久留米工業大学
KURUME INSTITUTE OF TECHNOLOGY

久留米工業大学における
全学的DS・AI教育の概要

2020.11.30

久留米工業大学AI応用研究所
小田まり子
mari@kurume-it.ac.jp

令和2年8月
数理・データサイエンス教育強化拠点コンソーシアム連携校

AI応用研究所：2020年4月開所

①AIによる地域課題解決

②AI技術者の輩出　→　DS・AIリテラシ教育

数理・データサイエンス・AIリテラシー教育

久留米工業大学
KURUME INSTITUTE OF TECHNOLOGY

教育改革に向けた主な取り組み

デジタル社会の「**読み・書き・そろばん**」である「**数理・データサイエンス・AI**」の基礎などの必要な力を**全ての国民**が育み、あらゆる分野で人材が活躍

- 本学：２０２０年後期から「AI概論」を開始
- 対象：全学科1年生を対象（必修科目）実施中
- 学共通教育のAI教育カリキュラム（シラバス）の作成・AI教育　→　AI応用研究所が担当

＜現在の実施方法＞「AI概論」　専任教員1名＋SA

- 全1年生を30人規模の14クラスに分けた少人数教育
- 遠隔ビデオ講義7回・対面講義8回を隔週で受講

モデルカリキュラムと教育方法を参考

- **モデルカリキュラムと教育方法**

導入	**1. 社会におけるデータ・AI利活用**	
	1-1. 社会で起きている変化	1-2. 社会で活用されているデータ
	1-3. データ・AIの活用領域	1-4. データ・AI利活用のための技術
	1-5. データ・AI利活用の現場	1-6. データ・AI利活用の最新動向
基礎	**2. データリテラシー**	
	2-1. データを読む	2-2. データを説明する
	2-3. データを扱う	
心得	**3. データ・AI利活用における留意事項**	
	3-1. データ・AIを扱う上での留意事項	3-2. データを守る上での留意事項
選択	**4. オプション**	
	4-1. 統計および数理基礎	4-2. アルゴリズム基礎
	4-3. データ構造とプログラミング基礎	4-4. 時系列データ解析
	4-5. テキスト解析	4-6. 画像解析
	4-7. データハンドリング	4-8. データ活用実践（教師あり学習）
	4-9. データ活用実践（教師なし学習）	

モデルカリキュラム：導入分野

導入	1. 社会におけるデータ・AI利活用	
	1-1. 社会で起きている変化	1-2. 社会で活用されているデータ
	1-3. データ・AIの活用領域	1-4. データ・AI利活用のための技術
	1-5. データ・AI利活用の現場	1-6. データ・AI利活用の最新動向

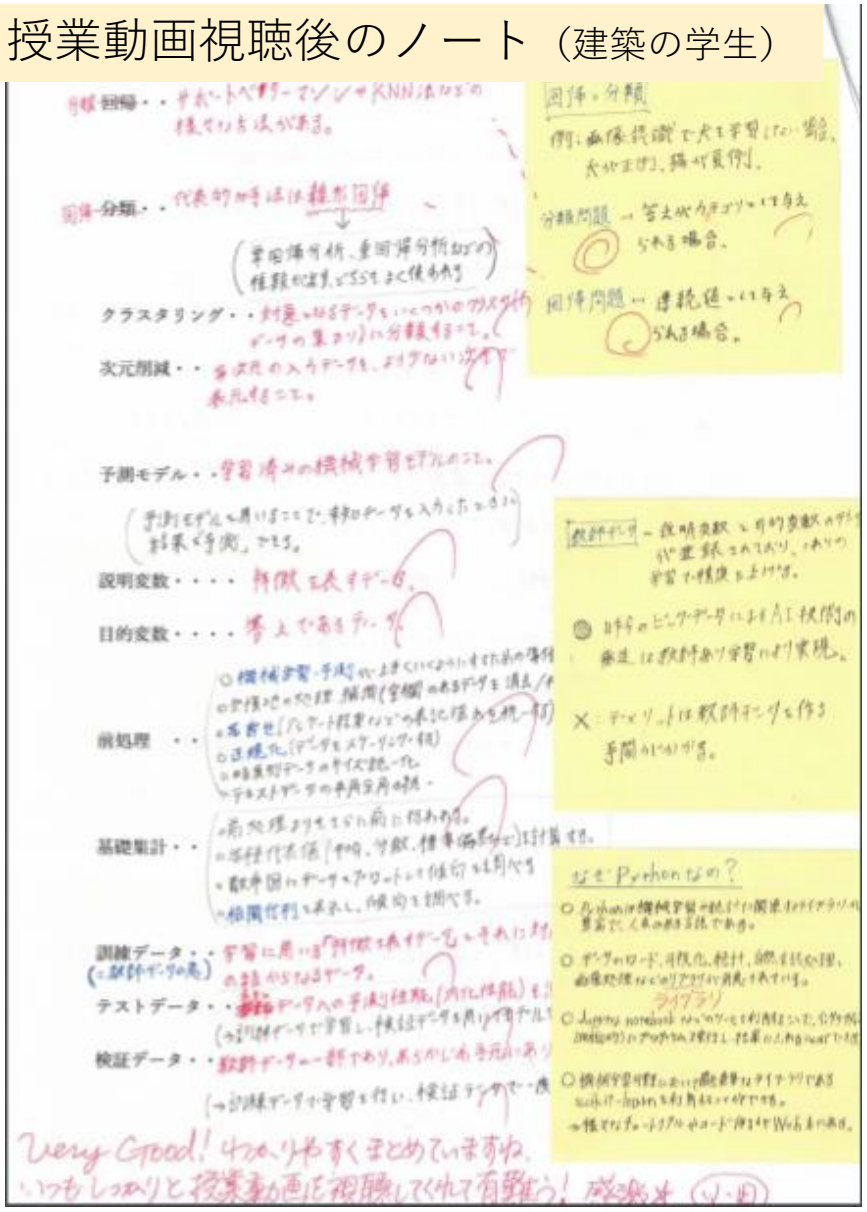
- データ・AI利活用事例を紹介した動画（MOOC等）を使った**反転学習**を取り入れ、講義ではデータ・AI活用領域の広がりや、技術概要の解説を行うことが望ましい。
- 学生がデータ・AI利活用事例を調査し発表する**グループワーク**等を行い、一方通行で事例を話すだけの講義にしないことが望ましい。

本学　→　座学（講義動画）

導入	1．人工知能（AI）とはなにか
導入	2．社会におけるデータ・AI利活用（**グループワーク：調査**） ・データ・AI 利活用における留意事項 ・データを守るうえでの留意事項　→　**個人レポート**
導入 導入	3．AI利活用における最新動向 (ビジネス・テクノロジー) 4．グループによる調査内容の発表→　**来年度「AI活用演習」**

授業動画視聴後の課題レポート例

授業動画視聴後のノート（建築の学生）



- 手書き記入用ノートを印刷・配布
- 授業用Moodleにファイルも UP
  → ワープロ入力して印刷も可
  → 9割以上の学生が手書きで提出

- 一言、必ずコメントを記入
- 隔週で行う対面講義の際に、学生の顔を見ながら返却
  → 学生へのフィードバック
  → 学生の顔を覚える

- 学生への質問には（できるだけ）親切に対応
  → 学生の理解度を知る。
  （何に困っているのか？）

心得（データ･AI利活用の留意事項）

心得

3. データ・AI利活用における留意事項

- 3-1. データ・AIを扱う上での留意事項
- 3-2. データを守る上での留意事項

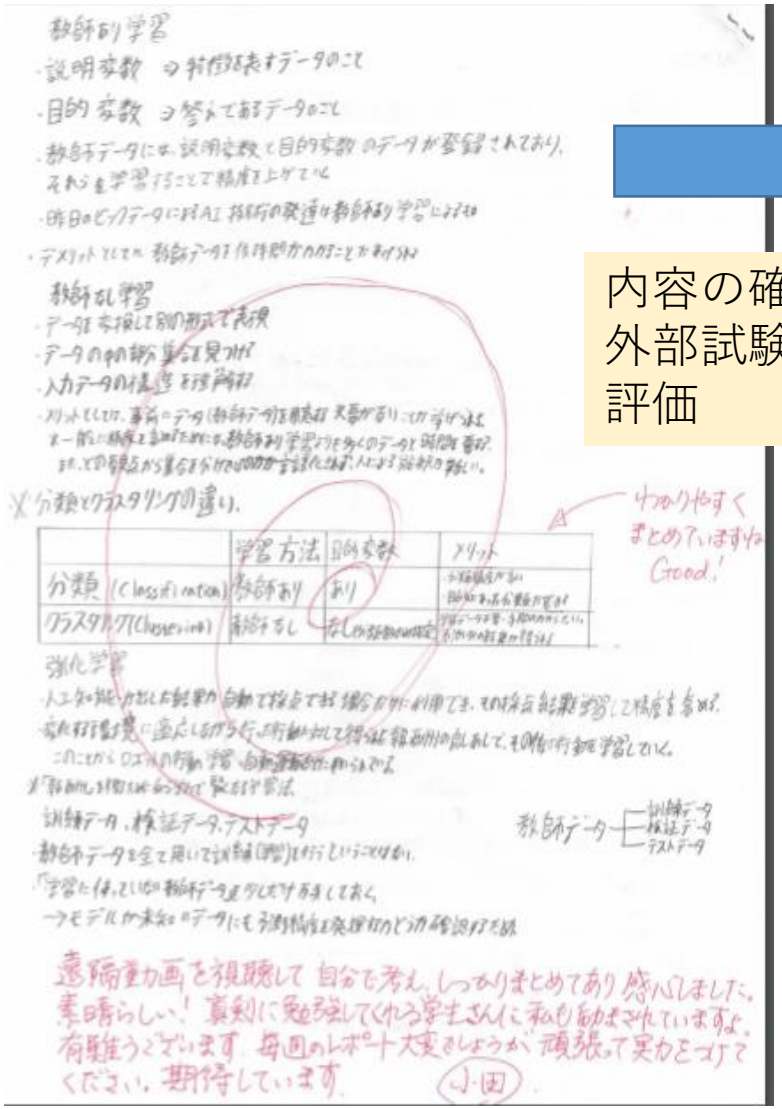
- データ駆動型社会のリスクを**自分ごと**として考えさせることが望ましい。
- データ・AIが引き起こす課題について**グループディスカッション**等を行い、一方通行で事例を話すだけの講義にしないことが望ましい。

本学　→　グループディスカッション・発表

心得 データ・AI利活用における留意事項	・データ・AI 利活用における留意事項 ・ データを守るうえでの留意事項 グループ発表　→ **来年度　AI活用演習で実施** 「AIの課題と今後の発展」

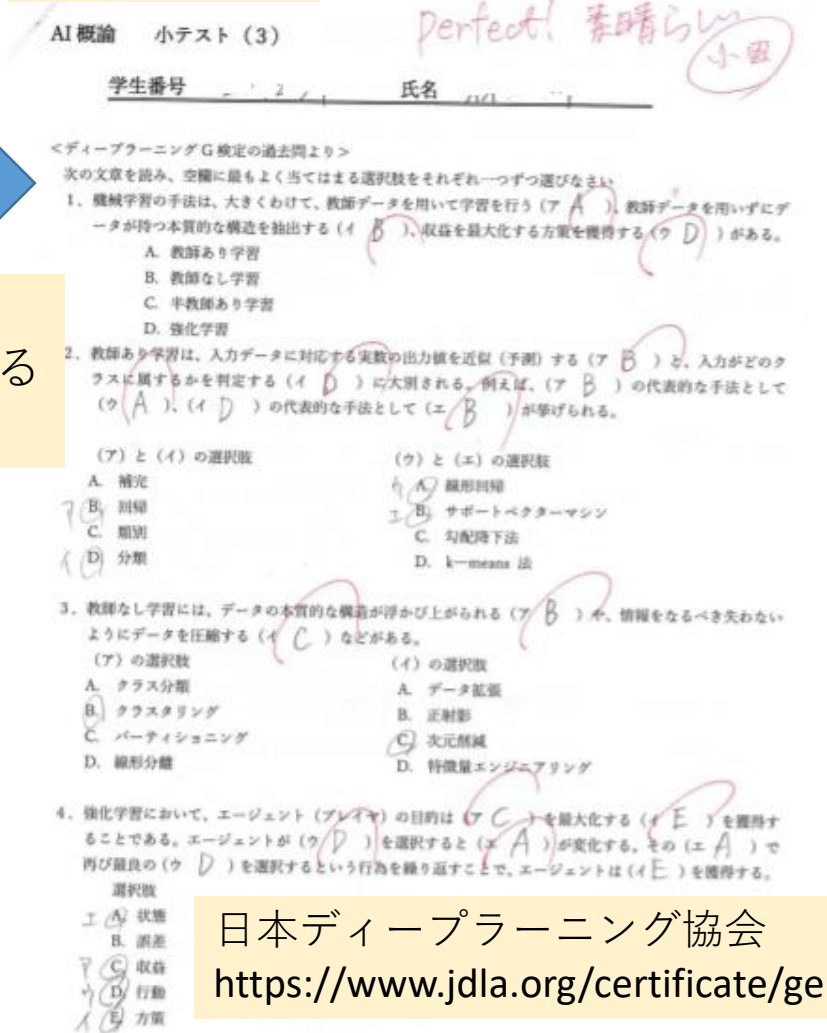
学生のノート例と小テスト解答例

授業動画視聴後のノート（交通の学生）



内容の確認
外部試験による
評価

G検定の問題



AI概論　小テスト（3）

学生番号　　氏名

＜ディープラーニングG検定の過去問より＞

次の文章を読み、空欄に最もよく当てはまる選択肢をそれぞれ一つずつ選びなさい。

1．機械学習の手法は、大きくわけて、教師データを用いて学習を行う（ア A ）、教師データを用いずにデータが持つ本質的な構造を抽出する（イ B ）、収益を最大化する方策を獲得する（ウ D ）がある。

A. 教師あり学習
B. 教師なし学習
C. 半教師あり学習
D. 強化学習

2．教師あり学習は、入力データに対応する実数の出力値を近似（予測）する（ア B ）と、入力がどのクラスに属するかを判定する（イ D ）に大別される。例えば、（ア B ）の代表的な手法として（ウ A ）、（イ D ）の代表的な手法として（エ B ）が挙げられる。

（ア）と（イ）の選択肢
A. 補完
B. 回帰
C. 順位
D. 分類

（ウ）と（エ）の選択肢
A. 線形回帰
B. サポートベクターマシン
C. 勾配降下法
D. k-means 法

3．教師なし学習には、データの本質的な構造が浮かび上がられる（ア B ）や、情報をなるべく失わないようにデータを圧縮する（イ C ）などがある。

（ア）の選択肢
A. クラス分類
B. クラスタリング
C. パーティショニング
D. 線形分離

（イ）の選択肢
A. データ拡張
B. 正射影
C. 次元削減
D. 特徴量エンジニアリング

4．強化学習において、エージェント（プレイヤ）の目的は（ア C ）を最大化する（イ E ）を獲得することである。エージェントが（ウ D ）を選択すると（エ A ）が変化する。その（エ A ）で再び最良の（ウ D ）を選択するという行為を繰り返すことで、エージェントは（イ E ）を獲得する。

選択肢
A. 状態
B. 誤差
C. 収益
D. 行動
E. 方策

＜裏も自由に利用してください＞　質問・要望などを書いてくれても OK です。

日本ディープラーニング協会
https://www.jdla.org/certificate/general/

基礎（データリテラシー）

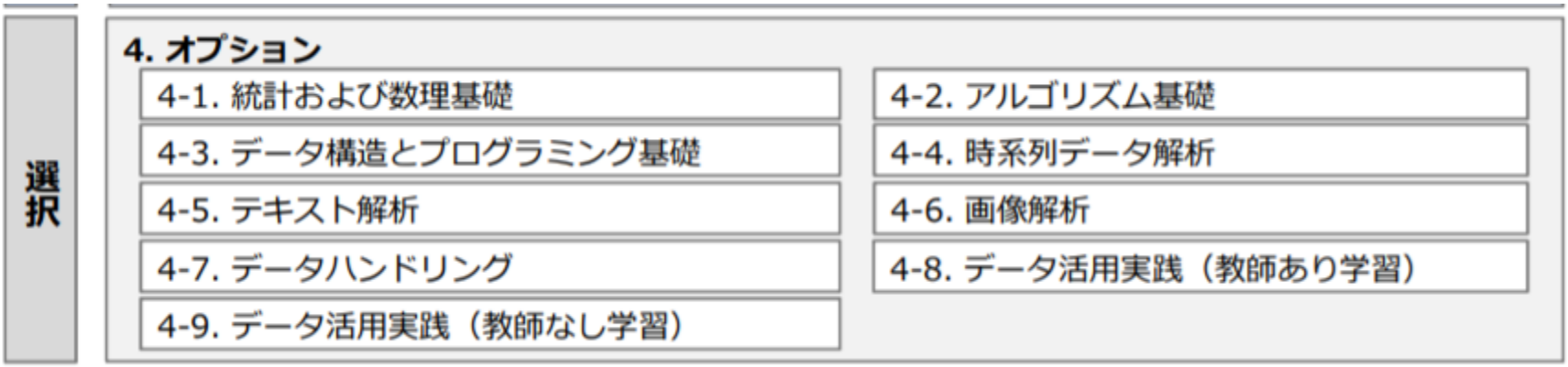
- 各大学・高専の特徴に応じて**適切なテーマ**を設定し、**実データ**（あるいは模擬データ）を用いた講義を行うことが望ましい。
- 実際に手を動かしてデータを可視化する等、学生自身がデータ利活用プロセスの一部を**体験**できることが望ましい。

必携PCの環境構築（インストール）からスタート

本学　→　プログラミング対面（人数：半分ずつ）

オプション　プログラミング基礎・アルゴリズム基礎	Pythonの基礎（１）データ・処理・関数
オプション　統計および数理基礎	主な統計量（平均・分散・標準偏差）
基礎　データリテラシー（プログラミング）	Pythonの基礎（２）Pandas(データの読み込み、データ抽出、統計量の算出） ・Matplotlib（ヒストグラム、箱ひげ図、相関関係）

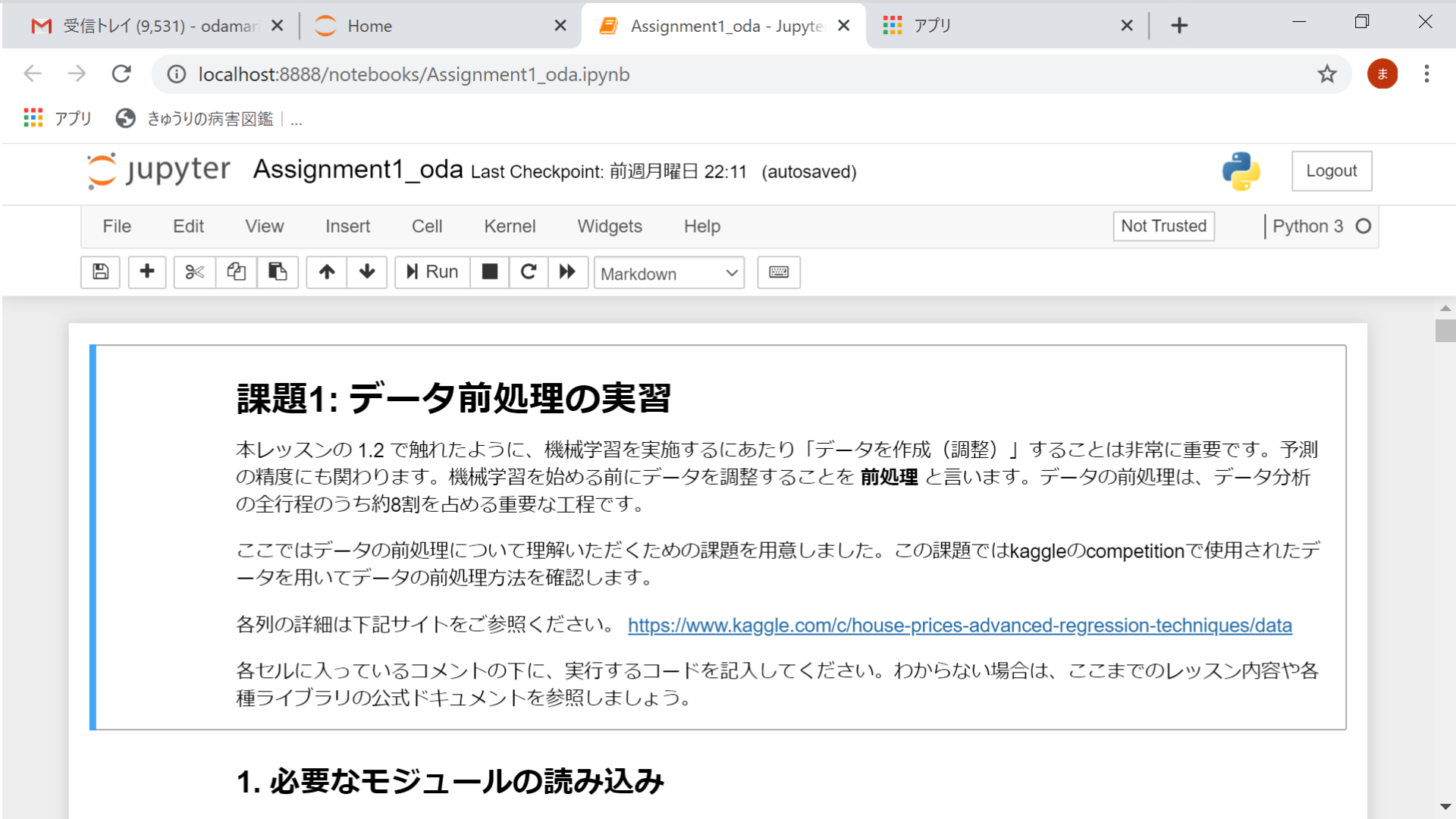
モデルカリキュラム（オプション）



- 本内容は**オプション**扱いとし、大学・高専の特徴に応じて学修内容を選択する。

４－１　４－３	AIのための基礎数学　　線形代数の基礎　Numpyを用いた演習
４－１　４－３	要約統計量、前処理
４－３　４－８	機械学習実践（１）演習「犬・猫の分類」 （２）　課題「手書き数字の分類」
４－３、６、８	機械学習実践　　分析データを基にした花の分類 →　来年度（復習として）
４－３，４，８	機械学習実践（３）「回帰」ビットコインの予測
４－３、６、８	機械学習実践　　「外れ値、スケーリング考慮の家賃予測」 →　来年度「AI活用演習」で

Pythonによる演習例



Jupyter notebook でファイルを配布→　その日の学習内容解説

必要なライブラリの読み込み

1. 必要なモジュールの読み込み

In [6]:
```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

matplotlib.style.use('ggplot')

%matplotlib inline
```

データの読み込み

2. データの読み込み

CSVファイル"kaggle_housing_price.csv"を読み込み、内容を確認します。

In [7]:
```
# データを変数datasetに読み込む
housing_data = pd.read_csv("kaggle_housing_price.csv")
```

In [8]:
```
# データを最初の5行だけ表示
housing_data.head()
```

Out[8]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utiliti
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPu
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPu
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPu
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPu
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPu

5 rows × 81 columns

DataFrameの `shape` プロパティで全データの行数と列数を取得できます。

参照：http://pandas.pydata.org/pandas-docs/version/0.23/generated/pandas.DataFrame.shape.html

要約統計量：

3. 要約統計量を出力する

データ数、平均や中央値、標準偏差などの統計量を確認し、データへの理解を深めます。

なお、DataFrameの `describe()` を使うと、様々な統計量の情報を要約として表示してくれます。

参考：http://pandas.pydata.org/pandas-docs/version/0.23/generated/pandas.DataFrame.describe.html

In [10]:
```
# 要約統計量を表示
housing_data.describe()
```

Out[10]:

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBu
count	1460.000000	1460.000000	1201.000000	1460.000000	1460.000000	1460.000000	1460.0000
mean	730.500000	56.897260	70.049958	10516.828082	6.099315	5.575342	1971.2678
std	421.610009	42.300571	24.284752	9981.264932	1.382997	1.112799	30.2029
min	1.000000	20.000000	21.000000	1300.000000	1.000000	1.000000	1872.0000
25%	365.750000	20.000000	59.000000	7553.500000	5.000000	5.000000	1954.0000
50%	730.500000	50.000000	69.000000	9478.500000	6.000000	5.000000	1973.0000
75%	1095.250000	70.000000	80.000000	11601.500000	7.000000	6.000000	2000.0000
max	1460.000000	190.000000	313.000000	215245.000000	10.000000	9.000000	2010.0000

8 rows × 38 columns

データの可視化：ヒストグラム

In [19]:
```
# datasetの'SalePrice'をヒストグラムで表示
housing_data = pd.read_csv("kaggle_housing_price.csv")
df=pd.DataFrame(housing_data)
df.hist('SalePrice')
```

Out[19]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000001A39C8FF5C8>]],
      dtype=object)
```

データの可視化：散布図

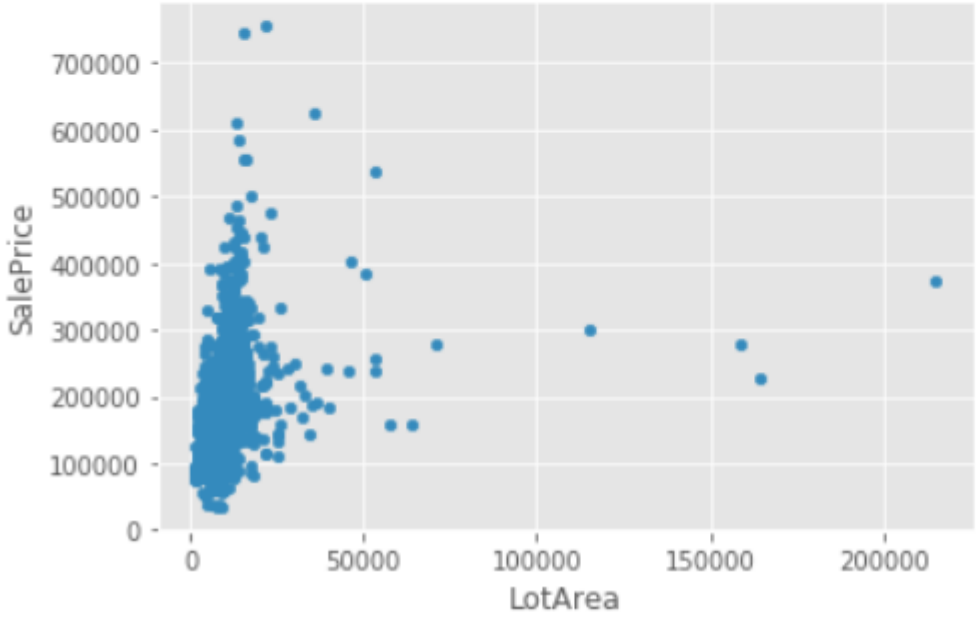
散布図

2つの変数の関係性を確認する際に有効です。DataFrameの `plot()` が使えます。

参考：http://pandas.pydata.org/pandas-docs/version/0.23/generated/pandas.DataFrame.plot.html

In [20]:
```
# datasetの'LotArea'と'SalePrice'を散布図で表示
df=pd.DataFrame(housing_data)
df.plot(kind='scatter', x='LotArea', y='SalePrice')
```

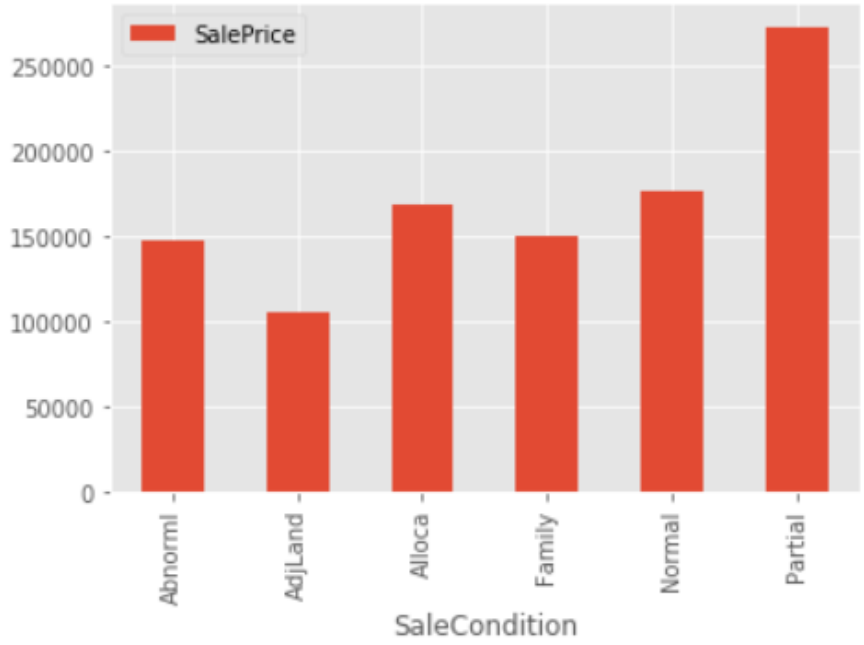
Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x1a39ccb9208>



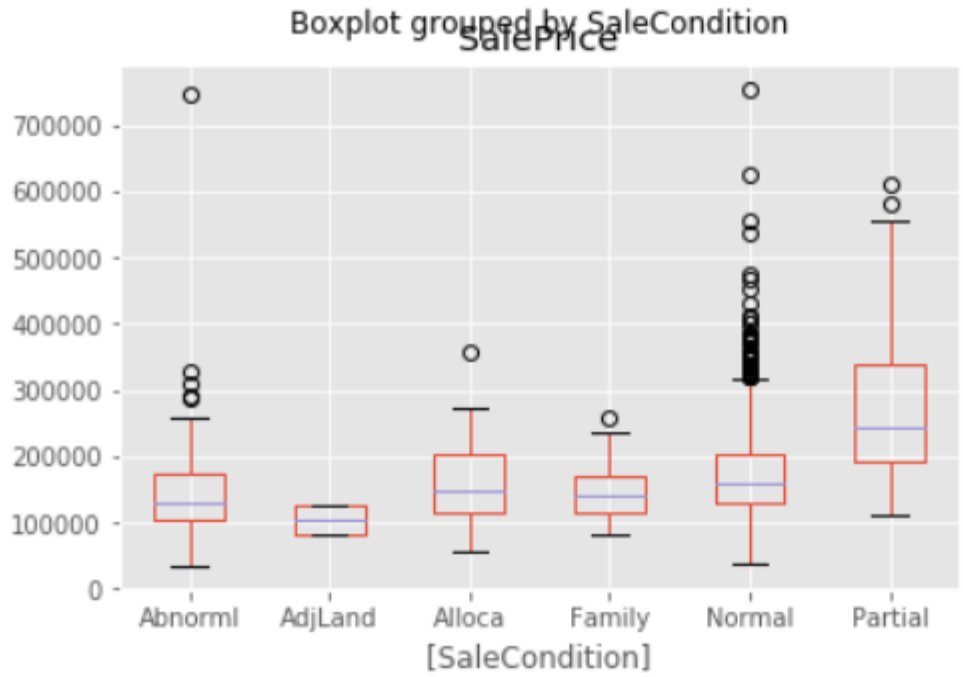
データの可視化：棒グラフ

In [21]:
```
#'SalePrice'のSaleCondition毎の平均を変数　price_by_conditionに格納
df=pd.DataFrame(housing_data)
grouped = df.groupby('SaleCondition')

price_by_condition=grouped.mean().SalePrice

df=pd.DataFrame(price_by_condition)

# price_by_conditionが持つ、棒グラフを表示する命令を実行
df.plot.bar()
```

Out[21]: <matplotlib.axes._subplots.AxesSubplot at 0x1a39dd05448>



データの可視化：箱ひげ図



箱ヒゲ図 (Boxplot)

複数の変数の分布を比較する際に有効です。（棒グラフでは平均の比較はできますが、分布全体の比較はできません)

DataFrameの `boxplot()` が使えます。

機械学習の例：犬と猫の分類

jupyter Lesson20_exam1_dogcat Last Checkpoint: 14分前 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

【機械学習1】写真に映る動物が犬か猫かを分類する

学習に使うデータセットをインポートして計測データと教師データに分ける

まずはeラーニングに記載したURLから犬と猫の画像が入ったzipファイルをダウンロードし、解凍してください。表示された dc_photos フォルダを直下（このノートブックと同じディレクトリ）にアップロードします。アップロードが完了した状態で、下記のコードを実行して、画像のデータセットを読み込んでください。

In [6]:
```
import imageio #画像処理のライブラリ。取り込んだ画像の情報（ピクセル単位）をndarray形式に変換
import numpy as np
import matplotlib.pyplot as plt       #図をプロットするためのライブラリー
from sklearn.metrics import accuracy_score     #モデルの正解率を計算するライブラリ

#図をJupyter Notebook内に表示させる
%matplotlib inline

# 写真は全て 75ピクセル x 75ピクセル のRGBカラー画像
PHOTO_SIZE = 75 * 75 * 3

# 空の配列（計測データ X と教師データ y）を用意する
X = np.empty((0, PHOTO_SIZE), np.uint8)
y = np.empty(0, np.uint8)

# 犬と猫の画像を配列形式で読み込んでXに格納（axis = 0で二次元配列の縦（行）に要素を追加する）
# y には 犬 なら 0, 猫 なら 1 で整数値のデータを追加
for i in range(1, 201):
    p1 = imageio.imread(f"dc_photos/dogs/dog-{i:03d}.jpg").reshape(1, PHOTO_SIZE)
    X = np.append(X, p1, axis = 0)
    y = np.append(y, np.array([0], dtype = np.uint8))

    p2 = imageio.imread(f"dc_photos/cats/cat-{i:03d}.jpg").reshape(1, PHOTO_SIZE)
    X = np.append(X, p2, axis = 0)
    y = np.append(y, np.array([1], dtype = np.uint8))

fig=plt.figure()
for i,j in enumerate(X[:10]):
    sp=fig.add_subplot(2,5,(i+1))
    sp.imshow(j.reshape(75,75,3),cmap="gray")
```

画像の表示・データの内部表現

```
fig=plt.figure()
for i,j in enumerate(X[:10]):
    sp=fig.add_subplot(2,5,(i+1))
    sp.imshow(j.reshape(75,75,3),cmap="gray")
```

`X` と `y` の要素数を確認してください。

In [3]:

```
# Xおよびyの要素数を確認する（命令を追記すること）
print(X.shape)
print(y.shape)
```

```
(400, 16875)
(400,)
```

In [50]:

```
print(X)
print(y)
```

```
[[199 197 123 ... 139 138  82]
 [ 62  53  46 ...  50  40  41]
 [  3   1   6 ... 182 134 130]
 ...
 [242 118  80 ... 122 101  82]
 [213 211 216 ...  47  47  49]
 [163 118  97 ... 230 204 207]]
[0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1]
```

訓練データとテストデータにわける

データを訓練データとテストデータに分ける

今回の課題において `train_test_split` は使用せず、単純に300件目で区切る形で訓練データとテストデータを分けましょう。

In [7]:

```
# 300件目までを訓練データ、301件目以降をテストデータに分割する（コメントを外した上で命令を追記すること）

X_train = X[:300]
X_test  = X[300:]
y_train = y[:300]
y_test  = y[300:]
print(X_train)
print(y_train)
print(y_test)
```

```
[[199 197 123 ... 139 138  82]
 [ 62  53  46 ...  50  40  41]
 [  3   1   6 ... 182 134 130]
 ...
 [  6  11  15 ...  69  96 113]
 [227 227 229 ...  46  46  44]
 [180 187 180 ...  34 131 138]]
[0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
 0 1 0 1]
[0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1]
```

Xの全ての要素は 0 から 255までのデータ（色の強さを255段階で表現したもの）になっていますので、スケーリングも不要です。

訓練データを用いた分類器の作成

jupyter Lesson20_exam1_dogcat Last Checkpoint: 20分前 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Markdown

0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1]

Xの全ての要素は 0 から 255までのデータ（色の強さを255段階で表現したもの）になっていますので、スケーリングも不要です。

訓練データを用いて分類器を作成する

このまま線形分類器を作成して、訓練データを設定しましょう。

サポートベクトルマシンは、2 つのカテゴリを識別する分類器です。与えられたデータを線形分離します。サポートベクトルマシンでは、境界線に最も近いサンプルとの距離（マージン）が最大となるように境界線が定義されます。

サポートベクターマシン(Support Vector Machines; SVMs)は教師あり学習 の手法としてクラス分けや回帰問題、そして外れ値検知(outlier detection)に使われている。

In [8]:
```
# 線形分類器を作成して訓練データを設定する（命令を追記すること）
from sklearn.svm import SVC
classifier = SVC(kernel = "linear")
classifier.fit(X_train, y_train)
```

Out[8]:
```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='linear',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
```

画像の分類・混合行列：結果表示

テストデータを分類器にかけて分類を実施する

テストデータを分類器にかけてください。

In [9]:
```
## テストデータを分類器にかける（命令を追記すること）
y_pred = classifier.predict(X_test)
```

結果を表示する

では、分類器が出力した結果と正解を比較してみましょう。

In [10]:
```
# 分類器の出力結果を表示する（命令を追記すること）
print(y_pred)
```

```
[0 1 0 1 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 0 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 1 0
 0 0 0 0 1 0 1 0 1 0 0 0 1 0 1 0 1 0 1 0 1 0 1 0 0 0 1 1 1 0 0 0 1 0 0 0 0
 0 0 0 1 0 1 0 1 0 1 0 0 0 1 0 0 0 1 0 1 0 0 0 1 0 1]
```

In [55]:
```
# 正解を表示する（命令を追記すること）
print(y_test)
```

```
[0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1]
```

In [11]:
```
# 混合行列を表示する（命令を追記すること）
from sklearn import metrics
print(metrics.confusion_matrix(y_test, y_pred))
```

```
[[47  3]
 [12 38]]
```

In [57]:
```
# 正答率を確認する（命令を追記すること）
print(metrics.classification_report(y_test, y_pred))
```

```
             precision    recall  f1-score   support

          0       0.80      0.94      0.86        50
          1       0.93      0.76      0.84        50

avg / total       0.86      0.85      0.85       100
```

課題：手書き文字」の分類へ

名称は「AI概論」ですが….

- 工学系学生：1年時にプログラミングの経験・必携PCの利用
  Pythonを選択　※開発環境Anacondaを利用
  Pythonで利用されるライブラリを含んだ開発環境
  Pandas（データ分析）、Matplotlib（グラフ描画）
  Numpy　　scikit-learn(サイキットラーン)
  ※オープンソースの機械学習ライブラリ

- 手を動かすプログラミングによる演習が半分

2年「AI活用演習」どこまでやれる？

① 導入(AI活用最新動向、活用例、学習理論)
② 基礎統計、予測・検定に関する理解（新井先生：座学）　演習①
③ 単回帰分析、重回帰分析の理解
④ SVM学習画像分類　（機械学習の演習例として）　演習②
⑤ 決定木、主成分分析　　演習③　execise3.pdf
⑥ クラスタリング　　演習④　execise4.pdf
⑦ ニューラルネットワークの基礎の理解
⑧ Pythonによるニューラルネットワーク演習　　演習⑤
⑨ CNNによる画像分類
⑩ PythonによるCNN画像分類　　演習⑥
⑪ RNN(LSTM)による時系列解析の理解
⑫ PythonによるRNN(LSTM)による時系列解析演習　演習⑦
⑬ DQNによる強化学習の理解
⑭ PythonによるDQNによる強化学習演習　　演習⑧
⑮ AIの課題と今後の発展

jupyter execise1 Last Checkpoint: 前週月曜日 11:36 (autosaved)

Logout

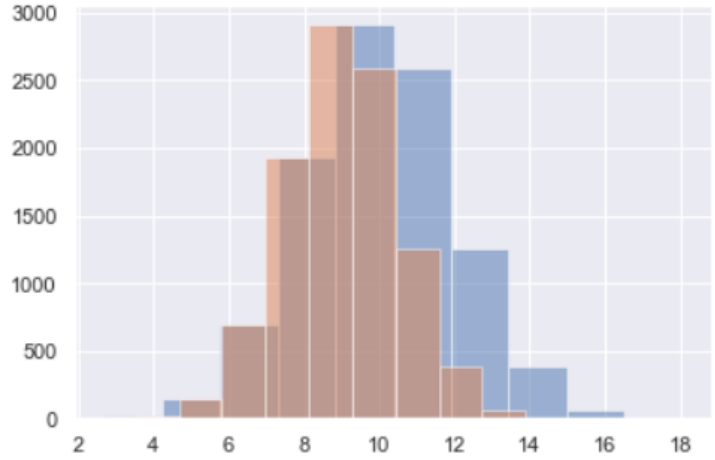
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Markdown

```
8.90595566,  6.84319516])
```

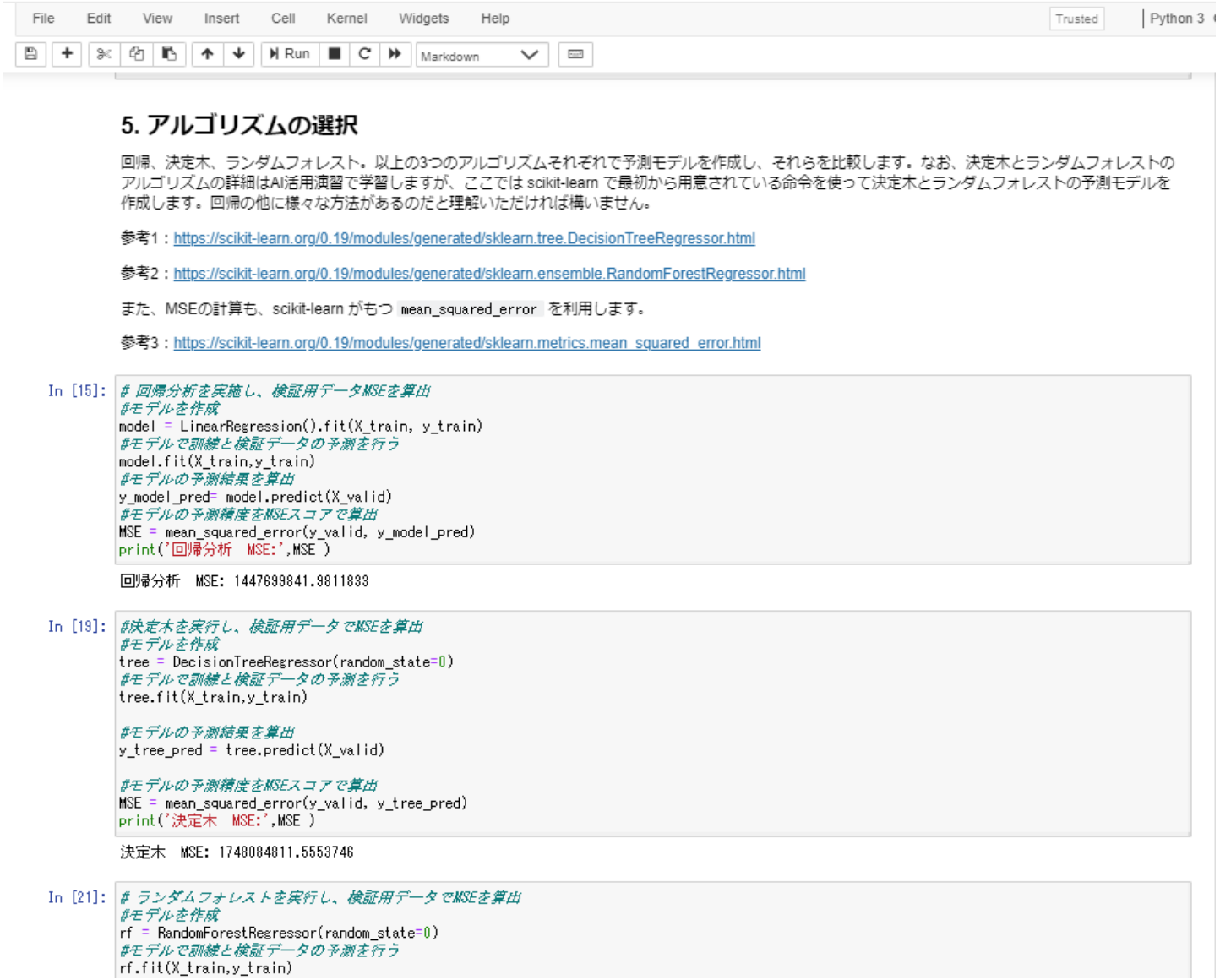
In [40]:
```
# data1とdata2をヒストグラムにてプロット
plt.figure()
plt.hist(data1,alpha=0.5)
plt.hist(data2,alpha=0.5)
```

Out[40]:
```
(array([  14.,  151.,  699., 1932., 2909., 2594., 1248.,  388.,   55.,
         10.]),
 array([ 3.51533985,  4.66783322,  5.82032659,  6.97281997,  8.12531334,
         9.27780671, 10.43030008, 11.58279345, 12.73528682, 13.8877802 ,
        15.04027357]),
 <a list of 10 Patch objects>)
```



In [42]:
```
# data1, data2それぞれの値を使って「対応のあるt検定」で計算し、結果を表示
stats.ttest_1samp(data1, data2)
```

AI活用演習　アルゴリズムの比較



5. アルゴリズムの選択

回帰、決定木、ランダムフォレスト。以上の3つのアルゴリズムそれぞれで予測モデルを作成し、それらを比較します。なお、決定木とランダムフォレストのアルゴリズムの詳細はAI活用演習で学習しますが、ここでは scikit-learn で最初から用意されている命令を使って決定木とランダムフォレストの予測モデルを作成します。回帰の他に様々な方法があるのだと理解いただければ構いません。

参考1：https://scikit-learn.org/0.19/modules/generated/sklearn.tree.DecisionTreeRegressor.html

参考2：https://scikit-learn.org/0.19/modules/generated/sklearn.ensemble.RandomForestRegressor.html

また、MSEの計算も、scikit-learn がもつ `mean_squared_error` を利用します。

参考3：https://scikit-learn.org/0.19/modules/generated/sklearn.metrics.mean_squared_error.html

In [15]:
```
# 回帰分析を実施し、検証用データMSEを算出
#モデルを作成
model = LinearRegression().fit(X_train, y_train)
#モデルで訓練と検証データの予測を行う
model.fit(X_train,y_train)
#モデルの予測結果を算出
y_model_pred= model.predict(X_valid)
#モデルの予測精度をMSEスコアで算出
MSE = mean_squared_error(y_valid, y_model_pred)
print('回帰分析　MSE:',MSE )
```

```
回帰分析　MSE: 1447699841.9811833
```

In [19]:
```
#決定木を実行し、検証用データでMSEを算出
#モデルを作成
tree = DecisionTreeRegressor(random_state=0)
#モデルで訓練と検証データの予測を行う
tree.fit(X_train,y_train)

#モデルの予測結果を算出
y_tree_pred = tree.predict(X_valid)

#モデルの予測精度をMSEスコアで算出
MSE = mean_squared_error(y_valid, y_tree_pred)
print('決定木　MSE:',MSE )
```

```
決定木　MSE: 1748084811.5553746
```

In [21]:
```
# ランダムフォレストを実行し、検証用データでMSEを算出
#モデルを作成
rf = RandomForestRegressor(random_state=0)
#モデルで訓練と検証データの予測を行う
rf.fit(X_train,y_train)
```

最終評価・エラー分析

6. モデルの最終評価

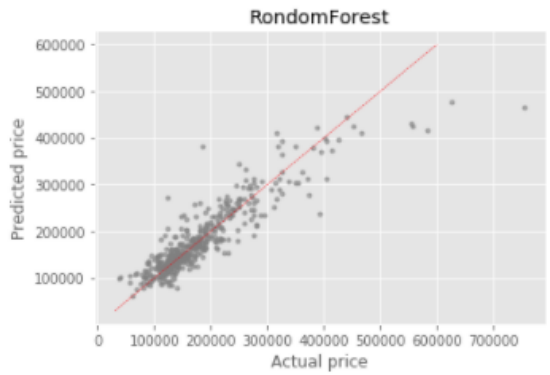
3つのモデルの中ではMSEの値が最も良かったランダムフォレストについて、テストデータでモデルの最終的な評価をしてください。

In [22]:
```
# Test dataを用いてMSEを算出し、予測精度を確認

#モデルを作成
#rf = RandomForestRegressor(random_state=0)
#モデルで訓練とtestデータの予測を行う
#rf.fit(X_train,y_train)
#モデルの予測結果を算出
y_rf_pred = rf.predict(X_test)
#モデルの予測精度をMSEスコアで算出
MSE = mean_squared_error(y_test, y_rf_pred)
print('ランダムフォレスト　MSE:',MSE )
```

ランダムフォレスト　MSE: 1410013593.182802

7. エラー分析

In [23]:
```
# 横軸：実際の成約価格、縦軸：予測した成約価格で散布図を作成し予測の誤差を確認# 予測データ（y_pred）と真値（y_test）を描画
plt.title('RondomForest')
plt.plot(y_test,y_rf_pred,'.', c='gray', alpha=.6)

minvalue = 30000
maxvalue = 600000
plt.plot((minvalue, maxvalue), (minvalue, maxvalue), 'r--', lw=0.5)
plt.xlabel("Actual price")
plt.ylabel("Predicted price")
plt.show()
```



In [24]:
```
# 実際の成約価格と予測価格の誤差率をヒストグラムで表示
```

アルゴリズムの選択・比較

5. アルゴリズムの選択

回帰、決定木、ランダムフォレスト。以上の3つのアルゴリズムそれぞれで予測モデルを作成し、それらを比較します。なお、決定木とランダムフォレストのアルゴリズムの詳細はレッスン7で学習しますが、ここでは scikit-learn で最初から用意されている命令を使って決定木とランダムフォレストの予測モデルを作成します。回帰の他に様々な方法があるのだと理解いただければ構いません。

参考1：https://scikit-learn.org/0.19/modules/generated/sklearn.tree.DecisionTreeRegressor.html

参考2：https://scikit-learn.org/0.19/modules/generated/sklearn.ensemble.RandomForestRegressor.html

また、MSEの計算も、scikit-learn がもつ `mean_squared_error` を利用します。

参考3：https://scikit-learn.org/0.19/modules/generated/sklearn.metrics.mean_squared_error.html

In [8]:
```
# 回帰分析を実施し、検証用データMSEを算出
#モデルを作成
model = LinearRegression().fit(X_train, y_train)
#モデルで訓練と検証データの予測を行う
model.fit(X_train,y_train)
#モデルの予測結果を算出
y_model_pred= model.predict(X_valid)
#モデルの予測精度をMSEスコア（平均二乗誤差）で算出
MSE = mean_squared_error(y_valid, y_model_pred)
print('回帰分析　MSE:',MSE )
```

```
回帰分析　MSE: 1160497759.0386035
```

アルゴリズムの比較


```
回帰分析　MSE: 1160497759.0386035
```

In [9]:
```
#決定木を実行し、検証用データでMSEを算出
#モデルを作成
tree = DecisionTreeRegressor(random_state=0)
#モデルで訓練と検証データの予測を行う
tree.fit(X_train,y_train)

#モデルの予測結果を算出
y_tree_pred = tree.predict(X_valid)

#モデルの予測精度をMSEスコアで算出
MSE = mean_squared_error(y_valid, y_tree_pred)
print('決定木　MSE:',MSE )
```

```
決定木　MSE: 1594185594.9869707
```

In [10]:
```
# ランダムフォレストを実行し、検証用データでMSEを算出
#モデルを作成
rf = RandomForestRegressor(random_state=0)
#モデルで訓練と検証データの予測を行う
rf.fit(X_train,y_train)
#モデルの予測結果を算出
y_rf_pred = rf.predict(X_valid)
#モデルの予測精度をMSEスコアで算出
MSE = mean_squared_error(y_valid, y_rf_pred)
print('ランダムフォレスト　MSE:',MSE )
```

```
C:¥Users¥hagoromo¥Anaconda3¥lib¥site-packages¥ipykernel_launcher.py:5: DataConversionWarning: A column-vec
tor y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example u
sing ravel().
  """
```

```
ランダムフォレスト　MSE: 1081123869.062658
```

モデルの最終評価

6. モデルの最終評価

3つのモデルの中ではMSEの値が最も良かったランダムフォレストについて、テストデータでモデルの最終的な評価をしてください。

In [11]:
```
# Test dataを用いてMSEを算出し、予測精度を確認

#モデルを作成
#rf = RandomForestRegressor(random_state=0)
#モデルで訓練とtestデータの予測を行う
#rf.fit(X_train,y_train)
#モデルの予測結果を算出
y_rf_pred = rf.predict(X_test)
#モデルの予測精度をMSEスコアで算出
MSE = mean_squared_error(y_test, y_rf_pred)
print('ランダムフォレスト　MSE:',MSE )
```

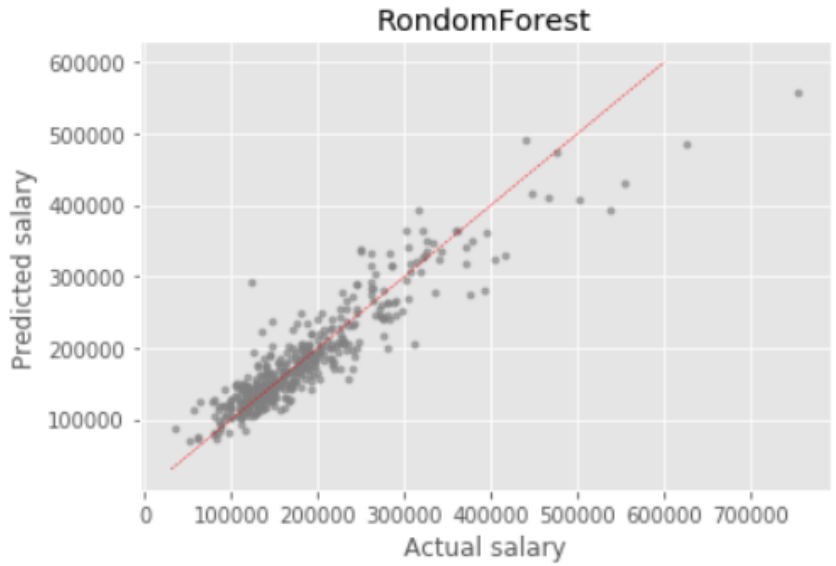
```
ランダムフォレスト　MSE: 1090494275.3098497
```

実際の値と予測値の比較

久留米工業大学
KURUME INSTITUTE OF TECHNOLOGY

7. エラー分析

In [12]:
```
# 横軸：実際の成約価格、縦軸：予測した成約価格で散布図を作成し予測の誤差を確認# 予測データ（y_pred）と真値
plt.title('RondomForest')
plt.plot(y_test,y_rf_pred,'.', c='gray', alpha=.6)

minvalue = 30000
maxvalue = 600000
plt.plot((minvalue, maxvalue), (minvalue, maxvalue), 'r--', lw=0.5)
plt.xlabel("Actual salary")
plt.ylabel("Predicted salary")
plt.show()
```



全学生対象の内容

- AI数学として数学の基礎を学ぶ
- Pythonによるプログラミングの基礎を学ぶ体験
- Pythonによるデータ解析・視覚的理解
- 機械学習（ディープラーニング）の体験

AIで地域課題などの解決ができる人材育成

「本学の地域性、特色を表す教育プロ グラム」

予定：2021年前期「AI活用演習」

■AI活用演習の１クラス： ５学科混成の優秀者クラスを作る

※AI技術を生かした地域課題解決に取り組む学生トップランナーを育成

■Udemyのアカデミックサブスクリプションを導入予定

成績上位層への教育リソースの提供→自ら学ぶ学生へ

■目指す大学像を実現するための地域課題解決力を持った学生を育成

■大学キャンパス外のデータを使った実装の実績

地域企業の生データを使ったAI教育プログラム