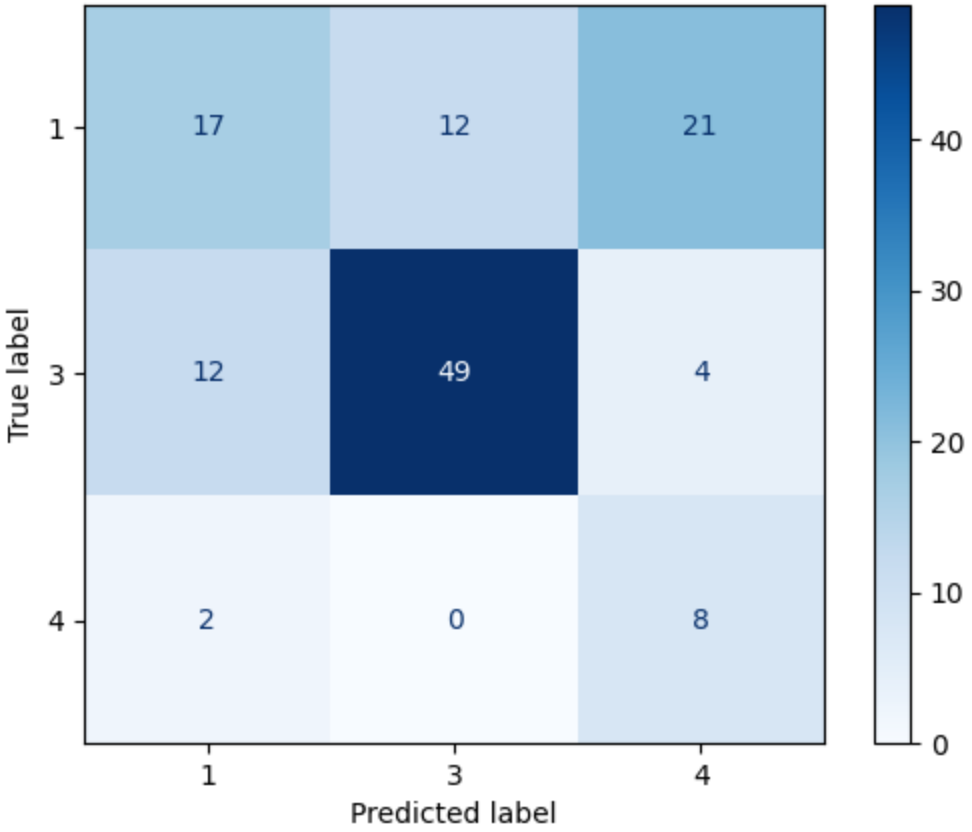


Figure 17: Example of a 3x3 confusion matrix

After running a Grid Search for each classifier using the GridSearchCV of scikit-learn library, the following hyperparameters are selected from the grids that are shown in the tables below. The best parameters were chosen for the best f1_macro score on a 5-fold cross-validation and are listed in Table. After finding the best parameters, they are used to train the classifiers, and then the models can be used to predict the manufacturing sites that need to be established for the problem.

Table 5: Grid of Hyperparameters for MLP Classifier

Parameters	Grid
hidden_layer_sizes	{100, 300, 500, (100,100), (300,300), (500,500), (100,100,100), (300,300,300), (500,500,500),
solver	{'sgd', 'adam', 'lbfgs'}
activation function	{'tanh', 'relu', 'sigmoid', 'linear'}
alpha	{1e-6, 1e-5, 1e-4, 0.001, 0.01, 0.1, 1}
learning_rate	{'constant', 'adaptive'}

Table 6: Grid of Hyperparameters for Random Forest Classifier

Parameters	Grid
n_estimators	{10,100,1000}
criterion	{'gini', 'entropy' }
max_depth	{3, 5, 10, None}
max_features	{1, 2, 3, 4}
class_weight	{'balanced', None}
min_samples_leaf	{1,2,3,4}
min_samples_split	{2,3,4,5}

Table 7:Grid of Hyperparameters for SVM Classifier

Parameters	Grid
C	{10, 100, 1000, 10000, 1e6}
Gamma	{1e-5, 1e-4, 0.001, 0.01, 0.1, 1}
Kernel	{'rbf', 'poly', 'logistic', 'sigmoid'}
Degree	{2,3,4}
Class_weight	{'balanced', None}

5.6. Computational Analysis

All the models have been implemented in Python 3.9. The MILP model was implemented in Pyomo 6.2 and solved with CPLEX 20.1.0. The ML models were implemented with scikit-learn library. The gathering of the data and the tests were run in a Windows 10 pro Pc with AMD Ryzen 5 3600 6-core 3.59 GHz Processor with 16.0GB of RAM

6. Results

Twelve different scenarios were constructed for testing, three each for total demands of 200, 500, 1000, and 2000 patients per year like some other articles, and six more ranging between the minimum and maximum demand, in order to test more the capabilities of the models. The saved model of each classifier was used to predict the manufacturing sites for each therapy in the scenarios, so the most probable facilities that need to be constructed are available to the problem. Then the data file for the scheduling problem instance is updated automatically so that the parameters related to the facilities that are not to be established, are removed from the file. This results in a smaller instance for the MILP model. The results are shown below.

Statistics concerning the classification results from the ML classifiers, the number of discrete and continuous variables as well as the number of constraints in the full space model and the model using the ML algorithms for both the unconstrained and constrained problems are displayed in Tables 11-35 and Figures 15-29.

To make an immediate comparison between the state-of-the-art model and the proposed work, demand profiles A and B from [14] were used here. The same statistics for the constraints, variables, and solutions for the profiles A, and B, as well as the difference in CPU time, can be found in tables 20-25 and figures 18-20. The demand profiles can be seen in figures 30-34 in Appendix B.

6.1. Results for Unconstrained model

Table 8: Hyperparameters' values of all three classifiers after grid search and cross-validation score

Classifier	Parameter	Value	c-v score
Multi-Layer Perceptron	hidden_layer_sizes	(300,300,300)	0.741
	solver	Adam	
	activation function	tanh	
	alpha	0.0001	
	learning_rate	adaptive	
Random Forest	n_estimators	10	0.702
	criterion	Gini	
	max_depth	10	
	max_features	2	
	class_weight	None (all have the same weight of 1)	
	min_samples_leaf	2	
	min_samples_split	2	
SVM	C	1000	0.710
	Gamma	1E+05	
	Kernel	rbf	
	Degree	-	
	Class_weight	None	

Table 9: Hyperparameters' values of all three classifiers after grid search and cross-validation score after removing the leukapheresis site as a feature

Classifier	Parameter	Value	c-v score
Multi-Layer Perceptron	hidden_layer_sizes	(300,300,300)	0.604
	solver	Adam	
	activation function	tanh	
	alpha	0.0001	
	learning_rate	adaptive	
Random Forest	n_estimators	50	0.614
	criterion	Entropy	
	max_depth	10	
	max_features	1	
	class_weight	balanced	
	min_samples_leaf	1	
	min_samples_split	2	
SVM	C	1000	0.642
	Gamma	1E+05	
	Kernel	rbf	
	Degree	-	
	Class_weight	None	

Table 10:Hyperparameters' values of all three classifiers after grid search and cross-validation score after adding the Peak in Demand feature

Classifier	Parameter	Value	c-v score
SVM	hidden_layer_sizes	(500,490,490)	0.793
	solver	Adam	
	activation function	tanh	
	alpha	0.00001	
	learning_rate	adaptive	
Random Forest	n_estimators	100	0.789
	criterion	Entropy	
	max_depth	10	
	max_features	3	
	class_weight	None	
	min_samples_leaf	2	
	min_samples_split	2	
SVM	C	1000	0.817
	Gamma	100	
	Kernel	rbf	
	Degree	-	
	Class_weight	None	

MLP Classifier

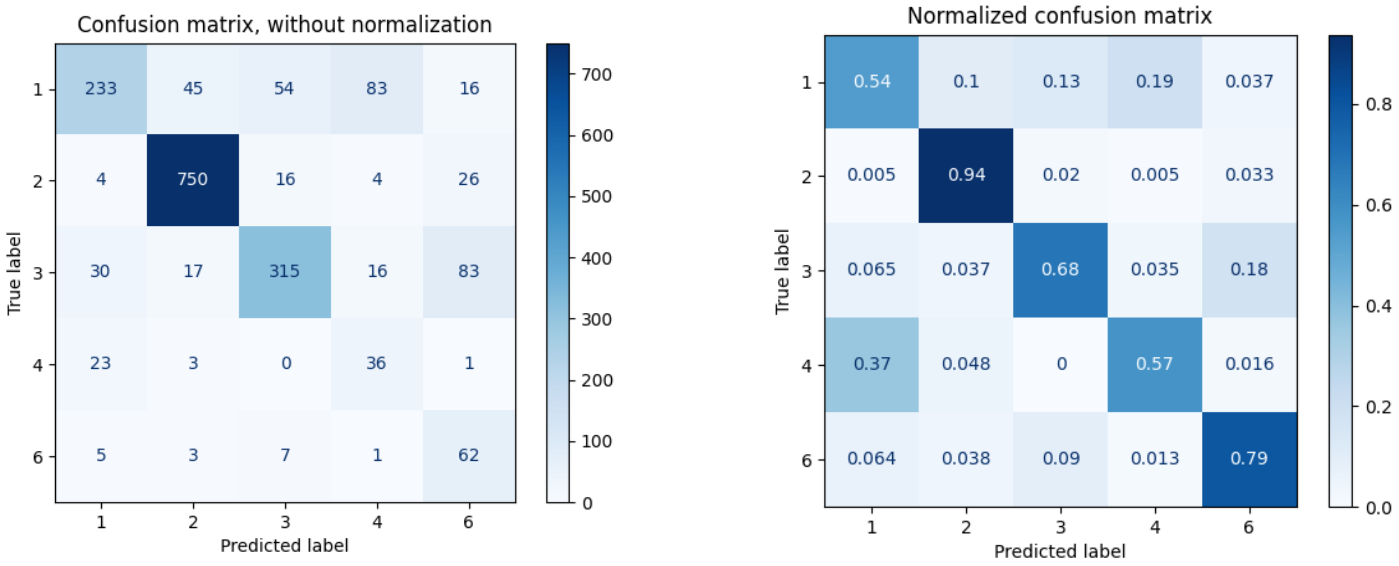


Figure 18: Normalized and not normalized Confusion matrices for the predicted manufacturing facilities for the MLP Classifier

Table 11: Classification report for MLP Classifier

Manufacturing Facility	Precision	Recall	F1_score
1	0.78	0.57	0.66
2	0.92	0.96	0.94
3	0.82	0.67	0.74
4	0.27	0.68	0.38
6	0.38	0.68	0.49
accuracy			0.78
Macro average	0.63	0.71	0.64
Weighted average	0.82	0.78	0.78

Training Time: 5 minutes, 57 seconds, 201 milliseconds

Random Forest Classifier

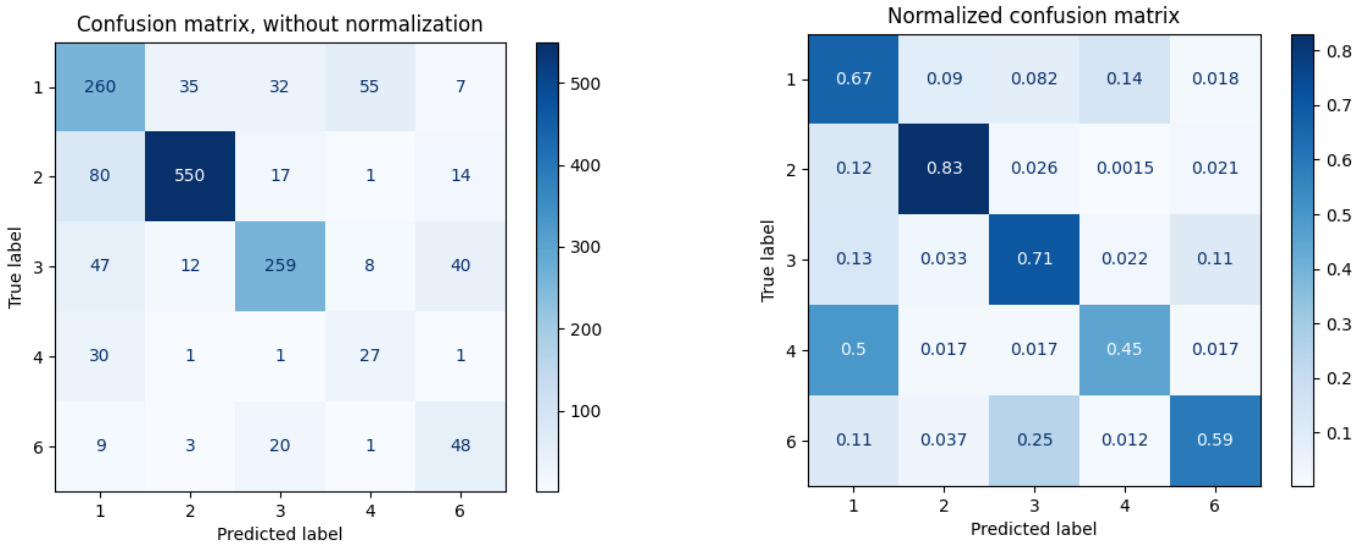


Figure 19: Normalized and not normalized Confusion matrices for the predicted manufacturing facilities for the Random Forest Classifier

Table 12: Classification report for Random Forest Classifier

Manufacturing Facility	Precision	Recall	F1_score
1	0.61	0.67	0.64
2	0.92	0.83	0.87
3	0.79	0.71	0.75
4	0.29	0.45	0.36
6	0.44	0.59	0.50
accuracy			0.73
Macro average	0.61	0.65	0.62
Weighted average	0.76	0.73	0.74

Training Time: 1 second, 98 milliseconds

SVM Classifier

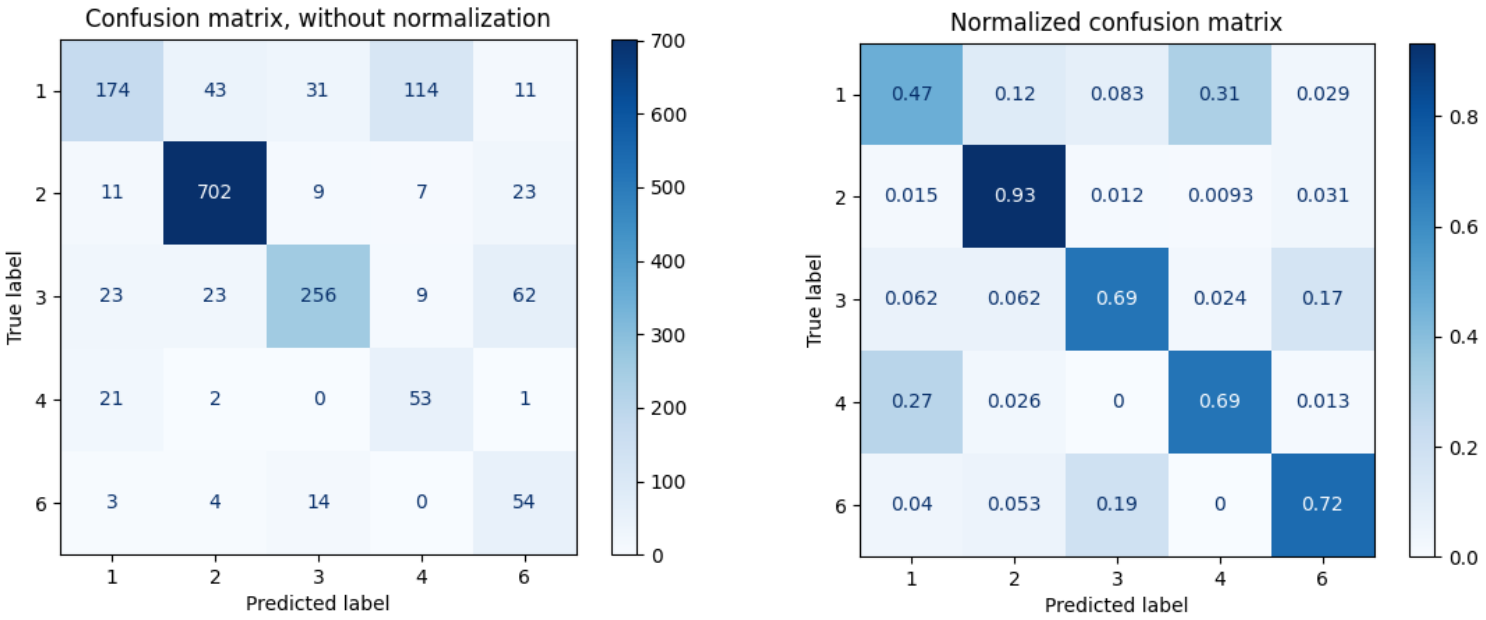


Figure 20: Normalized and not normalized Confusion matrices for the predicted manufacturing facilities for the SVM Clasifier

Table 13: Classification report for SVM Classifier

Manufacturing Facility	Precision	Recall	F1_score
1	0.75	0.47	0.58
2	0.91	0.93	0.92
3	0.83	0.69	0.75
4	0.29	0.69	0.41
6	0.36	0.72	0.48
accuracy			0.75
Macro average	0.63	0.70	0.63
Weighted average	0.80	0.75	0.76

Training Time: 47 seconds, 188 milliseconds

Table 14: Problem size for increasing problem instances without and with using the MLP Classifier for solving the planning problem

Model	Therapies per year	Demand Profile	Problem	Discrete variables	Continuous variables	Constraints	Cpu time
MLPClassifier	200	Test1	Full Space	6,24E+05	1,42E+06	2,72E+06	248,80
MLP			with ML	2,08E+05	5,08E+05	9,59E+05	95,84
		Test2	Full Space	6,24E+05	1,42E+06	2,72E+06	247,14
			with ML	2,08E+05	5,08E+05	9,59E+05	95,88
		Test3	Full Space	6,24E+05	1,42E+06	2,72E+06	247,84
			with ML	2,08E+05	5,08E+05	9,59E+05	91,28
	350	Test1	Full Space	1,09E+06	2,47E+06	4,73E+06	426,77
			with ML	3,62E+05	8,83E+05	1,67E+06	170,45
	500	Test1	Full Space	1,56E+06	3,54E+06	6,80E+06	617,92
			with ML	7,80E+05	1,84E+06	3,50E+06	326,17
		Test2	Full Space	1,56E+06	3,54E+06	6,80E+06	621,73
			with ML	7,80E+05	1,84E+06	3,50E+06	346,20
		Test3	Full Space	1,56E+06	3,54E+06	6,80E+06	629,52
			with ML	7,80E+05	1,84E+06	3,50E+06	417,45
	750	Test1	Full Space	2,33E+06	5,30E+06	1,02E+07	1019,34
			with ML	1,17E+06	2,75E+06	5,23E+06	519,19
	900	Test1	Full Space	2,81E+06	6,38E+06	1,22E+07	1244,00
			with ML	2,34E+06	5,36E+06	1,03E+07	968,94
	1000	Test1	Full Space	3,10E+06	7,03E+06	1,35E+07	1508,67
			with ML	5,16E+05	1,39E+06	2,57E+06	262,78
		Test2	Full Space	3,10E+06	7,03E+06	1,35E+07	1400,06
			with ML	1,03E+06	2,52E+06	4,76E+06	477,02
		Test3	Full Space	3,10E+06	7,03E+06	1,35E+07	1387,67
			with ML	5,16E+05	1,39E+06	2,57E+06	257,78
	1200	Test1	Full Space	3,74E+06	8,51E+06	1,63E+07	1986,36

			with ML	1,87E+06	4,41E+06	8,39E+06	961,05
	1350	Test1	Full Space	4,21E+06	9,55E+06	1,83E+07	2480,25
			with ML	1,40E+06	3,42E+06	6,46E+06	609,67
	1700	Test1	Full Space	5,30E+06	1,20E+07	2,31E+07	-
			with ML	2,65E+06	6,25E+06	1,19E+07	1435,41
	2000	Test1	Full Space	6,24E+06	1,42E+07	2,72E+07	-
			with ML	3,12E+06	7,35E+06	1,40E+07	1452,88

Table 15: Comparison of the solutions of the full space model without and with the help of the MLP Classifier for an increasing number of therapies

Model	Therapies per year	Demand Profile	Problem	Optimal Solution	Optimality gap%
MLPClassifier	200	Test1	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		Test2	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		Test3	Full Space	7,09E+06	0
			with ML	7,09E+06	0
	350	Test1	Full Space	1,24E+07	17,67
			with ML	1,24E+07	17,67
	500	Test1	Full Space	1,33E+07	0
			with ML	1,33E+07	0
		Test2	Full Space	1,63E+07	4,59
			with ML	1,63E+07	4,59
		Test3	Full Space	1,63E+07	13,89
			with ML	1,63E+07	13,61
	750	Test1	Full Space	2,22E+07	9,89
			with ML	2,22E+07	9,89
	900	Test1	Full Space	2,30E+07	0
			with ML	2,30E+07	2,6
	1000	Test1	Full Space	2,65E+07	8,22
			with ML	2,78E+07	12,83
		Test2	Full Space	2,65E+07	2,77
			with ML	2,78E+07	7,38
		Test3	Full Space	2,65E+07	4,68
			with ML	2,78E+07	9,29
	1200	Test1	Full Space	2,89E+07	0
			with ML	2,89E+07	0,00
	1350	Test1	Full Space	3,26E+07	0,00
			with ML	3,26E+07	16,00
	1700	Test1	Full Space	-	-
			with ML	3,90E+07	7,67
	2000	Test1	Full Space	-	-
			with ML	4,82E+07	6,13

Table 16: Problem size for increasing problem instances without and with using the RFC for solving the planning problem

RandomForest Classifier	Therapies per year	Demand Profile	Problem	Discrete variables	Continuous variables	Constraints	Cpu time
RF	200	Test1	Full Space	6,24E+05	1,42E+06	2,72E+06	248,80
			with ML	2,08E+05	5,08E+05	9,59E+05	85,73
		Test2	Full Space	6,24E+05	1,42E+06	2,72E+06	247,14
			with ML	2,08E+05	5,08E+05	9,59E+05	86,05
		Test3	Full Space	6,24E+05	1,42E+06	2,72E+06	247,84
			with ML	2,08E+05	5,08E+05	9,59E+05	85,63
	350	Test1	Full Space	1,09E+06	2,47E+06	4,73E+06	426,77
			with ML	3,62E+05	8,83E+05	1,67E+06	150,06
	500	Test1	Full Space	1,56E+06	3,54E+06	6,80E+06	617,92
			with ML	7,80E+05	1,84E+06	3,50E+06	314,22
		Test2	Full Space	1,56E+06	3,54E+06	6,80E+06	621,73
			with ML	7,80E+05	1,84E+06	3,50E+06	312,44
		Test3	Full Space	1,56E+06	3,54E+06	6,80E+06	629,52
			with ML	7,80E+05	1,84E+06	3,50E+06	312,39
	750	Test1	Full Space	2,33E+06	5,30E+06	1,02E+07	1019,34
			with ML	1,17E+06	2,75E+06	5,23E+06	473,42
	900	Test1	Full Space	2,81E+06	6,38E+06	1,22E+07	1244,00
			with ML	2,34E+06	5,36E+06	1,03E+07	948,48
	1000	Test1	Full Space	3,10E+06	7,03E+06	1,35E+07	1508,67
			with ML	5,16E+05	1,39E+06	2,57E+06	247,52
		Test2	Full Space	3,10E+06	7,03E+06	1,35E+07	1400,06
			with ML	1,03E+06	2,52E+06	4,76E+06	441,02
		Test3	Full Space	3,10E+06	7,03E+06	1,35E+07	1387,67
			with ML	5,16E+05	1,39E+06	2,57E+06	245,31
	1200	Test1	Full Space	3,74E+06	8,51E+06	1,63E+07	1986,36
			with ML	1,25E+06	3,04E+06	5,75E+06	557,55

	1350	Test1	Full Space	4,21E+06	9,55E+06	1,83E+07	2480,25
			with ML	1,40E+06	3,42E+06	6,46E+06	639,77
	1700	Test1	Full Space	5,30E+06	1,20E+07	2,31E+07	-
			with ML	2,65E+06	6,25E+06	1,19E+07	1439,27
	2000	Test1	Full Space	6,24E+06	1,42E+07	2,72E+07	-
			with ML	4160036	9623542	1,84E+07	2651,86

Table 17: Comparison of the solutions of the full space model without and with the help of the RF Classifier for an increasing number of therapies

Model	Therapies per year	Demand Profile	Problem	Optimal Solution	Optimality gap%
RandomForestClassifier	200	Test1	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		Test2	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		Test3	Full Space	7,09E+06	0
			with ML	7,09E+06	0
	350	Test1	Full Space	1,24E+07	17,67
			with ML	1,24E+07	17,67
	500	Test1	Full Space	1,33E+07	0
			with ML	1,33E+07	0
		Test2	Full Space	1,63E+07	4,59
			with ML	1,63E+07	4,59
		Test3	Full Space	1,63E+07	13,89
			with ML	1,63E+07	13,61
	750	Test1	Full Space	2,22E+07	9,89
			with ML	2,22E+07	9,89
	900	Test1	Full Space	2,30E+07	0
			with ML	2,30E+07	2,6
	1000	Test1	Full Space	2,65E+07	8,22
			with ML	2,78E+07	12,83
		Test2	Full Space	2,65E+07	0
			with ML	2,78E+07	4,61
		Test3	Full Space	2,65E+07	1,44
			with ML	2,78E+07	6,05
	1200	Test1	Full Space	2,89E+07	0
			with ML	2,89E+07	0,00
	1350	Test1	Full Space	3,26E+07	0,00

			with ML	3,26E+07	16,00
	1700	Test1	Full Space	-	-
			with ML	3,90E+07	7,67
	2000	Test2	Full Space	-	-
			with ML	4,82E+07	6.13

Table 18: Problem size for increasing problem instances without and with using the SVM for solving the planning problem

SVM	Therapies per year	Demand Profile	Problem	Discrete variables	Continuous variables	Constraints	Cpu time
	200	Test1	Full Space	6,24E+05	1,42E+06	2,72E+06	248,80
			with ML	2,08E+05	5,08E+05	9,59E+05	86,33
		Test2	Full Space	6,24E+05	1,42E+06	2,72E+06	247,14
			with ML	2,08E+05	5,08E+05	9,59E+05	88,28
		Test3	Full Space	6,24E+05	1,42E+06	2,72E+06	247,84
			with ML	2,08E+05	5,08E+05	9,59E+05	87,77
	350	Test1	Full Space	1,09E+06	2,47E+06	4,73E+06	426,77
			with ML	3,62E+05	8,83E+05	1,67E+06	153,23
	500	Test1	Full Space	1,56E+06	3,54E+06	6,80E+06	617,92
			with ML	7,80E+05	1,84E+06	3,50E+06	319,14
		Test2	Full Space	1,56E+06	3,54E+06	6,80E+06	621,73
			with ML	7,80E+05	1,84E+06	3,50E+06	319,13
		Test3	Full Space	1,56E+06	3,54E+06	6,80E+06	629,52
			with ML	7,80E+05	1,84E+06	3,50E+06	317,84
	750	Test1	Full Space	2,33E+06	5,30E+06	1,02E+07	1019,34
			with ML	1,56E+06	3,60E+06	6,88E+06	642,63
	900	Test1	Full Space	2,81E+06	6,38E+06	1,22E+07	1244,00
			with ML	1,87E+06	4,33E+06	8,28E+06	794,73
	1000	Test1	Full Space	3,10E+06	7,03E+06	1,35E+07	1508,67
			with ML	5,16E+05	1,39E+06	2,57E+06	252,55
		Test2	Full Space	3,10E+06	7,03E+06	1,35E+07	1400,06
			with ML	1,03E+06	2,52E+06	4,76E+06	450,13
		Test3	Full Space	3,10E+06	7,03E+06	1,35E+07	1387,67
			with ML	5,16E+05	1,39E+06	2,57E+06	251,14
	1200	Test1	Full Space	3,74E+06	8,51E+06	1,63E+07	1986,36
			with ML	1,87E+06	4,41E+06	8,39E+06	816,81
	1350	Test1	Full Space	4,21E+06	9,55E+06	1,83E+07	2480,25
			with ML	1,40E+06	3,42E+06	6,46E+06	626,38
	1700	Test1	Full Space	5,30E+06	1,20E+07	2,31E+07	-
			with ML	3,54E+06	8,18E+06	1,56E+07	1796,20
	2000	Test1	Full Space	6,24E+06	1,42E+07	2,72E+07	-
			with ML	4160036	9623542	1,84E+07	2122,67

Table 19:Comparison of the solutions of the full space model without and with the help of the SVM Classifier for an increasing number of therapies

Model	Therapies per year	Demand Profile	Problem	Optimal Solution	Optimality gap%
SVM	200	Test1	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		Test2	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		Test3	Full Space	7,09E+06	0
			with ML	7,09E+06	0
	350	Test1	Full Space	1,24E+07	17,67
			with ML	1,24E+07	17,67
	500	Test1	Full Space	1,33E+07	0
			with ML	1,33E+07	0
		Test2	Full Space	1,63E+07	4,59
			with ML	1,63E+07	4,59
		Test3	Full Space	1,63E+07	13,89
			with ML	1,63E+07	13,61
	750	Test1	Full Space	2,22E+07	9,89
			with ML	2,22E+07	9,89
	900	Test1	Full Space	2,30E+07	0
			with ML	2,30E+07	0
	1000	Test1	Full Space	2,65E+07	8,22
			with ML	2,78E+07	12,83
		Test2	Full Space	2,65E+07	0
			with ML	2,78E+07	4,61
		Test3	Full Space	2,65E+07	1,44
			with ML	2,78E+07	6,05
	1200	Test1	Full Space	2,89E+07	0
			with ML	2,89E+07	0,00
	1350	Test1	Full Space	3,26E+07	0,00
			with ML	3,26E+07	16,00
	1700	Test1	Full Space	-	-
			with ML	3,75E+07	3,95
	2000	Test1	Full Space	-	-
			with ML	4,82E+07	6.13

Table 20: Problem size for increasing problem instances without and with using the MLP Classifier for solving the planning problem

Model	Therapies per year	Profile	Problem	Discrete variables	Continuous variables	Constraints	Cpu time
MLPClassifier	200	A	Full Space	3,12E+05	7,95E+05	1,48E+06	148,92
			with ML	1,56E+05	4,24E+05	7,77E+05	81,06
		B	Full Space	3,12E+05	7,95E+05	1,48E+06	151,70
			with ML	1,04E+05	3,00E+05	5,44E+05	58,34
	500	A	Full Space	7,80E+05	1,98E+06	3,69E+06	379,75
			with ML	5,20E+05	1,37E+06	2,52E+06	262,78
		B	Full Space	7,80E+05	1,98E+06	3,69E+06	382,25
			with ML	5,20E+05	1,37E+06	2,52E+06	260,59
	1000	A	Full Space	1,55E+06	3,94E+06	7,31E+06	794,13
			with ML	7,74E+05	2,10E+06	3,85E+06	405,13
		B	Full Space	1,55E+06	3,94E+06	7,31E+06	781,67
			with ML	7,74E+05	2,10E+06	3,85E+06	405,48

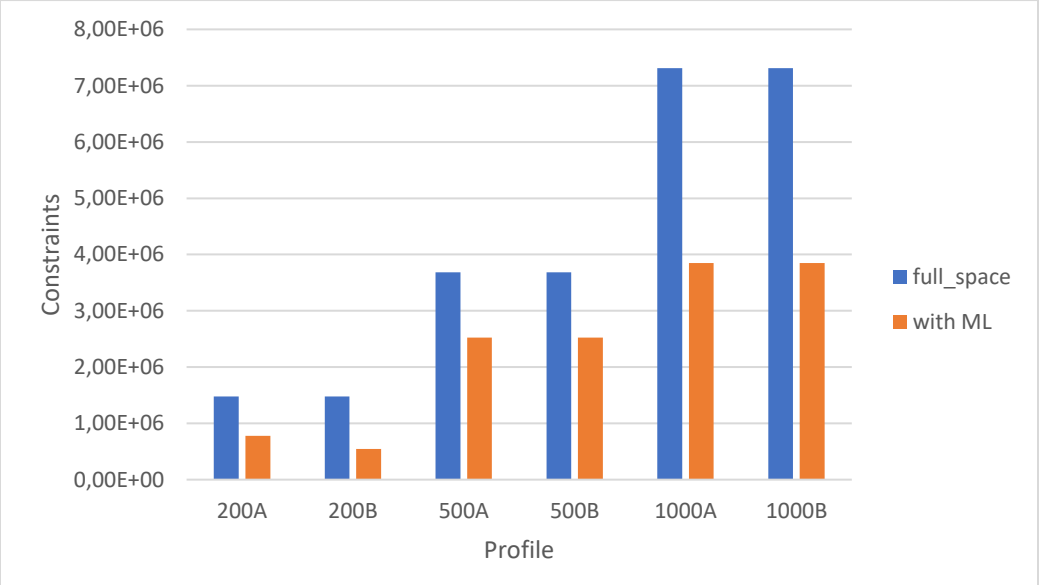


Figure 21: Comparison of the full space model and without and with the help of the MLP Classifier for an increasing number of therapies in the number of constraints

Table 21: Comparison of the solutions of the full space model without and with the help of the MLP Classifier for an increasing number of therapies

Model	Therapies per year	Demand Profile	Problem	Optimal Solution	Optimality gap%
MLPClassifier	200	A	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		B	Full Space	7,09E+06	0
			with ML	7,09E+06	0
	500	A	Full Space	1,63E+07	4,62
			with ML	1,63E+07	4,61
		B	Full Space	1,63E+07	0
			with ML	1,63E+07	0
	1000	A	Full Space	2,78E+07	0
			with ML	2,78E+07	0
		B	Full Space	2,65E+07	5,62
			with ML	2,78E+07	10,89

Table 22: Problem size for increasing problem instances without and with using the Random Forest Classifier for solving the planning problem

Model	Therapies per year	Profile	Problem	Discrete variables	Continuous variables	Constraints	Cpu time(s)
Random Forest Classifier	200	A	Full Space	3,12E+05	7,95E+05	1,48E+06	148,92
			with ML	1,56E+05	4,24E+05	7,77E+05	83,64
		B	Full Space	3,12E+05	7,95E+05	1,48E+06	151,70
			with ML	1,56E+05	4,24E+05	7,77E+05	82,41
	500	A	Full Space	7,80E+05	1,98E+06	3,69E+06	379,75
			with ML	3,90E+05	1,06E+06	1,94E+06	214,72
		B	Full Space	7,80E+05	1,98E+06	3,69E+06	382,25
			with ML	5,20E+05	1,37E+06	2,52E+06	264,94
	1000	A	Full Space	1,55E+06	3,94E+06	7,31E+06	794,13
			with ML	7,74E+05	2,10E+06	3,85E+06	417,86
		B	Full Space	1,55E+06	3,94E+06	7,31E+06	781,67
			with ML	7,74E+05	2,10E+06	3,85E+06	416,39

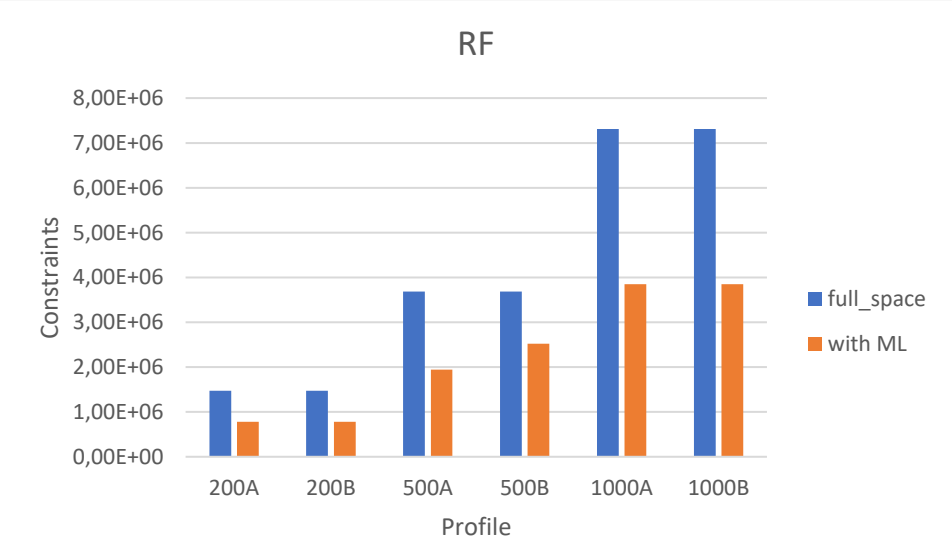


Figure 22: Comparison of the full space model and without and with the help of the Random Forest Classifier for an increasing number of therapies in the number of constraints

Table 23: Comparison of the solutions of the full space model without and with the help of the Random Forest Classifier for an increasing number of therapies

Model	Therapies per year	Demand Profile	Problem	Optimal Solution	Optimality gap%
Random Forest Classifier	200	A	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		B	Full Space	7,09E+06	0
			with ML	7,09E+06	0
	500	A	Full Space	1,63E+07	4,62
			with ML	1,63E+07	4,53
		B	Full Space	1,63E+07	0
			with ML	1,63E+07	0
	1000	A	Full Space	2,78E+07	0
			with ML	2,78E+07	0
		B	Full Space	2,65E+07	5,62
			with ML	2,78E+07	10,89

Table 24: Problem size for increasing problem instances without and with using the SVM for solving the planning problem

Model	Therapies per year	Profile	Problem	Discrete variables	Continuous variables	Constraints	Cpu time(s)
SVM	200	A	Full Space	3,12E+05	7,95E+05	1,48E+06	148,92
			with ML	1,56E+05	4,24E+05	7,77E+05	80,64
		B	Full Space	3,12E+05	7,95E+05	1,48E+06	151,70
			with ML	1,56E+05	4,24E+05	7,77E+05	83,09
	500	A	Full Space	7,80E+05	1,98E+06	3,69E+06	379,75
			with ML	5,20E+05	1,37E+06	2,52E+06	268,31
		B	Full Space	7,80E+05	1,98E+06	3,69E+06	382,25
			with ML	5,20E+05	1,37E+06	2,52E+06	263,58
	1000	A	Full Space	1,55E+06	3,94E+06	7,31E+06	794,13
			with ML	7,74E+05	2,10E+06	3,85E+06	412,94
		B	Full Space	1,55E+06	3,94E+06	7,31E+06	781,67
			with ML	5,16E+05	1,48E+06	2,70E+06	301,27

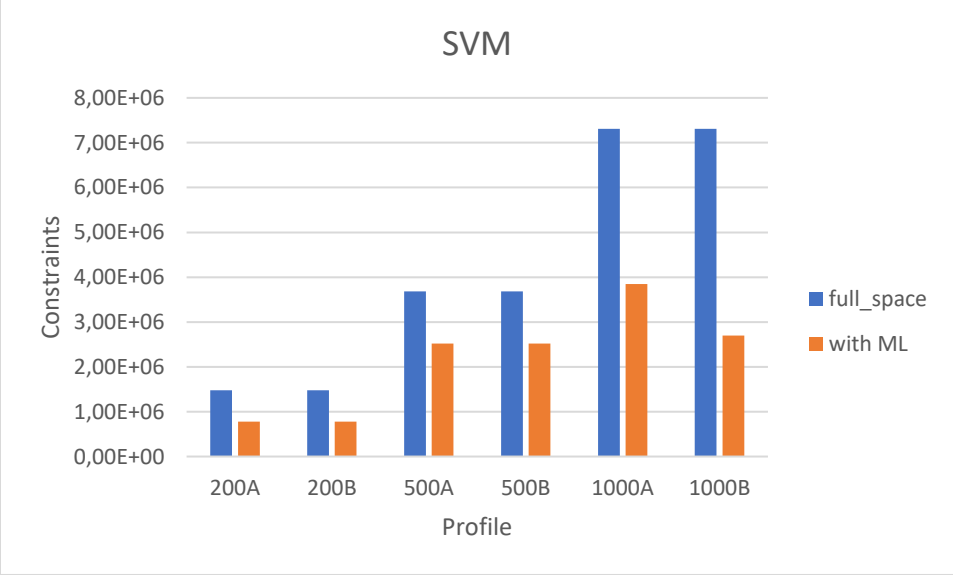
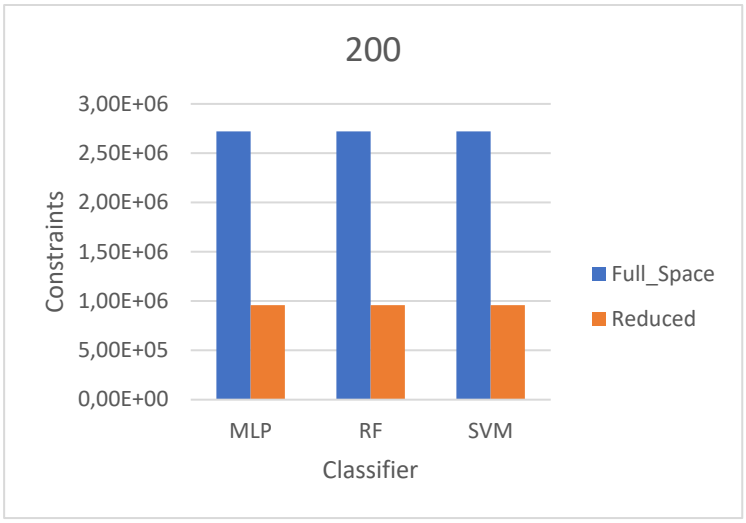
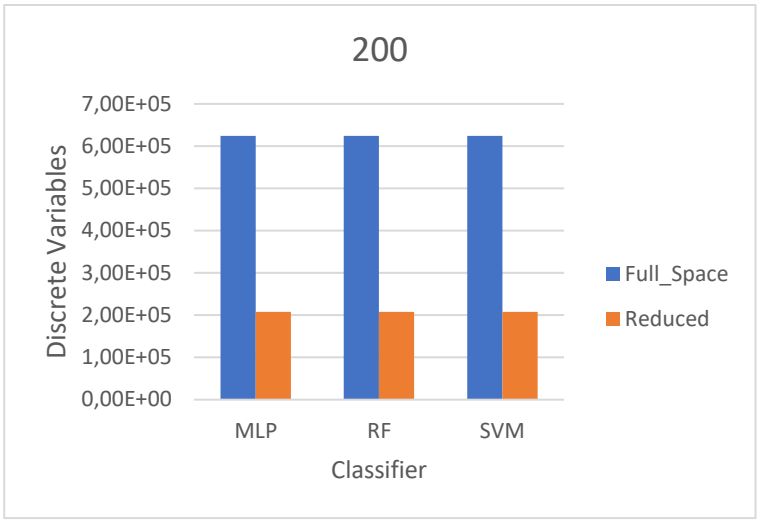


Figure 23: Comparison of the full space model and without and with the help of the SVM for an increasing number of therapies in the number of constraints

Table 25: Comparison of the solutions of the full space model without and with the help of the SVM Classifier for an increasing number of therapies

Model	Therapies per year	Demand Profile	Problem	Optimal Solution	Optimality gap%
SVM	200	A	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		B	Full Space	7,09E+06	0
			with ML	7,09E+06	0
	500	A	Full Space	1,63E+07	4,62
			with ML	1,63E+07	4,61
		B	Full Space	1,63E+07	0
			with ML	1,63E+07	0
	1000	A	Full Space	2,78E+07	0
			with ML	2,78E+07	0
		B	Full Space	2,65E+07	5,62
			with ML	2,78E+07	10,89


200
3,00E+06
2,50E+06
2,00E+06
1,50E+06
1,00E+06
5,00E+05
0,00E+00
Constraints
MLP
RF
SVM
Classifier
Full_Space
Reduced


200
7,00E+05
6,00E+05
5,00E+05
4,00E+05
3,00E+05
2,00E+05
1,00E+05
0,00E+00
Discrete Variables
MLP
RF
SVM
Classifier
Full_Space
Reduced


350
5,00E+06
4,00E+06
3,00E+06
2,00E+06
1,00E+06
0,00E+00
Constraints
MLP
RF
SVM
Classifier
Full_Space
Reduced


350
1,20E+06
1,00E+06
8,00E+05
6,00E+05
4,00E+05
2,00E+05
0,00E+00
Discrete Variables
MLP
RF
SVM
Classifier
Full_Space
Reduced

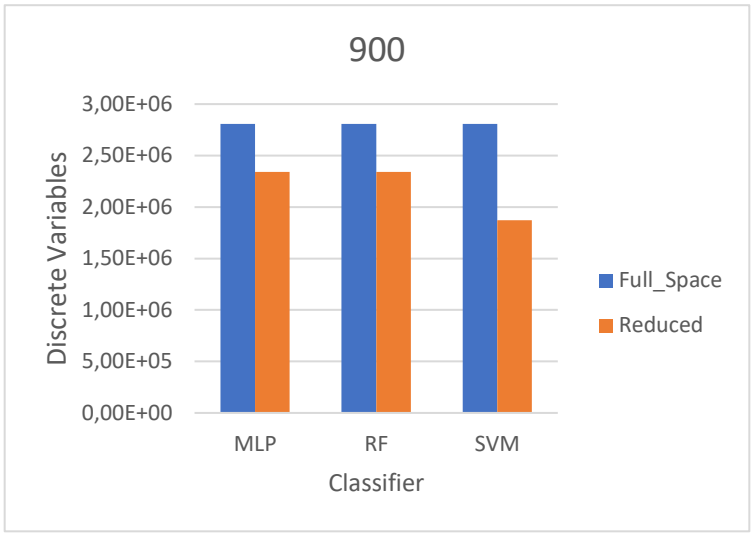

500
8,00E+06
7,00E+06
6,00E+06
5,00E+06
4,00E+06
3,00E+06
2,00E+06
1,00E+06
0,00E+00
Constraints
MLP
RF
SVM
Classifier
Full_Space
Reduced

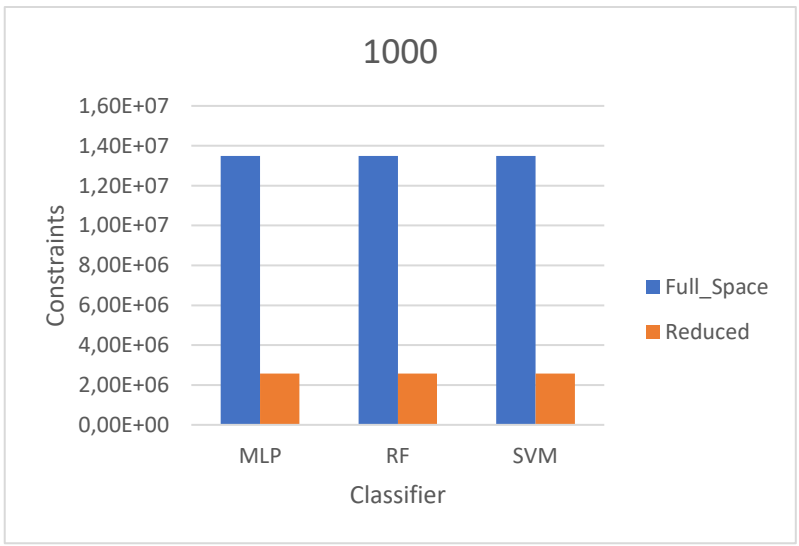

500
1,80E+06
1,60E+06
1,40E+06
1,20E+06
1,00E+06
8,00E+05
6,00E+05
4,00E+05
2,00E+05
0,00E+00
Discrete Variables
MLP
RF
SVM
Classifier
Full_Space
Reduced

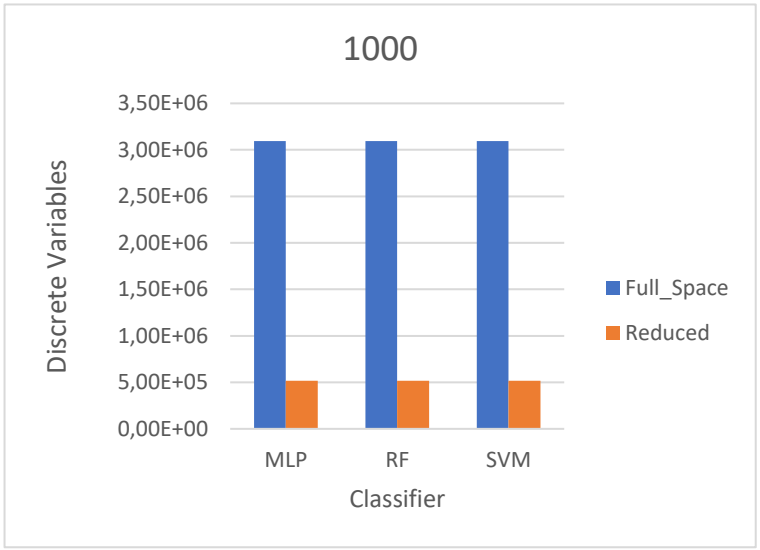

750
1,20E+07
1,00E+07
8,00E+06
6,00E+06
4,00E+06
2,00E+06
0,00E+00
Constraints
MLP
RF
SVM
Classifier
Full_Space
Reduced


750
2,50E+06
2,00E+06
1,50E+06
1,00E+06
5,00E+05
0,00E+00
Discrete Variables
MLP
RF
SVM
Classifier
Full_Space
Reduced


900
1,40E+07
1,20E+07
1,00E+07
8,00E+06
6,00E+06
4,00E+06
2,00E+06
0,00E+00
Constraints
MLP
RF
SVM
Classifier
Full_Space
Reduced

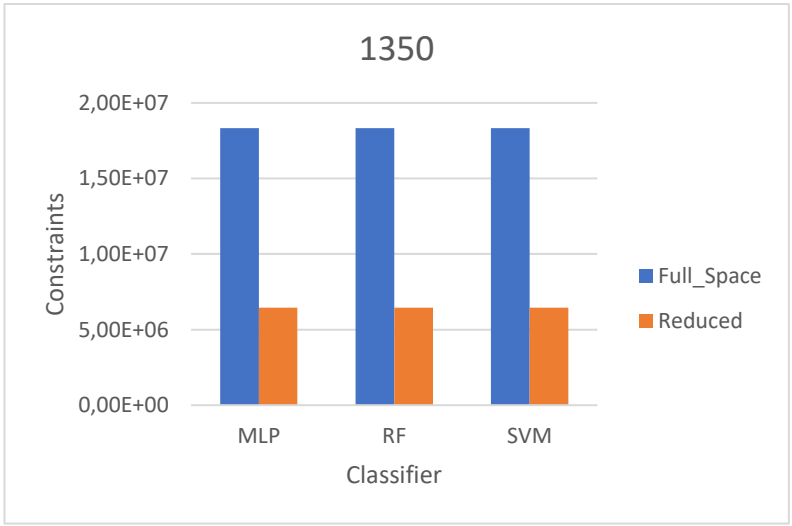

900
3,00E+06
2,50E+06
2,00E+06
1,50E+06
1,00E+06
5,00E+05
0,00E+00
Discrete Variables
MLP
RF
SVM
Classifier
Full_Space
Reduced

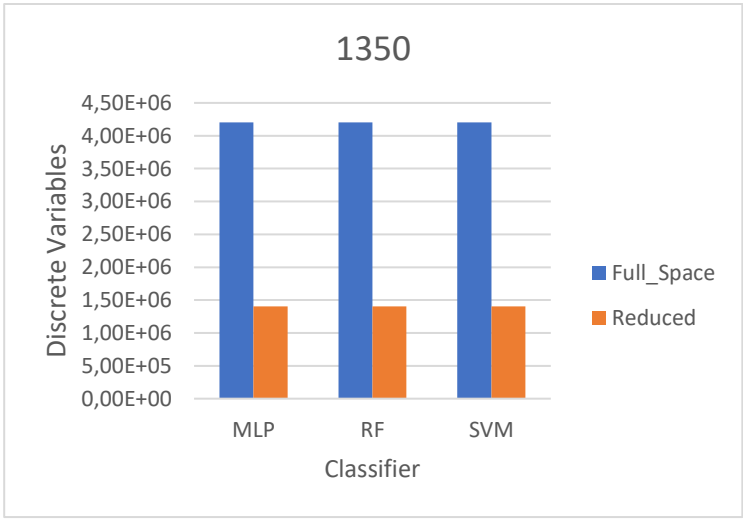

1000
1,60E+07
1,40E+07
1,20E+07
1,00E+07
8,00E+06
6,00E+06
4,00E+06
2,00E+06
0,00E+00
Constraints
MLP
RF
SVM
Classifier
Full_Space
Reduced


1000
3,50E+06
3,00E+06
2,50E+06
2,00E+06
1,50E+06
1,00E+06
5,00E+05
0,00E+00
Discrete Variables
MLP
RF
SVM
Classifier
Full_Space
Reduced


1200
1,80E+07
1,60E+07
1,40E+07
1,20E+07
1,00E+07
8,00E+06
6,00E+06
4,00E+06
2,00E+06
0,00E+00
Constraints
MLP
RF
SVM
Classifier
Full_Space
Reduced


1200
4,00E+06
3,50E+06
3,00E+06
2,50E+06
2,00E+06
1,50E+06
1,00E+06
5,00E+05
0,00E+00
Discrete Variables
MLP
RF
SVM
Classifier
Full_Space
Reduced


1350
2,00E+07
1,50E+07
1,00E+07
5,00E+06
0,00E+00
Constraints
MLP
RF
SVM
Classifier
Full_Space
Reduced


1350
4,50E+06
4,00E+06
3,50E+06
3,00E+06
2,50E+06
2,00E+06
1,50E+06
1,00E+06
5,00E+05
0,00E+00
Discrete Variables
MLP
RF
SVM
Classifier
Full_Space
Reduced


1700
2,50E+07
2,00E+07
1,50E+07
1,00E+07
5,00E+06
0,00E+00
Constraints
MLP
RF
SVM
Classifier
Full_Space
Reduced


1700
6,00E+06
5,00E+06
4,00E+06
3,00E+06
2,00E+06
1,00E+06
0,00E+00
Discrete Variables
MLP
RF
SVM
Classifier
Full_Space
Reduced

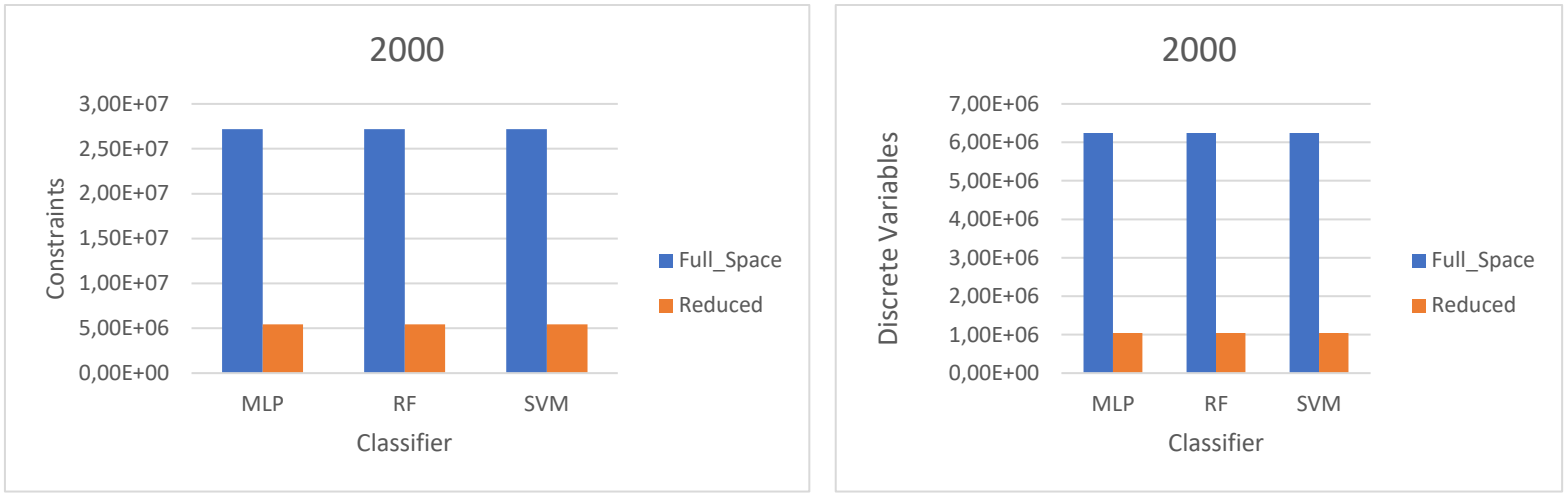


Figure 25: Reduction of constraints and discrete variables in the MILP for different demand scenarios

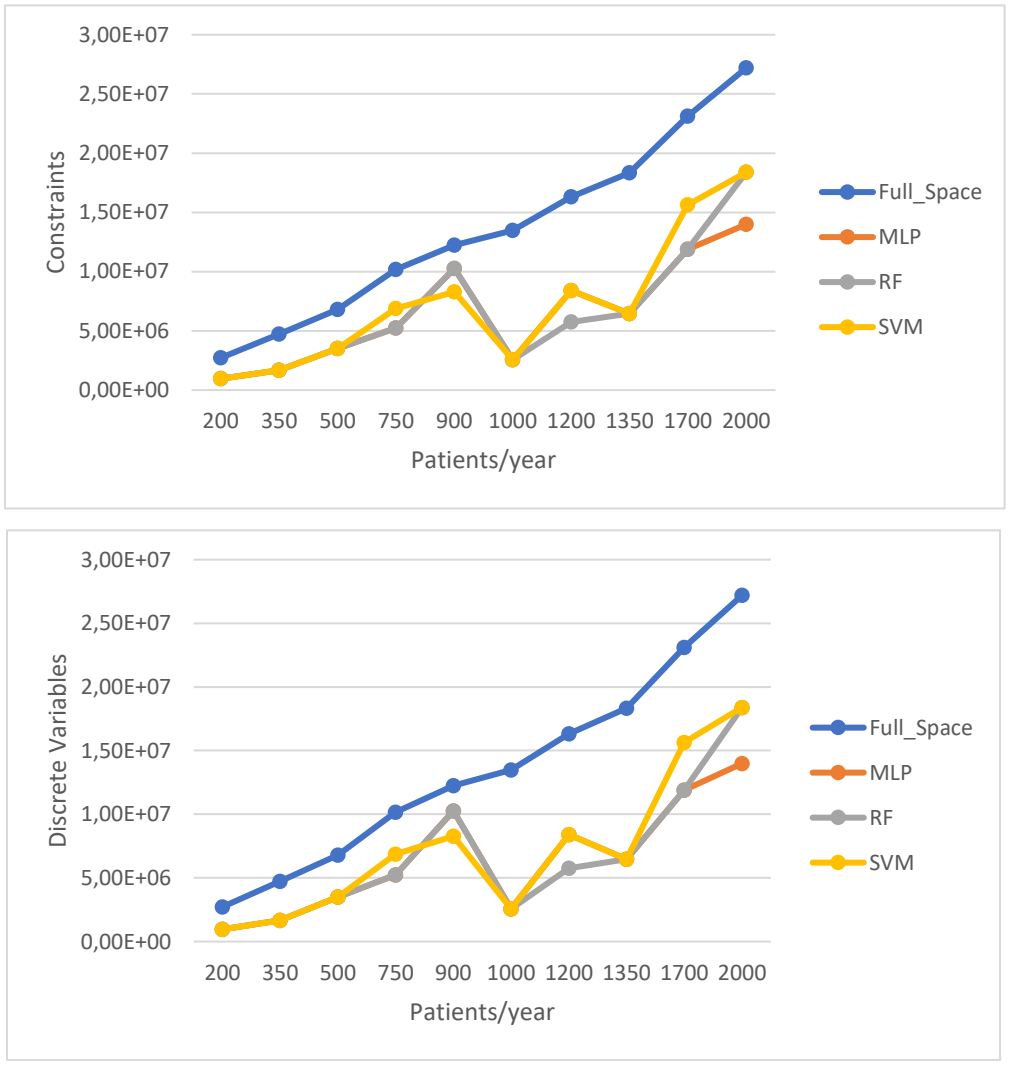


Figure 24:Number of constraints (a) and discrete variables(b) of the MILP model for full space and in combination with the data models for an increasing number of patients

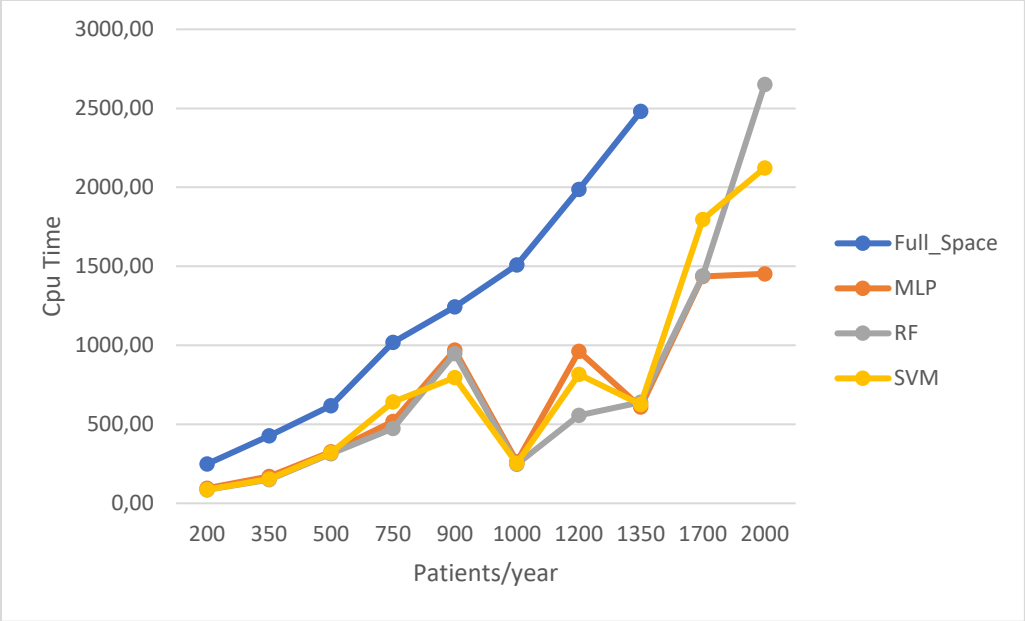


Figure 26:Cpu time (seconds) for solving the MILP model for full space and in combination with the data models for an increasing number of patients

From the tables and graphs above there are some observations that can be made. First of all, from tables 8-10 it is clear that the features that work best to fit the data are the Demand, Capacity, Leukapheresis site, and the Peak in Demand. Removing the Leukapheresis site from a feature removes the idea of the distance between the leukapheresis site and manufacturing facilities, as well as the upper bound of the daily demand. Adding the Peak in Demand can help the data models to understand the demand profile, which cannot be seen in the total demand feature.

From tables 11-13 all three classifiers have a higher recall macro average than a precision macro average, although, the weighted average, where the proportion of each label is considered, shows the opposite. The differences in both averages are small, so this suggests that the models are balanced in terms of bias and variance. However, Manufacturing facility 4 has the lowest F1-score across all classifiers. This could be due to the fact that most of the data for the specific facility are synthetic from oversampling.

It can also be seen that for every test scenario the data model can almost always predict the best manufacturing facilities that are needed for the specific demand since the

solutions after the reduction from the data model are the same as the full space model. Specifically, the data model's worst-performing demand scenarios are the ones with total demand of 1000 patients per year, with around 5% higher cost than the full space model. Moreover, for demands higher than 1500 patients/year Cplex cannot find a solution due to insufficient memory, in contrast with the help of the data model where a solution can be discovered.

In the tables and graphs that depict the statistics for constraints, variables, and CPU times (in seconds), it is evident that the data model does a really good job of reducing the complexity of the MILP model. For the most demand scenarios, 50-65% reduction in constraints and discrete variables can be seen, and it can reach as high as 83% (1000test1, 1000test3) but can be as low as 16% (900test1 – MLP). The time of the solutions in CPU time is decreased for every scenario as well as shown in figure 20, especially in the higher demand scenarios.

The ups and downs in figures 18 and 20 can be justified by the capacity of the manufacturing facilities. That is, until the demand of 900 therapies/year the mathematical model is choosing the smaller manufacturing facilities. When the demand reaches 1000 therapies/year the data model chooses one of the larger facilities to satisfy the demand. As the demand gets higher and higher more facilities are needed so the model starts picking more facilities and therefore the reduction is shortening again.

Lastly, all three classifiers yield approximately the same results. Even so, Random Forest classifier and SVM take a lot less time than the MLP classifier to train.

Generally, all ML algorithms tested can predict the manufacturing facilities that are needed for demands ranging from 200-2000 therapies/year and efficiently reduce the size of the MILP problem. However, seeing that the models are trained with data coming from the MILP solutions and the manufacturing capacity is limited, larger instances couldn't be tested.

6.2. Results for Constrained Model

Table 26: Hyperparameters' values of all three classifiers after grid search and cross-validation score for constrained model

Classifier	Parameter	Value	c-v score
Multi-Layer Perceptron	hidden_layer_sizes	(500,500,500)	0.898
	solver	Adam	
	activation function	tanh	
	alpha	0.0001	
	learning_rate	adaptive	
Random Forest	n_estimators	500	0.91
	criterion	Gini	
	max_depth	20	
	max_features	2	
	class_weight	None (all have the same weight of 1)	
	min_samples_leaf	2	
	min_samples_split	2	
SVM	C	4400	0.928
	Gamma	1	
	Kernel	rbf	
	Degree	-	
	Class_weight	None	

MLP Classifier

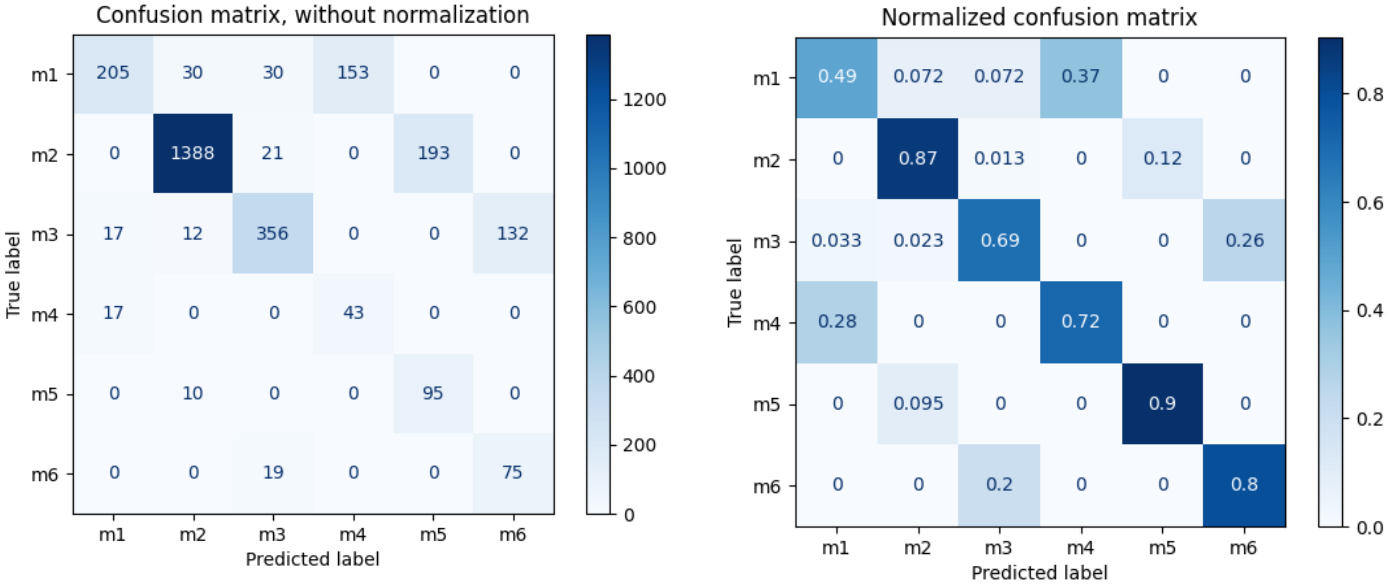


Figure 27: Normalized and not normalized Confusion matrices for the predicted manufacturing facilities for the MLP Classifier

Table 27: Classification report for MLP Classifier

Manufacturing Facility	Precision	Recall	F1_score
1	0.86	0.49	0.62
2	0.96	0.87	0.91
3	0.84	0.69	0.76
4	0.22	0.72	0.34
5	0.33	0.90	0.48
6	0.36	0.80	0.50
accuracy			0.77
Macro average	0.59	0.74	0.60
Weighted average	0.86	0.77	0.80

Training Time: 4 minutes, 32 seconds, 647 milliseconds

Random Forest Classifier

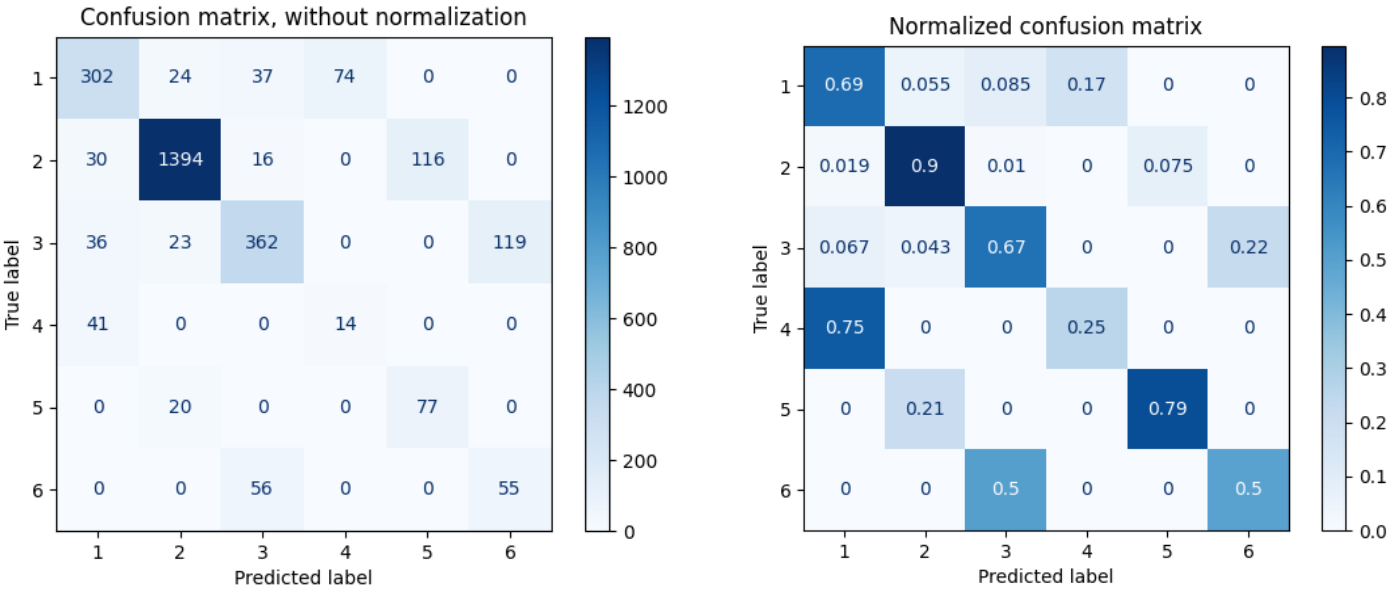


Figure 28: Normalized and not normalized Confusion matrices for the predicted manufacturing facilities for the Random Forest Classifier

Manufacturing Facility	Precision	Recall	F1_score
1	0.74	0.69	0.71
2	0.95	0.90	0.92
3	0.77	0.67	0.72
4	0.16	0.25	0.20
5	0.40	0.79	0.53
6	0.32	0.50	0.30
accuracy			0.79
Macro average	0.56	0.63	0.58
Weighted average	0.82	0.79	0.80

Table 28: Classification report for Random Forest Classifier

Training Time: 948 milliseconds

SVM Classifier

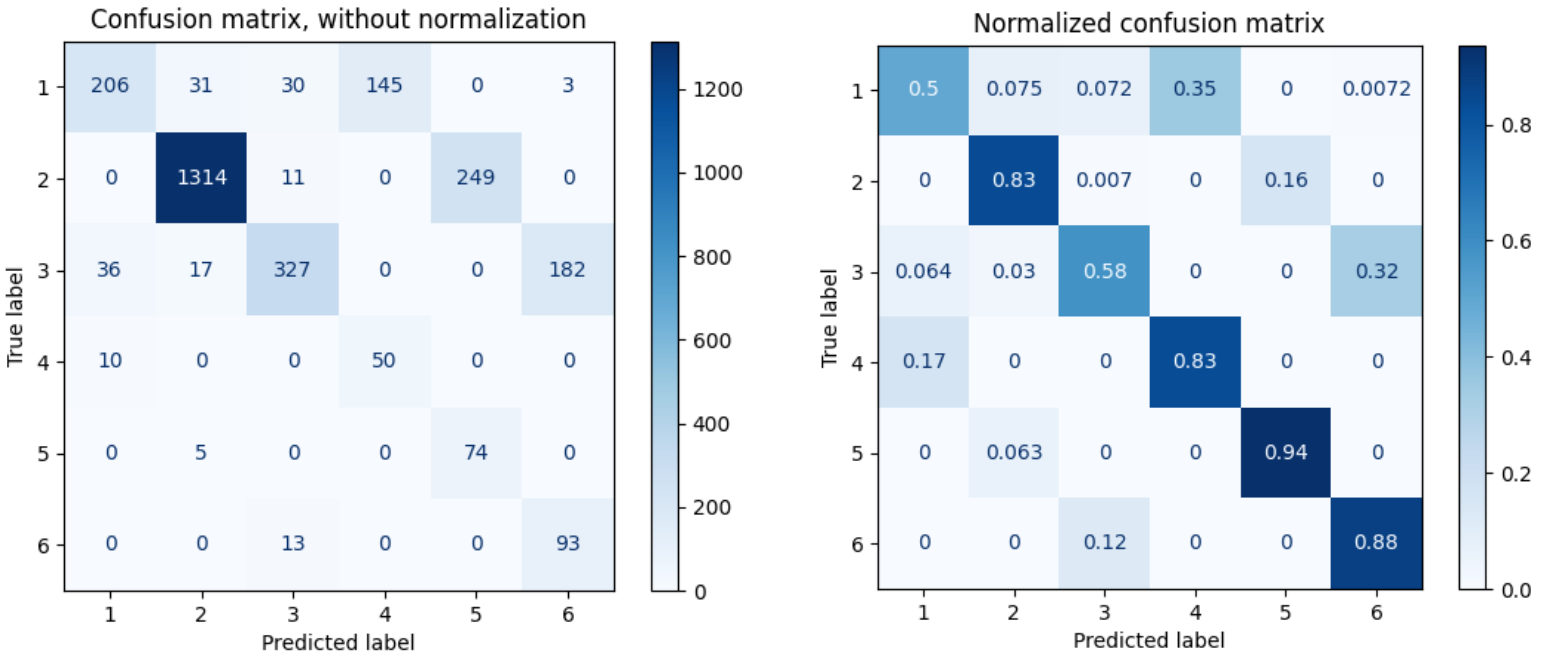


Figure 29: Normalized and not normalized Confusion matrices for the predicted manufacturing facilities for the SVM Classifier

Manufacturing Facility	Precision	Recall	F1_score
1	0.82	0.50	0.62
2	0.96	0.83	0.89
3	0.86	0.58	0.69
4	0.26	0.83	0.39
5	0.23	0.94	0.37
6	0.33	0.88	0.48
accuracy			0.74
Macro average	0.58	0.76	0.57
Weighted average	0.86	0.74	0.77

Table 29: Classification report for SVM Classifier

Training Time: 51 seconds, 191 milliseconds

Table 30: Problem size for increasing problem instances without and with using the MLP Classifier for solving the planning problem

Model	Therapies per year	Demand Profile	Problem	Discrete variables	Continuous variables	Constraints	Cpu time
MLPClassifier	200	Test1	Full Space	6,24E+05	1,42E+06	2,72E+06	248,80
MLP			with ML	2,08E+05	5,08E+05	9,59E+05	87,09
		Test2	Full Space	6,24E+05	1,42E+06	2,72E+06	251,33
			with ML	2,08E+05	5,08E+05	9,59E+05	87,38
		Test3	Full Space	6,24E+05	1,42E+06	2,72E+06	248,16
			with ML	2,08E+05	5,08E+05	9,59E+05	86,45
	350	Test1	Full Space	1,09E+06	2,47E+06	4,73E+06	442,89
			with ML	3,62E+05	8,83E+05	1,67E+06	153,36
	500	Test1	Full Space	1,56E+06	3,54E+06	6,80E+06	612,28
			with ML	5,20E+05	1,27E+06	2,40E+06	222,48
		Test2	Full Space	1,56E+06	3,54E+06	6,80E+06	604,48
			with ML	5,20E+05	1,27E+06	2,40E+06	220,34
		Test3	Full Space	1,56E+06	3,54E+06	6,80E+06	607,16
			with ML	5,20E+05	1,27E+06	2,40E+06	221,02
	750	Test1	Full Space	2,33E+06	5,30E+06	1,02E+07	1106,25
			with ML	3,89E+05	1,05E+06	1,94E+06	184,89
	900	Test1	Full Space	2,81E+06	6,38E+06	1,22E+07	1290,64
			with ML	4,68E+05	1,26E+06	2,33E+06	226,00
	1000	Test1	Full Space	3,10E+06	7,03E+06	1,35E+07	1437,44
			with ML	5,16E+05	1,39E+06	2,57E+06	253,41
		Test2	Full Space	3,10E+06	7,03E+06	1,35E+07	1399,69

			with ML	1,03E+06	2,52E+06	4,76E+06	450,03
		Test3	Full Space	3,10E+06	7,03E+06	1,35E+07	1399,23
			with ML	5,16E+05	1,39E+06	2,57E+06	250,84
	1200	Test1	Full Space	3,74E+06	8,51E+06	1,63E+07	1883,94
			with ML	1,25E+06	3,04E+06	5,75E+06	552,97
	1350	Test1	Full Space	4,21E+06	9,55E+06	1,83E+07	2369,95
			with ML	1,40E+06	3,42E+06	6,46E+06	602,09
	1700	Test1	Full Space	5,30E+06	1,20E+07	2,31E+07	-
			with ML	1,77E+06	4,31E+06	8,15E+06	790,97
	2000	Test1	Full Space	6,24E+06	1,42E+07	2,72E+07	-
			with ML	2,08E+06	5,07E+06	9,59E+06	956,11

Table 31: Comparison of the solutions of the full space model without and with the help of the MLP Classifier for an increasing number of therapies

Model	Therapies per year	Demand Profile	Problem	Optimal Solution	Optimality gap%
MLPClassifier	200	Test1	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		Test2	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		Test3	Full Space	7,09E+06	0
			with ML	7,09E+06	0
	350	Test1	Full Space	1,24E+07	0
			with ML	1,24E+07	0
	500	Test1	Full Space	1,33E+07	0
			with ML	1,33E+07	0
		Test2	Full Space	1,78E+07	0
			with ML	1,78E+07	0
		Test3	Full Space	1,78E+07	0
			with ML	1,78E+07	0
	750	Test1	Full Space	2,65E+07	0
			with ML	2,65E+07	0
	900	Test1	Full Space	2,73E+07	0
			with ML	2,73E+07	0
	1000	Test1	Full Space	2,78E+07	0
			with ML	2,78E+07	0
		Test2	Full Space	2,78E+07	0
			with ML	2,78E+07	0
		Test3	Full Space	2,78E+07	0,79
			with ML	2,78E+07	0,79
	1200	Test1	Full Space	2,89E+07	0
			with ML	2,89E+07	0
	1350	Test1	Full Space	3,26E+07	6,59
			with ML	3,26E+07	6,59
	1700	Test1	Full Space	-	-
			with ML	3,90E+07	7,67
	2000	Test1	Full Space	-	-
			with ML	5,57E+07	6,13

Table 32:Problem size for increasing problem instances without and with using the RF Classifier for solving the planning problem

Model	Therapies per year	Demand Profile	Problem	Discrete variables	Continuous variables	Constraints	Cpu time
Random Forest	200	Test1	Full Space	6,24E+05	1,42E+06	2,72E+06	248,80
			with ML	2,08E+05	5,08E+05	9,59E+05	89,30
		Test2	Full Space	6,24E+05	1,42E+06	2,72E+06	247,14
			with ML	2,08E+05	5,08E+05	9,59E+05	94,23
		Test3	Full Space	6,24E+05	1,42E+06	2,72E+06	247,84
			with ML	2,08E+05	5,08E+05	9,59E+05	93,00
	350	Test1	Full Space	1,09E+06	2,47E+06	4,73E+06	442,89
			with ML	3,62E+05	8,83E+05	1,67E+06	157,41
	500	Test1	Full Space	1,56E+06	3,54E+06	6,80E+06	612,28
			with ML	5,20E+05	1,27E+06	2,40E+06	222,75
		Test2	Full Space	1,56E+06	3,54E+06	6,80E+06	604,48
			with ML	5,20E+05	1,27E+06	2,40E+06	224,13
		Test3	Full Space	1,56E+06	3,54E+06	6,80E+06	607,16
			with ML	5,20E+05	1,27E+06	2,40E+06	225,13
	750	Test1	Full Space	2,33E+06	5,30E+06	1,02E+07	1019,34
			with ML	3,89E+05	1,05E+06	1,94E+06	184,19
	900	Test1	Full Space	2,81E+06	6,38E+06	1,22E+07	1290,64
			with ML	4,68E+05	1,26E+06	2,33E+06	230,48
	1000	Test1	Full Space	3,10E+06	7,03E+06	1,35E+07	1437,44
			with ML	5,16E+05	1,39E+06	2,57E+06	256,75
		Test2	Full Space	3,10E+06	7,03E+06	1,35E+07	1399,69

			with ML	1,03E+06	2,52E+06	4,76E+06	451,52
		Test3	Full Space	3,10E+06	7,03E+06	1,35E+07	1399,23
			with ML	5,16E+05	1,39E+06	2,57E+06	258,80
	1200	Test1	Full Space	3,74E+06	8,51E+06	1,63E+07	1883,94
			with ML	1,25E+06	3,04E+06	5,75E+06	530,63
	1350	Test1	Full Space	4,21E+06	9,55E+06	1,83E+07	2369,95
			with ML	1,40E+06	3,42E+06	6,46E+06	602,09
	1700	Test1	Full Space	5,30E+06	1,20E+07	2,31E+07	-
			with ML	1,77E+06	4,31E+06	8,15E+06	819,56
	2000	Test1	Full Space	6,24E+06	1,42E+07	2,72E+07	-
			with ML	2,08E+06	5,07E+06	9,59E+06	1040,00

Table 33: Comparison of the solutions of the full space model without and with the help of the RF Classifier for an increasing number of therapies

Model	Therapies per year	Demand Profile	Problem	Optimal Solution	Optimality gap%
Random Forest Classifier	200	Test1	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		Test2	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		Test3	Full Space	7,09E+06	0
			with ML	7,09E+06	0
	350	Test1	Full Space	1,24E+07	0
			with ML	1,24E+07	0
	500	Test1	Full Space	1,33E+07	0
			with ML	1,33E+07	0
		Test2	Full Space	1,78E+07	0
			with ML	1,78E+07	0
		Test3	Full Space	1,78E+07	0
			with ML	1,78E+07	0
	750	Test1	Full Space	2,65E+07	0
			with ML	2,65E+07	0
	900	Test1	Full Space	2,73E+07	0
			with ML	2,73E+07	0
	1000	Test1	Full Space	2,78E+07	0
			with ML	2,78E+07	0
		Test2	Full Space	2,78E+07	0
			with ML	2,78E+07	0
		Test3	Full Space	2,78E+07	0,79
			with ML	2,78E+07	0,79
	1200	Test1	Full Space	2,89E+07	0
			with ML	2,89E+07	0
	1350	Test1	Full Space	3,26E+07	6,59
			with ML	3,26E+07	6,59
	1700	Test1	Full Space	-	-
			with ML	3,90E+07	7,67
	2000	Test1	Full Space	-	-
			with ML	5,57E+07	6,13

Table 34: Problem size for increasing problem instances without and with using the SVM Classifier for solving the planning problem

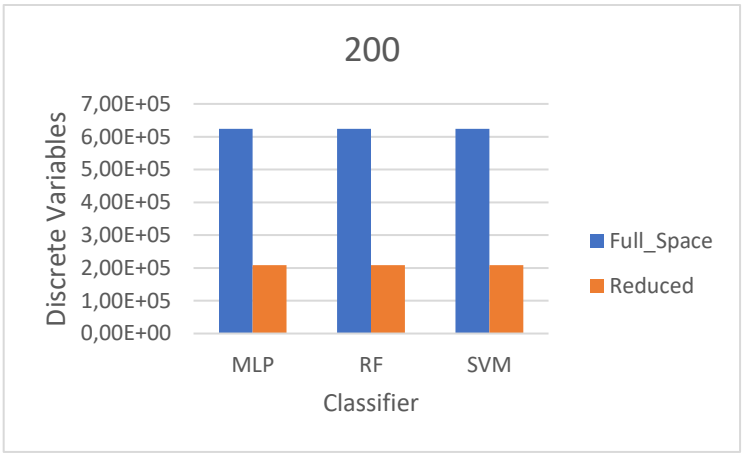
Model	Therapies per year	Demand Profile	Problem	Discrete variables	Continuous variables	Constraints	Cpu time
SVM	200	Test1	Full Space	6,24E+05	1,42E+06	2,72E+06	248,80
			with ML	2,08E+05	5,08E+05	9,59E+05	87,58
		Test2	Full Space	6,24E+05	1,42E+06	2,72E+06	247,14
			with ML	2,08E+05	5,08E+05	9,59E+05	88,48
		Test3	Full Space	6,24E+05	1,42E+06	2,72E+06	247,84
			with ML	2,08E+05	5,08E+05	9,59E+05	87,98
	350	Test1	Full Space	1,09E+06	2,47E+06	4,73E+06	442,89
			with ML	3,62E+05	8,83E+05	1,67E+06	150,44
	500	Test1	Full Space	1,56E+06	3,54E+06	6,80E+06	612,28
			with ML	5,20E+05	1,27E+06	2,40E+06	225,41
		Test2	Full Space	1,56E+06	3,54E+06	6,80E+06	604,48
			with ML	5,20E+05	1,27E+06	2,40E+06	222,19
		Test3	Full Space	1,56E+06	3,54E+06	6,80E+06	607,16
			with ML	5,20E+05	1,27E+06	2,40E+06	223,70
	750	Test1	Full Space	2,33E+06	5,30E+06	1,02E+07	1019,34
			with ML	3,89E+05	1,05E+06	1,94E+06	184,48
	900	Test1	Full Space	2,81E+06	6,38E+06	1,22E+07	1290,64
			with ML	4,68E+05	1,26E+06	2,33E+06	223,47
	1000	Test1	Full Space	3,10E+06	7,03E+06	1,35E+07	1437,44
			with ML	5,16E+05	1,39E+06	2,57E+06	246,39
		Test2	Full Space	3,10E+06	7,03E+06	1,35E+07	1399,69

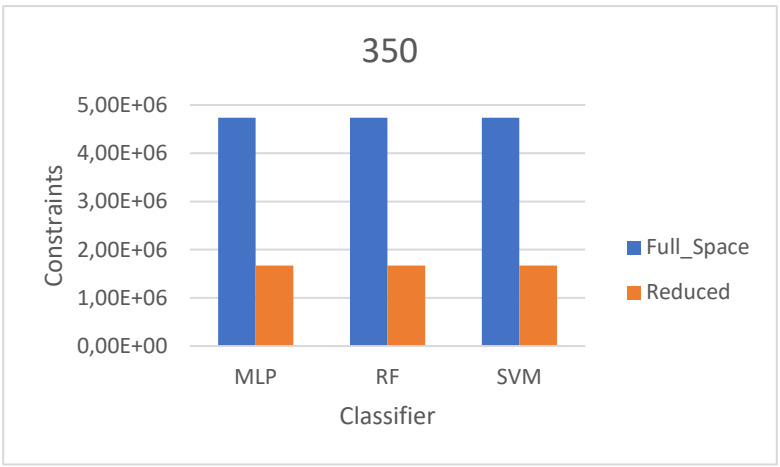
			with ML	1,03E+06	2,52E+06	4,76E+06	450,59
		Test3	Full Space	3,10E+06	7,03E+06	1,35E+07	1399,23
			with ML	5,16E+05	1,39E+06	2,57E+06	253,84
	1200	Test1	Full Space	3,74E+06	8,51E+06	1,63E+07	1883,94
			with ML	1,25E+06	3,04E+06	5,75E+06	551,19
	1350	Test1	Full Space	4,21E+06	9,55E+06	1,83E+07	2369,95
			with ML	1,40E+06	3,42E+06	6,46E+06	611,91
	1700	Test1	Full Space	5,30E+06	1,20E+07	2,31E+07	-
			with ML	1,77E+06	4,31E+06	8,15E+06	765,13
	2000	Test1	Full Space	6,24E+06	1,42E+07	2,72E+07	-
			with ML	2,08E+06	5,07E+06	9,59E+06	1145,98

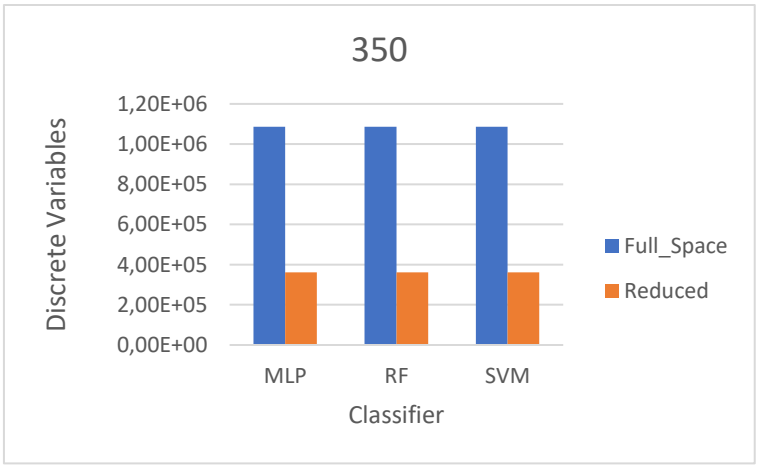
Table 35: Comparison of the solutions of the full space model without and with the help of the SVM Classifier for an increasing number of therapies

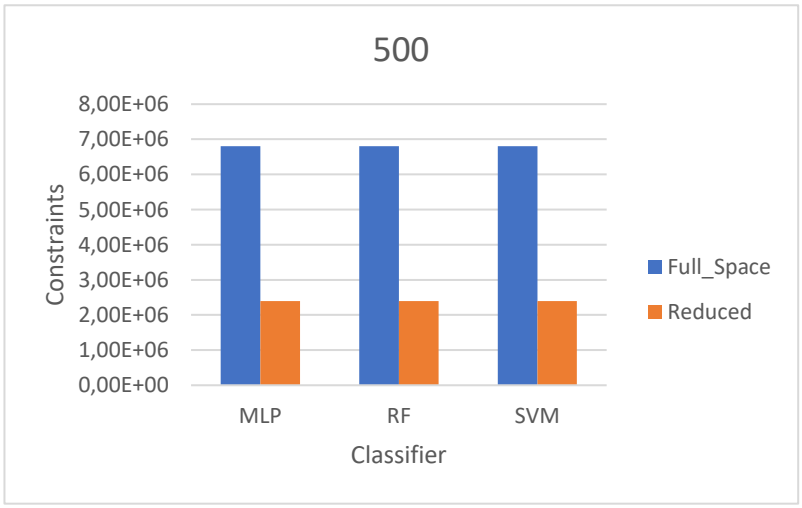
Model	Therapies per year	Demand Profile	Problem	Optimal Solution	Optimality gap%
SVM	200	Test1	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		Test2	Full Space	7,09E+06	0
			with ML	7,09E+06	0
		Test3	Full Space	7,09E+06	0
			with ML	7,09E+06	0
	350	Test1	Full Space	1,24E+07	0
			with ML	1,24E+07	0
	500	Test1	Full Space	1,33E+07	0
			with ML	1,33E+07	0
		Test2	Full Space	1,78E+07	0
			with ML	1,78E+07	0
		Test3	Full Space	1,78E+07	0
			with ML	1,78E+07	0
	750	Test1	Full Space	2,65E+07	0
			with ML	2,65E+07	0
	900	Test1	Full Space	2,73E+07	0
			with ML	2,73E+07	0
	1000	Test1	Full Space	2,78E+07	0
			with ML	2,78E+07	0
		Test2	Full Space	2,78E+07	0
			with ML	2,78E+07	0
		Test3	Full Space	2,78E+07	0,79
			with ML	2,78E+07	0,79
	1200	Test1	Full Space	2,89E+07	0
			with ML	2,89E+07	0
	1350	Test1	Full Space	3,26E+07	6,59
			with ML	3,26E+07	6,59
	1700	Test1	Full Space	-	-
			with ML	3,90E+07	0
	2000	Test1	Full Space	-	-
			with ML	5,57E+07	0


200
Constraints
3,00E+06
2,50E+06
2,00E+06
1,50E+06
1,00E+06
5,00E+05
0,00E+00
MLP
RF
SVM
Classifier
Full_Space
Reduced

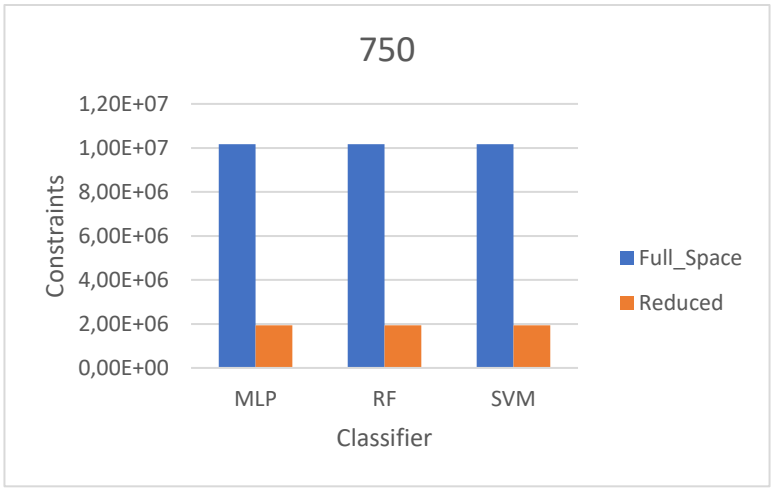

200
Discrete Variables
7,00E+05
6,00E+05
5,00E+05
4,00E+05
3,00E+05
2,00E+05
1,00E+05
0,00E+00
MLP
RF
SVM
Classifier
Full_Space
Reduced

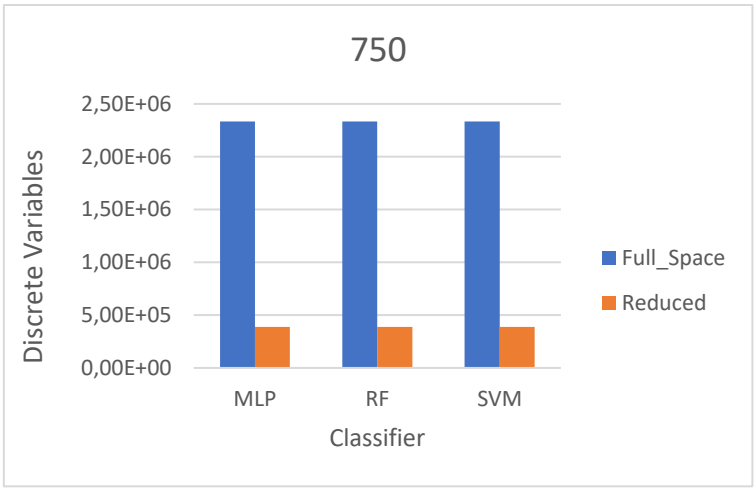

350
Constraints
5,00E+06
4,00E+06
3,00E+06
2,00E+06
1,00E+06
0,00E+00
MLP
RF
SVM
Classifier
Full_Space
Reduced

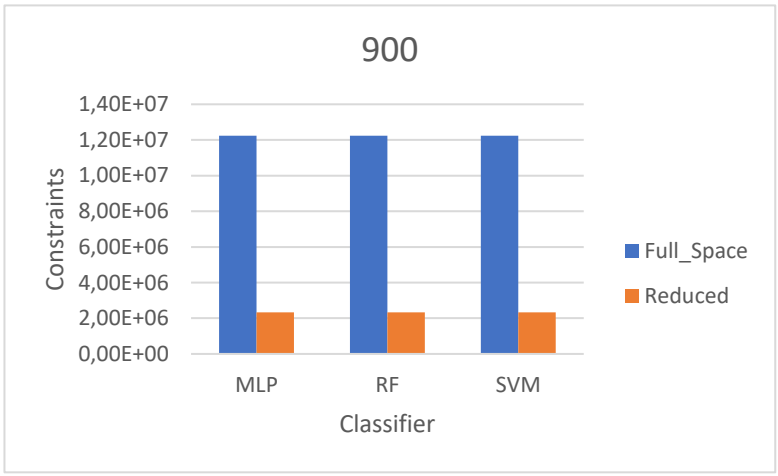

350
Discrete Variables
1,20E+06
1,00E+06
8,00E+05
6,00E+05
4,00E+05
2,00E+05
0,00E+00
MLP
RF
SVM
Classifier
Full_Space
Reduced

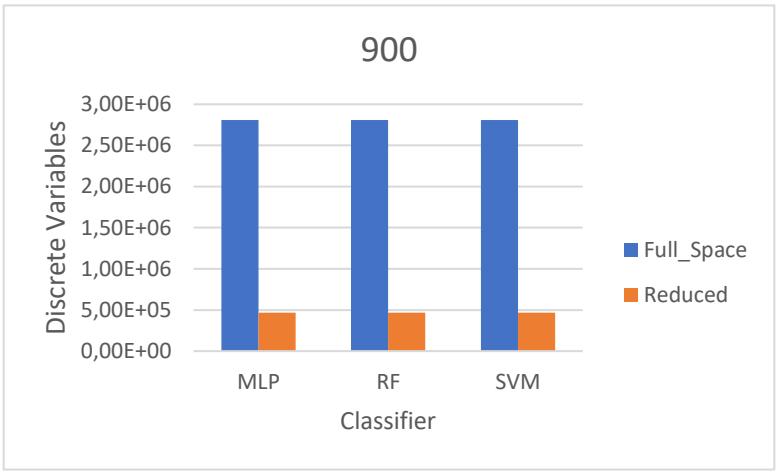

500
Constraints
8,00E+06
7,00E+06
6,00E+06
5,00E+06
4,00E+06
3,00E+06
2,00E+06
1,00E+06
0,00E+00
MLP
RF
SVM
Classifier
Full_Space
Reduced

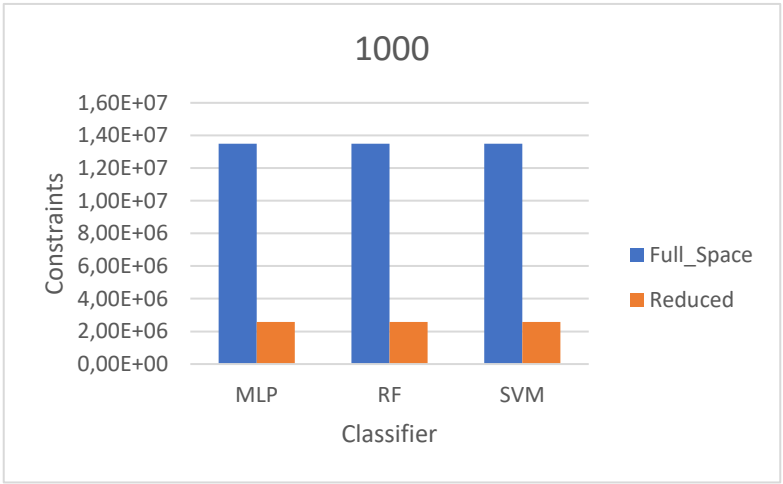

500
Discrete Variables
1,80E+06
1,60E+06
1,40E+06
1,20E+06
1,00E+06
8,00E+05
6,00E+05
4,00E+05
2,00E+05
0,00E+00
MLP
RF
SVM
Classifier
Full_Space
Reduced

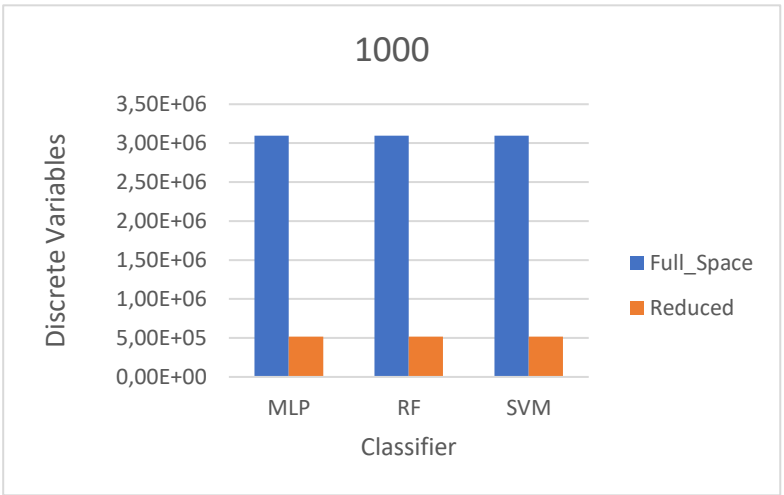

750
1,20E+07
1,00E+07
8,00E+06
6,00E+06
4,00E+06
2,00E+06
0,00E+00
Constraints
MLP
RF
SVM
Classifier
Full_Space
Reduced


750
2,50E+06
2,00E+06
1,50E+06
1,00E+06
5,00E+05
0,00E+00
Discrete Variables
MLP
RF
SVM
Classifier
Full_Space
Reduced

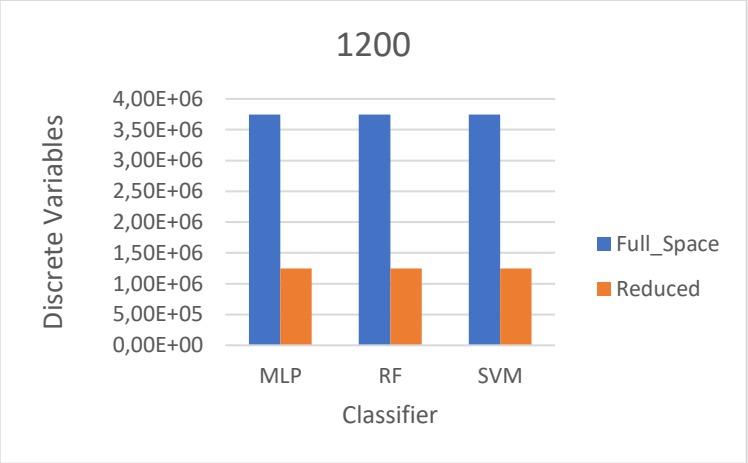

900
1,40E+07
1,20E+07
1,00E+07
8,00E+06
6,00E+06
4,00E+06
2,00E+06
0,00E+00
Constraints
MLP
RF
SVM
Classifier
Full_Space
Reduced

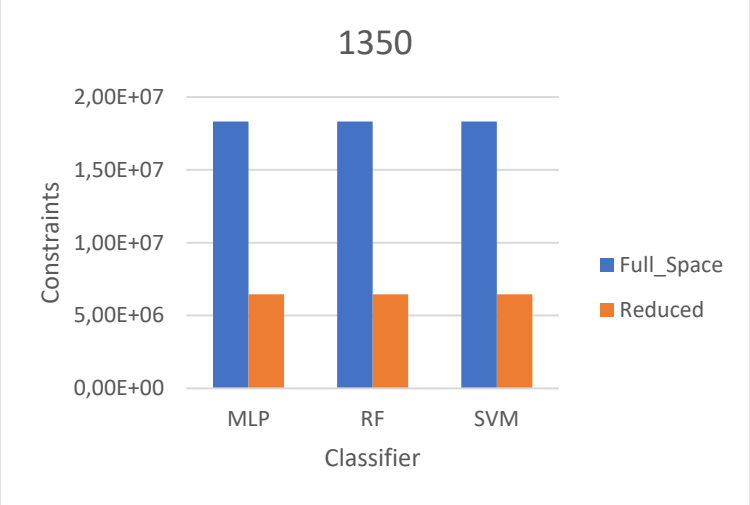

900
3,00E+06
2,50E+06
2,00E+06
1,50E+06
1,00E+06
5,00E+05
0,00E+00
Discrete Variables
MLP
RF
SVM
Classifier
Full_Space
Reduced

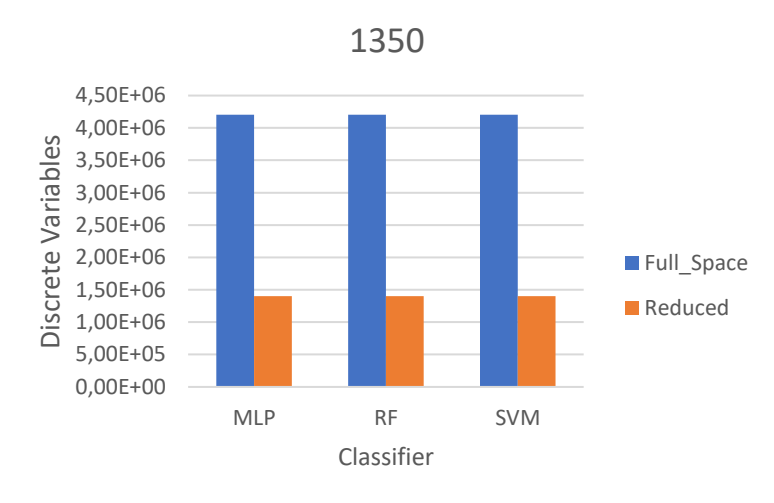

1000
1,60E+07
1,40E+07
1,20E+07
1,00E+07
8,00E+06
6,00E+06
4,00E+06
2,00E+06
0,00E+00
Constraints
MLP
RF
SVM
Classifier
Full_Space
Reduced

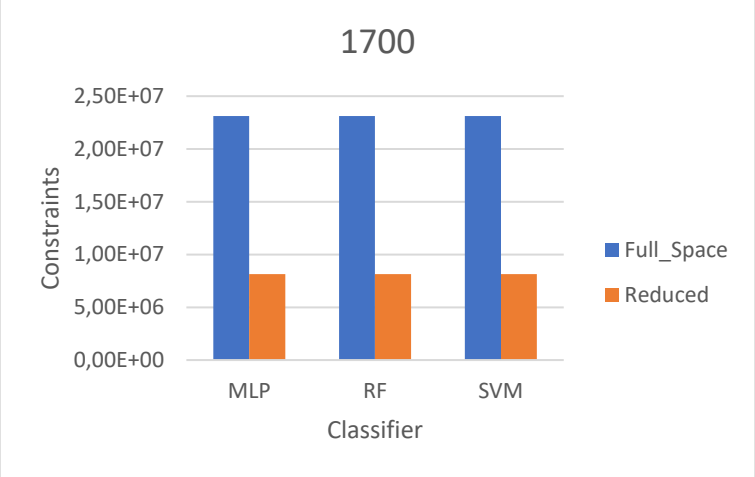

1000
3,50E+06
3,00E+06
2,50E+06
2,00E+06
1,50E+06
1,00E+06
5,00E+05
0,00E+00
Discrete Variables
MLP
RF
SVM
Classifier
Full_Space
Reduced


1200
Constraints
2,00E+07
1,50E+07
1,00E+07
5,00E+06
0,00E+00
MLP
RF
SVM
Classifier
Full_Space
Reduced


1200
Discrete Variables
4,00E+06
3,50E+06
3,00E+06
2,50E+06
2,00E+06
1,50E+06
1,00E+06
5,00E+05
0,00E+00
MLP
RF
SVM
Classifier
Full_Space
Reduced


1350
Constraints
2,00E+07
1,50E+07
1,00E+07
5,00E+06
0,00E+00
MLP
RF
SVM
Classifier
Full_Space
Reduced


1350
Discrete Variables
4,50E+06
4,00E+06
3,50E+06
3,00E+06
2,50E+06
2,00E+06
1,50E+06
1,00E+06
5,00E+05
0,00E+00
MLP
RF
SVM
Classifier
Full_Space
Reduced


1700
Constraints
2,50E+07
2,00E+07
1,50E+07
1,00E+07
5,00E+06
0,00E+00
MLP
RF
SVM
Classifier
Full_Space
Reduced

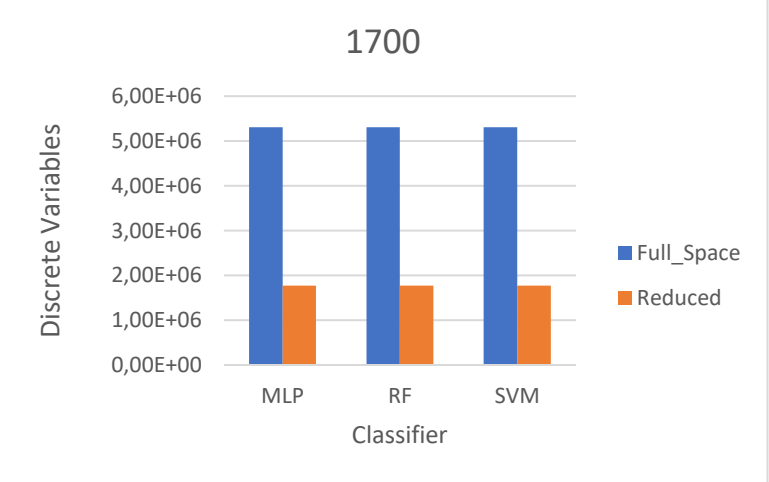

1700
Discrete Variables
6,00E+06
5,00E+06
4,00E+06
3,00E+06
2,00E+06
1,00E+06
0,00E+00
MLP
RF
SVM
Classifier
Full_Space
Reduced

Figure 30:Figure 19: Reduction of constraints and discrete variables in the MILP for different demand scenarios

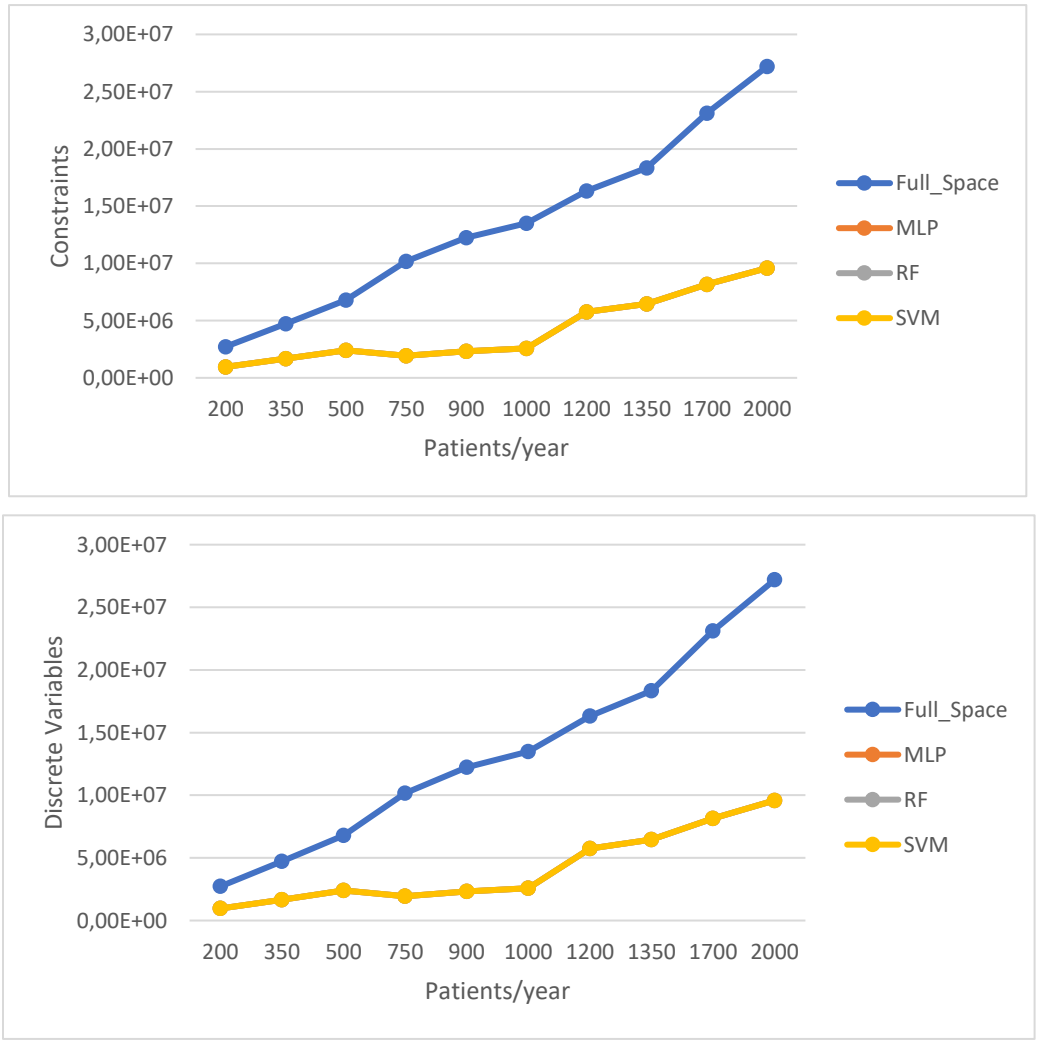


Figure 31: Number of constraints and discrete variables of the MILP model for full space and in combination with the data models for an increasing number of patients

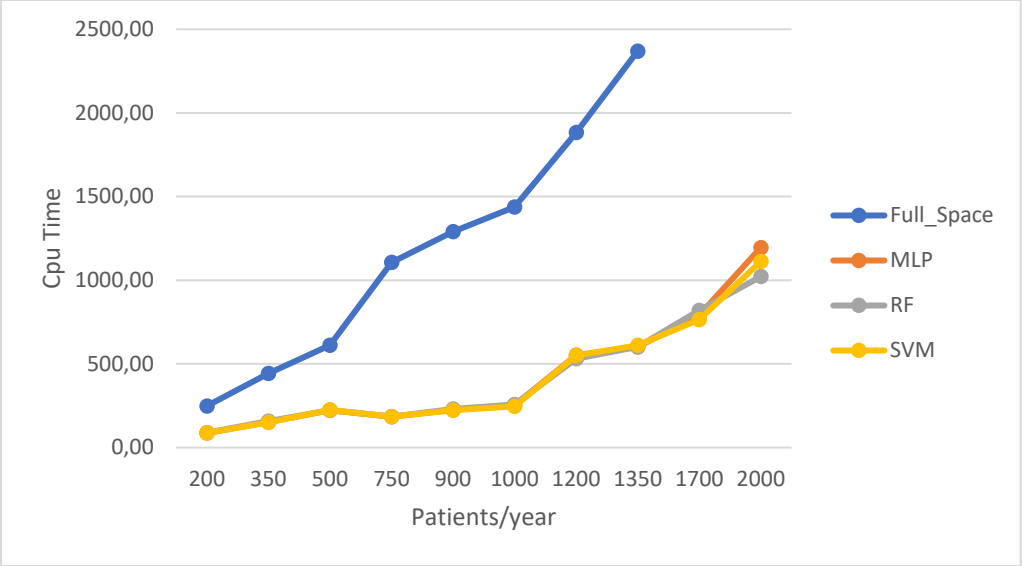


Figure 32: Cpu time (seconds) for solving the MILP model for full space and in combination with the data models for an increasing number of patients

The tables and graphs in section 4.2. show the results for the data models when they are fed, with the constrained in manufacturing facilities, MILP solutions. The input features that are used to fit the data are the Demand, Capacity, Leukapheresis site, and the Peak in Demand like before. The classifiers behave in the same manner but have better results than the unconstrained model. The data for the constrained model is easier for the data models to fit, due to the fact that the ML algorithms have to pick only two from the six manufacturing facilities rather than from all six, which is the case in the unconstrained model.

From the solution tables, it can be seen that for every test scenario the data model can always predict the best manufacturing facilities that are needed for the specific demand since the solutions after the reduction from the data model are the same as the full space model. Specifically, the MILP solutions after the prediction of the planning problem by the data models have the same cost as the full space model. Moreover, like the unconstrained model, for demands higher than 1500 patients/year, Cplex cannot find a solution due to insufficient memory. Again, after reducing the complexity of the MILP model with the ML models, solutions can be discovered. In the tables and graphs that depict the statistics for constraints, variables, and CPU times (in seconds), it is evident that

the data model does a really good job of reducing the complexity of the MILP model. For the most demand scenarios, an average 65% reduction in constraints and discrete variables can be seen, and it can reach as high as 83% (750test1, 900test1, 1000test1, 1000test3). The time of the solutions in CPU time is decreased for every scenario as well, especially in the higher demand scenarios. Lastly, all three classifiers yield approximately the same results.

Still, same as before, larger instances than 2000 therapies/year couldn't be tested, because there are no data to train the data models.

7. Conclusions and Future work

In this paper, in order to combat the high complexity and therefore the really high computational cost of CAR T-cells' supply chain optimization, which makes the solution of the optimization problem impossible to be done in high-demand scenarios, in personal computers, machine learning approaches were used. Data from the original MILP model were used to train three different classification models and these were used to solve the planning aspect of the problem (predict the manufacturing facilities that need to be established for a specific demand scenario).

Analyzing the results of the previous section can reach several conclusions. Firstly, the ML classifiers can solve the planning problem with high accuracy, especially in the constrained approach, giving the same solution as the full space model. So, the lower level (scheduling) of the problem always comes to a feasible or optimal solution for the scenarios tested. Furthermore, due to the reduction of the complexity, instances where high demand is present and the full space model cannot solve the optimization problem, the problem can be compacted to smaller instances where fewer manufacturing facilities are available and then can be solved from the MILP model. This shows that using ML algorithms to decompose the problem can lead to solving higher demand instances with the right data.

In addition, seeing as all three of the classifiers work really well with similar results, it can be concluded that the classifier and therefore the way that machine learning is being utilized does not have a big impact on the results. However, Random Forest can be trained in way less time than the other two classifiers so it can be considered that it has the advantage over the three, although when the trained model is being used, the differences are small to non-existent.

From this work, we can conclude that Machine Learning techniques can be utilized in the CAR T-cell's supply chain optimization. In this work, the classifiers were used for the planning of the supply chain, although it is a good indication that later, different ML models that predict the different decision variables of the problem can replace the MILP model altogether, when sufficient data for the supply chain are available.

Meta-Heuristic approaches can also be utilized in the optimization of the supply chain problem as well. These techniques can guide the search process strategically to find near-optimal solutions and they should be tried for optimizing this supply chain in the future.

8. Appendices

Appendix A

Mathematical Formulation of MILP model used to solve the CAR T-cell therapy supply chain optimization problem, the corresponding nomenclature, and the capacity of the manufacturing facilities

$$minTOTCOST = \sum_p CTM_p + \sum_p TTC_p + \sum_p CQC_p \quad \text{(A-1)}$$

$$CTM_p = \frac{NT * \sum_m (E1_m * (CIM_m + CVM_m))}{NP} \ , \forall p \quad \text{(A-2)}$$

$$RATIO_{m,t} = \frac{\sum_p DURM_{p,m,t}}{FCAP_m} \ , \forall \ m, t \quad \text{(A-3)}$$

$$TTC_p = \sum_{c,m,j,t} Y1_{p,c,m,j,t} * U1_{c,m,j} + \sum_{m,h,j,t} Y2_{p,m,h,j,t} * U2_{m,h,j} \ , \forall p \quad \text{(A-4)}$$

$$INC_{p,c,t} = OUTC_{p,c,t+TLS} \ , \forall p, c, t \quad \text{(A-5)}$$

$$LSR_{p,c,m,j,t} = LSA_{p,c,m,j,t+TT1_j}, \forall p, c, m, j, t \quad \text{(A-6)}$$

$$OUTC_{p,c,t} = \sum_{m,j} LSR_{p,c,m,j,t} \ , \forall p, c, t \quad \text{(A-7)}$$

$$INM_{p,m,t} = \sum_{c,j} LSA_{p,c,m,j,t} \ , \forall p, m, t \quad \text{(A-8)}$$

$$INM_{p,m,t} = OUTM_{p,m,t+TMFE+TQC}, \forall p, m, t \quad \text{(A-9)}$$

$$OUTM_{p,m,t} = \sum_{h,j} MSO_{p,m,h,j,t} \ , \forall p, m, t \quad \text{(A-10)}$$

$$FTD_{p,m,h,j,t} = MSO_{p,m,h,j,t+TT2_j} \ , \forall p, m, h, j, t \quad \text{(A-11)}$$

$$INH_{p,h,t} = \sum_{m,j} FTD_{p,m,h,j,t} \ , \forall p, h, t \quad \text{(A-12)}$$

$$CAP_{m,t} = FCAP_m - \sum_p INM_{p,m,t} \ , \forall p, m, t \quad \text{(A-13)}$$

$$\sum_p INM_{p,m,t} - \sum_p OUTM_{p,m,t} \leq CAP_{m,t} \ , \forall p, m, t \quad \text{(A-14)}$$

$$X1_{c,m} \leq E1_m \ , \forall c, m \quad \text{(A-15)}$$

$$X2_{m,h} \leq E1_m \ , \forall m, h \quad \text{(A-16)}$$

$$\sum_{c,m,j,t} Y1_{p,c,m,j,t} \leq 1, \forall p, c, m, j, t \quad \text{(A-17)}$$

$$\sum_{m,h,j,t} Y2_{p,m,h,j,t} \leq 1, \forall p, m, h, j, t \quad \text{(A-18)}$$

$$\sum_{m,j,t} Y2_{p,m,h1,j,t} \leq \sum_t INC_{p,c1,t} \ , \forall p \quad \text{(A-19)}$$

$$\sum_{m,j,t} Y2_{p,m,h2,j,t} \leq \sum_t INC_{p,c2,t} \ , \forall p \quad \text{(A-20)}$$

$$\sum_{m,j,t} Y2_{p,m,h3,j,t} \leq \sum_t INC_{p,c3,t} \ , \forall p \quad \text{(A-21)}$$

$$\sum_{m,j,t} Y2_{p,m,h4,j,t} \leq \sum_t INC_{p,c4,t} \ , \forall p \quad \text{(A-22)}$$

$$\sum_{p,h,t} INH_{p,h,t} = NP \ , \forall p, h, t \quad \text{(A-23)}$$

$$Y1_{p,c,m,j,t} \leq X1_{c,m} \ , \forall p, c, m, j, t \quad \text{(A-24)}$$

$$Y2_{p,m,h,j,t} \leq X2_{m,h} \ , \forall p, m, h, j, t \quad \text{(A-25)}$$

$$LSR_{p,c,m,j,t} \geq Y1_{p,c,m,j,t} * FMIN, \ \forall p, c, m, j, t \quad \text{(A-26)}$$

$$LSR_{p,c,m,j,t} \leq Y1_{p,c,m,j,t} * FMAX, \ \forall p, c, m, j, t \quad \text{(A-27)}$$

$$MSO_{p,m,h,j,t} \geq Y2_{p,m,h,j,t} * FMIN, \forall p, m, h, j, t \quad \text{(A-28)}$$

$$MSO_{p,m,h,j,t} \leq Y2_{p,m,h,j,t} * FMAX, \forall p, m, h, j, t \quad \text{(A-29)}$$

$$DURM_{p,m,t} = \sum_t INM_{p,m,t-1} - \sum_t OUTM_{p,m,t} + OUTM_{p,m,t} \ , \forall p, m, t \quad \text{(A-30)}$$

$$STT_p = \sum_{c,t} INC_{p,c,t} * t \ , \forall p \quad \text{(A-31)}$$

$$CTT_p = \sum_{h,t} INH_{p,h,t} * t \ , \forall p \quad \text{(A-32)}$$

$$STT_p \leq CTT_p \ , \forall p \quad \text{(A-33)}$$

$$TRT_p = CTT_p - STT_p \ , \forall p \quad \text{(A-34)}$$

$$TRT_p \leq 19 \quad \text{(A-35)}$$

$$ATRT_p = \sum_p \frac{TRT_p}{NP} \quad \text{(A-36)}$$

Equation (A-1) gives the objective function of the MILP problem, which minimizes the total cost while satisfying a series of constraints.

Equation (A-2) gives the manufacturing cost per therapy *p*.

Equation (A-3) calculates the percentage of utilization of facility *m* at time *t*.

Equation (A-4) gives the total transportation cost of all therapies *p* from leukapheresis c to manufacturing m and from manufacturing m to hospital h.

Equations (A-5) to (A-12) are the material balances of the model

Equations (A-13) to (A-14) give the capacity constraints

Equations (A-15) to (A-22) show the constraints of the network structure.

Equation (A-23) ensures that the total rate of flow of every therapy *p* arriving at hospital *h* is equal to the corresponding demand.

Equations (A-24) to (A-29) are the logical constraints for transportation flows.

Finally, Equations (A-30) to (A-36) are the time constraints.

Nomenclature:

Indices:

c	Leukapheresis sites
h	Hospitals
j	Transport modes
m	Manufacturing sites

p	Patients
t	Time points

Parameters:

$TOTCOST$	Total cost of all the therapies p
$CMIm$	Capital investment for manufacturing facility m
$CQCp$	QC cost when in house
$CVMm$	Fixed variable cost for manufacturing facility m
$TT1j$	Transport time from leukapheresis to manufacturing site via transport mode j
$TT2j$	Transport time from manufacturing site m to hospital h via transport mode j
$U1c,m,j$	Unit transport cost from leukapheresis site c to manufacturing site m via transport mode j
$U2m,h,j$	Unit transport cost from manufacturing site m to hospital h via transport mode j
$FCAPm$	Total capacity of manufacturing facility m
$INCp,c,t$	Demand therapy p arriving for leukapheresis c at time t
$FMIN$	Minimum flow
$FMAX$	Maximum flow
NP	Number of therapies
NT	Number of time points
TLS	Duration of leukapheresis
$TMFE$	Duration of manufacturing excluding QC
TAD	Duration of administration

Variables

$CTMp$	Total manufacturing cost of therapy p
$TTCp$	Total transport cost per therapy p
$OUTCp,c,t+TLS$	Therapy p leaving leukapheresis site c at time t
$LSRp,c,m,j,t$	Therapy p that is leaving leukapheresis site c and is transported to manufacturing site m via transport mode j at time t
$LSAp,c,m,j,t+TT1j$	Therapy p that left leukapheresis site c arriving at manufacturing site m via transport mode j at time t
$INMp,m,t$	Therapy p arriving at manufacturing site m at time t
$OUTMp,m,t+TMFE$	Therapy p leaving manufacturing site m at time t
$MSOp,m,q,j,t$	Therapy p leaving manufacturing site m and is transported to hospital h via transport mode j at time t
$FTDp,m,h,j,t$	Final therapy that left from manufacturing site m arriving at hospital h via transport mode j at time t
$INHp,h,t$	Therapy p arriving at hospital h at time t
$DURMp,m,t$	1 only for the time points t at which a therapy p is manufactured in facility m
$RATIOm,t$	Percentage of utilization of manufacturing site m at time t
$CAPm,t$	Capacity of manufacturing facility m at time t

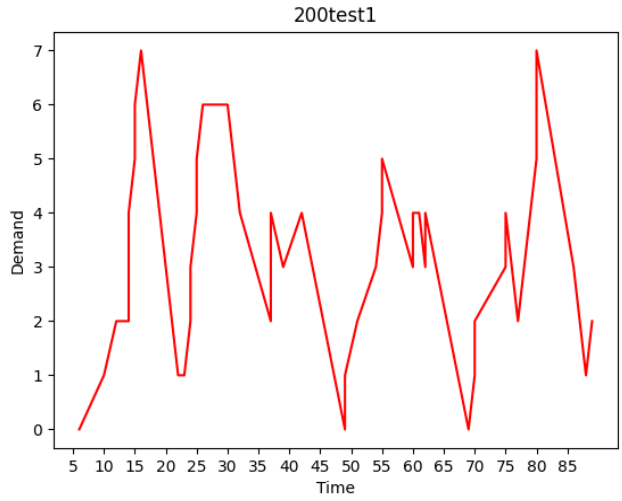
$STTp$	Starting time of treatment for patient p
$CTTp$	Completion time of treatment for patient p
$TRTp$	Total return time of therapy p
$ATRTp$	Average return time of all the therapies p
$E1m$	Binary variable to denote if manufacturing facility m is established
$X1c,m$	Binary variable to denote if a match between leukapheresis site c and manufacturing facility m is established
$X2m,h$	Binary variable to denote if a match between manufacturing facility m and hospital h is established
$Y1p,c,m,j,t$	Binary variable to denote if sample p is transferred from leukapheresis site c to manufacturing facility m via transport mode j at time t
$Y2p,m,h,j,t$	Binary variable to denote if sample p is transferred from manufacturing facility m to hospital h via transport mode j at time t

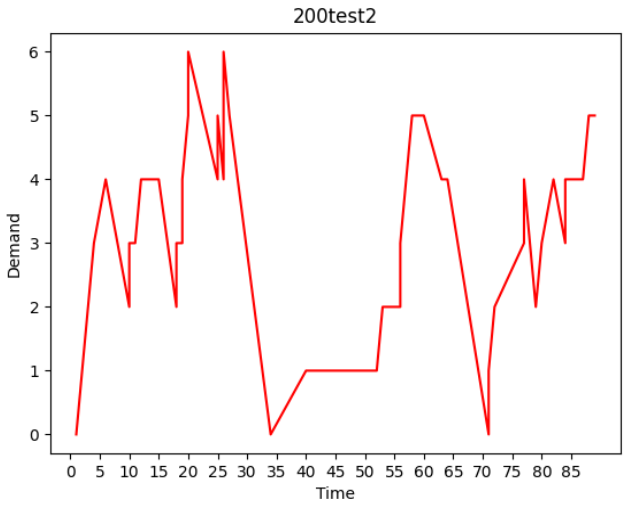
Table 36: Capacity of manufacturing facilities

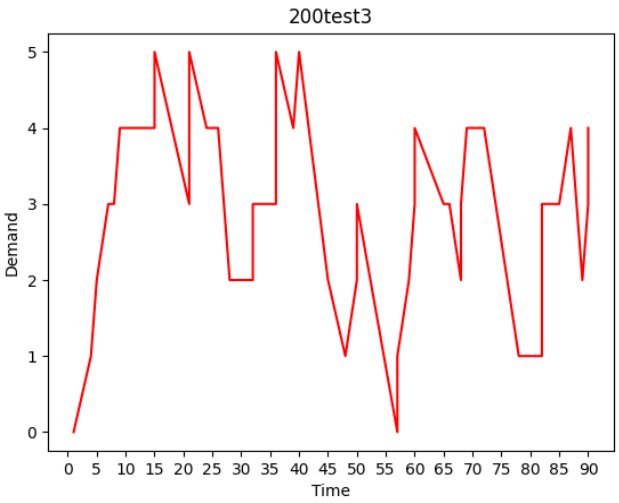
Manufacturing Facility	Capacity
m1	4
m2	31
m3	10
m4	4
m5	31
m6	10

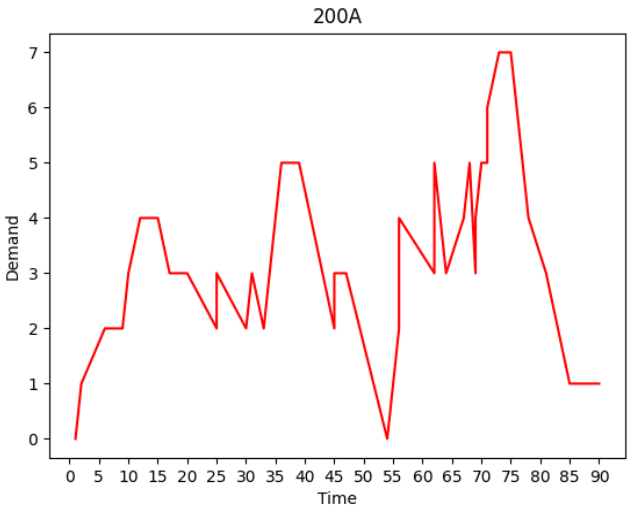
Appendix B

Demand Profiles used for testing the ML algorithms on the prediction of manufacturing facilities and the reduction of the MILP model.









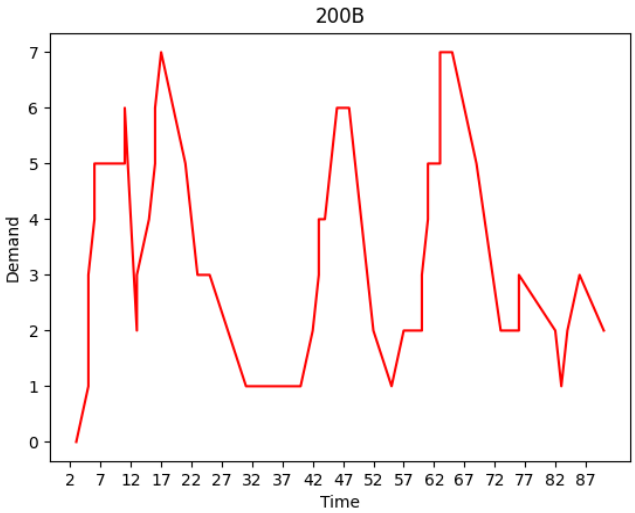
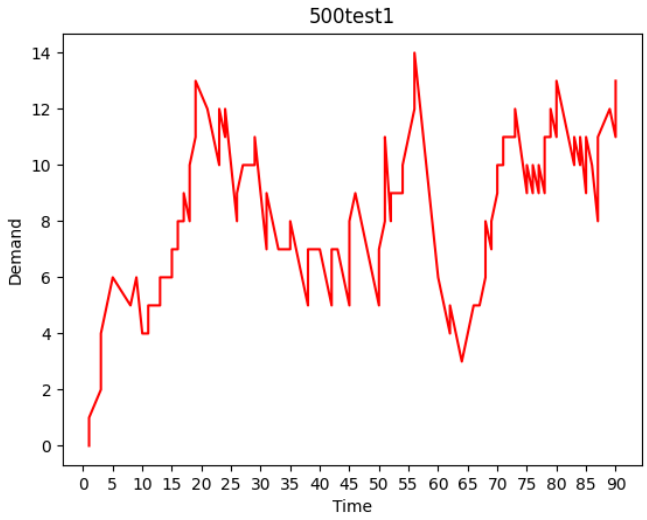
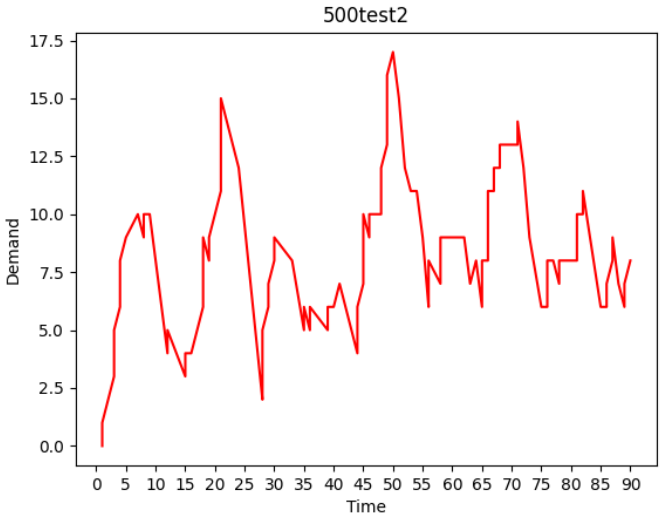
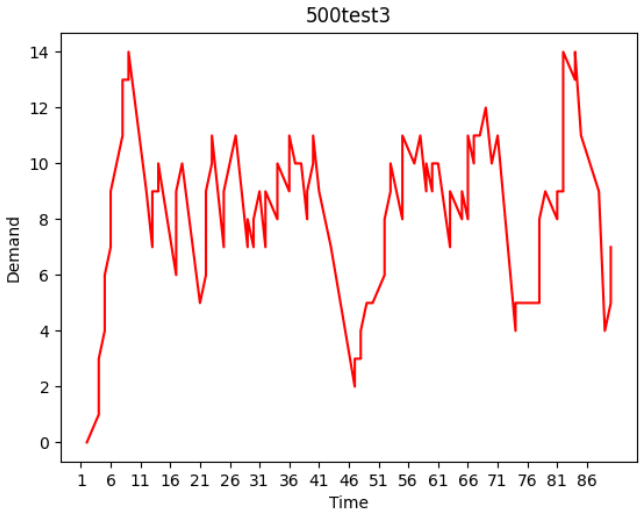
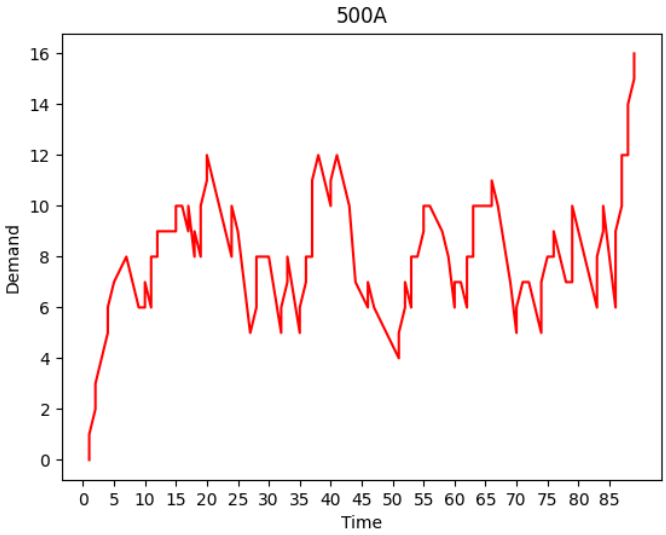


Figure 33: Demand Profiles for 200 Therapies per year









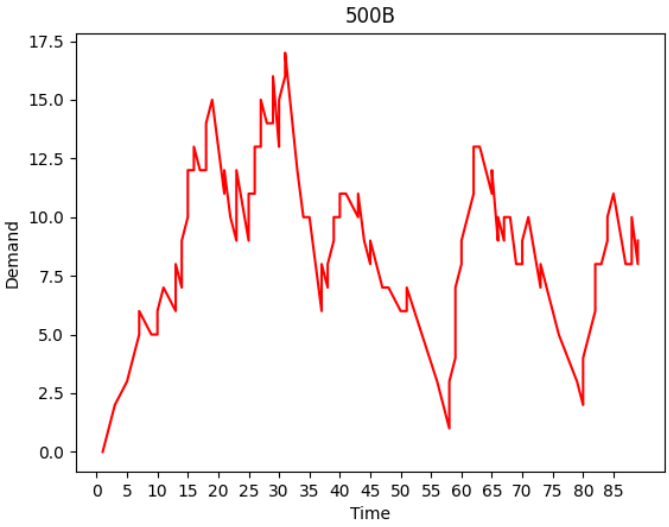
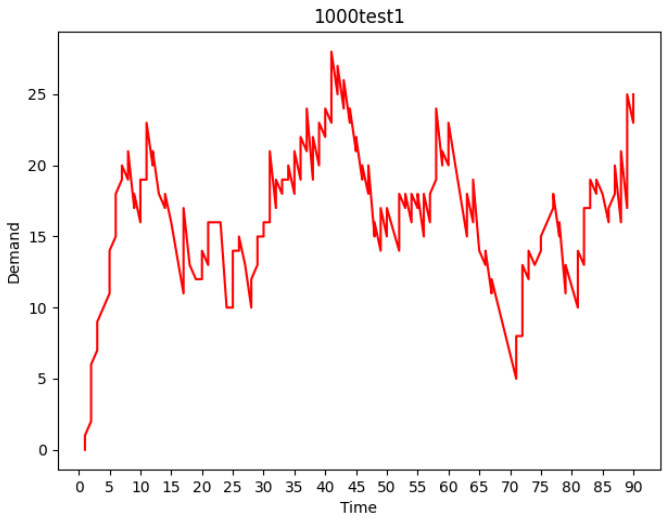
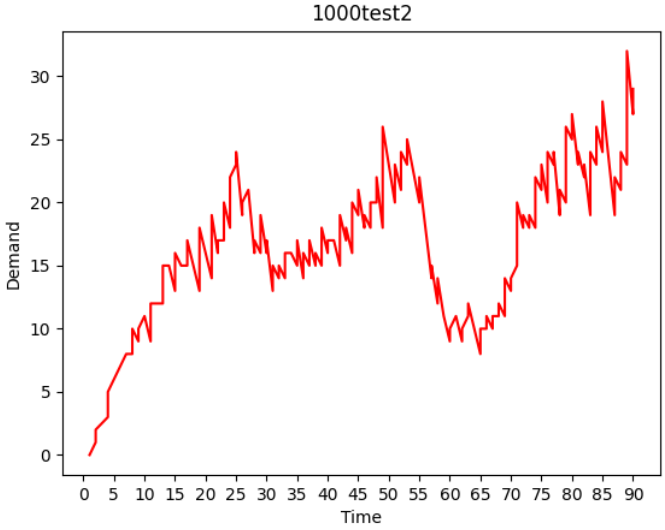
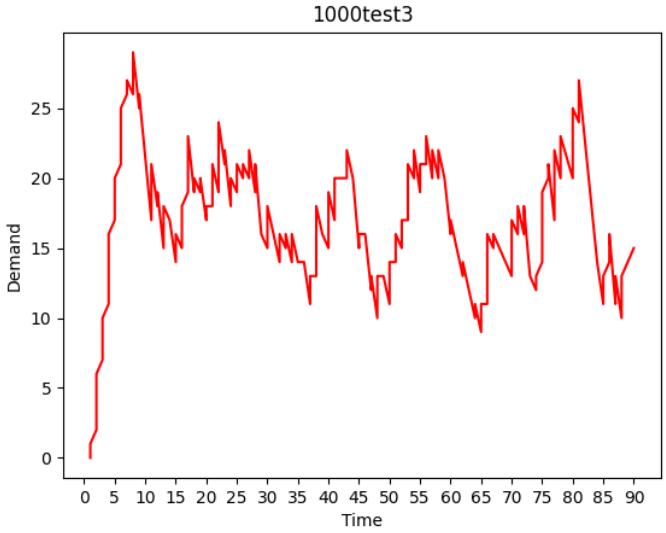
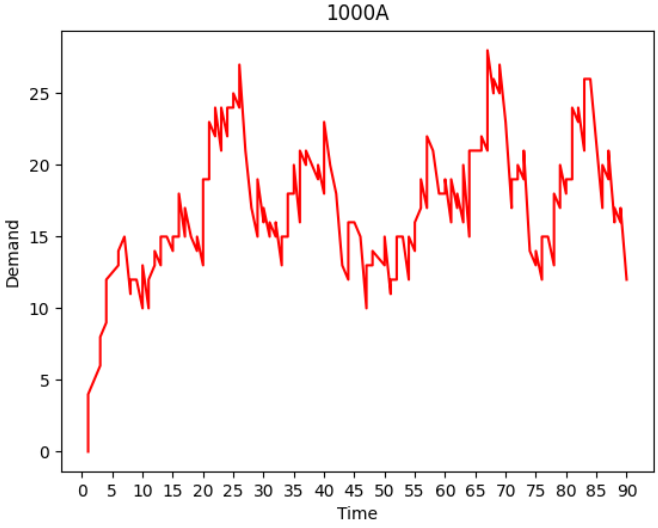


Figure 34: Demand Profiles for 500 Therapies per year









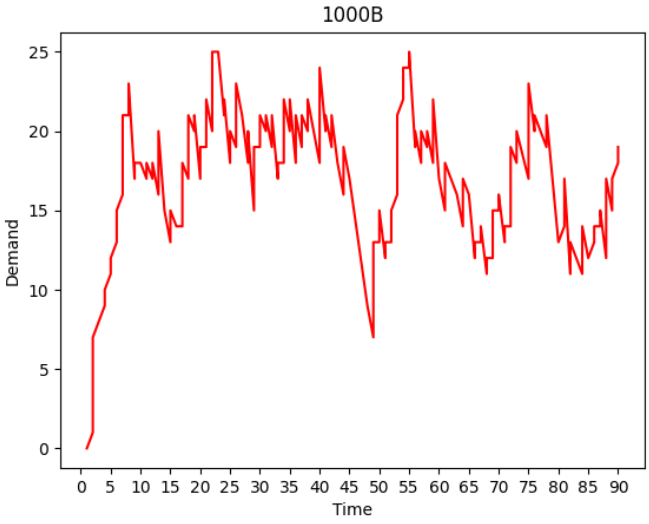


Figure 35: Demand Profiles for 1000 Therapies per year

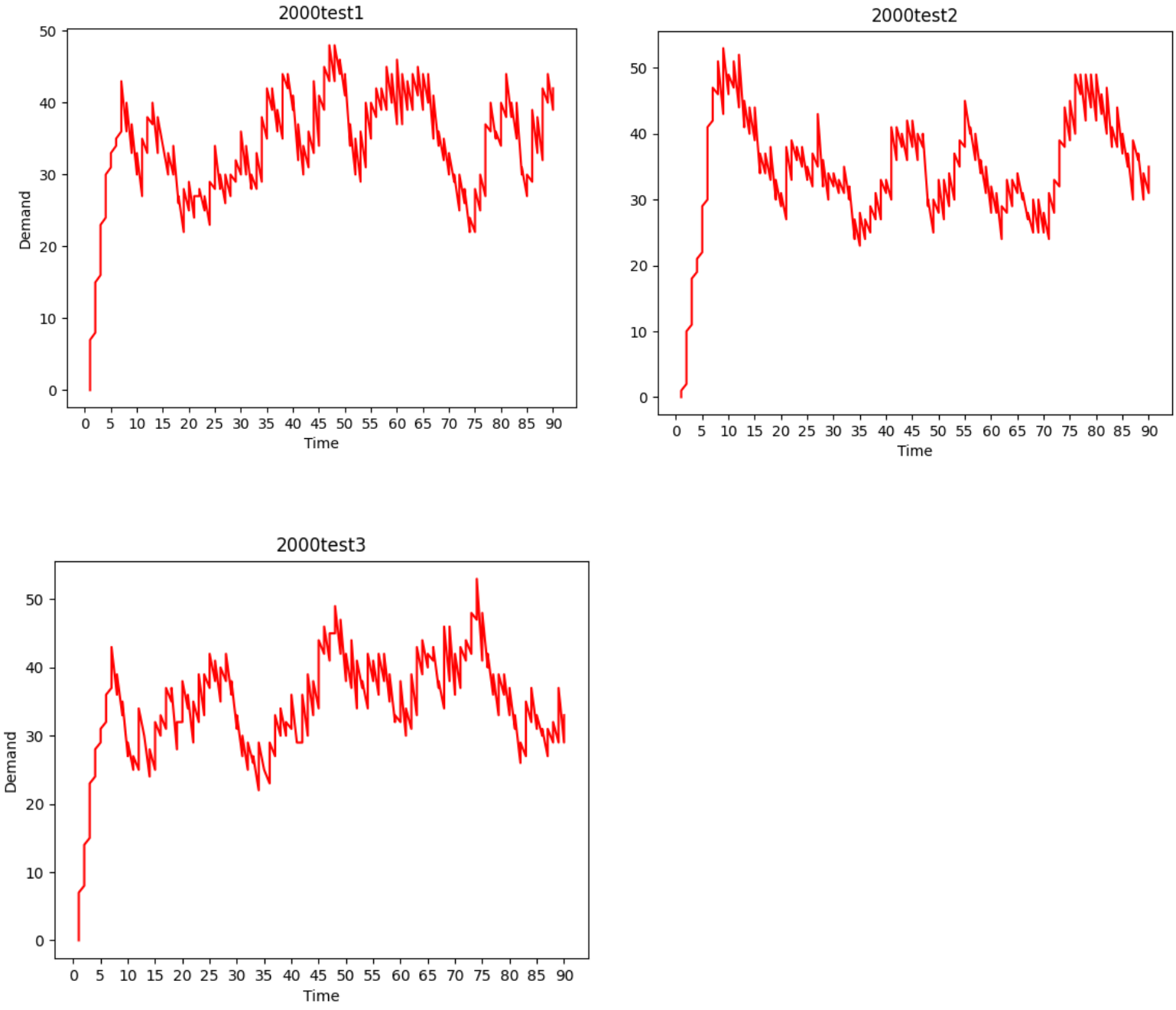
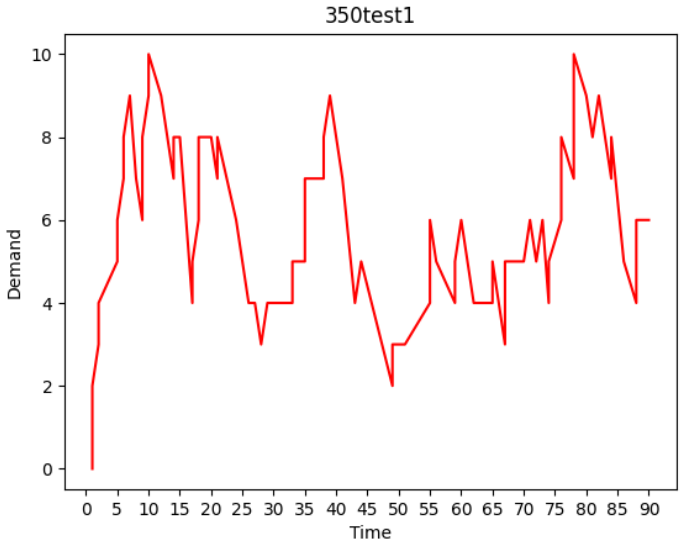
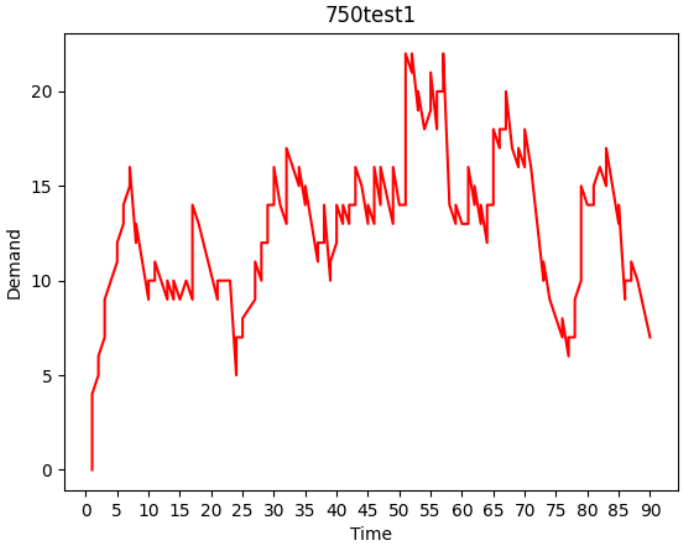
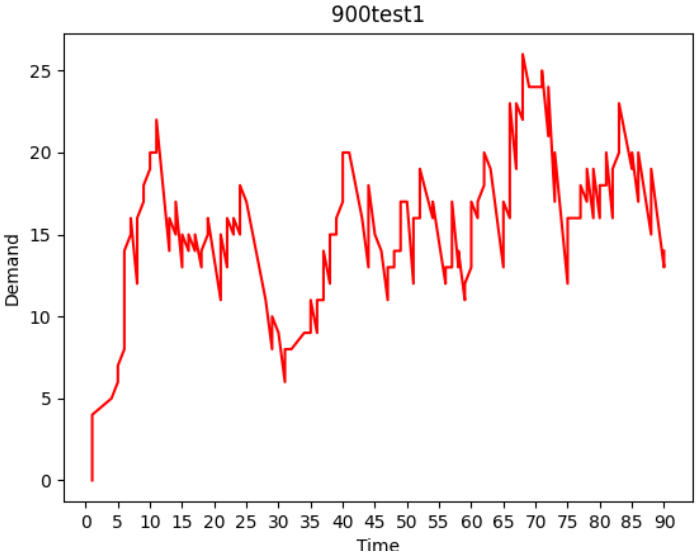
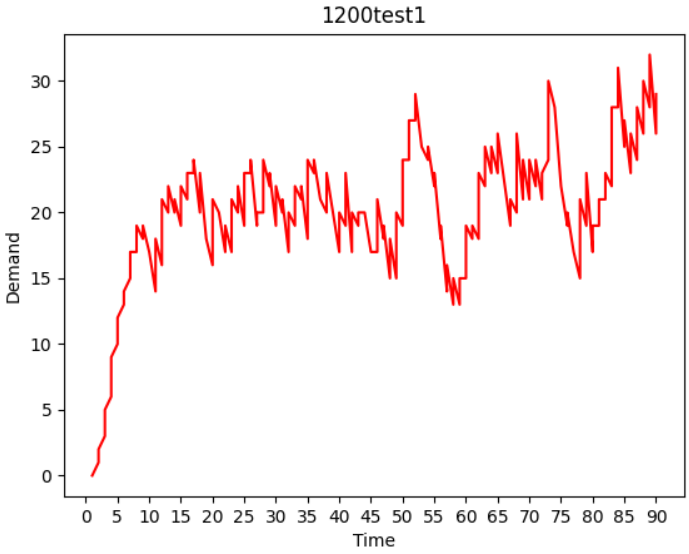


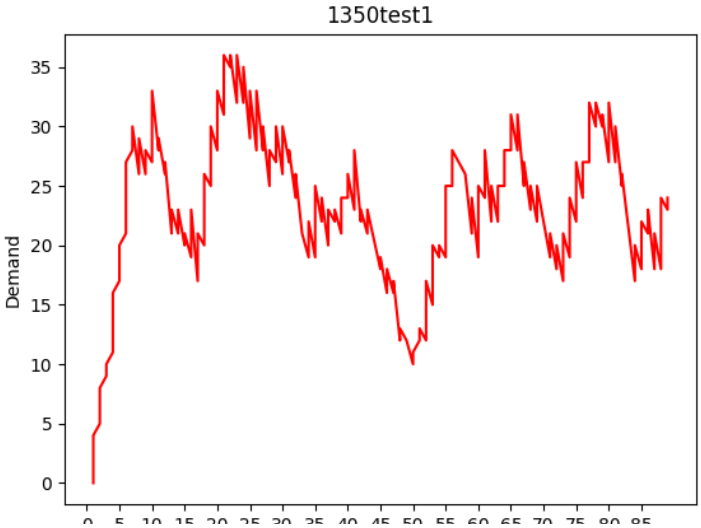
Figure 36: Demand Profiles for 2000 Therapies per year











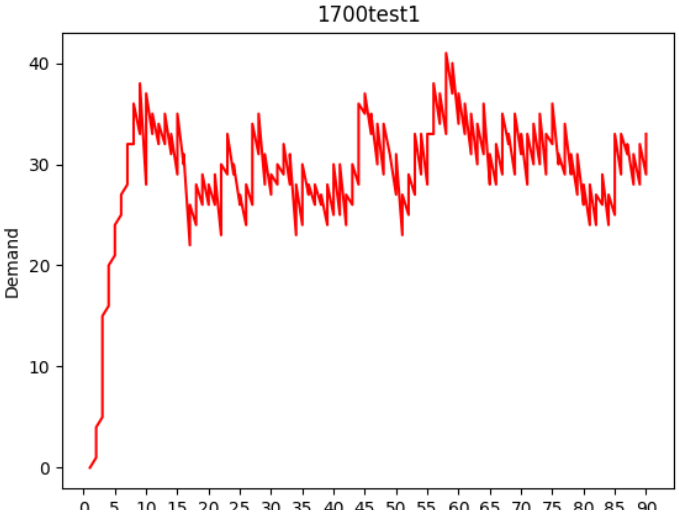


Figure 37:Demand Profiles for the rest of test scenarios

References

[1] H. Almåsbak, T. Aarvak and M. C. Vemuri, "CAR T Cell Therapy: A Game Changer in Cancer Treatment," *Journal of Immunology Research,* vol. 2016, pp. 1-10, 2016.

[2] T. R. Abreu, N. A. Fonseca, N. Gonçalves and J. N. Moreira, "Current challenges and emerging opportunities of CAR-T cell therapies," *Journal of Controlled Release,* vol. 319, pp. 246-261, 2020.

[3] S. Schuster, M. Bishop, C. Tam and e. al, "Global pivotal phase 2 trial of the CD19-targeted therapy CTL019 in adult patients with relapsed or refractory (R/R) diffuse large B-cell lymphoma (DLBCL)-an interim analysis.," *Hematological Oncology,* vol. 35, p. 27, 2017.

[4] N. C. Institute, "CAR T Cells: Engineering Patients' Immune Cells to Treat Their Cancers," National Cancer Institute, 10 3 2020. [Online]. Available: https://www.cancer.gov/about-cancer/treatment/research/car-t-cells. [Accessed 12 5 2022].

[5] M. M. Papathanasiou, C. Stamatis, M. Lakelin, S. Farid, N. Titchener-Hooker and N. Shah, "Autologous CAR T-cell therapies supply chain: challenges and opportunities?," *Cancer Gene Therapy,* vol. 27, no. 10-11, pp. 799-809, 2020.

[6] P. Karakostas, N. Panoskaltsis, A. Mantalaris and M. C. Georgiadis, "Optimization of CAR T-cell therapies supply chains," *Computers & Chemical Engineering,* vol. 139, 2020.

[7] K. Wang, Y. Liu, J. Li, B. Wang, R. Bishop, C. White, A. Das, A. D. Levine, L. Ho, B. L. Lenive and A. D. Fensak, "A multiscale simulation framework for the manufacturing facility and supply chain of autologous cell therapies," *Cytotherapy,* vol. 21, no. 10, pp. 1081-1093, 2019.

[8] S. J. W., "Apheresis Techniques and Cellular Immunomodulation," *Therapeutic Apheresis,* vol. 1, no. 3, pp. 203-206, 1997.

[9] B. L. Levine, J. Miskin, K. Wonnacott and C. Keir, "Global Manufacturing of CAR T Cell Therapy," *Molecular Therapy - Methods & Clinical Development,* vol. 4, pp. 92-101, 2017.

[10] B. L. Levine, "Performance-enhancing drugs: design and production of redirected chimeric antigen receptor (CAR) T cells," *Cancer Gene Therapy,* vol. 22, no. 2, pp. 79-84, 2015.

[11] K. Swiech, K. C. R. Malmegrim, V. Picanço-Castro, L. Chicaybam, L. Abdo and M. H. Bonamino, Generation of CAR+ T Lymphocytes Using the Sleeping Beauty Transposon System, New York: Springer US, 2020, pp. 131-137.

[12] N. Watanabe, F. Mo and M. K. McKenna, "Impact of Manufacturing Procedures on CAR T Cell Functionality," *Frontiers in Immunology,* vol. 13, 2022.

[13] S. Tyagarajan, T. Spencer and J. Smith, "Optimizing CAR-T Cell Manufacturing Processes during Pivotal Clinical Trials," *Molecular Therapy - Methods & Clinical Development,* vol. 16, pp. 136-144, 2020.

[14] A. Bernardi, M. Sarkis, N. Triantafyllou, M. Lakelin, N. Shah and M. M. Papathanasiou, "Assessment of intermediate storage and distribution nodes in personalised medicine," *Computers & Chemical Engineering,* vol. 157, 2022.

[15] S. Datta and J. P. Davim, Optimization in Industry: Present Practices and Future Scopes, Cham: Springer International Publishing, 2019.

[16] I. Griva, S. Nash and A. Sofer, Linear and nonlinear optimization, Philadelphia: Society for Industrial and Applied Mathematics, 2009.

[17] A. Jahan, K. L. Edwards and M. Bahraminasab, Multi-criteria decision analysis: for supporting the selection of engineering materials in product design, Kidlington, Oxford, UK ; Cambridge, MA, USA: Butterworth-Heinemann, 2016.

[18] I. Grossmann, L. Biegler and A. Westerberg, Systematic Methods of Chemical Process Design, New Jersey: Prentice Hall PTR, 1997.

[19] M.-H. Lin, J.-F. Tsai and C.-S. Yu, "A Review of Deterministic Optimization Methods in Engineering and Management," *Mathematical Problems in Engineering,* vol. 2012, pp. 1-15, 2012.

[20] J. Westerlund, M. Hästbacka, S. Forssell and T. Westerlund, "Mixed-Time Mixed-Integer Linear Programming Scheduling Model," *Industrial & Engineering Chemistry Research,* vol. 46, no. 9, pp. 2781-2796, 2007.

[21] "IBM ILOG CPLEX Optimization Studio CPLEX User's Manual," IBM Corporation, 2017. [Online]. Available: https://www.ibm.com/docs/en/SSSA5P_12.8.0/ilog.odms.studio.help/pdf/usrcplex.pdf.

[22] "Gurobi Optimizer Reference Manual," Gurobi Optimization, LLC, 2022. [Online]. Available: https://www.gurobi.com/documentation/9.5/refman/index.html.

[23] "FICO Xpress Optimization Help," Fair Isaac Corporation, 2022. [Online]. Available: https://www.fico.com/fico-xpress-optimization/docs/latest/solver/optimizer/HTML/chapter1.html.

[24] P. Bonissone, R. Subbu, N. Eklund and T. Kiehl, "Evolutionary algorithms + domain knowledge = real-world evolutionary computation," *IEEE Transactions on Evolutionary Computation,* vol. 10, no. 3, pp. 256-280, 2006.

[25] P. Attaviriyanupap, H. Kita, E. Tanaka and J. Hasegawa, "A hybrid EP and SQP for dynamic economic dispatch with nonsmooth fuel cost function," *IEEE Transactions on Power Systems,* vol. 17, no. 2, pp. 411-416, 2002.

[26] M. Francisco, S. Revollar, P. Vega and R. Lamanna, "A COMPARATIVE STUDY OF DETERMINISTIC AND STOCHASTIC OPTIMIZATION METHODS FOR INTEGRATED DESIGN OF PROCESSES," *IFAC Proceedings Volumes,* vol. 38, no. 1, pp. 335-340, 2005.

[27] S. Boyd, L. Xiao, A. Mutapcic and J. Mattingley, "Notes on Decomposition Methods," 2008.

[28] J. J. Cochran, L. A. Cox, P. Keskinocak, J. P. Kharoufeh and J. C. Smith, Wiley Encyclopedia of Operations Research and Management Science, Hoboken, NJ, USA: John Wiley & Sons, Inc., 2011.

[29] W. Laesanklang and D. Landa-Silva, "Decomposition techniques with mixed integer programming and heuristics for home healthcare planning," *Annals of Operations Research,* vol. 256, no. 1, pp. 93-127, 2017.

[30] V. Kesavan, R. Kamalakannan, R. Sudhakarapandian and P. Sivakumar, "Heuristic and meta-heuristic algorithms for solving medium and large scale sized cellular manufacturing system NP-hard problems: A comprehensive review," *Materials Today: Proceedings,* vol. 21, pp. 66-72, 2020.

[31] M. Christopher, Logistics and supply chain management: creating value-added networks, Harlow, England: FT Prentice Hall, 2005.

[32] N. Shah, "Process industry supply chains: Advances and challenges," *Computers & Chemical Engineering,* vol. 29, no. 6, pp. 1225-1235, 2005.

[33] B. Zahiri, J. Zhuang and M. Mohammadi, "Toward an integrated sustainable-resilient supply chain: A pharmaceutical case study," *Transportation Research Part E: Logistics and Transportation Review,* vol. 103, pp. 109-142, 2017.

[34] I. Grossmann, "Enterprise-wide optimization: A new frontier in process systems engineering," *AIChE Journal,* vol. 51, no. 7, pp. 1846-1857, 2005.

[35] E. Savadkoohi, M. Mousazadeh and S. A. Torabi, "A possibilistic location-inventory model for multi-period perishable pharmaceutical supply chain network design," *Chemical Engineering Research and Design,* vol. 138, pp. 490-505, 2018.

[36] J. Laínez, E. Schaefer and G. Reklaitis, "Challenges and opportunities in enterprise-wide optimization in the pharmaceutical industry," *Computers & Chemical Engineering,* vol. 47, pp. 19-28, 2012.

[37] R. P. Harrison, S. Ruck, Q. A. Rafiq and N. Medcalf, "Decentralised manufacturing of cell and gene therapy products: Learning from other healthcare sectors," *Biotechnology Advances,* vol. 36, no. 2, pp. 345-357, 2018.

[38] D. Moschou, M. M. Papathanasiou, M. Lakelin and N. Shah, "Investment Planning in Personalised Medicine," *Computer Aided Chemical Engineering,* vol. 48, pp. 49-54, 2020.

[39] M. Sarkis, A. Bernardi, N. Shah and M. M. Papathanasiou, "Emerging Challenges and Opportunities in Pharmaceutical Manufacturing and Distribution," *Processes,* vol. 9, no. 3, p. 457, 2021.

[40] P. M. Castro, I. E. Grossmann and Q. Zhang, "Expanding scope and computational challenges in process scheduling," *Computers & Chemical Engineering,* vol. 114, pp. 14-42, 2018.

[41] M. Mohamed-Iliasse, B. Loubna and B. Abdelaziz, "Is Machine Learning Revolutionizing Supply Chain?," in *2020 5th International Conference on Logistics Operations Management (GOL)*, Rabat, Morocco, 2020.

[42] M. Akbari and T. N. A. Do, "A systematic review of machine learning in logistics and supply chain management: current trends and future directions," *Benchmarking: An International Journal,* vol. 28, no. 10, pp. 2977-3005, 2021.

[43] D. Goettsch, K. K. Castillo-Villar and M. Aranguren, "Machine-Learning Methods to Select Potential Depot Locations for the Supply Chain of Biomass Co-Firing," *Energies,* vol. 13, no. 24, p. 6554, 2020.

[44] D. N. Duc and N. Nananukul, "A Hybrid Methodology Based on Machine Learning for a Supply Chain Optimization Problem," *Journal of Physics: Conference Series,* vol. 1624, no. 5, 2020.

[45] J. Feizabadi, "Machine learning demand forecasting and supply chain performance," *International Journal of Logistics Research and Applications,* vol. 25, no. 2, pp. 119-142, 2022.

[46] M. A. Villegas, D. J. Pedregal and J. R. Trapero, "A support vector machine for model selection in demand forecasting applications," *Computers & Industrial Engineering,* vol. 121, pp. 1-7, 2018.

[47] S. Manyathi, "An Analysis of Various Types of Supply Chain Management Systems: Case of Global Public Sector versus Private Sector Procurement," *Asian Journal of Social Sciences and Management Studies,* vol. 4, no. 1, pp. 10-19, 2017.

[48] B. Abbasi, T. Babaei, Z. Hosseinifard, K. Smith-Miles and M. Dehghani, "Predicting solutions of large-scale optimization problems via machine learning: A case study in blood supply chain management," *Computers & Operations Research,* vol. 119, 2020.

[49] F. Pedregosa, G. Varoquaux, A. Gramfort and et.al., Scikit-learn: Machine Learning in Python, JMLR 12, 2011, pp. 2825-2830.

[50] D. Paper, Hands-on Scikit-Learn for Machine Learning Applications, Logan, UT, USA: Apress , 2020.

[51] R. Garreta and G. Moncecchi, Learning scikit-learn: machine learning in Python: experience the benefits of machine learning techniques by applying them to real-world problems using Python and the open source scikit-learn library, Birmingham, UK: Packt Publishing Ltd, 2013.

[52] Y. LeCun, L. Bottou, G. Orr and K.-R. Muller, "Efficient BackProp," *Neural Networks: tricks of the trade,* 1998.

[53] T. Yiu, "Understanding Random Forest," *Towards Data Science,* 2019.

[54] D. Glenn and F. Katharina E., "CLASSIFICATION AND REGRESSION TREES: A POWERFUL YET SIMPLE TECHNIQUE FOR ECOLOGICAL DATA ANALYSIS," *Ecology,* vol. 81, no. 11, pp. 3178-3192, 2000.

[55] Breiman, "Random Forests," *Machine Learning,* vol. 45, no. 1, pp. 5-32, 2001.

[56] W. Ting-Fan, L. Chih-Jen and W. Ruby C., "Probability Estimates for Multi-class Classification by Pairwise Coupling," *Journal of Machine Learning Research 5,* pp. 975-1005, 2004.

[57] K. Crammer and Y. Singer, "On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines," *Journal of Machine Learning Research 2,* pp. 265-292, 2001.

[58] C. Chih Chung and L. Chih-Jen, "A Library for Support Vector Machines," Taipei, Taiwan, 2021.

[59] X. Wang, "Influence of feature scaling on convergence of gradient iterative algorithm," *Journal of Physics: Conference Series,* p. 032021, 2019.

[60] A. Zheng and A. Casari, in *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*, Sebastopol, O'Reilly Media, Inc, 2018.

[61] G. E. Batista, M.-C. Monard and A. L. C. Bazzan, "Balancing Training Data for Automated Annotation of Keywords: a Case Study," in *II Brazilian Workshop on Bioinformatics*, Macaé, 2003.

[62] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research 16 ,* vol. 16, pp. 321-357, 2002.

[63] D.-S. Huang, X.-P. Zhang and G.-B. Huang, "Borderline-SMOTE: A New Over-Sampling Method in Imbalanced," in *Advanced in Intelligent Computing*, Hefei, China, Springer, 2005, pp. 878-887.

[64] E. At, M. Aljourf, F. Al-Mohanna and M. Shoukri, "Classification of Imbalance Data using Tomek Link (T-Link) Combined with Random Under-sampling (RUS) as a Data Reduction Method," *Global Journal of Technology and Optimization,* vol. 01, no. S1, 2016.

[65] R. M. Pereira, Y. M. Costa and C. N. Silla Jr., "MLTL: A multi-label approach for the Tomek Link undersampling algorithm," *Neurocomputing,* vol. 383, pp. 95-105, 2020.

[66] D. Devi, S. k. Biswas and B. Purkayastha, "Redundancy-driven modified Tomek-link based undersampling: A solution to class imbalance," *Pattern Recognition Letters,* vol. 93, pp. 3-12, 2017.

[68] C. Han and Q. Zhang, "Optimization of supply chain efficiency management based on machine learning and neural network," *Neural Computing and Applications,* vol. 33, no. 5, pp. 1419-1433, 2021.

[69] H. Kartal, A. Oztekin, A. Gunasekaran and F. Cebi, "An integrated decision analytic framework of machine learning with multi-criteria decision making for multi-attribute inventory classification," *Computers & Industrial Engineering,* vol. 101, pp. 599-613, 2016.

[70] S. Rafiq, C. S. Hackett and R. J. Brentjens, "Engineering strategies to overcome the current roadblocks in CAR T cell therapy," *Nature Reviews Clinical Oncology,* vol. 17, no. 3, pp. 147-167, 2020.

[71] M. Sadelain, R. Brentjens and I. Rivière, "The Basic Principles of Chimeric Antigen Receptor Design," *Cancer Discovery,* vol. 3, no. 4, pp. 388-398, 2013.

[72] M. Sadelain, R. Brentjens, I. Rivière and J. Park, "CD19 CAR Therapy for Acute Lymphoblastic Leukemia," *American Society of Clinical Oncology Educational Book,* no. 35, pp. 360-363, 2015.

[73] G. Zhou and H. Levitsky, "Towards Curative Cancer Immunotherapy: Overcoming Posttherapy Tumor Escape," *Clinical and Developmental Immunology,* vol. 2012, pp. 1-12, 2012.

[74] R. C. Bansal, "Optimization Methods for Electric Power Systems: An Overview," *International Journal of Emerging Electric Power Systems,* vol. 2, no. 1, 2005.

[75] L. G. Papageorgiou and E. S. Fraga, "A mixed integer quadratic programming formulation for the economic dispatch of generators with prohibited operating zones," *Electric Power Systems Research,* vol. 77, no. 10, pp. 1292-1296, 2007.

[76] J. Brownlee, Probability for Machine Learning: Discover How To Harness Uncertainty With Python, 2019.