ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
Қ.И.Сәтбаев атындағы Қазақ ұлттық техникалық университеті
Ақпараттық және теллекоммуникациялық технологиялар институты
Математатика кафедрасы

Қалмырзаев Уалихан Нұржанұлы

Машинды оқыту алгоритмдері және оның қосымшалары. Кванттық машинды оқыту.

ДИПЛОМДЫҚ ЖҰМЫС

5В070500-Математикалық және компьютерлік модельдеу

Алматы 2019

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Қ.И.Сәтбаев атындағы Қазақ ұлттық техникалық университеті

Ақпараттық және теллекоммуникациялық технологиялар институты

Математатика кафедрасы

ҚОРҒАУҒА ЖІБЕРІЛДІ

Кафедра меңгерушісі

ф.-м.ғ.к, доцент

_____Кельтенова Р.Т

«23» мамыр 2019ж.

ДИПЛОМДЫҚ ЖҰМЫС

Тақырыбы: Машинды оқыту алгоритмдері және оның қосымшалары. Кванттық машинды оқыту.

5В070500-Математикалық және компьютерлік модельдеу

Орындаған: Қалмырзаев Уалихан Нұржанұлы

Рецензент,

т.,ғ.к.,профессор

_____ Сатыбалдиева Р.Ж.

«23» мамыр 2019ж.

Ғылыми жетекші,

сениор-лектор

_____ Ергазина Р.А.

«23» мамыр 2019ж.

Алматы 2019

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
Сәтбаев университеті
Ақпараттық телокоммуникациялық технологиялар институты
Математика кафедрасы
5В070500-«Математикалық және компьютерлік модельдеу»

ҚОРҒАУҒА ЖІБЕРІЛДІ

кафедра меңгерушісі
ф.-м.ғ.к, доцент
_____Р.Т.Кельтенова

«23» 05 _____2019ж.

Дипломдық жұмысты орындауға
ТАПСЫРМА

Білім алушы: Қалмырзаев Уалихан Нұржанұлы
Тақырыбы: «Машинды оқыту алгоритмдері және оның қосымшалары. Кванттық машинды оқыту»
Университеттің бұйрығымен бекітілген №___ «__» ____2019 ж.
Жұмысты тапсыру мерзімі:«25» 05 2019 ж.

Дипломдық жұмысқа бастапқы деректер:
1) Жасанды интеллекттің бөлімі – машинды оқытуға кіріспе. Оның қазіргі әлемдегі рөлі және қолдану аясын зерттеу.
2) Машинды оқытудың support vector machine алгоритмінің математикалық базасына талдама.
3) Support vector machine алгоритмінің қолданысы мен оның python тілінде реализациясы.
4) Кванттық компьютер және оның қасиеттері.
5) Support vector machine алгоритмінің – кванттық компьютердегі реализациясы.

Графикалық материал тізбесі: Жоғары дәрежелі Python программалау тілінде және оның кітапханаларында жұмыс.
Ұсынылатын негізгі әдебиеттер:

[1]. Wittek, Peter, “Quantum Machine Learning: What Quantum Computing Means to Data Mining”, Academic Press; Paperback reprint of hardcover 1st ed., 2014 edition (September 2, 2016)

[2]. Nielsen, Michael A.; Chuang, Isaac L. (2010). Quantum Computation and Quantum Information (2nd ed.). Cambridge: Cambridge University Press.

[3]. Вапник В. Н. Восстановление зависимостей по эмпирическим данным. — М.: Наука, 1979. — 448 с.

Дипломдық жұмысты дайындау

КЕСТЕСІ

Бөлімдердің атауы,әзірленетін сұрақтар тізбесі	Ғылыми жетекші мен кеңесшіге көрсету мерзімі	Ескертпелер
1 Кіріспе	13.02.2019	
2 Негізгі бөлім	25.03.2019	
3 Қорытынды	15.04.2019	

Дипломдық жұмыс бөлімдерінің кеңесшілері мен норма бақылаушының аяқталған жұмысқа қойған

қолтаңбалары

Бөлімдердің атаулары	Ғылыми кеңесші, аты-жөні (ғылыми дәреже,атағы)	Қол қойған мерзімі	Қол таңбасы
Нормабақылаушы	к.ф.-м.н., сениор-лектор Шатманов Ж.Ж.	23.05.2019г.	

Ғылыми жетекшісі _____/Р.А.Ергазина /

Тапсырманы орындауға алған білім алушы _____/У.Н.Қалмырзаев

Күні “15” мамыр 2019 ж.

АҢДАТПА

Дипломдық жұмыс мақсаты тірек векторлар методын қолданып классификация есебін шешу арқылы объекті анықтамасымен айналысатын кроссплатформды қосымша құру және тірек векторлар методының кванттық компьютердегі реализациясын ядролық матрица құру және классификация есебін шешу арқылы жасау.

Сонымен қатар, python тілін және оның кітапханаларын қолдана отырып нәтижелерді визуалдау.

АННОТАЦИЯ

Целью дипломной работы является создание кроссплатформенного дополнения к объектно-ориентированному объекту путем решения классификации с использованием метода опорных векторов и создания области выпрямительных векторов на квантовом компьютере путем создания ядерной матрицы и решения задачи классификации.

Кроме того, визуализация результатов, используя язык Python и его библиотеки.

ANNOTATION

The aim of this diploma work is to create a cross-platform application to an object-oriented object by solving the classification using the support vector method and creating a region of rectifier vectors on a quantum computer by creating a nuclear matrix and solving the classification.

Also, visualizing the results of researches by using python language and its libraries.

МАЗМҰНЫ

Кіріспе 8
1 Машинды оқыту алгоритмдеріне жалпы сипаттама 10
1.1 Машинды оқыту. Оның категориялары, түрлері мен типтері 10
1.2 Машинды оқыту негізгі түсініктері 13
1.3 Машинды оқытудың шешетін тапсырмалары 14
2 Классикалық тірек векторлар машинасы (svm) алгоритмі 16
2.1 SVM алгоритмінің негіздері 16
2.2 SVM алгоритмі мен grayscale-классификациясын қолданып қосымша мен оның интерфейсін жасау 22
3 SVM алгоритмінің кванттық компьютерде реализациясы 26
3.1 Кванттық компьтерлер түсінігі және оның болашақтағы рөлі 26
3.2 Кванттық есептеулер 28
3.3 Quantum SVM және оның питон тілінде және IBMQ кванттық компьютерінде реализациясы 31
Қорытынды 38
Пайдаланылған әдебиеттер 39

КІРІСПЕ

Қазіргі заманда бүкіл әлемде жоғары технологиялар даму кезеңі жүріп жатыр. Осы заманда адамзат өз алдына қойған мақсаттарды тек адам миын қолдана ғана қоймай оған қоса машинаның ойлау қасиеттерін қолдануды іске асыру аса тиімді болып тұр. Жасанды интеллект пен машинды оқыту салалары аса қарқынды дамуда, ол оған дейін тек адам прерогативасы болып саналған тапсырмаларды шешуге мүмкіндік береді. Машинды оқыту – өндірістегі және басқа да салалардағы процестерді оптимизациялау мен автоматтандырудың басты жолы болып табылады.

Жаңа тенденцияларға қарайтын болсақ жоғары технологиялардың адамзат өміріндегі рөлі күннен күнге артып келе жатыр. Жоғары технологияларға күрделі электротехникалық құрылғылар, микропроцессорлар және т.б. жатады. Соның ішінде аса айта кететін зат – микропроцессорлар. Қазіргі әлемде микропроцессорлар саласында біраз мәселелер пайда болды. XX – ғасырдың 50-60-жылдарынан бастап 2010-жылдарға дейін микропроцессорлар әлемінде Мур заңы жұмыс істеп келген, яғни ол бойынша – әрбір 24 ай сайын интегралды схемадағы транзисторлар саны 2 есе көбееді. Оны атақты американ ғалымы – Джон Мур енгізген. Алайда транзисторлар саны көбейген сайын, олардың өлшемі кішірейген сайын – классикалық Ньютон заңдары істен шыға бастайды да, микромир, яғни кванттық әлем заңдары істей бастайды. Оның себебі – транзисторлардың тым кішірейгені, яғни макромир заңдары істемей, микромир заңдары, яғни кванттық физиканың істей бастауы болып табылады. Сондықтан да қазіргі бүкіл әлемдегі ғылыми институттар мен IT-корпорациялардың назары – кванттық компьютерлер мен кванттық технологияларда жатыр.

Сол үшін кванттық компьютерлерде атқарылатын кванттық есептеулер мен оның жеке жағдайы – кванттық машинды оқыту – нағыз болашақ технологиясы болып табылады. Алайда бүкіл артықшылықтарына қарамастан бұл технологиялар қазіргі таңда нашар зерттелген болып табылады.

Кванттық машинды оқытудың басты артықшылықтарына – классикалық аналогтарына қарағанда әжептеуір уақыт ұтуға мүмкіндік береді. Яғни классикалық алгоритмдерге қарағанда экспоненциалды немесе квадратты түрде жылдамдық арттыруын беруі мүмкін.

Дипломдық жұмыстың негізгі міндеттері:

1) Машинды оқыту туралы толық мәлімет жинақтай отырып, оның қазіргі әлемдегі қолдану мүмкіндіктерін қарастыру.

2) SVM машинды оқыту алгоритмін зерттеп, оның математикалық негіздемесіне баға беру.

3) SVM алгоритмі мен grayscale-классификациясын қолданып қосымша жасау.

4) Кванттық компьютерлер мен кванттық есептеулерді және олардың қолдану перспективаларын зерттеу.

5) SVM алгоритмінің кванттық компьютердегі реализациясын python тілі арқылы құру.

6) Тірек векторлар методының кванттық нұсқасының – классикалық нүсұасынан айырмашылықтарын зерттеу.

Жобаның өзектілігі: Машинды оқыту алгоритмдері мен оның категорияларына, шешетін есептеріне анализ жасап, оның қолданылу аяларын зерттеп, машинды оқыту алгоритмінің бірі – тірек векторлар методын(Support Vector Machine) қолданып қосымша жасау, оған интерфейс ойлап табу. Кванттық машинды оқытудың қазіргі әлемдегі рөлі мен оның мүмкіндіктерін зерттеу, кванттық тірек векторлар методын қолданып бинарлы классификация есептерін зерттеу және оның кванттық компьютерде атқарылатын реализациясын құру. Оның классикалық нұсқасынан айырмашылығын зерттеу.

1 МАШИНДЫ ОҚЫТУ АЛГОРИТМДЕРІНЕ ЖАЛПЫ СИПАТТАМА

1.1 Машинды оқыту. Оның категориялары, түрлері мен типтері

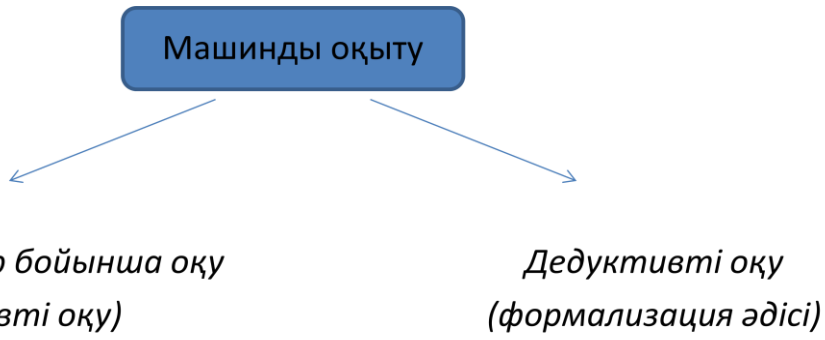
Қазіргі жоғары технологиялар заманында өзінің алдына қойылған белгілі бір есептерді, тапырмаларды адамдар тек өз миымен ғана емес, оған қоса компьютер арқылы, яғни компьютер өзін-өзі оқыту алгоритмдерін қолдана отырып та шешуге мүмкіндіктер пайда бола бастады. Болашақта бұл адам өмірінің көптеген салаларын автоматтандыруға, оптимизациялауға септігін тигізеді.

Жасанды интеллект – деп әдетте адамның интеллектуалды түсінігін талап ететін, бұған дейін тек адам прерогативасы болып табылған тапсырмаларды шешуге мүмкіндік беретін методтар мен операцияларды дамытуға арналған ғылым мен технология саласы болып табылады.

Машинды оқыту – жасанды интеллекттің белгілі бір тапсырманы тура шешумен емес, оны шешу кезінде оқи алатын арнайы методтерінің класы, саласы болып табылады. Басқаша сөзбен айтқанда бұл оқуға қабілетті алгоритмдерді құру әдістерін зерттейтін жасанды интеллекттің кең тарауы.

Машинды оқытудың қолданылу аялары өте көп болып табылады: суретте адам бетін анықтаудан бастап, шахмат, го ойындарында адамды жеңуге қолданылатын алгоритмдерге дейін. Оған қоса бұл сала коммерциялық жағынан өте перспективті болып табылады, яғни машинды оқыту алгоритмдерін – веб-қосымшаларда, нативті немесе кроссплатформды қосымшаларда да алуан түрлі тапсырмаларды орындау кезінде қолдануға болады.

Машинды оқытуды классикалық классификация бойынша 2 категорияға бөлуге болады:



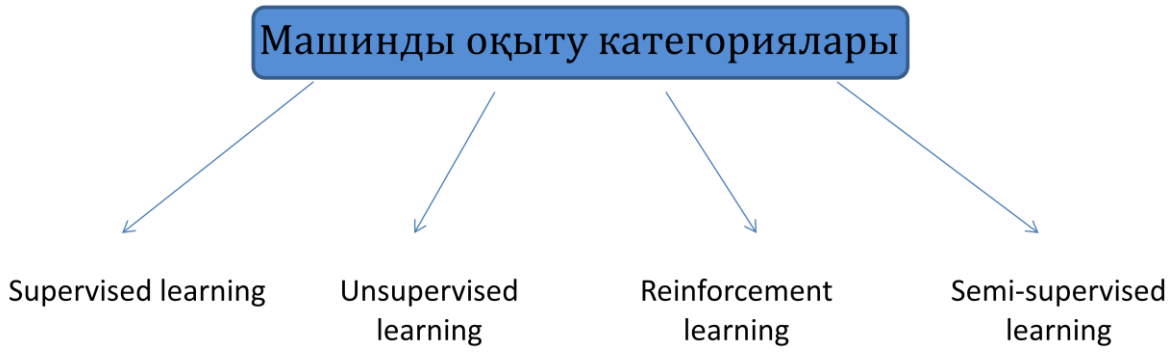
1.1 Сурет – Машинды оқыту категориялары.

Прецеденттер бойынша оқу дегеніміз – белгілі бір жеке жағдайлар бойынша белгілі бір жалпы заңдылықтарды іздейтін алгоритмдер тобы.

Оның екінші атауы – индуктивті оқу, яғни логиканың индукция методын қолдану. Эмпирикалық жолмен табылған мәліметтен бүкіл системаға тән жалпы заңдылықтарды анықтау.

Дедуктивті оқу немесе формализация деп – прецедентті оқуға қарама-қарсы, яғни жалпыдан – жеке заңдылықтарды таба білетін алгоритмдер класы. логиканың дедукция әдісін қолданады.

Мәселенің жалпы тұжырымдамасы: белгілі бір объектілер (жағдайлар) жиыны және мүмкін жауаптар (реакциялар) жиыны бар делік. Жауаптар мен объектілер арасындағы өзара байланыс, заңдылық, тәуелділік бар, бірақ ол белгісіз. Алғашында бізге тек прецденттердің түпкілікті жиынтығы - оқыту іріктемесі деп аталатын «объект, жауап» жұптары ғана белгілі. Осы алғашқы деректерді қолдана отырып екі жиын арасындағы тәуелділікті қалпына келтіру керек, яғни кез-келген кіріс мәлімет үшін максималды дәл классификациялық жауап табуға қабілетті алгоритм салу.



1.2 Сурет – Машинды оқыту категориялары.

Supervised learning (Мұғаліммен оқыту) – келесі есепті шешетін машинды оқыту саласы: белгілі бір объектілер жиыны болсын және белгілі бір жауаптар жиыны болсын. Және де осы екі жиын арасында белгілі бір заңдылық, тәуелділік бар, бірақ ол белгісіз. Бізге тек оқыту іріктемесі деп аталатын жауап-объект жиындары болып табылатын прецеденттер ғана белгілі. Бізге осы тәуелділікті қайта құру керек, яғни кез-келген жаңа объектіге дәлдігі жеткілікті түрде болатындай жауап жұбын табатын алгоритм құру. Ең соңында жауаптардың дәлдігін тексеру үшін – сапа функционалын, яғни метриканы енгізу керек. Классификация және регрессия есептерін шығару үшін қолданылады.

Есептің жалпы тұжырымдамасы:

1. Объектілер жиыны бар. Оны X деп белгілейміз.
2. Жауаптар(реакциялар) жиыны бар. Оны Y деп белгілейміз.
3. Осы екі жиын арасында тәуелділік бар, яғни X жиынының әрбір элементіне Y жиынының бір элементі сәйкес келеді.

Осы тәуелділікті *y* функциясы арқылы жазуға болады.

4. Шектеулі элементтерден тұратын X жиынының ішкі жиыны болып табылатын және оған Y жауаптары белгілі болатын бір жиын бар. Оны оқыту іріктемесі деп атаймыз.

5. X объектісіне Y жауабы сәйкес келетін мәнді табу үшін бізге белгілі бір *a* функциясы қажет. *a* функциясын табу – машинды оқытудың басты мақсаты.

Сапа метрикасы ретінде көбінесе дәлдік(accuracy) қолданылады. Дәлдік деп – алгоритм іріктеме ішіндегі жауаптарды дұрыс табу пайызын айтамыз.

Unsupervised learning(Мұғалімсіз оқыту) – машинды оқытудың екінші бөлімі. Бұл жерде модельге мәліметтер жиынтығы берілген, және онымен не істеу керектігі туралы нақты нұсқаулар жоқ. Машина бұл жерде мәліметтер арасында белгілі бір заңдылықтарды, корреляцияларды өзі табуға тиісті. Мұғалімсіз оқыту – мұғаліммен оқытуға қарама-қарсы метод болып табылады. Яғни мұғаліммен оқытуда – бастапқы мәлімет ретінде дұрыс жауаптар жиыны берілсе, мұғалімсіз оқытуда ешқандай дұрыс жауаптар берілмейді, алгоритм өзімен-өзі оқу керек. Кластеризация есептерін шығару кезінде қолданылады.

Объектілердің іріктемесі – бір-бірімен қиылыспайтын жиындарға, яғни локальды кластерлерге бөлінеді. Әр кластердің элементтерінің екінші кластер элементтерінен айырмашылығы үлкен болып табылады. Мәлімет қашықтық матрицасы ретінде беріледі. Қашықтық матрицасы – объектіден оқыту ірітемесінің әр элементіне дейінгі қашықтық ретінде сипатталатын матрица.

Кіріс деректер түрлері:

1. Объектілердің қасиеттік сипаттамасы. Әрбір объект өзінің қасиеттері деп аталатын характеристикалар жиынымен сипатталады. Қасиеттер – сандық немесе сандық емес болуы мүмкін.

2. Объектілер арасындағы қашықтық матрицасы. Объектіден оқыту ірітемесінің әр элементіне дейінгі қашықтық ретінде сипатталатын матрица.

Қорыта келгенде мұғалімсіз оқу деп – экспериментатордан ешқандай көмек алмайтын, бүкіл тәуелділікті өзі анықтайтын, өзімен-өзі оқитын машинаны айтамыз.

Reinforcement learning(обучение с подкреплением, бекітілетін оқу) – белгілі бір системада(ортада) оқытылып жатқан машина(агент) – ол система туралы ештеңке білмейді, алайда машина бұл ортада кездейсоқ іс-әрекеттер, операциялар жасауы мүмкін. Әрбір жасалған операцияға орта тарапынан әрқашанда белгілі бір реакция жүреді, яғни машинаның дұрыс әрекеттері үшін сыйақы беріледі, ал керісінше дұрыс емес іс-әрекет үшін ешқандай сыйақы берілмейді.

Бұл жерде модель қоршаған орта жайлы ешқандай мәлімет білмей тұрып, сол орта тарапынан максимал сыйақы, оң реакция алатындай етіп машинаны оқытуға тырысады.

Және орта тарапынан агентке(машинаға) оң реакция, яғни сыйақы бірден берілмейді.

Тек қана бір іс-әрекет немесе бірнеше іс-әрекеттің реттелген жиынына жауап ретінде беріледі.

Агент ортаға әсер етеді, орта агентке әсер етеді. Мұндай жүйені кері байланысы бар деп те атайды.

Біріші рет мұндай кері байланысы бар алгоритм жайлы әйгілі совет математигі М.Л. Цетлиннің еңбегінде жазылған болатын. Оның еңбегі бойынша

- автоматтың(агенттің) қандай іс-әрекет жасағанына байланысты – орта автоматқа не сыйақы беруі мүмкін, не жазалауы мүмкін. Уақыт өте келе автомат тарапынан кететін қателіктер саны азаяды, яғни автомат-машина оқиды.

Semi-supervised lerning(Мұғалімнің жартылай қатысуымен оқыту) – модель құру кезінде таңбаланған және таңбаланбаған мәліметтерді қолданатын машиныд оқыту алгоритмдерінің саласы. Әдетте кішкене ғана таңбаланған және көп таңбаланбаған мәліметтерді қолдану арқылы оқытылады. Бұл алгоритмдер саласын – мұғаліммен оқыту мен мұғалімсіз оқыту арасындағы компромис ретінде қарастыруға болады. Тәжірибелер бойынша аралас оқыту – қарапайым алгоритмдерге қарағанда оқыту сапасының әлдеқайда үлкен өсімін беруі мүмкін екені анықталды.

1.2 Машинды оқыту негізгі түсініктері

Іріктеме – жалпы жиын деп аталатын прецеденттер жиынынан модельді құруға және оны тексеруге қолдану үшін қажет арнайы іріктеліп алынған прецеденттер ішкі жиыны.

Оқыту іріктемесі(training sample) — моделдің құрылуы кезінде қолданылатын іріктеме. Моделдің оптимизациясы жасалатын негізгі іріктеме.

Бақылау(тесттік) іріктеме - (test sample) — құрастырылған модельдің сапасын бағалауға болатын іріткеме. Егер жаттығулар мен сынақ үлгілері тәуелсіз болса, онда сынақ үлгісінен алынған бағалау объективті болып табылады..

1. 3 Сурет – Іріктеме түрлері.

Аядан тыс оқу, жаттығу(overtraining, overfitting) – орысша “перетреннированность, переподгонка”.
Яғни модельдің аядан тыс, аса жаттығып – оның бақылау іріктемесіндегі дәлдік ықтималдығы оқыту іріктемесіндегі дәлдігіне қарағанда әжептеуір аз болған жағдайды айтамыз. Прецеденттермен оқу кезіндегі жағымсыз жағдай болып табылады. Көбінесе аса күрделі алгоритмдерді қолданған кезде пайда болады.

Кросс-валидация (кросс-тексеріс) – аналитикалық модельдің және оның жаңа, тәуелсіз мәндермен жұмыс істеу қабілетін зерттеуге мүмкіндік беретін тәсіл. Оның мағынасы: алғашқыда мәліметті n бөлікке бөлеміз, одан кейін оның

n-1 бөлігін оқыту іріктеме ретінде қолданып, модель құрамыз да – қалған n-ші бөлігін бақылау іріктемесі ретінде қолданамыз. Сөйтіп n рет қайталаймыз. Нәтижесінде мәліметтердің максимал бірыңғай терілуін қолданып таңдалған моделдің продуктивтілік деңгейін анықтауға мүмкіндік береді.

Метрика – машинды оқытуда модельдің сапасын бағалау үшін және әртүрлі алгоритмдерді салыстыру үшін қолданылатын басты параметрлердің бірі болып табылады.

1.3 Машинды оқытудың шешетін тапсырмалары

Машинды оқытуды қолданудың негізгі жолы – арнайы жасанды интеллекттің бір бөлігі болып табылаты есептерді шешу, тапсырмаларды орындау болып табылады. Машинды оқытудың негізгі шешетін тапсырмалары мен есептеріне жатады:

1.4 Сурет – Машинды оқыту есептері, шешетін тапсырмалары.

Соның ішінде классифкация мен регрессия – мұғаліммен оқытудың, ал кластеризация – мұғалімсіз оқытудың есептері болып табылады.

Классификация – белгілі бір класстарға бөлінген объектілер жиыны бар делік. Осы жиынның ішінде объект-класс жауаптары белгілі жұптардан тұратын ішкі жиын бар делік. Бұл ішкі жиынды – оқыту іріктемесі деп атаймыз. Қалған объектілердің қай классқа жататыны белгісіз.

Бізге – жалпы жиын ішіндегі кез-келген объектіні классификациялауға мүмкіндік беретін алгоритм құру керек.

Классификациялау деп – объектінің қай классқа жататынын көрсету, немесе класстың номерін, атауын көрсету болып табылады, басқаша айтқанда құрылған алгоритмнің немесе оның моделінің объектіге қарасты класс атауын не нөмерін беруі.

Классификация есебінің жалпы тұжырымдамасы: белгілі бір X объектілер жиынтығы болсын, және Y нөмерлер, класстар жиынтығы болсын. Осы екі жиын арасында тәуелділік бар, бірақ ол белгісіз. Оны y*: X->Y деп жазуға болады. Тәуелділік тек Xm = {(x1,y1),…,(xm, ym)} оқыту іріктемесінде ғана белгілі. Табу керек: кез-келген x∈X элементін классификациялауға қабілетті a: X-> Y алгоритмін құру.

Ескерту: классификацияны тек бір ғана қасиет бойынша жүргізу керек. Олай болмаса – класстар бір-бірімен қиылысып кетуі мүмкін.

Кластеризация деп – объектілер жиынтығын кластерлер деп аталатын группаларға бөлу процесін атайды. Кластерлер – класстарға қарағанда локальды болып келеди, яғни машина – локальды, бір-біріне жақын объектілер арасында заңдылықтарды тауып оларды группалау керек. шешу методы – мұғалімсіз оқу болып табылғанан кейін оқыту іріктемесінде дайын жауаптар берілмейді, яғни машина, алгоритм заңдылықтарды өзі шығару керек, өзімен-өзі оқу керек.

Алгоритмді құру этаптары:

1. Кластеризацияға керекті объектілер іріктемесін құру.
2. Объектілердің қандай қасиеттері бойынша сипатталатынын анықтайтын айнымалыларды енгізу.
3. Объектіллер арасындағы ұқсастық мәнін есептеу.
4. Бір-біріне ұқсас объектілер группасын – кластерлерді құру үшін – кластерлік анализ методтарын қолдану.
5. Нәтижелерді шығару.

Қазіргі программалау саласында ең көп қолданылатын алгоритмдердің бірі тірек векторлар методы болып табылады, ал оның көбінесе шешетін есебі – классификация есебі болып табылады. Тірек векторлар методының басты артықшылығы – салыстырмалы түрде оңайлығы және оның кез-келген есепке ядро қолданып шешім таба алуы болып табылады.

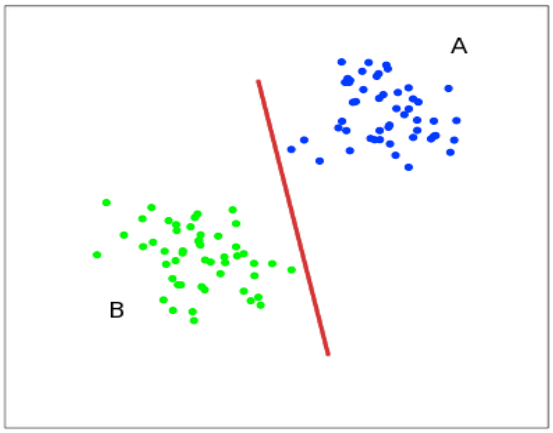
2 ТІРЕК ВЕКТОРЛАР МАШИНАСЫ (SVM) АЛГОРИТМІ

2.1 SVM алгоритмінің негіздері

Машинды оқытудың ең жақсы зерттелген алгоритмдерінің бірі – тірек векторлар методы немесе тірек векторлар машинасы болып табылады. Оның басты ерекшелігі – ядроларды көп жағдайда қолдана алу. Бұл дипломдық жұмыста осы метод қолданылады.

Тірек векторлар методы(машинасы) – классификация мен регрессия есептерін шешетін мұғаліммен оқуды қолданатын машинды оқыту алгоритмдерінің класы. Классификация есебі – объекттің екі кластың қайсысына жататынын анықтау. Әдетте тірек векторлар методында объект – n-өлшемді кеңістіктегі вектор болып табылады.

Алгоритмнің басты мақсаты – класстарды бір-бірінен бөлетін гипержазықтық құру және жүйені жоғары дәрежелі кеңістікке ауыстыру. Ең жақсы алгоритм болып гипержазықтық бойымен өтетін саңылау ең үлкен болатын алгоритм болып табылады, сол үшін де оны – максимал саңылау классификатор есебі деп те аталады. Өйткені класстар арасындағы саңылау үлкейген сайын – эмпирикалық қате саны төмендейді.

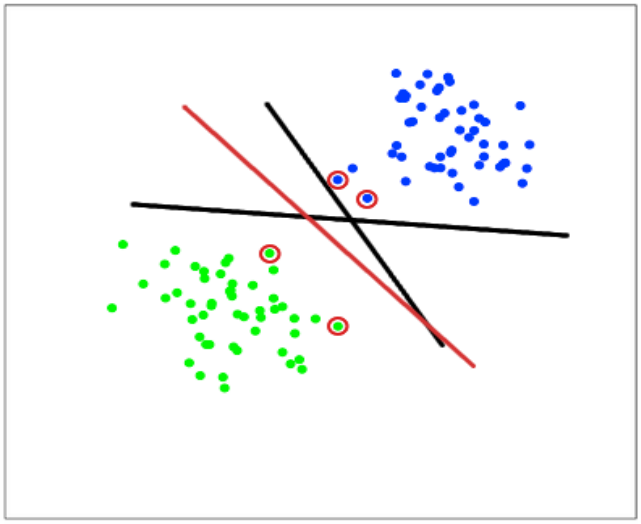


2. 1 Сурет – Гипержазықтық арқылы классификациялау.

Гипержазықтықтар саны көп болуы мүмкін, бірақ бізге оның ішіндегі ең оптимал бір гипержазықтықты табу керек. Басты гипержазықтықты – оптимал бөлуші гипержазықтық деп атайды. Басты гипержазықтықтың екі жағында қосалқы, параллель екі гипержазықтық салынады. Олар тірек векторларының бойымен өтеді. Алгоритмнің басты мақсаты осы екі қосалқы гипержазықтық арасындағы арақашықтықты максимал ету, яғни системаның орташа қателігін минимал ету болып табылады. Бұл шарттар орындалса – классификациялау дұрыс өтті деп саналады.

Тірек векторлары деп – алгоритмді құруға қолданатын, әр класстың екінші классқа ең жақын болатын векторларын айтады.

Тірек векторлар методында кеңістіктегі нүктелерді – p-өлшемді векторлар ретінде қарастырамыз. Яғни системадағы әр объектіні - p-өлшемді векторлар ретінде қарастырамыз.



2. 2 Сурет – Тірек векторларын қолданып бірнеше гипержазықтықтарды табу және олардың ішіндегі оптимал вариантын таңдау.

Гипержазықтық деп (p-1)-өлшемді кеңістікті айтамыз. Мысалы үшін 2-өлшемді жүйе(қарапайым жазықтық) үшін гипержазықтық – қарапайым түзу болады. 3-өлшемді жүйе үшін гипержазықтық – қарапайым 2-өлшемді жазықтық болады, және т.с.с. Бүкіл өлшеулер Евклид кеңістігінде жүргізіледі.

Тірек векторлар машинасын, методын классификация есептерін шешу үшін қолданған кезде бірнеше айта кетерлік зат бар. Олардың бірі – p-өлшемді кеңістікте әрбір координат осі – объектінің белгілі бір аттрибуттарын сипаттайды. Мысалы үшін – RGB моделі бойынша құрылған C түсінің 3 аттрибуты болады, және ол 3-өлшемді координаттар жүйесіндегі вектор болады: C = (red, green, blue).

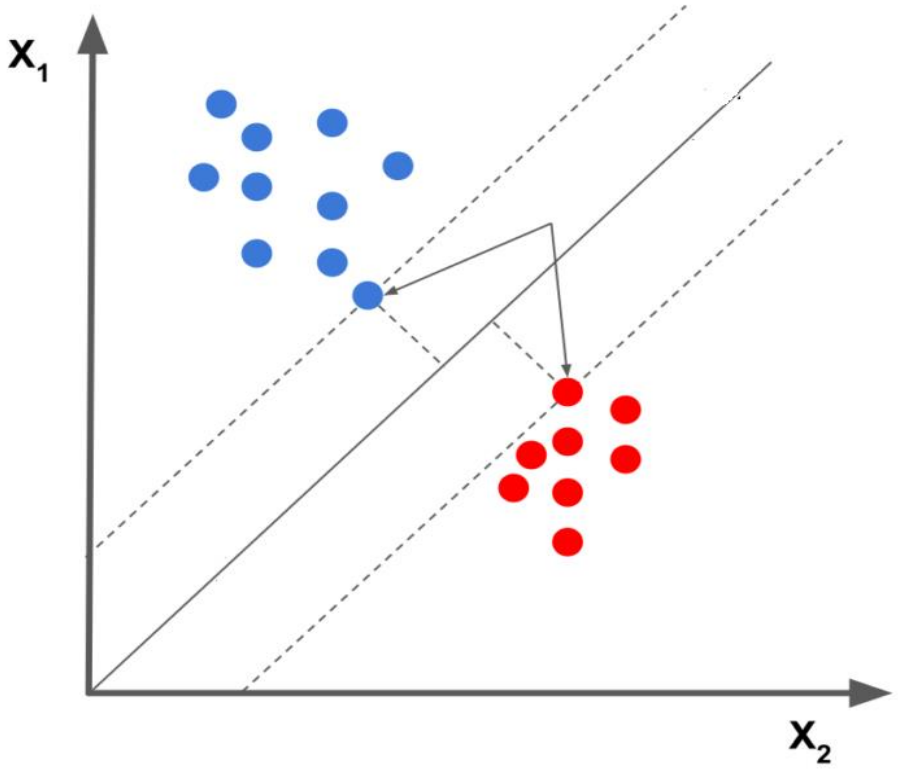
Егер класстар тек екеу ғана болса – (спам/спам емес), (ақ/қара), (қолдану/қолданбау) – мұндай есепті бинарлы классифкация деп те атаймыз.

Егер класстар алдын-ала берілмесе бұл – кластеризация есебі деп саналады.

Тірек векторлар машинасын қолданып классифкациялау есебінің жалпы тұжырымдамасы:

1. X объектілер кеңістігі болсын, және Y класстар жиыны болсын;
2. Бізде 2 класс бар деп есептейік, онда Y = {-1,1};
3. Бізге бастапқы оқыту іріктемесі берілген.
4. Кез-келген x-объектісі үшін y-класын табатын функция(классификатор) жүргізу керек.

Демек SVM(Support Vector Machine) алгоритмі – оқыту іріктемесі деп аталатын мағлұматты кіріс мәлімет ретінде алып, шығарда – дайын функция, модель құрастырып беретін алгоритмді атайды.



2. 3 Сурет – Классификация процесі.

Гипержазықтықты теңдеуі арқылы табуға болады:

$$g(x) = \vec{w}*\vec{x} + w0 \quad (2.1)$$

Әр қосалқы гипержазықтықтың басты гипержазықтыққа қарама қарсы жағында – екі класстың элементтері басталады. Және бізде 2 класс болғандықтан Y = {-1,1} болып табылады. Оптимал гипержазықтықтың мәні 0-ге тең болады, ал екі қосалқы гипержазықтықтардың мәндері 1 және -1 болады. Яғни кездейсоқ объектінің, нүктенің, элементке сәйкес у-мәні 1-ге тең болса, ол элемент бірінші классқа жатады, ал керісінше болса – екінші классқа жатады. Яғни, бірінші қосалқы, оптимал және екінші қосалқы гипержазықтық мәндері сәйкесінше:

$$g(x) = \vec{w}*\vec{x} + w0 = 1 \quad (2.2)$$

$$g(x) = \vec{w}*\vec{x} + w0 = 0 \quad (2.3)$$

$$g(x) = \vec{w}*\vec{x} + w0 = -1 \quad (2.4)$$

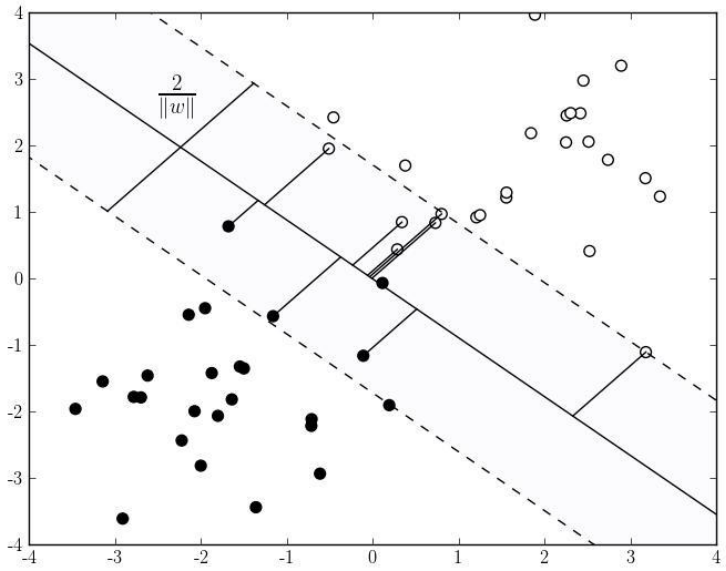
Объектінің бірінші немесе екінші классқа жататынын мына өрнек бойынша табуға болады:

$$g(x)>=1;\ x \in class1, \qquad g(x)<=1;\ x \in class2 \quad (2.5)$$

Келесі бір айта кететін ұғым – z(margin), яғни басты гипержазықтық пен қосалқы гипержазықтық арасындағы саңылауды, оның өлшемін айтады. Ол басты оптимал гипержазықтыққа параллель болып келеді, және оны келесі формуламен табуға болады:

$$Z = |g(x)|/||w|| = 1/||w|| \tag{2.6}$$

бұл жердегі w – басты гипержазықтыққа параллель болатын нормаль вектор болып табылады. Ал ||w|| - осы нормаль вектордың ұзындығы.



2.4 Сурет – Margin-саңылау өлшемін анықтау.

SVM алгоритмінің есептерді шешу жолының математикалық базасы. Нүктелерді $\{(x_1,c_1), (x_2,c_2),\ldots,(x_n, c_n)\}$ өрнегі ретінде жазуға болады. Мұндағы x_i – элемент, объект индексі, ал c_i – сол объектіге сәйкес класс. c_i мәні {-1,1} қабылдай алады.

Гипержазықтықты мына өрнекпен де жазуға болады:

$$w*x - b = 0 \tag{2.7}$$

мұндағы w – бөлуші гипержазықтыққа перпендикуляр, ал басты гипержазықтықтан қосалқы жазықтыққа дейінгі арақашықтықты осылай табуға болады:

$$z = \frac{b}{||w||} \tag{2.8}$$

Біздің басты мақсатымыз z арақашықтығы максимал гипержазықтық табу керек болғандықтан, (2.8) формуланы қолдана отырып – бізге ||w|| минимизация проблемасын шешу керек екені айқын.

Классификация шарттарын қолданып өзіміз қарапайым жүйе жаза аламыз:

$$\begin{cases} w * xi - b \geq 1, & ci = 1 \\ w * xi - b \leq 1, & ci = -1 \end{cases} \tag{2.9}$$

және де осыны бір өрнекке біріктірсек шығады:

$$c_i(w*x_i - b) \geq 1, \quad 1 \leq i \leq n \tag{2.10}$$

Тірек векторлар методының сызықты түрде бөлінетін кезіндегі шешілуі.

Алғашында ||w|| минимизация проблемасын және (2.10) қолданып система құрамыз:

$$\begin{cases} \|w\|^2 \rightarrow \min \\ ci(w * xi - b) \geq 1, \quad 1 \leq i \leq n \end{cases} \tag{2.11}$$

Бұл есеп Лагранж функциясының ершік нүктелерін(седловая точка) табудың қосарлы есебіне эквивалентті болып келеді. Оны көрсету үшін Лагранж функциясы мен оның көбейткіштері туралы біраз айтып кету керек.

Лагранж көбейткіштер методы – математикалық программалау, және оның дербес жағдайы сызықты программалау есептерін шешуге арналған метод болып табылады. Оның негізі: f(x), x ∈ R^n функциясының φi (x) = 0 негізінде *m* шарт бойынша(i = (1,…,m)) шартты экстремум табу болып табылады.

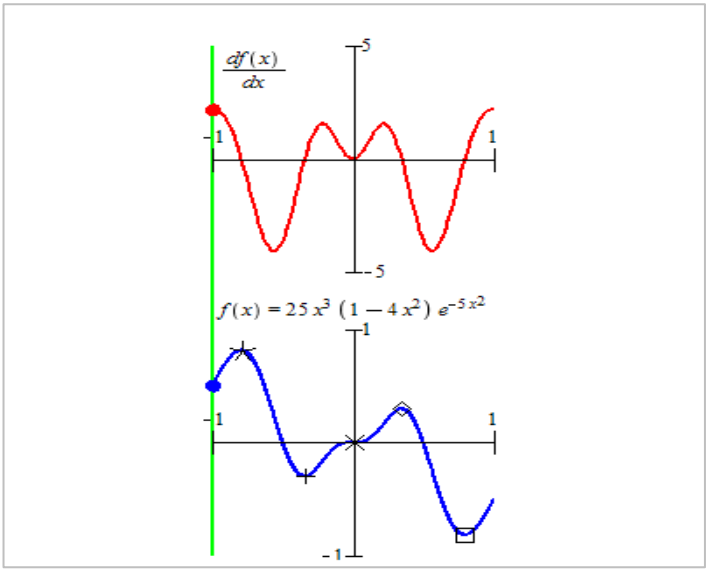
Жалпы жазылуы:

$$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda i * \varphi i(x), \tag{2.12}$$

$$\lambda = (\lambda_1, \ldots, \lambda_m)$$

және де Лагранж функциясының жеке туындыларын нөлге теңестіреміз. Егер алынған системаның бастапқы аргументтері бойынша шешімі бар болса – x’-шартты экстремум болуы мүмкін.

Ершік нүкте(Седловая точка) деп – берілген функцияның стационар нүктесі болатын және оның локальды экстремумы болмайтын нүктені айтады. Оның жеке туындылары нөлге тең және функция графигі осы нүктеде горизонталь болады.



2.5 Сурет – Локалды және глобалды экстремумдер.

Енді осы Лагранж әдісін тірек векторлар методына қолданатын болсақ шығады:

$$\begin{cases} L(w,b;\lambda) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \lambda i(ci(w * xi - b) - 1) \to \min(w,b), \max(\lambda) \\ \lambda i \geq 0,\ 1 \leq i \leq n \end{cases}, \quad (2.13)$$

$$\lambda = (\lambda_1, \ldots, \lambda_n)$$

яғни Лагранж функциясының ершік нүктесін табуға эквивалентті болып табылады.

Бұл есепті шештің деп қарастырайық, онда w пен b мәндерін мына формулалар бойынша табуға болады:

$$w = \sum_{i=1}^{n} \lambda_i c_i x_i, \quad b = w^* x_i - c_i, \quad \lambda_i > 0 \quad (2.14)$$

Нәтижесінде классификация алгоритмінің жалпы түрі:

$$a(x) = \text{sign}(\sum_{i=1}^{n} \lambda_i c_i x_i * x - b) \quad (2.15)$$

Классификацияны барлық нүктелер бойынша емес, тек тірек нүктелер бойынша ғана жүргіземіз, яғни $\lambda_i \neq 0$.

2.2 SVM алгоритмі мен grayscale-классификациясын қолданып қосымша мен оның интерфейсін жасау.

Бұл жоба бойынша адам қолымен жазылған сандарды тірек векторлар методы мен grayscale-жүйесі бойынша классификациялайтын алгоритм құру және оның қосымшасын жасау болып табылады.

Ең біріншіден керек утилиталар мен методтар импортталды, яғни біріншіден 2-өлшемді графиканы визуализациялауға арналған matplotlib кітапханасының методы, scikit-learn кітапханасынан керекті мәліметтер базасын және тірек векторлар методын.

```
import matplotlib.pyplot as plt
from sklearn import datasets,svm
dig = datasets.load_digits()
```

Бұл мәліметтер базасында барлығы 1700-ден астам объект бар. Оның біразы оқыту іріктемесі ретінде алынады. Датасетті дайындаймыз:

```
print("digits :",dig.target)
images_and_labels=list(zip(dig.images,dig.target))
print("len(images_and_labels)",len(images_and_labels))
for index,[image,label] in enumerate(images_and_labels[:5]):
    print("index :",index,"image :\n",image,"label :",label)
n_samples=len(dig.images)
print("n_samples :",n_samples)
imageData = dig.images.reshape((n_samples,-1))
print("After Reshaped : len(imageData[0]) :",len(imageData[0]))
```

Датасет дайын болғаннан кейін – support vector machine ішіндегі support vector classifier атты классификатор 8*8 бөлікке бөлінген суреттің әр пикселіне тағайындалған 0-ден 16-ға дейінгі мәнді қабылдайтын grayscale-сұр түс идентификатор мәндеріне қарап санды анықтауды бастайды:

```
classify=svm.SVC(gamma=0.001)
classify.fit(imageData[:n_samples//2],dig.target[:n_samples//2])

expected=digits.target[n_samples//2:]
predicted=classify.predict(imageData[n_samples//2:])
image_and_predictions = list(zip(dig.images[n_samples//2:],predicted))
print("original values:",dig.target[n_samples//2:(n_samples//2)+5])
classify=svm.SVC(gamma=0.001)
classify.fit(imageData[:],digits.target[:])

from scipy.misc import  imread,imresize,bytescale
img=imread("34.png")
img=imresize(img,(8,8))
img=img.astype(digits.images.dtype)#to change image into numbers
img=bytescale(img,high=16.0,low=0)
```

```
x_testData =[]
for c in img:
    for r in c:
        x_testData.append(sum(r)/3.0)
x_testData=[x_testData]
print("len(x_testData) :",len(x_testData))
print("Prediction :",classify.predict(x_testData))
plt.show()
```

Оған қоса осы алгоритмге Microsoft Windows платформаларында қолдануға болатындай және оны басқа платформаларға оңай порттауға болатындай – арнайы python тіліне арналған kivy-фреймворкін қолданып кроссплатформды қосымша жасалды.

Нәтижесінде шығады:

```
index : 3 image :
 [[ 0.  0.  7. 15. 13.  1.  0.  0.]
 [ 0.  8. 13.  6. 15.  4.  0.  0.]
 [ 0.  2.  1. 13. 13.  0.  0.  0.]
 [ 0.  0.  2. 15. 11.  1.  0.  0.]
 [ 0.  0.  0.  1. 12. 12.  1.  0.]
 [ 0.  0.  0.  0.  1. 10.  8.  0.]
 [ 0.  0.  8.  4.  5. 14.  9.  0.]
 [ 0.  0.  7. 13. 13.  9.  0.  0.]] label : 3
index : 4 image :
 [[ 0.  0.  0.  1. 11.  0.  0.  0.]
 [ 0.  0.  0.  7.  8.  0.  0.  0.]
 [ 0.  0.  1. 13.  6.  2.  2.  0.]
 [ 0.  0.  7. 15.  0.  9.  8.  0.]
 [ 0.  5. 16. 10.  0. 16.  6.  0.]
 [ 0.  4. 15. 16. 13. 16.  1.  0.]
 [ 0.  0.  0.  3. 15. 10.  0.  0.]
 [ 0.  0.  0.  2. 16.  4.  0.  0.]] label : 4
n_samples : 1797
After Reshaped : len(imageData[0]) : 64
original values: [8 8 4 9 0]
len(x_testData) : 1
Machine Output : [8]
>>>
```

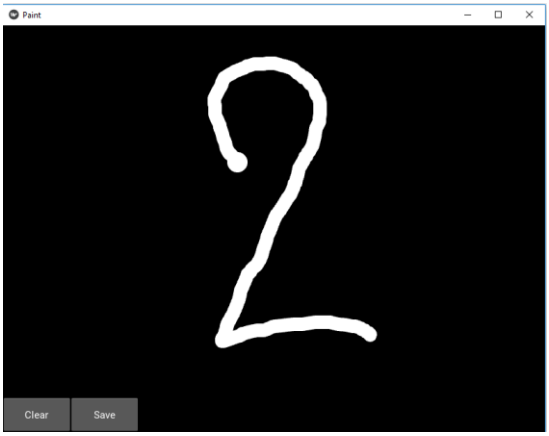
2.7 Сурет – Тірек векторлар классификаторы нәтижесі.

Құрылысы: Киви-фреймворкі мен оның басты утилиталарын(виджет, батырма, қосымша) импорттау, содан кейін 2 класс ашу. Басты класста(PaintApp) екі батырманы анықтайтын функция мен басты метод орналасқан. Екінші класста курсорды басқан кезде және оны қозғалтқан кездегі іс-әрекетті сипаттайтын методтар орналасқан.

```
from kivy.app import App
from kivy.uix.widget import Widget
from kivy.uix.button import Button
from kivy.core.window import Window
from kivy.graphics import (Color, Ellipse, Rectangle, Line)
```

```
class PainterWidget(Widget):
    def on_touch_down(self, touch):
        with self.canvas:
            Color(255, 255, 255, 1)
            rad = 30
            Ellipse(pos = (touch.x - rad/2, touch.y - rad/2), size =
(rad, rad))
            touch.ud['line'] = Line(points = (touch.x, touch.y), width =
10)

    def on_touch_move(self, touch):
        touch.ud['line'].points += (touch.x, touch.y)

class PaintApp(App):
    def build(self):
        parent = Widget()
        self.painter = PainterWidget()
        parent.add_widget(self.painter)
        parent.add_widget(Button(text = "Clear", on_press =
self.clear_canvas, size = (100, 50)))
        parent.add_widget(Button(text = "Save", on_press = self.save,
size = (100, 50), pos = (100, 0)))
        return parent

    def clear_canvas(self, instance):
        self.painter.canvas.clear()

    def save(self, instance):
        self.painter.size = (Window.size[0], Window.size[1])
        self.painter.export_to_png('11.png')

if __name__ == '__main__':
    PaintApp().run()
```

Нәтижесінде SVM алгоритмі мен grayscale-классификациясын қолданылып қолдан жазылған араб сандарын анықтайтын алгоритм құрылды және оның интерфейсі мен қосымшасы жасалды. python тіліне арналған kivy-фреймворкі мен оның инструменттері, кроссплатформды қосымшалар идеясы зерттелді.



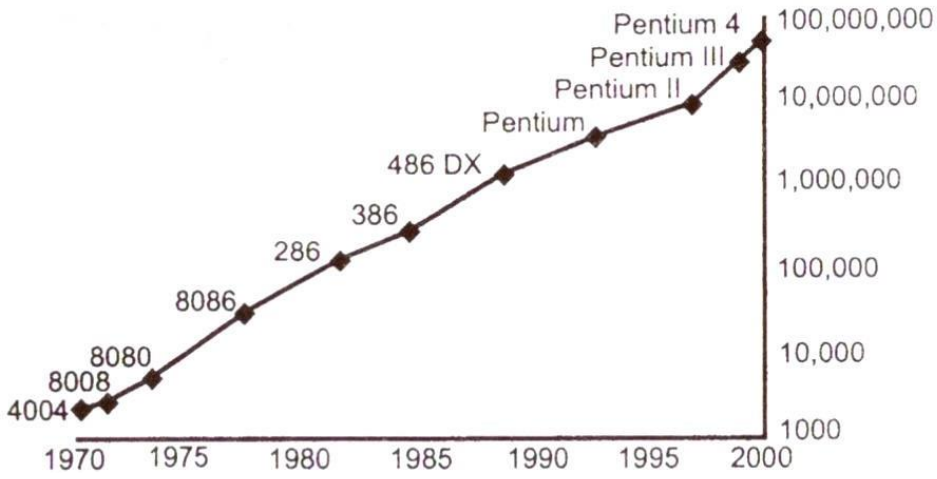
2.8 Сурет – Қосымша.

Қолданылған негізгі алгоритм – тірек векторлар методы ішіндегі – тірек векторлар классификаторы. Қосымша функционалына жатады: қолмен араб сандарын салуға болатын немесе санды қолмен теруге болатын редактор,
тазарту және сақтау батырмалары. Сақталған суретті алгоритм өзі көріп қай сан екенін анықтайды.

3 SVM АЛГОРИТМІНІҢ КВАНТТЫҚ КОМПЬЮТЕРДЕ РЕАЛИЗАЦИЯСЫ

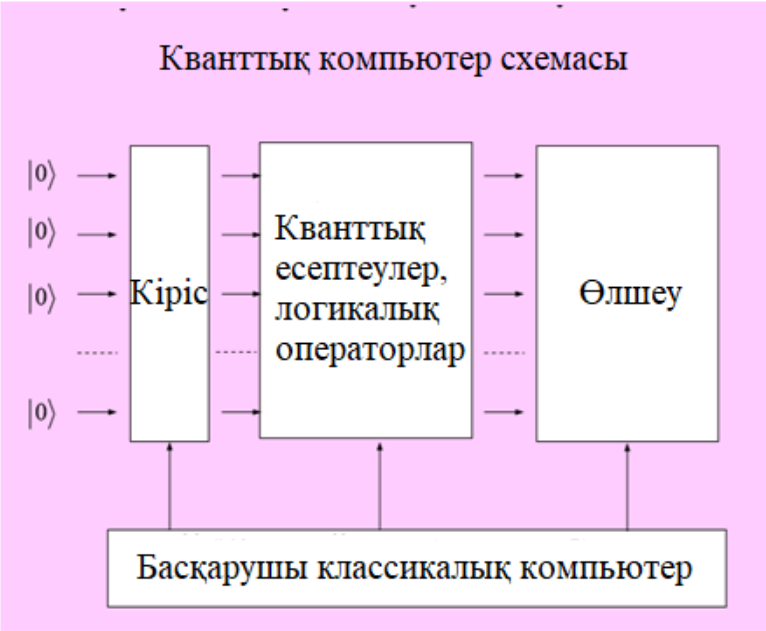
3.1 Кванттық компьтерлер түсінігі және оның болашақтағы рөлі

Қазіргі заманда жоғары технологиялардың адамзат өміріндегі рөлі күннен күнге артып келе жатыр. Жоғары технологияларға күрделі электротехникалық құрылғылар, микропроцессорлар және т.б. жатады. Соның ішінде аса айта кететін зат – микропроцессорлар. XX – ғасырдың 50-60-жылдарынан бастап қазіргі заманға дейін микропроцессорлар әлемінде Мур заңы жұмыс істеп келген, яғни ол бойынша – әрбір 24 ай сайын интегралды схемадағы транзисторлар саны 2 есе көбееді. Оны атақты американ ғалымы – Джон Мур енгізген. Бірақ қазіргі кездегі жартылай-өткізгіштердің аса үлкен жылдамдықпен дамып келе жатқантан Мур заңы істен шыға бастады. Оның себебі – транзисторлардың тым кішірейгені, яғни макромир заңдары істемей, микромир заңдары, яғни кванттық физиканың істей бастауы болып табылады. Сондықтан да қазіргі бүкіл әлемдегі ғылыми институттар мен IT-корпорациялардың назары – кванттық компьютерлер мен кванттық технологияларда жатыр.



3.1 Сурет – Мур заңы(x-осі – жыл, y-осі – транзисторлар саны).

Кванттық компьютер деп – кванттық әлемнің, яғни кванттық физиканың құбылыстарын(кванттық шиеленіс, кванттық суперпозиция) есептеулер операциялары кезінде, деректерді өңдеу үшін және тарату үшін қолдана алатын компьютерді айтады. Классикалық компьютерге қарағанда – кванттық компьютер – биттармен емес(ақпарат минимал өлшем бірлігі, 0 немесе 1 мәнін ғана қабылдай алады), кубиттер жұмыс істейді.



3. 2 Сурет – Кванттық компьютер жалпы схемасы.

Черч-Тюринг-Дойч тезисі – кез-келген классикалық физикалық процесті – кванттық компьтерде эффективті түрде модельдеуге болады.

Кванттық шиеленіс деп – екі немесе одан да көп кванттық объектілердің кванттық жағдайларының бір-бірімен өзара байланыста болуымен сипатталатын кванттық механиканың құбылысы. Мысалы үшін екі кванттық шиеленіскен фотонды алатын болсақ, оның біреуінің спинын анықтайтын болсақ – екіншісінікі де бірден анықталады. Кванттық шиеленіскендік – объектілер бір-бірінен өте үлкен, ешқандай әрекеттесу түрлері істемейтін арақашықта да сақтала береді. Бір кванттық объектінің параметрі анықталатын болса – екінші объектінің параметрі шиеленіскендік жағдайын тура сол мезетте айырылады.

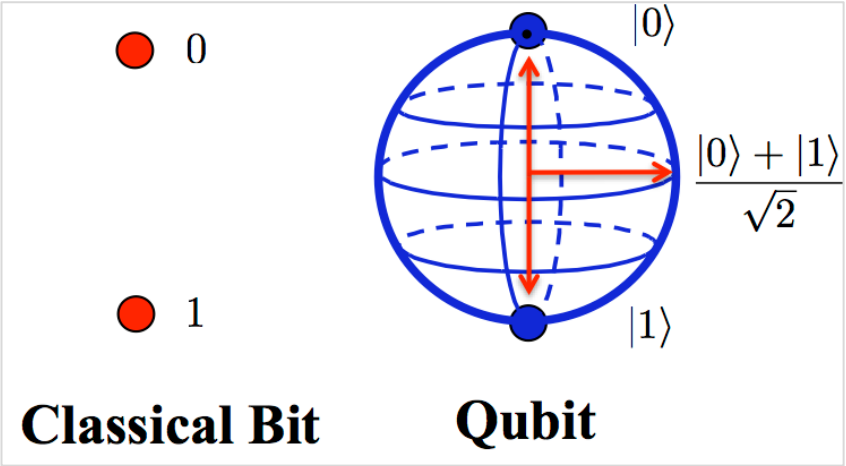
Кубит(qubit, quantum bit) деп – кванттық разрядты немесе кванттық компьютердегі ақпараттың минимал өлшем бірлігін айтады. “qubit” терминін ең бірінші рет 1995ж. Кеньон-колледж ғалымы Бен Шумахер қолданған.

Кубиттер өзара шиеленіскен болуы мүмкін.

Кубит - битқа қарағанда 0, 1 және олардың суперпозициясы(бір уақытта екі қалыпте де болу мүмкіндігі) жағдайларында болуы мүмкін. Оны осылай белгілейміз:

$$|\psi\rangle = A|0\rangle + B|1\rangle \qquad (3.1)$$

мұндағы A мен B: $|A|^2 + |B|^2 = 1$ шартын қанағаттандыратын комплекс сандар болып табылады. $|0\rangle$ және $|1\rangle$ - нөлінші және бірінші кубиттың жазылуы.



3. 3 Сурет – Бит пен Кубит айырмашылығы.

Кванттық параллелизм деп – кванттық компьютерлердің болашақта классикалық компьютерлерді өнімділігі мен продуктивтігі бойынша озып кетуге мүмкіндік беретін принцип. Бұл құбылыстың негізінде – кванттық жағдайлардың суперпозициясын қолдану, яғни бір уақытта өте көп деректі өңдеу принципі жатыр. Мысалы үшін 64-кубитті кванттық компьютер бір уақытта 2^{64} жағдайды қарастыра алады. Алайда мұндай артықшылықтарға қарамастан кванттық компьютерлерде қолдану саласында әлі көптеген қиындықтар бар.

Қазіргі әлемде толығымен кванттық үстемділік, яғни классикалық компьютерлерден толығымен озу туралы айту ерте.

3.2 Кванттық есептеулер

Кванттық есептеулер деп – кванттық механиканың шиеленіс және параллелизм құбылыстарын есептеулер жүргізу кезінде қолданылуын айтады. Кванттық информатиканың бір саласы болып табылады. Ең бірінші 1980-жылдары Ричард Фейнман мен Юрий Маниннің классикалық компьютерлердің бүкіл процесстерді модельдей алмайтыны туралы сөздерінен кейін дами бастады.

Кванттық есептеулердің қазіргі кездегі маңыздылығы себептері:

1. Классикалық компьютерлер әлі де криптография мен кванттық механика есептерін толығымен шеше алмайды.
2. Классикалық компьютерлердің қуат өсімінің физикалық шектеулері.

Кванттық есептеулердің негізгі түсініктері:

Кванттық алгоритм деп – белгілі бір унитарлы операциялардың(кванттық гейттер немесе вентильдердің) ретін сипаттайтын классикалық алгоритмді айтады.

Кванттық алгоритмдердің басты артықшылығы – мүлдем жаңа алгоритмдерді қолдана алу емес, бірақ баяғыдан белгілі классикалық алгоритмдерді өте үлкен жылдамдықпен шешу болып табылады.

Кванттық алгоритммен шешілетін кез-келген есеп – классикалық компьютермен де шешілуі мүмкін. Алайда классикалық есептерді кванттық алгоритмдерді қолданып шешкен кезде – экспоненциалды немесе квадратты түрде уақыт ұтуға болады.

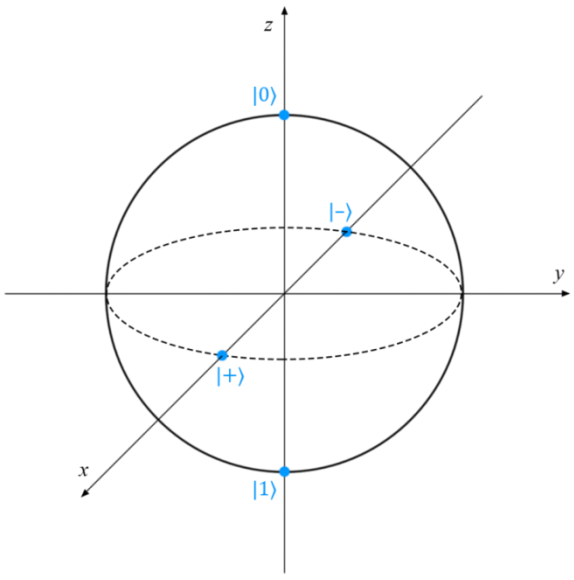
Кванттық компьютерде уақыт ұтысы классикалық компьютердегідей тактті жиілікке байланысты емес, бірақ кванттық параллелизмді қолданады.

Кванттық вентиль деп – кубиттердің кіріс мәнін қолданып шығардағы мәнін белгілі бір заңдылық бойынша есептеуге мүмкіндің беретін кванттық компьютердің басты элементтерінің бірі, оны кванттық логикалық элемент деп те атайды. Қарапайым логикалық вентильдерге қарағанда кубиттермен жұмыс істейді және кванттық механика заңдарына бағынады. Қайта орындалатын болып келеді.

Кубитті екіөлшемді кеңістіктегі вектор ретінде қарастыруға болғандықтан вентиль әрекетін – кубит векторына унитарлы матрица көбейтіндісі ретінде де өрнектеуге болады. Мысалы үшін, біркубитты вентильді 2*2 матрицасымен, ал екікубитты вентильді 4*4 матрицасымен, ал n-кубитты вентильді $2^n * 2^n$ матрицасымен сипаттауға болады.

|0> және |1> кубиттерінің барлық мүмкін жағдайлары 4 болады, яғни |0>|0>, |0>|1>, |1>|0> және |1>|1>.

Блох сферасы деп – кванттық жағдайларды үшөлшемді кеңістіктегі вектор ретінде немесе сфера бетіндегі нүкте етіп сипаттау жолы. Яғни кубиттің мәнін анықтауға көмектеседі. Атақты ғалым Феликс Блохтың атымен аталған.



3. 4 Сурет – Блох сферасы.

Кванттық вентильдер мысалдары:

Адамар вентилі - $|0\rangle$ и $|1\rangle$ базистік күйлерінің суперпозициясын құрастырады. Оның жалпы түрі:

$$H \equiv \frac{1}{sqrt(2)}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \tag{3.2}$$

яғни $|0\rangle$ және $|1\rangle$ кубиттері үшін Адамар вентилі сәйкесінше:

$$|0\rangle \rightarrow \frac{|0\rangle+|1\rangle}{\sqrt{2}}, \tag{3.3}$$

$$|1\rangle \rightarrow \frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

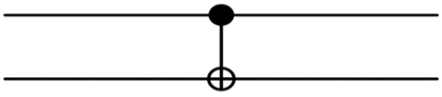
яғни оның $|0\rangle$ кубитімен әрекеттесуінен пайда болады:

$$X|0\rangle = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}0*1+1*0\\ 1*1+0*0\end{bmatrix} = \begin{bmatrix}0\\1\end{bmatrix} = |1\rangle \tag{3.4}$$

ал $|1\rangle$ кубитімен әрекеттесуінен пайда болады:

$$X|1\rangle = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}1\\0\end{bmatrix} = |0\rangle \tag{3.5}$$

CNOT – бақылаушы ЖОҚ, яғни қарапайым жоқ вентилі, бірақ алғашқы кубит мәні бір болса – екінші кубитке ЖОҚ операторы қолданылады, ал мәні ноль болса – қолданылмайды. Оны графикалық бейнесі:



3.5 Сурет – CNOT кванттық вентилі.

ал оның матрицалық түрін осылай өрнектеуге болады:

$$U_{CN} \equiv \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix} \tag{3.6}$$

мұндағы матрицаның шығу жолы – үстіңгі сол жағына бірлік матрицаны жазу, ал астыңғы оң жақ бөлігіне ЖОҚ матрицасын жазудан пайда болады, яғни

бақылаушы кубит үшін бірлік матрица орындалса – екінші кубитте ЖОҚ матрицасы орындалады:

$$II = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad NOT = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \tag{3.7}$$

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

яғни кванттық вентильдің матрицасын құрастыру кезінде осы жолды қолдануға болады.

NOT-вентилі, яғни жоқ деген мағынаны білдіреді, матрицалық түрі:

$$X \equiv \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{3.8}$$

Қорыта келгенде кванттық есептеулер – кванттық машинды оқытудың негізгі, бастапқы алғабастар инструменттерінің бірі болып табылады. Кванттық есептеулер сызықты алгебра элементтері мен оның кванттық механика құбылыстарын қолданып есептеулер жасауға көмектеседі.

3.3 Quantum SVM және оның питон тілінде және IBMQ кванттық компьютерінде реализациясы

Қазіргі әлемде микропроцессорлар саласында біраз мәселелер пайда болды. Оның ең бастысы алдында айтылып кеткен – Мур заңының істен шығуы. Бұл заң 1950-жылдардан 2010-жылдарға дейін істеп келген. Алайда транзисторлар саны көбейген сайын, олардың өлшемі кішірейген сайын – классикалық Ньютон заңдары істен шыға бастайды да, яғни кванттық әлем заңдары істей бастайды. Сол үшін кванттық компьютерлерде атқарылатын кванттық есептеулер мен оның жеке жағдайы – кванттық машинды оқыту – нағыз болашақ технологиясы болып табылады. Кванттық машинды оқытудың басты артықшылығы – ол классикалық компьютерлерден есептеу кезінде уақыттан әжептеуір үлкен ұтыс бере алады. Алайда бүкіл артықшылықтарына қарамастан бұл технологиялар нашар зерттелген.

Кванттық машинды оқыту деп – кванттық физика мен информатика ғылымдарының қиылысу нүктесінде орналасқан саланы айтамыз.

Яғни бұл саланың басты мақсаты – кванттық механика құбылыстарын(кванттық шиеленіс, кванттық параллелизм) жасанды интеллект, машинды оқыту есептерін шешу кезінде қолдану, кейбір алгоритмдер бойынша классикалық компьютерлерге қарағанда үстемдік көрсету.

Кванттық машинды оқытуда көбінесе – классикалық ақпаратты кванттық компьютерде қолдануға болатындай арнайы кодировкадан өткізуді қолданады. Одан кейін кванттық ақпарат өңдеу методтары қолданылады да ең соңында нәтиже есептеледі. Көп кванттық алгоритмдер универсалды компьютерде әлі де толық зерттеліп қолданылмаған.

Кванттық машинды оқытуда негізгі 3 оқыту моделі ерекшеленеді:

1. Нақты оқыту(exact learning) деп – табу керек функциямыз белгісіз функцияға максимал жақын және де белгісіз функцияның әртүрлі аргументтер қойған кезіндегі мәндерін анықтауға болатын оқытуды айтамыз. Алгоритмдердің бұл түрі максимал өніміділікті – кванттық суперпозицияда жатқан запростарды қолдану кезінде көрінеді.
2. PAC-оқыту – бұл жерде де белгісіз функцияға максимал жақын функция ізделінеді, алайда кванттық запростар жоқ. Оның орнына алдын-ала берілген үлгілер жиынтығы берілген және осы үлгілерге сай белгісіз функцияны табу. Классикалық PAC-оқытудан айырмашылығы – кванттық суперпозицияны қолданады.
3. Агностикалық оқыту деп – бізге n кубит реті берілген, бізге n+1 кубитті табу керек.

Кванттық қателерді дұрыстау деп – мажоритарлы ықтималдық теориясын қолдануды айтады. Кубиттер стабильды болмауы мүмкін, яғни қателесуі мүмкін. Сол жағдайда мысалы үшін 3-кубиттік система үшін біреуінің мәні 0 болса, ал екеуінің мәні 1 болса, онда қайсысының саны көп, ықтималдығы үлкен, яғни мәні 1 болады деп таңдаймыз.

Кванттық машинды оқыту реализациясы(QSVM):

Жобаның басты мақсаттарының бірі – кванттық компьютерде тірек векторлар машинасының бинарлы классификатор есебін шешу арқылы реализациясы және оны классикалық алгоритммен салыстырылуы болып табылады.

QSVM(Quantum Support Vector Machine) деп – классификация мен регрессия есептерін тірек векторлар методын кванттық компьютерде есептейтін алгоритмді айтады. Классификациялық алгоритмдер – ақпараттардың интеллектуалды анализы саласында өте үлкен рөль атқарады. Классификация методындағы басты ұғымдардың бірі – ядро. Объектілер берілген кеңістікте классификацияланбауы мүмкін, яғни қанағаттандыратын гипрежазықтық табылмауы мүмкін, осы жағдайда ядроларды қолданады. Мұндай гипержазықтықты табу үшін сызықты емес түрлендіру функциясын қолданады, яғни кеңістік өлшемін арттыруды айтамыз. Бұл функцияны характеристикалар картасы дейміз.

QSVM методы классикалық жағдайдағы ядроларды қолдану тиімсіз болған жағдайда, характеристикалар картасын құру керек классификация есептерін шешу кезінде қолданылады. Бұл метод мұғаліммен оқыту әдісіне жатады, яғни оқыту(ядро мен тірек векторларды табу кезеңі) және тестілеу немесе классификация(жаңа, оқытуға қатыспаған объектілерді классификациялау) бөлімдерінен тұрады.

Feature map деп – жаңа өлшемі көбейтілген кеңістіктегі объектілердің, векторлардың орналасуын көрсететін схема.

QSVM алгоритмін – системаны классикалық жолмен шешуге болмайтын кезде қолданамыз, яғни мәселе қиындығы өскен сайын – қажетті есептеуіш ресурстарының экспоненциалды өсуі. Кванттық ядроны қолданған кезде – классификация бірден характеристикалар картасынан анықталады.

Kernel matrix немесе distance matrix деп – классификация есебін шешу үшін – объектілердің, векторлардың бір-біріне қаншалықты жақын орналасқанын анықтау жолын айтады. Яғни системада n объект болса – n * n матрицы құрылады да, әр жұп объектінің бір-бірінен арақашықтығы анықталады. Бұл арақашықтар коллекциясын – ядро деп атаймыз.

Бұл методты іске асыру үшін IBM Q системасындағы кванттық компьютер қолданылды. IBM Q - IBM компаниясының open access берілген кванттық компьютерлеріне онлайн қолдануға мүмкіндік беретін жүйе. Бұл сервисті алгоритмдер мен эксперименттерді іске қосу үшін, сондай-ақ кванттық есептеулер мүмкіндіктеріне қатысты Оқу құралдары мен симуляцияларды зерттеу үшін пайдалануға болады. Жүйеде бірнеше компьютер бар, және олардың ішіндегі бұқараға ашық ең қуатты компьютер – 5-кубиттен тұрады және біз өзіміздің жобамызға соны қолдандық. Python тілін қолданып кванттық компьютерге қосылу үшін арнайы Qiskit-кітапханасын қолданамыз және бізді қарастыратынымыз – бинарлы классификатор, яғни екі класстан тұратын системаны шешу.

Бинарлы тірек векторлар методын кванттық компьютерде реализациялау:
Ең бірінші бүкіл керек методтар мен утилиталарды, qiskit-кітапханасы элементтерін импорттаймыз:

```
from qsvm_datasets import *

from qiskit import Aer
from qiskit_aqua.utils import split_dataset_to_data_and_labels,
map_label_to_class_name
from qiskit_aqua import QuantumInstance
from qiskit_aqua.input import SVMInput
from qiskit_aqua import run_algorithm
from qiskit_aqua.algorithms import QSVMKernel
from qiskit_aqua.components.feature_maps import SecondOrderExpansion
```

яғни бұл жердегі бізге негізгі керектісі: SVMInput(яғни тірек методының негізгі методы), QSVMKernel – тірек векторлар методы ядросының кванттық версиясы, run_algorithm – кванттық алгоритмді іске қосатын утилита.

Және кванттық компьютерге қосылу үшін qiskit кітапханасынан IBMQ утилитасын импорттап, өзіміздің IBMQ Experience аккаунт верификаторын қолдануымыз керек.

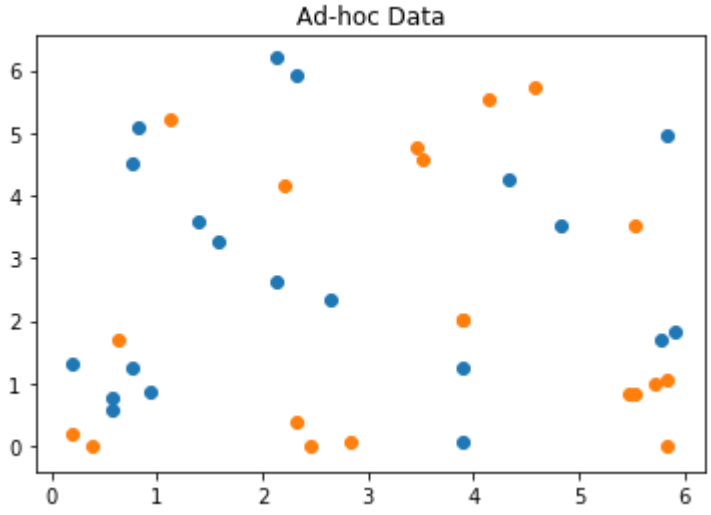
```
import logging
from qiskit_aqua import set_aqua_logging
```

```
from qiskit import IBMQ
IBMQ.enable_account('373cb4daa02a0f316690943385a0e5a37d99e6a81e4f4cdaa8b4
506da6b3a04170b8a7808991ef0ebf965f4ac287d55c1ff50d14eb60e23d74c9e2764231e
418')

from timeit import default_timer as timer
start = timer()
```

Одан кейін біз датасетті – басында оқытуға, тестілеуге және соңында болжауға қабілетті болатындай етіп дайындау керекпіз:

```
#print(class_to_label)
feature_dim=2
sample_Total, training_input, test_input, class_labels =
Breast_cancer(training_size=20, test_size=10, n=feature_dim, gap=0.3,
PLOT_DATA=True)

extra_test_data = sample_ad_hoc_data(sample_Total, 10, n=feature_dim)
datapoints, class_to_label =
split_dataset_to_data_and_labels(extra_test_data)
```

Нәтижесінде бізде мынадай объектілер картасы шығады, бұл объектілерді сызықты түрде классификациялау мүмкін емес болғандықтан ядроларды қолданамыз(QSVMKernel), яғни өлшем санын көбейтеміз. Бастапқы екіөлшемді кеңістіктегі объектілер жиынтығы:



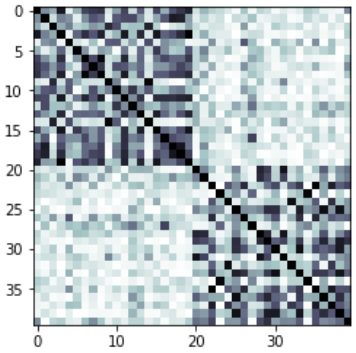
3.6 Сурет – Объектілер жиынтығы, {'A': 0, 'B': 1}

Және соңында нәтижелерді шығару үшін print методымен керек заттардың бәрін шығарамыз:

```
print("testing success ratio: {}".format(result['testing_accuracy']))
print("prediction of datapoints:")
```

```
print("ground truth: {}".format(map_label_to_class_name(datapoints[1],
qsvm.label_to_class)))
print("prediction:   {}".format(result['predicted_classes']))
print("kernel matrix during the training:")
kernel_matrix = result['kernel_matrix_training']
img =
plt.imshow(np.asmatrix(kernel_matrix),interpolation='nearest',origin='upp
er',cmap='bone_r')
plt.show()
```

Нәтижесінде мынадай ядро матрицасы немесе характеристикалар картасы шығады:



3.7 Сурет – QSVM арқылы шешілген ядро матрицасы.

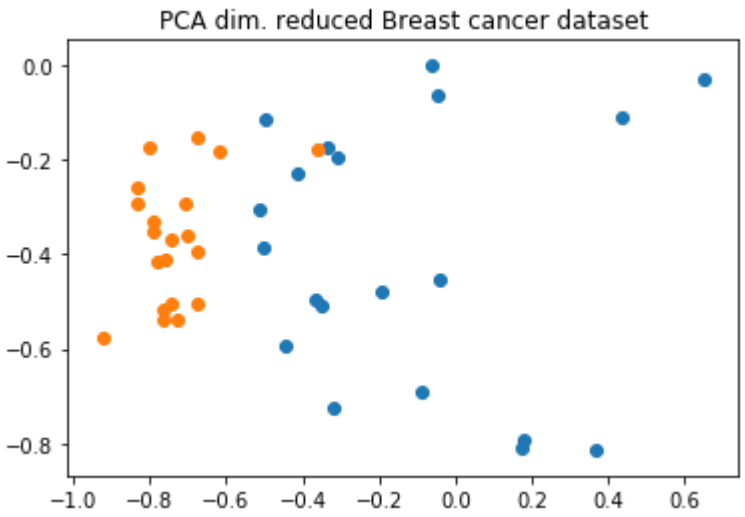
Тура осы тірек векторлар классификаторының классикалық нұсқасы: Керек методтарды импорттаймыз:

```
from datasets import *
from qiskit_aqua.utils import split_dataset_to_data_and_labels,
map_label_to_class_name
from qiskit_aqua.input import SVMInput
from qiskit_aqua import run_algorithm
```

Датасетті – басында оқытуға, тестілеуге және соңында болжауға қабілетті болатындай етіп дайындау керекпіз:

```
sample_Total, training_input, test_input, class_labels =
Breast_cancer(training_size=20, test_size=10, n=2, PLOT_DATA=True)
# n =2 is the dimension of each data point

datapoints, class_to_label = split_dataset_to_data_and_labels(test_input)
label_to_class = {label:class_name for class_name, label in
class_to_label.items()}
print(class_to_label, label_to_class)
```

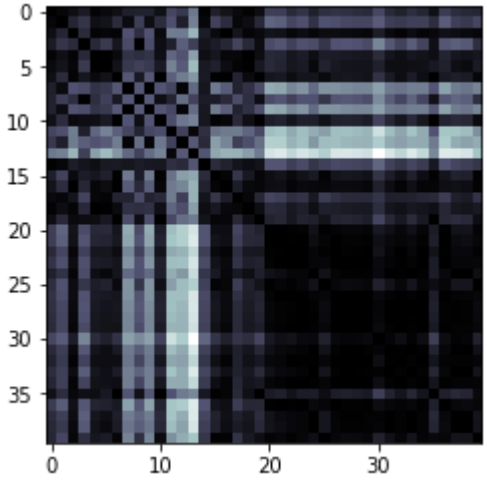
Нәтижесінде бізде мынадай объектілер картасы шығады, бұл объектілерді сызықты түрде классифкациялау үшін тірек векторлар методын қолданамыз:



3.8 Сурет – Классикалық SVM арқылы шешетін объектілер жиынтығы

```
algo_input = SVMInput(training_input, test_input, datapoints[0])
result = run_algorithm(params, algo_input)
# print(result)
print("kernel matrix during the training:")
kernel_matrix = result['kernel_matrix_training']
img =
plt.imshow(np.asmatrix(kernel_matrix),interpolation='nearest',origin='upp
er',cmap='bone_r')
plt.show()

print("testing success ratio: ", result['testing_accuracy'])

print("ground truth: {}".format(map_label_to_class_name(datapoints[1],
label_to_class)))
print("predicted:    {}".format(result['predicted_classes']))
```

Және нәтижесінде мынадай ядро матрицасы немесе характеристикалар картасы шығады:



3.9 Сурет – SVM арқылы шешілген ядро матрицасы

Нәтижесінде, қазір кванттық кванттық компьютердегі тірек векторлар методының ядро матрицасын, яғни арақашықтық матрицасын табу арқылы бинарлы классификация есебін шешуге болатынын дәлелдедік. Ол үшін ең бірінші – екіөлшемді кеңістіктегі сызықты түрде бөлінбейтін объектілер жиынтығын алып, оның өлшемін өсіріп – алынған характеристикалар картасына кванттық ядро методын қолдандық. Және соіында ядро матрицасын шығардық. Кванттық компьютерде реализациялау себебі – есеп мәселе қиындығы өскен сайын – оның есептеуіш қуат ресурстарындағы қажеттілік де экспоненциалды түрде ұлғайып отырады. Алайда әлі де толық кванттық үстемділік туралы айту ерте болып тұр.

Бұл дипломдық жобада – кванттық тірек векторлар ядросы арқылы бинарлы классификация жасау мүмкіндігі зерттелді. Және оның программасы IBMQ кванттық компьютерінде орындалды. Оған қоса классикалық тірек векторлар методын қолданып кроссплатформды қосымша жасалды.

ҚОРЫТЫНДЫ

Диплом жұмысы барысында машинды оқыту алгоритмдері мен оның негізгі категорияларлы, шешетін есептері зерттелді. Оның қазіргі әлемдегі рөліне баға берілді. Тірек векторлар методының алгоритмі мен оның математикалық базасы зерттелді.

Оған қоса SVM алгоритмі мен grayscale-классификациясын қолданып алгоритм жасалды және оның интерфейсі мен қосымшасы жасалды. python тіліне арналған kivy-фреймворкі мен оның инструменттері, кроссплатформды қосымшалар идеясы зерттелді.

Бұл дипломдық жұмыста кванттық программалаудың жеке жағдайы – кванттық машинды оқытудың реализациясы, оның классикалық машинды оқытудан айырмашылығы және қолданылу перспективалары зерттелді. Оған қоса кванттық есептеулер элементтері қарастырылды. Реализация ретінде – 5-кубитті IBMQ кванттық компьютерінде python тілі мен оның qiskit кітапханасын қолдана отырып бинарлы классификатор көрсетілді. Және оның классикалық аналогі көрсетілді. Қолданылған басты әдіс – QSVM kernel немесе кванттық тірек векторлар ядросы. Сызықты бөлінбейтін класстар үшін – кеңістіктің өлшемін арттыру және жаңа характерискалар картасының бүкіл объектілерінің дистанция матрицасын, яғни ядро матрицасын тауып – классификация жасалды. Экспоненциалды қиындықты талап ететін – классификация есебі үшін – кванттық компьютер қолданылды.

Классикалық процессорлардың даму үдерісінің тоқтауына байланысты кванттық компьютерлер мен кванттық есептеулер – нағыз болашақ технологиясы болып табылады, және бұл саладағы зерттеулер – өте маңызды болып табылады.

Жұмыс актуалдығына дәлел ретінде қазіргі Google, IBM, Facebook және т.б. ірі IT-компаниялардың жыл сайын кванттық компьютерлер салу мен кванттық алгоритмдерді оптимизациялауға миллиондаған доллар жаратады. Яғни үлкен ақпараттармен жұмыс істеу кезіндегі кванттық артықшылықты бүкіл әлемде де бағалайды.

Нәтижесінде айтуға болатыны – кванттық есептеулер XX ғасырдың 60-жылдарынан бастап дамып келе жатқанына қарамастан - қазіргі кезде кванттық компьютерлер әлі де нашар дамыған болып келеді, бүкіл артықшылықтарына да қарамастан – кванттық үстемділік әлі толық орнаған жоқ(бір-екі компьютерді айтпасақ). Кванттық үстемділік орнау үшін, яғни классикалық компьютерлерден озу үшін кванттық компьютерде шамамен 49-51 кубиттей болу керек. Алайда – есептеу кезінде квадраттық немесе экспоненциалды жылдамдық өсуін беретін болғандықтан кванттық компьютерлер перспективада өте жоғары есептеуіш қуаттарға ие болады.

ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ

1 Nielsen, Michael A.; Chuang, Isaac L. (2010). Quantum Computation and Quantum Information (2nd ed.). Cambridge: Cambridge University Press.

2 Wittek, Peter, “Quantum Machine Learning: What Quantum Computing Means to Data Mining”, Academic Press; Paperback reprint of hardcover 1st ed., 2014 edition (September 2, 2016)

3 Айвазян С. А., Енюков И. С., Мешалкин Л. Д. Машинное обучение. URL: https://machinlearning.ru/ - профессиональный русскоязычный информационно-аналитический ресурс по машинному обучению, распознаванию образов и интеллектуальному анализу данных.

4 Вапник В. Н. Восстановление зависимостей по эмпирическим данным. — М.: Наука, 1979. — 448 с.

5 Баумейстер Д., Экерт А., Цайлингер А. Физика квантовой информации. — М.: Постмаркет, 2002. — 376 с.

6 К. А. Валиев, «Квантовая информатика: компьютеры, связь икриптография» URL: http://vivovoco.astronet.ru/VV/JOURNAL/VRAN/QUBIT/QUBIT.HTM - вестник российской академии наук

7 Schölkopf B., Smola A.J. Learning with Kernels. Support Vector Machines, Regularization, Optimization, and Beyond. — MIT Press, Cambridge, MA, 2002.