Цифрлық әдіскер
ақпараттық-білім беру ортасы

ПРАКТИКАЛЫҚ САБАҚТАР

Жасанды интеллект (машиналық оқыту негіздері)

Машиналық оқытудың толық циклі: Python кітапханалары (NumPy, Pandas, Scikit-learn) арқылы деректерді өңдеу, модель құру және бағалау.

- Машиналық оқыту циклі
- NumPy, Pandas, Scikit-learn
- Жіктеу және болжау
- Деректерді визуализациялау

Пән санаты
ЖИ және математика

adisker.kz
Әдістемелік қор

ӘДІСТЕМЕЛІК НҰСҚАУ

Қысқаша ұйымдастыру-әдістемелік сипаттама «Жасанды интеллект негіздері» пәні бойынша тәжірибелік сабақтарға арналған әдістемелік нұсқаулық – бұл студенттердің теориялық білімін іс жүзінде қолдана отырып, Python тілін, мәліметтерді өңдеу, визуализациялау және машиналық оқыту модельдерін құру дағдыларын қалыптастыруға бағытталған оқу құралы. Негізгі мақсат:

- Жасанды интеллект және машиналық оқыту саласындағы базалық практикалық дағдыларды дамыту;
- Python, NumPy, Pandas, Matplotlib, Scikit-learn, Anaconda, JupyterLab құралдарын пайдалану арқылы студенттерге нақты тапсырмаларды орындау мүмкіндігін беру. Ұйымдастыру ерекшеліктері:
- Әрбір практикалық жұмыс 15 аптаға жоспарланған және тақырыптық модульдерге бөлінген.
- Сабақтар жеке және топтық форматта, компьютерлік сыныпта немесе қашықтан (Jupyter/Google Colab) форматында өткізіледі.
- Әр сабақ нұсқау (инструкция), тапсырма, мысалдар, бақылау сұрақтары және қорытынды рефлексиядан тұрады. Оқыту әдістері:
- Интерактивті демонстрация (ноутбук немесе тақтада кодпен жұмыс);
- "Code-along" практикумы — оқытушымен қатар студенттің коды жазуы;
- Жұптық/топтық mini-hackathon;
- ЖИ көмегімен кодты жазу және талдау (GitHub Copilot, ChatGPT, Replit AI);
- Автоматтандырылған бағалау құралдары (Kaggle, Google Colab Tests, nbgrader). Студент үшін қажетті құралдар:
- Орнатылған Anaconda ортасы немесе Google Colab;
- Python тілінің негізгі кітапханалары (numpy, pandas, sklearn, matplotlib, seaborn);
- GitHub немесе Google Drive – жұмыс портфолиосын сақтау үшін;
- Жеке жұмыс дәптері немесе электрондық notebook – код пен түсіндірмелерді біріктіру үшін. Бағалау:
- Әр практикалық жұмыстың нәтижесі жұмыстың орындалу сапасына, нақты нәтижеге жетуге және кодтың дұрыстығына қарай бағаланады;
- Қосымша ретінде өзіндік талдау, жобалық элементтер және автоматтандырылған тестілеу нәтижелері ескеріледі.

Практикалық сабақ №1

Тақырыбы: Python орнату және Jupyter Notebook пайдалану

1. Мақсаты:
2. Студенттерге Python бағдарламалау ортасын таныстыру;
3. Anaconda дистрибутивін орнату арқылы Python мен Jupyter Notebook құралын іске қосуға үйрету;
4. Jupyter-де қарапайым Python бағдарламаларын орындауды меңгерту.

1. Міндеттері:
2. Anaconda платформасын жүктеу және орнату;
3. Jupyter Notebook интерфейсін игеру;
4. Ячейкалармен жұмыс істеу әдістерін үйрену;
5. Айнымалылар, арифметикалық операциялар, print() функциясын қолдану;
6. Markdown форматында мәтін жазуды үйрену.

1. Жоспар:
2. Python және Anaconda платформасы туралы жалпы түсінік;
3. Anaconda дистрибутивін орнату тәртібі;
4. Jupyter Notebook-ты іске қосу жолдары;

5. Кодтық және мәтіндік ячейкалармен жұмыс істеу;
6. Python-да қарапайым есептер шығару.

1. Қысқаша теориялық деректер: Python – оқуға жеңіл, кеңінен қолданылатын жоғары деңгейлі бағдарламалау тілі. Anaconda – Python және R тілдерімен жұмыс істеуге арналған, құрамында көптеген кітапханалар мен Jupyter, Spyder секілді құралдары бар платформа. Jupyter Notebook – деректерді өңдеу, код жазу және визуализация үшін қолданылатын интерактивті орта. Markdown – Jupyter-де мәтінді форматтауға арналған жеңіл тіл.

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. https://www.anaconda.com/ сайтына өтіңіз;
3. ОЖ сәйкес Anaconda орнату файлын жүктеңіз;
4. Орнату барысында барлық параметрді әдепкіде қалдырып, орнату аяқталғаннан кейін Anaconda Navigator-ды ашыңыз;
5. Jupyter Notebook қосымшасын іске қосыңыз;
6. Жаңа Notebook ашып, келесі әрекеттерді орындаңыз:
7. Бір айнымалыға мән меншіктеп, арифметикалық амалдар орындаңыз;
8. Нәтижені print() арқылы шығарыңыз;

1. Орындауға арналған тапсырмалар:
2. Anaconda орнату сәтінің скриншотын алыңыз;
3. Jupyter Notebook-ты іске қосып, жаңа файл жасаңыз;
4. Код ячейкасында келесі кодты орындаңыз: x = 7 y = 3 z = x + y print("Қосынды нәтижесі:", z)
5. Markdown ячейкасында келесі тақырыпты жазыңыз: # Python-да қарапайым есептеу мысалы
6. Бақылау сұрақтары:
7. Python тілінің қандай артықшылықтары бар?
8. Anaconda платформасының ерекшелігі қандай?
9. Jupyter Notebook дегеніміз не?
10. Ячейкалардың қандай түрлері бар және олардың қолданылуы?
11. Markdown қандай мақсатта қолданылады?

1. Үй тапсырмасы:
2. Жеке орта құрыңыз: python_intro_env;
3. Жаңа Jupyter notebook ашып, келесі әрекеттерді орындаңыз:
4. input() функциясымен пайдаланушыдан мәлімет алып, оны int-ке айналдырып, арифметикалық операция жасаңыз;
5. Нәтижені экранға шығарыңыз;
6. Түсіндірмесін Markdown форматында жазыңыз;
7. Орындалған жұмысты скриншоттап Google Drive немесе GitHub-та жүктеңіз.

1. Ұсынылатын әдебиеттер:
2. Wes McKinney — "Python for Data Analysis"
3. Jake VanderPlas — "Python Data Science Handbook"
4. https://docs.anaconda.com/
5. https://jupyter.org/
6. https://realpython.com/jupyter-notebook-introduction/

Практикалық сабақ №2

Тақырыбы: NumPy массивтерімен жұмыс істеу және есептер шығару

1. Мақсаты:
2. Студенттерді NumPy кітапханасымен таныстыру;
3. Бір және көп өлшемді массивтермен жұмыс жүргізуді үйрету;

4. Негізгі математикалық операцияларды массивтерге қолдану дағдыларын қалыптастыру.

1. Міндеттері:
2. NumPy кітапханасын жүктеп, Python ортасында пайдалану;
3. Массивтерді құру, индекстеу, қайта пішіндеу (reshape), біріктіру амалдарын орындау;
4. Статистикалық және арифметикалық есептеулер жүргізу;
5. Практикалық тапсырмалар арқылы массивтерді қолдануды бекіту.

1. Жоспар:
2. NumPy кітапханасын импорттау;
3. array, arange, linspace, zeros, ones, random функцияларын пайдаланып массив құру;
4. Индекстеу мен слайсинг тәсілдері;
5. reshape, flatten, transpose әдістері;
6. Арифметикалық операциялар мен статистикалық функциялар (mean, std, sum);
7. Массивтерді біріктіру (concatenate, hstack, vstack).

1. Қысқаша теориялық деректер: NumPy (Numerical Python) — ғылыми есептеулерге арналған Python кітапханасы. Оның негізгі құрылымы — ndarray типті массив. Массивтер тізімдерге қарағанда жылдамырақ және аз жады тұтынады. Олармен векторлық операциялар, логикалық сүзу, статистикалық есептеулер орындауға болады.

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. pip install numpy немесе Anaconda ортасында импорттау: import numpy as np
3. Бір өлшемді массив жасау: a = np.array([1, 2, 3, 4])
4. Екі өлшемді массив жасау: b = np.array([[1, 2], [3, 4]])
5. Массивтерге арифметикалық операция қолдану: x = np.array([10, 20, 30]) y = np.array([1, 2, 3]) print(x + y)
6. Статистикалық функциялар қолдану: print(np.mean(x), np.std(x))

1. Орындауға арналған тапсырмалар:
2. np.arange() көмегімен 0-ден 20-ға дейінгі сандардан тұратын массив құрыңыз;
3. Сол массивтен жұп сандарды слайсинг арқылы бөліп алыңыз;
4. 3×3 өлшемді кездейсоқ сандар матрицасын құрыңыз (np.random.randint());
5. reshape қолданып массивтің пішінін өзгертіңіз;
6. Екі массивті тігінен және көлденеңінен біріктіріңіз;
7. mean, max, min, sum, std функцияларын қолдана отырып сипаттама есептеңіз.

1. Бақылау сұрақтары:
2. NumPy массивінің ерекшелігі қандай?
3. Python тізімдері мен NumPy массивтерінің айырмашылығы қандай?
4. reshape() функциясы не үшін қажет?
5. np.mean() және np.std() нені есептейді?
6. Массивтерді біріктірудің қандай әдістері бар?

1. Үй тапсырмасы:
2. Кездейсоқ сандардан тұратын 5x5 матрица құрыңыз;
3. Диагональ элементтерін бөліп шығарыңыз;
4. Екінші және төртінші жолдарды алып тастаңыз;
5. Қалған матрицаның орташа мәнін есептеңіз;
6. Google Colab немесе Jupyter-де жазып, GitHub немесе Google Drive-та сақтаңыз.

Практикалық сабақ №3

Тақырыбы: Pandas кітапханасы: DataFrame және Series қолдану

1. Мақсаты:
2. Студенттерге Pandas кітапханасының негізгі құрылымдары — Series және DataFrame-мен жұмыс істеуді үйрету;
3. Деректерді құрылымдау, фильтрациялау және агрегаттау дағдыларын қалыптастыру.

1. Міндеттері:
2. Pandas кітапханасын жүктеп, Python ортасында пайдалану;
3. Series және DataFrame объектілерін құру және түрлендіру;
4. Индекстеу, сүзу (filtering), сұрыптау (sorting), топтау (groupby) амалдарын орындау;
5. Статистикалық сипаттама есептеу.

1. Жоспар:
2. Pandas кітапханасына шолу;
3. Series объектісі: құрылуы және қолданылуы;
4. DataFrame объектісі: бағандар мен қатарлар құрылымы;
5. Индекстеу, слайсинг, loc[], iloc[] әдістері;
6. Негізгі функциялар: describe(), head(), tail(), info();
7. Қатарлар мен бағандарды қосу, өшіру, өзгерту.

1. Қысқаша теориялық деректер: Pandas – Python тілінде кестелік деректермен (дерекқорлар, Excel, CSV) жұмыс істеуге арналған кітапхана. Series – бір өлшемді индексі бар құрылым (бір баған деректері). DataFrame – екі өлшемді құрылым, кестелік пішінде деректер сақтайды (жолдар мен бағандар).

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. pip install pandas арқылы Pandas орнату немесе Anaconda ортасында пайдалану;
3. Импорттау: import pandas as pd
4. Series жасау: s = pd.Series([10, 20, 30], index=['a', 'b', 'c'])
5. DataFrame жасау: data = {'Аты': ['Айдос', 'Жанар', 'Марат'], 'Жасы': [23, 21, 25]} df = pd.DataFrame(data)
6. Негізгі ақпаратты шығару: df.info() df.describe()
7. Индекстеу және сұрыптау: df['Жасы'] > 22 df.sort_values(by='Жасы')

1. Орындауға арналған тапсырмалар:
2. 5 студенттің аты, курсы және GPA-сы бар DataFrame құрыңыз;
3. GPA > 3.0 болатын студенттерді таңдаңыз;
4. Жаңа баған «Жетістік» қосып, GPA > 3.5 болса – «Жоғары», болмаса – «Орташа» деп белгілеңіз;
5. mean(), max(), min() функцияларымен GPA-ға статистикалық сипаттама жасаңыз;
6. Жасалған DataFrame-ді CSV-ге сақтап, қайта оқыңыз.

1. Бақылау сұрақтары:
2. Series пен DataFrame-нің айырмашылығы неде?
3. loc[] пен iloc[] айырмашылығы қандай?
4. describe() және info() функциялары не үшін қажет?
5. DataFrame-ге жаңа баған қалай қосылады?
6. CSV файлымен жұмыс істеу қадамдары қандай?

1. Үй тапсырмасы:
2. Тақырып: Университет кітапханасы
3. pandas көмегімен келесі деректерден тұратын DataFrame құрыңыз: кітап атауы, автор, басылым жылы, рейтинг;
4. 2010 жылдан кейін шыққан кітаптарды бөліп алыңыз;
5. Рейтингі > 4.5 кітаптарды сұрыптаңыз;

6. Жаңа баған қосыңыз: «Қолжетімділік» (иә/жоқ);
7. Жұмысты Jupyter-де орындап, нәтижесін скриншоттап жіберіңіз.

1. Ұсынылатын әдебиеттер:
2. Wes McKinney — "Python for Data Analysis"
3. https://pandas.pydata.org/
4. https://realpython.com/pandas-dataframe/
5. https://www.geeksforgeeks.org/python-pandas-tutorial/
6. https://jupyter.org/

Практикалық сабақ №4

Тақырыбы: Matplotlib көмегімен графиктер мен диаграммалар құру

1. Мақсаты:
2. Студенттерге Python-да деректерді визуализациялау негіздерін үйрету;
3. Matplotlib кітапханасын қолданып сызықтық графиктер, гистограммалар, дөңгелек диаграммалар салуды меңгерту;
4. Графиктерді баптау және түсіндіру дағдыларын дамыту.

1. Міндеттері:
2. Matplotlib кітапханасын орнату және импорттау;
3. Әртүрлі типтегі графиктер мен диаграммаларды құру;
4. Графиктердің атауын, осьтердің атын, аңызын (legend) беру;
5. Графиктерге түс, сызық стилі, маркер секілді баптаулар енгізу.

1. Жоспар:
2. Matplotlib туралы шолу;
3. pyplot модулін пайдалану;
4. Сызықтық график (plot()), бағандық диаграмма (bar()), гистограмма (hist()), дөңгелек диаграмма (pie());
5. Графикке тақырып, осьтер атауы, тор (grid()), аңыз (legend()) қосу;
6. Бірнеше графикті бір координат жүйесінде салу;
7. Графиктерді файл ретінде сақтау.

1. Қысқаша теориялық деректер: Matplotlib — Python-дағы ең кең таралған визуализация кітапханасы. Оның pyplot модулі MATLAB стилінде жұмыс істеуге мүмкіндік береді. График құру негізгі қадамдары:
2. import matplotlib.pyplot as plt
3. plt.plot(x, y) — деректерді көрсету;
4. plt.title(), plt.xlabel(), plt.ylabel() — графикті баптау;
5. plt.show() — графикті шығару.

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. Matplotlib кітапханасын орнату: pip install matplotlib
3. Импорттау: import matplotlib.pyplot as plt
4. Сызықтық график жасау: x = [1, 2, 3, 4, 5] y = [10, 20, 25, 30, 40] plt.plot(x, y) plt.title('Сызықтық график') plt.xlabel('X oci') plt.ylabel('Y oci') plt.grid(True) plt.show()
5. Дөңгелек диаграмма: labels = ['A', 'B', 'C'] values = [30, 45, 25] plt.pie(values, labels=labels, autopct='%1.1f%%') plt.title('Дөңгелек диаграмма') plt.show()

1. Орындауға арналған тапсырмалар:
2. 1-ден 10-ға дейінгі сандар үшін y = x^2 функциясының графигін салыңыз;
3. Студенттердің бағаларын көрсететін бағандық диаграмма құрыңыз;
4. Кітап жанрларының таралуын көрсету үшін дөңгелек диаграмма салыңыз;

5. Бір графикте екі функцияны бейнелеңіз: y1 = x^2 және y2 = x^3;
6. Барлық графикке атау, осьтер атауы және аңыз қосыңыз.

1. Бақылау сұрақтары:
2. Matplotlib кітапханасының басты мақсаты қандай?
3. plot(), bar(), pie() функцияларының қолдану ерекшеліктері?
4. legend() не үшін қажет?
5. Графикке тақырып пен тор қалай қосылады?
6. Графикті файлға сақтау тәсілі қандай?

1. Үй тапсырмасы:
2. Кез келген CSV файлдан деректер оқып, сол деректер негізінде 2 түрлі график құрыңыз;
3. Бірі сызықтық график, екіншісі – гистограмма болуы тиіс;
4. Графиктерге атау, түс, сызық түрі және белгілер қосыңыз;
5. Жұмысты Google Colab немесе Jupyter-де орындап, нәтижені PDF немесе PNG форматында сақтаңыз.

1. Ұсынылатын әдебиеттер:
2. https://matplotlib.org/stable/contents.html
3. https://realpython.com/python-matplotlib-guide/
4. https://www.geeksforgeeks.org/python-matplotlib-tutorial/
5. Jake VanderPlas — "Python Data Science Handbook"
6. https://jupyter.org/

Практикалық сабақ №5

Тақырыбы: Қатынас диаграммалары мен жұптық визуализация (Seaborn)

1. Мақсаты:
2. Студенттерге Seaborn кітапханасы арқылы деректер арасындағы байланысты визуализациялауды үйрету;
3. Жұптық диаграмма (pairplot) және қатынас диаграммаларын (scatterplot, heatmap) құру дағдыларын дамыту.

1. Міндеттері:
2. Seaborn кітапханасымен танысу;
3. Деректер арасындағы өзара байланысты анықтау;
4. Түсін, маркерді, категорияны визуалдауда қолдану;
5. Графиктерге қосымша түсіндірмелер беру.

1. Жоспар:
2. Seaborn кітапханасына кіріспе;
3. Қатынас диаграммасы (scatterplot, relplot);
4. Жұптық визуализация (pairplot);
5. Корреляция матрицасы және heatmap;
6. Категориялық түстерді қолдану, hue және style параметрлері;
7. Seaborn графиктерін Matplotlib көмегімен толықтыру.

1. Қысқаша теориялық деректер: Seaborn – Matplotlib негізінде құрылған, деректердің күрделі құрылымдарын визуалдауға арналған Python кітапханасы.
2. scatterplot() — екі үздіксіз айнымалы арасындағы байланысты көрсетеді.
3. pairplot() — бірнеше айнымалылардың арасындағы өзара байланыс диаграммалары мен гистограммаларды бірден көрсетеді.

4. heatmap() — корреляция матрицасын визуалдауға арналған.

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. Seaborn орнату: pip install seaborn
3. Импорттау: import seaborn as sns import matplotlib.pyplot as plt
4. Кіріктірілген деректер жинағын жүктеу: df = sns.load_dataset('iris')
5. Қатынас диаграммасын салу: sns.scatterplot(x='sepal_length', y='sepal_width', hue='species', data=df) plt.title('Сепал ұзындығы мен ені арасындағы қатынас') plt.show()
6. Жұптық визуализация: sns.pairplot(df, hue='species') plt.show()
7. Корреляция және жылу картасы: corr = df.corr(numeric_only=True) sns.heatmap(corr, annot=True, cmap='coolwarm') plt.title('Корреляция матрицасы') plt.show()

1. Орындауға арналған тапсырмалар:
2. iris немесе tips деректер жинағын жүктеңіз;
3. Екі үздіксіз айнымалы үшін қатынас диаграммасын салыңыз (scatterplot);
4. pairplot көмегімен барлық сандық айнымалылар арасындағы байланысты бейнелеңіз;
5. Корреляция матрицасын есептеп, heatmap визуализациясын жасаңыз;
6. Барлық графиктерге тақырып, ось атауы және аңыз (legend) қосыңыз.

1. Бақылау сұрақтары:
2. Seaborn кітапханасының ерекшелігі неде?
3. pairplot қандай мақсатта қолданылады?
4. hue параметрі не үшін қажет?
5. Корреляция матрицасы дегеніміз не?
6. heatmap пен scatterplot айырмашылығы қандай?

1. Үй тапсырмасы:
2. penguins деректер жинағын жүктеп, 4 үздіксіз айнымалы бойынша pairplot құрыңыз;
3. species бағанын hue ретінде пайдаланыңыз;
4. Корреляцияны есептеп, heatmap арқылы бейнелеңіз;
5. Жұмысты Jupyter-де орындап, PDF не PNG ретінде сақтап, GitHub/Google Drive-қа жүктеңіз.

1. Ұсынылатын әдебиеттер:
2. https://seaborn.pydata.org/
3. https://realpython.com/seaborn-python-visualization/
4. https://www.geeksforgeeks.org/seaborn-library-in-python/
5. Jake VanderPlas — "Python Data Science Handbook"
6. https://jupyter.org/

Практикалық сабақ №6

Тақырыбы: Мәліметтерді тазалау және шкалалау

1. Мақсаты:
2. Студенттерге деректерді алдын ала өңдеудің маңыздылығын түсіндіру;
3. Таза, шкалаланған деректердің машиналық оқытуда қаншалықты маңызды екенін көрсету;
4. Pandas, NumPy және Scikit-learn кітапханаларын пайдалана отырып деректерді өңдеуге үйрету.

1. Міндеттері:
2. Жоғалған мәндерді (missing values) табу және толтыру;
3. Аномалияларды анықтау және өңдеу;
4. Шкалалау әдістерін (Min-Max, StandardScaler) қолдану;

5. Категориялық айнымалыларды сандық форматқа ауыстыру (encoding).

1. Жоспар:
2. Деректердегі бос мәндерді іздеу және өңдеу (isnull(), fillna(), dropna());
3. Шығарылған мәндер мен аномалияларды өңдеу;
4. Статистикалық сипаттама арқылы деректерді бағалау;
5. StandardScaler және MinMaxScaler көмегімен шкалалау;
6. Категориялық деректерді get_dummies және LabelEncoder арқылы кодтау.

1. Қысқаша теориялық деректер: Мәліметтерді тазалау — деректерді анализге немесе модельге дайындау үшін бос мәндерден, аномалиялардан арылу процесі. Шкалалау — әртүрлі өлшемдегі мәндерді бір шкалаға келтіру (мысалы, 0-1 немесе стандартты нормалау). Encoding — мәтіндік (категориялық) айнымалыларды машиналық оқытуға жарамды сандық түрге айналдыру.

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. Кітапханаларды импорттау: import pandas as pd import numpy as np from sklearn.preprocessing import StandardScaler, MinMaxScaler, LabelEncoder
3. Жасанды деректер жинағын жасау немесе CSV жүктеу;
4. Жоғалған мәндерді анықтау: df.isnull().sum() df.fillna(method='ffill', inplace=True)
5. Аномалияларды анықтау және алып тастау: q1 = df['баға'].quantile(0.25) q3 = df['баға'].quantile(0.75) iqr = q3 - q1 filtered_df = df[(df['баға'] >= q1 - 1.5*iqr) & (df['баға'] <= q3 + 1.5*iqr)]
6. Шкалалау: scaler = StandardScaler() df[['жасы', 'табысы']] = scaler.fit_transform(df[['жасы', 'табысы']])
7. Категориялық деректерді кодтау: le = LabelEncoder() df['жыныс'] = le.fit_transform(df['жыныс'])

1. Орындауға арналған тапсырмалар:
2. Жасанды немесе дайын деректер жинағын (мысалы, titanic, tips) жүктеңіз;
3. Бос мәндердің санын есептеп, оларды орташа мәнмен немесе алдыңғы мәнмен толтырыңыз;
4. 1-2 бағаң бойынша шкалалау жүргізіңіз (StandardScaler, MinMaxScaler);
5. 1-2 категориялық бағанды сандық форматқа ауыстырыңыз;
6. Өңделген деректерді .csv файлына сақтаңыз.

1. Бақылау сұрақтары:
2. Мәліметтерді тазалау не үшін қажет?
3. Қандай жағдайларда dropna() қолдануға болады?
4. StandardScaler мен MinMaxScaler айырмашылығы қандай?
5. Категориялық деректерді кодтаудың қандай әдістерін білесіз?
6. Шкалаланбаған деректер модельдің нәтижесіне қалай әсер етеді?

1. Үй тапсырмасы:
2. titanic деректер жиынын қолдана отырып:
3. жасы, тариф және жынысы бойынша деректерді алдын ала өңдеңіз;
4. шкалалау және кодтау тәсілдерін қолданыңыз;
5. өңделген деректерді жеке .csv файлға сақтап, нәтижесін Google Drive немесе GitHub-қа жүктеңіз.

1. Ұсынылатын әдебиеттер:
2. https://pandas.pydata.org/docs/
3. https://scikit-learn.org/stable/modules/preprocessing.html
4. https://realpython.com/python-data-cleaning-numpy-pandas/
5. https://www.geeksforgeeks.org/data-cleaning/
6. https://jupyter.org/

Тақырыбы: Жаңа айнымалы құру (Feature Engineering)

1. Мақсаты:
2. Студенттерге деректерден жаңа, мәнді айнымалылар жасау арқылы модель сапасын жақсартуды үйрету;
3. Feature engineering ұғымы мен қолдану тәсілдерін практикада меңгерту.

1. Міндеттері:
2. Қолданыстағы деректерден жаңа белгілерді (features) құру;
3. Логикалық, математикалық, санаттық негізде айнымалы жасау;
4. Айнымалылар арасындағы өзара қатынасты қолдану арқылы жаңа мағыналы көрсеткіштер жасау;
5. Уақыттық және мәтіндік деректерден айнымалылар шығару.

1. Жоспар:
2. Feature engineering түсінігі және маңызы;
3. Ағымдағы айнымалылардан арифметикалық жолмен жаңа айнымалы құру;
4. Санаттық айнымалылардан индикаторлық айнымалылар жасау;
5. Мәтіндік айнымалылардан ұзындық, символдар саны сияқты белгілер алу;
6. Уақыт айнымалыларынан күн, ай, апта, т.б. шығару;
7. Feature engineering нәтижесін визуалдап бағалау.

1. Қысқаша теориялық деректер: Feature engineering — машиналық оқыту моделіне ең тиімді ақпаратты беру үшін жаңа айнымалыларды құру процесі. Жаңа айнымалыларды келесі жолдармен жасауға болады:
2. Бағандар арасындағы математикалық операциялар;
3. Логикалық шарттардан derived feature жасау;
4. Уақыт пен мәтіндік деректерден құрамдас белгілерді шығару.

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. Кітапханаларды импорттау: import pandas as pd import numpy as np
3. Деректер жинағын жүктеу: df = pd.read_csv('https://raw.githubusercontent.com/mwaskom/seaborn-data/master/tips.csv')
4. Мысал – жалпы шотқа пайыздық тип бойынша жаңа айнымалы қосу: df['tip_percent'] = df['tip'] / df['total_bill'] * 100
5. Мысал – таңертең/кеш айнымалысы: df['is_lunch'] = df['time'] == 'Lunch'
6. Мәтіннен ұзындық алу: df['name_length'] = df['sex'].astype(str).apply(len)
7. Уақыт айнымалыларынан жыл, ай, күн шығару: df['date'] = pd.to_datetime(df['date']) df['month'] = df['date'].dt.month

1. Орындауға арналған тапсырмалар:
2. tips немесе titanic деректер жинағын қолданыңыз;
3. Жаңа бағаң ретінде «tip процентін» есептеңіз;
4. total_bill > 20 болған кезде «High bill» деген индикатор жасаңыз;
5. sex және smoker бағандарынан біріктірілген profile атты жаңа бағаң жасаңыз;
6. Құрылған айнымалылардың мәнін describe() көмегімен сипаттаңыз.

1. Бақылау сұрақтары:
2. Feature engineering-нің машиналық оқытудағы рөлі қандай?
3. Қай кезде жаңа айнымалы құру қажет?
4. Қандай айнымалы түрлерінен жаңа белгілер жасауға болады?
5. Уақыттық айнымалылармен қандай операциялар жүргізуге болады?
6. Feature engineering нәтижесін қалай бағалауға болады?

1. Үй тапсырмасы:
2. titanic деректер жиынынан:
3. «family_size» айнымалысын жасаңыз (sibsp + parch);

4. age бағаны негізінде age_group санаттарын құрыңыз: бала, жас, ересек, қарт;
5. Жаңа айнымалыларды визуалдап (гистограмма, boxplot) көрсетіңіз;
6. Жұмысты Jupyter-де орындап, нәтижесін сақтап, Google Drive не GitHub-та жариялаңыз.

1. Ұсынылатын әдебиеттер:
2. https://towardsdatascience.com/feature-engineering
3. https://scikit-learn.org/stable/modules/preprocessing.html
4. https://realpython.com/python-data-cleaning-numpy-pandas/
5. https://www.kaggle.com/learn/feature-engineering
6. https://jupyter.org/

Практикалық сабақ №8

Тақырыбы: Классификация әдістері: Decision Tree және K-Nearest Neighbors (KNN)

1. Мақсаты:
2. Decision Tree және KNN алгоритмдерінің жұмыс істеу принципін үйрету;
3. Scikit-learn кітапханасын пайдаланып модель құру, жаттықтыру және болжау;
4. Нәтижелерді интерпретациялау және визуализациялау дағдыларын қалыптастыру.

1. Міндеттері:
2. Decision Tree және KNN алгоритмдерімен танысу;
3. Деректер жиынын бөліп, модель жаттықтыру;
4. Болжау, дәлдік пен қателіктерді бағалау;
5. Графикалық түсіндірулер арқылы модельді түсіндіру.

1. Жоспар:
2. Классификация ұғымы мен түрлері;
3. Decision Tree алгоритмінің логикасы (шарттарға негізделген бөлу);
4. K-Nearest Neighbors логикасы (қашықтыққа негізделген);
5. Scikit-learn көмегімен модельдерді құру;
6. train_test_split, fit(), predict() функцияларын пайдалану;
7. confusion_matrix, accuracy_score, classification_report көмегімен бағалау.

1. Қысқаша теориялық деректер: Decision Tree — деректерді шарттар арқылы бөліп, шешім ағашы негізінде болжау жасайды. K-Nearest Neighbors (KNN) — жаңа деректі ең жақын көршілеріне қарап классификациялайды. Ең жиі кездесетін класс таңдалады. Осы әдістер классификациялық мәселелерде кеңінен қолданылады: мәтін тану, медициналық диагноз, маркетингтік талдау.

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. Кітапханаларды импорттау: import pandas as pd from sklearn.model_selection import train_test_split from sklearn.tree import DecisionTreeClassifier from sklearn.neighbors import KNeighborsClassifier from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
3. Деректерді жүктеу: from sklearn.datasets import load_iris iris = load_iris() X = iris.data y = iris.target
4. Деректерді бөлу: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
5. Decision Tree моделі: dt_model = DecisionTreeClassifier() dt_model.fit(X_train, y_train) y_pred_dt = dt_model.predict(X_test)
6. KNN моделі: knn_model = KNeighborsClassifier(n_neighbors=3) knn_model.fit(X_train, y_train) y_pred_knn = knn_model.predict(X_test)
7. Бағалау: print(confusion_matrix(y_test, y_pred_dt)) print(accuracy_score(y_test, y_pred_knn))

1. Орындауға арналған тапсырмалар:
2. iris немесе titanic деректер жинағымен жұмыс істеңіз;

3. Decision Tree классификаторын жаттықтырып, дәлдігін есептеңіз;
4. 3, 5, 7 көршілер саны бойынша KNN классификаторын қолданып салыстырыңыз;
5. Нәтижелерді confusion_matrix және classification_report көмегімен талдаңыз;
6. Классификация нәтижесін визуалдаңыз (бар болса, PCA немесе 2D plot).

1. Бақылау сұрақтары:
2. Decision Tree қалай жұмыс істейді?
3. KNN әдісінің негізгі идеясы қандай?
4. train_test_split функциясы не үшін қажет?
5. Модельдің сапасын бағалау үшін қандай метрикалар қолданылады?
6. Decision Tree мен KNN алгоритмдерінің айырмашылығы неде?

1. Үй тапсырмасы:
2. wine немесе breast_cancer деректер жиынын жүктеп:
3. Decision Tree және KNN модельдері арқылы болжау жасаңыз;
4. Екі модельдің нәтижесін салыстырып, қайсысы жақсы екенін сипаттаңыз;
5. Нәтижені .csv файлда сақтап, GitHub немесе Google Drive-та бөлісіңіз.

1. Ұсынылатын әдебиеттер:
2. https://scikit-learn.org/stable/modules/tree.html
3. https://scikit-learn.org/stable/modules/neighbors.html
4. https://realpython.com/decision-tree-classifier-python/
5. https://www.geeksforgeeks.org/ml-decision-tree-classifier-in-python/
6. https://jupyter.org/

Практикалық сабақ №9

Тақырыбы: Регрессиялық модельдер: Linear Regression және Ridge Regression

1. Мақсаты:
2. Студенттерге регрессиялық модельдердің жұмыс істеу принципін үйрету;
3. Жай сызықтық регрессия және Ridge регрессиясының айырмашылығын көрсету;
4. Модель нәтижелерін интерпретациялау мен бағалау.

1. Міндеттері:
2. LinearRegression және Ridge алгоритмдерін қолдану;
3. Модельге деректерді жаттықтыру және болжау жасау;
4. Нәтижені визуалдау және сапасын бағалау;
5. mean_squared_error, r2_score сияқты метрикаларды қолдану.

1. Жоспар:
2. Регрессия ұғымы және түрлері;
3. Сызықтық регрессия формуласы мен модельдің мағынасы;
4. Ridge регрессиясының артықшылығы (регуляризация);
5. Scikit-learn көмегімен модель құру;
6. Деректерді жаттықтыру және тестілеу;
7. Нәтижені визуализациялау және интерпретациялау.

1. Қысқаша теориялық деректер: Сызықтық регрессия (Linear Regression) — мақсатты айнымалы мен тәуелсіз айнымалылар арасындағы сызықтық байланысты модельдейді. Ridge Regression — сызықтық регрессияның

кеңейтілген түрі, ол модельдің күрделілігін шектейтін (регуляризациялайтын) тәсілді қолданады. Бұл артық сәйкестік (overfitting) жағдайын болдырмауға көмектеседі.

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. Қажетті кітапханаларды импорттау: import numpy as np import pandas as pd import matplotlib.pyplot as plt from sklearn.linear_model import LinearRegression, Ridge from sklearn.model_selection import train_test_split from sklearn.metrics import mean_squared_error, r2_score
3. Деректерді жүктеу (мысалы, үй бағасы, студенттің нәтижесі): from sklearn.datasets import load_diabetes data = load_diabetes() X = data.data y = data.target
4. Деректерді бөлу: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
5. Linear Regression: lr = LinearRegression() lr.fit(X_train, y_train) y_pred_lr = lr.predict(X_test)
6. Ridge Regression: ridge = Ridge(alpha=1.0) ridge.fit(X_train, y_train) y_pred_ridge = ridge.predict(X_test)
7. Бағалау және салыстыру: print("LR R2:", r2_score(y_test, y_pred_lr)) print("Ridge R2:", r2_score(y_test, y_pred_ridge))

1. Орындауға арналған тапсырмалар:
2. load_diabetes немесе басқа CSV файлды қолданып, деректер жүктеңіз;
3. Сызықтық регрессия моделін құрып, болжау жасаңыз;
4. Ridge регрессиясын құрып, салыстырмалы бағалау жүргізіңіз;
5. Нәтижені кесте және график арқылы көрсетіңіз;
6. r2_score пен mean_squared_error арқылы модель сапасын салыстырыңыз.

1. Бақылау сұрақтары:
2. Сызықтық регрессияның негізгі формуласы қандай?
3. Ridge регрессиясы қандай жағдайларда қолданылады?
4. r2_score не көрсетеді?
5. Overfitting дегеніміз не?
6. Регрессиялық модельдер қай салаларда қолданылады?

1. Үй тапсырмасы:
2. Өзіңіз таңдаған CSV деректер жинағымен жұмыс істеп:
3. Linear және Ridge регрессия моделін құрыңыз;
4. Екі модельдің нәтижесін салыстырып, R^2 және MSE мәндерін жазыңыз;
5. Графиктік түрде болжау нәтижесін бейнелеңіз (шынайы және болжау сызықтары).

1. Ұсынылатын әдебиеттер:
2. https://scikit-learn.org/stable/modules/linear_model.html
3. https://realpython.com/linear-regression-in-python/
4. https://www.geeksforgeeks.org/ml-linear-regression/
5. https://towardsdatascience.com/understanding-ridge-regression
6. https://jupyter.org/

Практикалық сабақ №10

Тақырыбы: Модель параметрлерін оңтайландыру: GridSearchCV және Cross-validation

1. Мақсаты:
2. Студенттерге модель сапасын жақсарту үшін гиперпараметрлерді таңдаудың маңызын түсіндіру;
3. GridSearchCV және кросс-валидация әдістерін пайдалану арқылы оңтайлы модель параметрлерін табуға үйрету.

1. Міндеттері:
2. Cross-validation ұғымымен таныстыру және түрлерін меңгерту;
3. GridSearchCV арқылы параметрлерді іздеуді жүзеге асыру;

4. Модельдің артық үйрену (overfitting) мен кем үйрену (underfitting) жағдайларын бақылау;
5. Ең жақсы параметрлермен жұмыс істейтін модель құру.

1. Жоспар:
2. Cross-validation түсінігі және не үшін қажет;
3. KFold, StratifiedKFold тәсілдерінің айырмашылығы;
4. GridSearchCV-мен гиперпараметрлер таңдау;
5. Сызықтық немесе Decision Tree моделі мысалында қолдану;
6. Нәтижені бағалау және кесте түрінде көрсету;
7. Үздік параметрмен алынған модельді қолдану.

1. Қысқаша теориялық деректер: Кросс-валидация (Cross-validation) — деректер жиынын бірнеше бөлікке (fold) бөліп, модельді бірнеше рет жаттықтыру мен тестілеу арқылы жалпы сапаны бағалау әдісі. GridSearchCV — барлық мүмкін параметр комбинацияларын тексеріп, ең тиімдісін таңдауға мүмкіндік беретін автоматты іздеу әдісі.

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. Қажетті кітапханаларды импорттау: from sklearn.datasets import load_iris from sklearn.model_selection import GridSearchCV, cross_val_score from sklearn.tree import DecisionTreeClassifier from sklearn.model_selection import train_test_split, KFold
3. Деректерді жүктеу және бөлу: data = load_iris() X = data.data y = data.target
4. GridSearchCV конфигурациясы: param_grid = {'max_depth': [2, 3, 4, 5], 'criterion': ['gini', 'entropy']} model = DecisionTreeClassifier() gs = GridSearchCV(model, param_grid, cv=5) gs.fit(X, y)
5. Нәтижені шығару: print("Үздік параметрлер:", gs.best_params_) print("Үздік дәлдік:", gs.best_score_)
6. Cross-validation мысалы: scores = cross_val_score(model, X, y, cv=5) print("CV дәлдік нәтижелері:", scores) print("Орташа дәлдік:", scores.mean())

1. Орындауға арналған тапсырмалар:
2. Decision Tree немесе KNN моделіне GridSearchCV қолданып параметр таңдаңыз;
3. 3 түрлі гиперпараметр комбинациясын қолданыңыз (K, depth, criterion т.б.);
4. Әрбір конфигурация үшін cross_val_score мәндерін салыстырыңыз;
5. Үздік параметр негізінде модель құрып, нәтижесін визуалдаңыз;
6. Әртүрлі cv мәндерін қолдану әсерін сипаттаңыз.

1. Бақылау сұрақтары:
2. GridSearchCV дегеніміз не және ол не үшін қажет?
3. Cross-validation көмегімен не тексеріледі?
4. Артық үйрену (overfitting) мен кем үйрену (underfitting) деген не?
5. cv=5 дегеніміз нені білдіреді?
6. Ең жақсы параметрлерді таңдаудың басқа жолдары қандай?

1. Үй тапсырмасы:
2. wine немесе digits деректер жиынын қолдана отырып:
3. GridSearchCV арқылы Decision Tree немесе KNN параметрлерін оңтайландырыңыз;
4. Үздік параметрмен модель жаттықтырып, дәлдігін есептеңіз;
5. Алынған нәтижелерді .csv немесе .xlsx форматында кесте түрінде сақтаңыз.

1. Ұсынылатын әдебиеттер:
2. https://scikit-learn.org/stable/modules/grid_search.html
3. https://realpython.com/grid-search-python-machine-learning/
4. https://towardsdatascience.com/cross-validation-explained
5. https://www.geeksforgeeks.org/ml-hyperparameter-tuning/

6. https://jupyter.org/

Практикалық сабақ №11

Тақырыбы: Model Training Pipeline – модельді дайындау мен жаттықтыру процесін автоматтандыру

1. Мақсаты:
2. Машиналық оқытуда толық дайындық және жаттықтыру процесін құрылымдау;
3. Pipeline көмегімен деректерді өңдеу, модель жаттықтыру және болжауды біріздендіру;
4. Кодты тиімді және қайта пайдалануға ыңғайлы түрде ұйымдастыру.

1. Міндеттері:
2. Pipeline ұғымымен танысу және оның құрылымын үйрену;
3. StandardScaler, LabelEncoder, SimpleImputer, модельді біріктіріп pipeline құру;
4. Pipeline негізінде модельді жаттықтырып, болжау жасау;
5. GridSearchCV арқылы pipeline ішінде параметр оңтайландыруды меңгеру.

1. Жоспар:
2. Pipeline деген не және оның артықшылықтары;
3. sklearn.pipeline.Pipeline синтаксисі;
4. Импьютер, шкалалау, кодтау және модельдерді біріктіру;
5. Тестілеу және болжау;
6. GridSearchCV көмегімен pipeline ішіндегі параметрді таңдау;
7. Финал нәтижесін бағалау.

1. Қысқаша теориялық деректер: Pipeline — деректерді алдын ала өңдеу мен модельді құру кезеңдерін тізбектеп біріктіретін құрал. Бұл тәсіл модельді қайталап оқыту процесін автоматтандырады және қатесіз орындалуын қамтамасыз етеді. Pipeline компоненттері:
2. Импьютер (SimpleImputer);
3. Шкалалау (StandardScaler);
4. Кодтау (OneHotEncoder);
5. Модель (Classifier/Regressor).

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. Қажетті кітапханаларды импорттау: import pandas as pd from sklearn.datasets import load_titanic from sklearn.model_selection import train_test_split, GridSearchCV from sklearn.pipeline import Pipeline from sklearn.impute import SimpleImputer from sklearn.preprocessing import StandardScaler from sklearn.linear_model import LogisticRegression from sklearn.metrics import classification_report
3. Деректерді дайындау: df = pd.read_csv('https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.csv') X = df[['Age', 'Fare']] y = df['Survived']
4. Деректерді бөлу: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
5. Pipeline құру: pipeline = Pipeline([('imputer', SimpleImputer(strategy='mean')), ('scaler', StandardScaler()), ('classifier', LogisticRegression())])
6. Модельді жаттықтыру: pipeline.fit(X_train, y_train)
7. Нәтижені бағалау: y_pred = pipeline.predict(X_test) print(classification_report(y_test, y_pred))

1. Орындауға арналған тапсырмалар:
2. titanic, iris, немесе өз деректеріңізді пайдаланып Pipeline жасаңыз;
3. 3 компоненттен (импьютер, шкалалау, модель) тұратын Pipeline құрыңыз;
4. Pipeline көмегімен модельді жаттықтырып, тест нәтижесін бағалаңыз;
5. GridSearchCV қолдана отырып classifier параметрлерін оңтайландырыңыз;

6. Pipeline-мен алынған модельді .pkl файлына сақтаңыз.

1. Бақылау сұрақтары:
2. Pipeline деген не және не үшін қажет?
3. Pipeline қандай модульдерден тұрады?
4. Қарапайым код пен pipeline негізіндегі кодтың айырмашылығы неде?
5. GridSearchCV қалай pipeline ішінде жұмыс істейді?
6. Модельді .pkl файлында сақтау не үшін қажет?

1. Үй тапсырмасы:
2. Өзіңіз таңдаған деректер жиынымен жұмыс істеп:
3. 4 сатыдан тұратын pipeline жасаңыз (импьютер, шкалалау, кодтау, модель);
4. Модельді жаттықтырып, дәлдік метрикаларын есептеңіз;
5. Үздік параметрлерді GridSearchCV арқылы анықтап, нәтижені .pkl форматында сақтаңыз.

1. Ұсынылатын әдебиеттер:
2. https://scikit-learn.org/stable/modules/compose.html
3. https://realpython.com/python-pipelines-machine-learning/
4. https://towardsdatascience.com/sklearn-pipeline-clearly-explained
5. https://www.geeksforgeeks.org/ml-pipeline-using-scikit-learn/
6. https://jupyter.org/

Практикалық сабақ №12

Тақырыбы: Жұмыс ортасын баптау – Python, Jupyter, Anaconda және кітапханаларымен жұмыс

1. Мақсаты:
2. Студенттерді Python бағдарламалау ортасын дұрыс орнату мен баптауға үйрету;
3. Jupyter Notebook ортасында тәжірибе жасауға мүмкіндік беру;
4. Қажетті кітапханалар мен құралдармен таныстыру және орнатуды меңгерту.

1. Міндеттері:
2. Anaconda платформасын орнату;
3. Виртуалды орта құру және конфигурациялау;
4. Jupyter Notebook іске қосу;
5. Қажетті кітапханаларды (numpy, pandas, matplotlib, scikit-learn) орнату және тестілеу.

1. Жоспар:
2. Anaconda платформасы және оның компоненттері;
3. Anaconda көмегімен виртуалды орта құру;
4. Jupyter Notebook пен VS Code жұмыс принциптері;
5. pip және conda арқылы кітапхана орнату;
6. Python скрипті мен notebook құру;
7. Кітапханаларымен сынама код орындау.

1. Қысқаша теориялық деректер: Anaconda — Python тіліндегі ғылыми есептеулерге арналған толық платформа. Оның құрамында Jupyter, Spyder, VS Code, Conda сияқты құралдар бар. Jupyter Notebook — Python кодын интерактивті түрде жазуға, өңдеуге және орындауға мүмкіндік беретін құрал. Conda — кітапханалар мен орталарды басқаруға арналған пакет менеджері.

1. Практикалық жұмысты орындауға арналған нұсқаулықтар: 1-қадам: Anaconda орнату
2. https://www.anaconda.com/products/distribution сайтына өтіп, операциялық жүйеңізге сәйкес Anaconda дистрибутивін жүктеп орнатыңыз. 2-қадам: Виртуалды орта құру bash КопироватьРедактировать conda create -n ml_env python=3.11

conda activate ml_env 3-қадам: Қажетті кітапханаларды орнату bash КопироватьРедактировать conda install numpy pandas matplotlib scikit-learn jupyter 4-қадам: Jupyter іске қосу bash КопироватьРедактировать jupyter notebook 5-қадам: Жаңа notebook ашу және сынама код: python КопироватьРедактировать import numpy as np import pandas as pd import matplotlib.pyplot as plt from sklearn.linear_model import LinearRegression

```
print("Барлығы дұрыс жұмыс істеп тұр!")
```

1. Орындауға арналған тапсырмалар:
2. Anaconda платформасын орнатыңыз;
3. ml_env виртуалды ортасын құрып, қажетті кітапханаларды орнатыңыз;
4. Jupyter Notebook іске қосып, жаңа notebook жасаңыз;
5. Python кітапханаларын импорттап, шағын кодпен тексеріңіз;
6. VS Code пен Jupyter арасындағы айырмашылықтарды зерттеңіз.

1. Бақылау сұрақтары:
2. Anaconda деген не және не үшін қажет?
3. conda мен pip командаларының айырмашылығы қандай?
4. Jupyter Notebook қандай жағдайда қолайлы?
5. Виртуалды орта не үшін қолданылады?
6. VS Code пен Jupyter айырмашылығы неде?

1. Үй тапсырмасы:
2. Anaconda көмегімен жеке орта жасап, scikit-learn кітапханасын орнатыңыз;
3. Ортаның скриншоттарын жасап, Jupyter ішіндегі шағын мысал кодының нәтижесін көрсетіңіз;
4. Қай кітапхананың орнатылуы қиын болды? Себебін сипаттаңыз.

1. Ұсынылатын әдебиеттер:
2. https://www.anaconda.com/
3. https://jupyter.org/
4. https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html
5. https://realpython.com/installing-python/
6. https://scikit-learn.org/stable/install.html

Практикалық сабақ №13

Тақырыбы: Confusion Matrix, ROC және AUC көрсеткіштері арқылы модельді бағалау

1. Мақсаты:
2. Классификация моделінің нәтижелерін нақты және дәл бағалау әдістерімен таныстыру;
3. Confusion Matrix, ROC curve және AUC көрсеткіштерін түсіндіру және қолдану;
4. Бұл метрикаларды Python көмегімен есептеуді үйрету.

1. Міндеттері:
2. Confusion Matrix ұғымымен және құрылымымен танысу;
3. True Positive, False Positive, True Negative, False Negative анықтамаларын түсіну;
4. ROC curve және AUC мәнінің маңызын анықтау;
5. Scikit-learn көмегімен аталған метрикаларды есептеу және визуализациялау.

1. Жоспар:
2. Классификация метрикаларына шолу;
3. Confusion Matrix құрылымы және есептелуі;

4. Precision, Recall, F1-score түсінігі;
5. ROC curve түсінігі және оның графигі;
6. AUC (Area Under Curve) мәнінің интерпретациясы;
7. Python көмегімен есептеу және визуализациялау.

1. Қысқаша теориялық деректер: Confusion Matrix — классификация нәтижелерін 4 негізгі элементке бөледі: TP, TN, FP, FN.
2. TP (True Positive): дұрыс позитив болжам
3. TN (True Negative): дұрыс негатив болжам
4. FP (False Positive): қате позитив болжам
5. FN (False Negative): қате негатив болжам ROC (Receiver Operating Characteristic) — нақты позитивтік көрсеткіш пен жалған позитивтік көрсеткіштің арасындағы байланыс графигі. AUC (Area Under Curve) — ROC астындағы аудан. AUC мәні 1-ге неғұрлым жақын болса, модель соғұрлым жақсы жұмыс істейді.

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. Қажетті кітапханаларды импорттау: from sklearn.datasets import load_breast_cancer from sklearn.model_selection import train_test_split from sklearn.linear_model import LogisticRegression from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay from sklearn.metrics import roc_curve, roc_auc_score, RocCurveDisplay
3. Деректерді жүктеу және бөлу: data = load_breast_cancer() X = data.data y = data.target X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
4. Модельді жаттықтыру: model = LogisticRegression(max_iter=10000) model.fit(X_train, y_train) y_pred = model.predict(X_test) y_proba = model.predict_proba(X_test)[:, 1]
5. Confusion Matrix есептеу: cm = confusion_matrix(y_test, y_pred) ConfusionMatrixDisplay(cm).plot()
6. ROC және AUC есептеу: fpr, tpr, thresholds = roc_curve(y_test, y_proba) roc_auc = roc_auc_score(y_test, y_proba) RocCurveDisplay(fpr=fpr, tpr=tpr, roc_auc=roc_auc).plot()

1. Орындауға арналған тапсырмалар:
2. Breast Cancer немесе басқа деректер жиынын пайдаланып логистикалық регрессия моделін құрыңыз;
3. Confusion Matrix есептеңіз және интерпретация жасаңыз;
4. Precision, Recall, F1-score мәндерін анықтаңыз;
5. ROC curve сызып, AUC есептеңіз;
6. Нәтижелерді график және кесте түрінде көрсетіңіз.
7. Бақылау сұрақтары:
8. Confusion Matrix деген не және ол не үшін қолданылады?
9. TP, FP, TN, FN нені білдіреді?
10. Precision мен Recall арасындағы айырмашылық қандай?
11. ROC curve қандай мағына береді?
12. AUC мәні 0.5 болған жағдайда модель қаншалықты жақсы?

1. Үй тапсырмасы:
2. Өз деректер жиыныңызда классификация модель жасап:
3. Confusion Matrix құрыңыз;
4. Precision, Recall, F1 және AUC көрсеткіштерін есептеңіз;
5. ROC curve диаграммасын салыңыз және түсіндіріңіз;
6. Барлық нәтижелерді .ipynb немесе .pdf ретінде сақтап тапсырыңыз.

1. Ұсынылатын әдебиеттер:
2. https://scikit-learn.org/stable/modules/model_evaluation.html
3. https://realpython.com/logistic-regression-python/
4. https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/

5. https://www.geeksforgeeks.org/confusion-matrix-machine-learning/

Практикалық сабақ №14

Тақырыбы: Жобалық жұмыс жоспары және деректер жиынтығын дайындау

1. Мақсаты:
2. Машиналық оқыту жобасын өз бетінше жоспарлап, құрылымдау;
3. Деректерді таңдау, талдау және дайындау дағдыларын қалыптастыру;
4. Жобаны аяқтау үшін қажетті кезеңдерді анықтап, логикалық тәртіппен орналастыру.
5. Міндеттері:
6. Жобаның мақсаттарын нақтылау;
7. Деректер көздерін таңдау (ашық деректер жиыны);
8. Деректерді жүктеу, зерттеу және алдын ала өңдеу;
9. Модель таңдауға дайындық;
10. Жобаны құжаттау құрылымын жасау.
11. Жоспар:
12. Жоба идеяларын талқылау;
13. Мақсат пен міндеттерді анықтау;
14. Деректер жиыны мен сипаттамаларын таңдау;
15. Деректерді жүктеу, тазалау, сипаттау;
16. Алдын ала визуализация жасау;
17. Жоба кезеңдерін бөлу (pipeline);
18. GitHub немесе Google Drive арқылы жобаны сақтау.

1. Қысқаша теориялық деректер: Жобалық жұмыс — нақты мәселені шешу үшін машиналық оқытуды қолданатын практикалық зерттеу. Ол идеядан бастап нәтижені талдауға дейінгі барлық кезеңдерді қамтиды. Деректер көздері:
2. Kaggle.com
3. UCI Machine Learning Repository
4. Datahub.io
5. Google Dataset Search Жақсы деректер жиыны:
6. Таза және толық;
7. Мән-мағынасы бар белгілерден тұрады;
8. Қисынды мақсаттық айнымалысы болады (target).

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. 2–3 жоба идеяларын ойластырыңыз (мысалы: үй бағасын болжау, клиентті жоғалту, ауруды анықтау т.б.);
3. Әр идея бойынша:
4. Мақсат (нені болжау/анықтау керек?)
5. Жинақ (қандай деректер қажет?)
6. Күтілетін нәтиже мен бағалау метрикасы
7. Таңдалған идея үшін деректер жиынын жүктеңіз;
8. Pandas көмегімен сипаттамалық статистика жүргізіңіз: import pandas as pd url = 'https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.csv' df = pd.read_csv(url) df.info() df.describe()
9. Белгілерді визуализациялау (histogram, scatter, boxplot);
10. Деректерді тазарту және сақтап қою.

1. Орындауға арналған тапсырмалар:
2. Жоба идеяларын генерациялау (кемінде 2);
3. Әр жоба үшін деректер жиынын тауып, сипаттаңыз;

4. Жоба кезеңдерін 5–6 сатыға бөліп жоспарлаңыз (pipeline);
5. Деректерді Pandas арқылы жүктеп, тексеріңіз;
6. Дайын деректерді .csv немесе .xlsx файл ретінде сақтаңыз.
7. Бақылау сұрақтары:
8. Машиналық оқыту жобасы қалай басталады?
9. Жақсы деректер жиынының негізгі белгілері қандай?
10. Жоба мақсаты мен target айнымалысы қалай байланысты?
11. Деректерді қайдан табуға болады?
12. Жобаны құжаттау үшін қандай құрылым тиімді?

1. Үй тапсырмасы:
2. Таңдалған жобаға сәйкес деректер жиынын жүктеп, өңдеңіз;
3. Жоба жоспарын құрыңыз: мақсат, деректер, кезеңдер;
4. Python көмегімен деректерді талдап, алдын ала визуализация жасаңыз;
5. Файлдарды Google Drive немесе GitHub-та сақтап, сілтемесін тапсырыңыз.

1. Ұсынылатын әдебиеттер:
2. https://kaggle.com
3. https://scikit-learn.org/stable/
4. https://towardsdatascience.com/data-science-project-structure
5. https://pandas.pydata.org/
6. https://matplotlib.org/
7. https://seaborn.pydata.org/

Практикалық сабақ №15

Тақырыбы: Жоба қорғау және презентация жасау

1. Мақсаты:
2. Студенттердің жеке немесе топтық жобаларын қорғай алу дағдыларын қалыптастыру;
3. Жобаның логикалық құрылымын түсінікті түрде ұсыну;
4. Модель нәтижелерін визуалды және статистикалық тұрғыдан дәлелдеу.

1. Міндеттері:
2. Жобаны нақты, қысқа және түсінікті форматта таныстыру;
3. Жоба кезеңдерін логикалық тәртіппен баяндау;
4. Негізгі шешімдер мен нәтижелерді визуализациялау;
5. Қорғау кезінде қойылған сұрақтарға дәлелмен жауап беру.

1. Жоспар:
2. Жоба мақсаты мен өзектілігін сипаттау;
3. Деректер жиынына шолу жасау;
4. Алдын ала өңдеу мен визуализация нәтижелері;
5. Таңдалған модельдер мен гиперпараметрлер;
6. Бағалау метрикалары: Accuracy, F1, AUC т.б.;
7. Қорытынды: нәтижелер мен ұсыныстар;
8. Сұрақ-жауап кезеңі.

1. Қысқаша теориялық деректер: Жоба қорғау — студенттің машиналық оқыту жобасы бойынша жүргізген жұмысының нәтижесін қысқаша және көрнекі түрде таныстыру процесі. Жақсы презентация мынадай қасиеттерге ие:
2. Мақсат нақты;

3. Құрылымы логикалық;
4. Графиктер мен диаграммалар арқылы көрнекі;
5. Нәтижелер нақты санмен берілген;
6. Ұсыныстар негізделген.

1. Практикалық жұмысты орындауға арналған нұсқаулықтар:
2. PowerPoint, Google Slides немесе Jupyter Notebook негізінде жоба слайдтарын дайындаңыз (8–12 слайд немесе ұяшықтан тұратын блокнот);
3. Кіріспеде жоба мақсаты мен деректер жиыны сипатталуы керек;
4. Деректерді өңдеу кезеңдерін, модельдерді, және нәтижелерді диаграммалармен көрсетіңіз;
5. Әрбір нәтиже бойынша қысқаша қорытынды жазыңыз;
6. Қосымшада қолданылған кітапханалар мен код үзінділері көрсетілуі мүмкін;
7. Топпен жасалған жоба болса, әр қатысушы өз үлесін баяндасын.

1. Орындауға арналған тапсырмалар:
2. Презентация үшін слайдтар дайындаңыз немесе Jupyter блокнотты интерактивті түрде реттеңіз;
3. Модельді бағалау метрикаларын нақты графиктермен көрсетіңіз;
4. Ең маңызды нәтиже мен қиындықты нақты сипаттаңыз;
5. Жобалық жұмысты аудитория алдында 5–7 минут ішінде қорғауға дайындалыңыз.

1. Бақылау сұрақтары:
2. Жобаңыздың басты мақсаты не болды?\n2. Модельді неге дәл осы түрін таңдадыңыз?\n3. Қандай қиындықтарға тап болдыңыз?\n4. Нәтижелер қандай шешімдерге алып келді?\n5. Егер жобаны кеңейтетін болсаңыз, қандай жақсартулар енгізер едіңіз?

1. Үй тапсырмасы:
2. Жобалық жұмысты финалдық форматқа келтіріңіз (слайдтар, .ipynb файл, деректермен бірге .zip немесе репозиторий сілтемесі);
3. Презентацияңызды қорғау алдында репетиция жасаңыз;
4. Қысқаша бейнематериал түсіруге немесе плакат жасауға болады (қалауыңызша).

1. Ұсынылатын әдебиеттер:
2. https://www.kaggle.com/learn/data-science-projects
3. https://realpython.com/python-data-visualization-guide/
4. https://www.tidyverse.org/
5. https://scikit-learn.org/stable/visualizations.html
6. https://jupyter.org/